Algorithms and Complexity Results for Discrete Probabilistic Reasoning Tasks

Doctoral Dissertation submitted to the
Faculty of Informatics of the Università della Svizzera Italiana
in partial fulfillment of the requirements for the degree of
Doctor of Philosophy

presented by
Denis Deratani Mauá

under the supervision of
Jürgen Schmidhuber and Marco Zaffalon and Cassio Polpo de Campos

September 2013

Dissertation Committee

Fabio Crestani	Università della Svizzera Italiana, Switzerland
Fabian Kuhn	Università della Svizzera Italiana, Switzerland
Thomas D. Nielsen	Aalborg University, Denmark
Serafín Moral	Univesity of Granada, Spain

Dissertation accepted on 17 September 2013

Research Advisor

Jürgen Schmidhuber

Co-Advisor

Marco Zaffalon and Cassio Polpo de Campos

PhD Program Director

Antonio Carzaniga

I certify that except where due acknowledgement has been given, the work presented in this thesis is that of the author alone; the work has not been submitted previously, in whole or in part, to qualify for any other academic award; and the content of the thesis is the result of work which has been carried out since the official commencement date of the approved research program.

Denis Deratani Mauá
Lugano, 17 September 2013

To Karina

Abstract

Many solutions to problems in machine learning and artificial intelligence involve solving a combinatorial optimization problem over discrete variables whose functional dependence is conveniently represented by a graph. This thesis addresses three types of these combinatorial optimization problems, namely, the *maximum a posteriori* inference in *discrete probabilistic graphical models*, the selection of optimal strategies for *limited memory influence diagrams*, and the computation of upper and lower probability bounds in *credal networks*.

These three problems arise out of seemingly very different situations, and one might believe that they share no more than the graph-based specification of their inputs or the underlying probabilistic treatment of uncertainty. However, correspondences among instances of these problems have long been noticed in the literature. For instance, the computation of probability bounds in credal networks can be reduced either to the problem of maximum a posteriori inference in graphical models, or to the selection of optimal strategies in limited memory influence diagrams. Conversely, both the maximum a posteriori inference and the strategy selection problems can be reduced to the computation of a probability bound in a credal network. These reductions suggest that much insight can be gained by carrying out a joint study of the practical and theoretical computational complexity of these three problems.

This thesis describes algorithms and complexity results for these three classes of problems. In particular, we develop a new anytime algorithm for the maximum a posteriori problem. Not only the algorithm is of practical relevance, as we show that it compares favorably against a state-of-the-art method, but it is the base of the proof of polynomial-time approximability of the two other problems. We characterize the tractability of the strategy selection problem according to the input parameters, and we show that the strategy selection problem can be solved in polynomial time in singly connected diagrams over binary variables and univariate utility functions, and that relaxing any of these assumptions makes the problem NP-hard to solve or even approximate within any bound.

We also investigate the theoretical complexity of computing upper and lower

probability bounds in credal networks. We show that the complexity of the problem depends on the irrelevance concept adopted, but is in general NP-hard even in polytree-shaped networks, and even in trees if we assume strong independence. We also show that there is a particular type of inference that can be solved in polynomial time in imprecise hidden Markov models, whether we assume epistemic irrelevance or strong independence.

Acknowledgements

This work would not have been possible without the financial aid of the Swiss National Science Foundation through the grants no. 200020_134759/1 and 200020_116674/1, and without the consent and supervision of Jürgen Schmidhuber. It would not have been as fruitful as I believe it was without the close guidance, the patience and the expertise of Cassio Polpo de Campos and Marco Zaffalon. It certainly would not have been as pleasant without the companies of Alessandro Antonucci, Alessio Benavoli, Giogio Corani, Alessandro Giusti, Cassio Polpo de Campos, Francesca Mangili and Fu Shuai, with whom I have shared not only an office and with some co-authorship in papers, but ideas, laughs, doubts and experiences. Indeed, I own to the whole group – PhD students, Researchers, Engineers and Staff – my very enjoyable four-year stay at IDSIA, and I would (and perhaps should) explicitly acknowledge many other people whose company provided me with great pleasure, and whose help was at times much needed, but I wanted to be brief here; excuse me for that. The final version of this manuscript received much valuable feedback from the dissertation committee members, for which I am very grateful. At last, I would have not been able to accomplish this work without all the financial, emotional and motivational support that I received from my parents, and for that I thank them.

Contents

Chapter 1

Introduction

Many solutions to problems in machine learning and artificial intelligence involve solving a combinatorial optimization problem over variables whose functional dependence is conveniently represented by a graph. Three of these optimization problems are the *maximum a posteriori* inference in *discrete probabilistic graphical models* (specifically, in Bayesian and Markov networks, [Park and Darwiche, 2004]), the selection of optimal strategies for *limited memory influence diagrams* [Lauritzen and Nilsson, 2001], and the computation of upper and lower probability bounds in *credal networks* [Cozman, 2000].

Arguably, the most studied of these three classes of problems is the problem of maximum a posteriori inference, which consists in finding an assignment of the values of a given subset of the variables in the model that maximizes their marginal joint probability. The theoretical computational complexity of this task has been largely determined by Park and Darwiche [2004] and de Campos [2011], in terms of the shape of the underlying graph, the cardinality of variable domains and the number of latent variables (i.e., variables which are marginalized out from the model). In particular, de Campos [2011] showed that the problem is NP-hard to approximate even in tree-shaped Bayesian networks, unless the cardinality of the variables is assumed bounded, in which case it admits a fully polynomial-time approximation scheme. On the practical side, many efficient approximate algorithms have been devised [Dechter and Rish, 2003; Park and Darwiche, 2004; Liu and Ihler, 2011; Jiang et al., 2011; Meek and Wexler, 2011]. These algorithms however are not able to provide a solution with a given accuracy using a given limited amount of computational resources (i.e., time and memory). To remedy this situation, we develop in Chapter 2 an anytime algorithm to the problem, which we show empirically to compare favorably against the only other anytime procedure for the problem we are aware of. The key

feature of our algorithm is the trade-off between efficiency and accuracy, which is achieved by designing a multiplicative approximation scheme whose accuracy and complexity is determined by the user (hence, part of the input). Besides its practical relevance, the anytime algorithm is the base for proving approximability results regarding the complexity of the two other classes of problems. Indeed, the fully polynomial-time approximation scheme of de Campos [2011] can be seen as a special case of our anytime algorithm for networks of bounded treewidth and bounded variable cardinality.

The second class of problems we study is the selection of optimal strategies for limited memory influence diagrams. Influence diagrams are intuitive and concise representations of decision making situations [Howard and Matheson, 1984]. A decision-making problem usually involves both controllable and non-controllable quantities, which in the formalism of influence diagrams are represented, respectively, by action and state variables. A strategy is a mapping from state variables into action variables, which completely determines the behavior of an agent (or a team of cooperative agents) acting under the model. The specification of an influence diagram and a suitable strategy uniquely determines a joint probability distribution over the state variables, and a rational agent tries to select a strategy that maximizes expected utility over these probability distributions. In principle, an optimal action at a given decision step might depend on all previous actions and observations, which leads to an exponentially large strategy. To avoid such an exponential complexity, Lauritzen and Nilsson [2001] proposed using limited memory influence diagrams, in which the information available to each local decision is explicitly determined as part of the input, and hence under the control of the model builder. They showed the existence of a class of limited memory influence diagrams for which the optimal strategy remains the same if we relax the constraint on the size of admissible strategies, that is, allowing each local decision to be made based on the full history of actions and observations does not increase expected utility. They named those diagrams *soluble*, showed that membership of a diagram in the class of soluble diagrams can be verified in polynomial time, and developed an algorithm that finds an optimal strategy for a soluble diagram in time exponential in the treewidth of the underlying graph; the algorithm is thus polynomial time on soluble diagrams of bounded treewidth. In Chapter 3, we investigate the theoretical complexity of this problem in terms of the diagram shape, the cardinality of variable domains, and the structure of the utility function. We conclude that optimal strategies can be found in polynomial time in diagrams of bounded treewidth over binary state variables and with a univariate utility function, and that relaxing any of these conditions results in an NP-hard problem. This shows that the class of soluble

diagrams is a very restrictive one, and even structurally simple diagrams can be difficult to solve. We also show that relaxing optimality does make the problem easier, as we prove that a provably good strategy can be found in polynomial-time in (non-soluble) diagrams of bounded treewidth over variables of bounded cardinality, and that the same task is NP-hard in case variables assume arbitrarily many values.

The third class of problems we consider involves credal networks, which are graphical models whose numerical parameters are imprecisely specified through sets of probabilities [Cozman, 2000]. The set-valued quantification allows for the distinction of uncertainty and indeterminacy, the former being the result of (partial) ignorance about facts, and the latter being the incapability of acting under severe uncertainty. Arguably, this distinction facilitates the knowledge elicitation from experts [Walley, 2000; Antonucci et al., 2007], and allows for more realistic models of one's beliefs [Walley, 1991]. A credal network determines a closed and convex set of joint probability distributions of the variables in the model. The updating problem is to compute the maximum and minimum values for the posterior probability of a given variable taking on a certain value over all the joint distributions determined by the network. The theoretical complexity of the updating problem depends on the precise characterization of the set of joint distributions induced by a credal network. This joint set is usually precisely characterized by assigning a proper semantics to the arcs of the network graph. Two common approaches are to assume that the graph encodes a set of either strong independence or epistemic irrelevance assessments. The former implies that the joint set arises out of a finite combination of precisely specified Bayesian networks all of which have a graph structure that coincides with that of the credal network. The latter takes imprecision as a core feature of the model, and describes the joint set as a rational extension to a larger domain of the assessments about local domains provided by the set-valued specifications in the input; the distributions in the joint set need not be related to any Bayesian network sharing the exact same graph structure of the credal network.

De Campos and Cozman [2005] showed that under strong independence the updating problem is NP-hard to approximate even in singly connected networks of bounded treewidth, unless all the variables are binary, in which case the problem can be solved in polynomial time by the 2U algorithm of Fagiuoli and Zaffalon [1998b]. Under epistemic irrelevance, de Cooman et al. [2010] showed that the problem can be solved in polynomial time if the underlying graph is a tree. In Chapter 4, we extend these results and show that updating singly connected credal networks is NP-hard under either type of irrelevance, even if the root variables are binary and non-root variables are ternary. We also

show that under strong independence the problem is NP-hard already in trees, but that a particular type of query can be answered in polynomial time in hidden Markov models, a special class of tree-shaped credal networks well-suited for the analysis of time series. Finally, we show that a fully polynomial-time approximation scheme for the problem exists if we assume that both the network treewidth and the variable cardinalities are bounded, and adopt strong independence.

These three classes of problems arise out of seemingly very different situations, and one might believe that they share no more than the graph-based specification of the models or the probabilistic treatment of uncertainty. However, correspondences among instances of these problems have long been noticed in the literature. Provided that the specification of the credal network satisfies certain conditions such as sets being represented by their extreme points and strong independence being adopted, Cano et al. [1994] showed that this problem can be efficiently reduced to the computation of a maximum posteriori inference in a properly designed Bayesian network. De Campos and Cozman [2005] showed that the converse is also true, that is, maximum a posteriori inference in Bayesian networks can be efficiently reduced to an updating problem in a credal network that satisfies the conditions of the previous reduction. Antonucci and Zaffalon [2006, 2008] and de Campos and Ji [2008] studied the correspondence between updating credal networks and selecting optimal strategies in limited memory influence diagrams. In particular, they showed that the updating problem in credal networks satisfying the same conditions and the strategy selection problem in limited memory influence diagrams can be efficiently reduced one to another.

The existence of efficient reductions between instances of these three different classes of problems brings important consequences, both practical and theoretical. On the practical side, the reductions enable algorithms developed for one problem to be directly used to solve the other, thus enriching the toolsets for solving each of the three problems. On the theoretical side, they lead to a different interpretation of each problem, as seen from the viewpoint of the reduced problem. For instance, Antonucci and Zaffalon [2008] used the reduction of the updating problem in credal networks into the strategy selection problem in limited memory influence diagrams to propose a decision-theoretic representation of credal networks, by which the imprecision in the numerical parameters is seen as a decision problem. This decision-theoretic representation of credal networks allows for the logical constraints in the set-valued specification of parameters to be graphically and intuitively expressed in the language of influence diagrams. Also, the reductions allow complexity results for one problem to be immediately derived from results about another problem. For example, by reducing the updating problem into a maximum a posteriori inference problem, we can show

that the former is an NP^{PP}-hard problem, as this is the case for the latter [Park and Darwiche, 2004]. These correspondences however do not render redundant the parametrized complexity results obtained for each class of problems, as the reductions usually do not preserve all of the structure of the original problem. For example, the reduction from an updating problem in credal networks into a maximum a posteriori inference transforms a tree-shaped credal network into a graphical model whose graph structure is not a tree. Other reductions insert cycles that were previously absent, or do not preserve approximation.

To sum up, this dissertation describes a new anytime algorithm for the computation of maximum a posteriori inference in discrete probabilistic graphical models (more precisely, in Bayesian and Markov networks), and contains a study about the computational complexity of the problems of selecting optimal strategies in limited memory influence diagrams, and computing upper and lower posterior probabilities in credal networks. Correspondence between these three different problems are used to prove both positive and negative complexity results, justifying their joint treatment.

1.1 Contributions

The following is a summary of the most important contributions of this work, annotated with references to the publications where they (partially) appeared previously.

1. We develop a new anytime algorithm for the maximum a posteriori inference problem in graphical models. The algorithm is carefully analyzed and is shown empirically to be very competitive. Most of this part appeared previously in

   > Mauá, D. D. and de Campos, C. P. [2012]. Anytime marginal MAP inference, *Proceedings of the 28th International Conference on Machine Learning (ICML)*.

2. We develop an algorithm to find optimal strategies in limited memory influence diagrams. We show empirically that the algorithm, which has exponential worst-case running time performance, is able to solve exactly much bigger instances than the current state-of-the-art solvers. By using this algorithm, we are able to evaluate the performance of approximate algorithms on more realistic inputs. This work appeared in

> Mauá, D. D. and de Campos, C. P. [2011]. Solving decision problems with limited information, *Advances in Neural Information Processing Systems 24 (NIPS)*, pp. 603–611.

3. We show that the problem of selecting an optimal strategy in limited memory influence diagram is NP-hard to solve in multiply connected diagrams over binary variables and a single value node, and in singly connected diagrams with ternary variables and a single value node. Moreover, we prove the problem to be NP-hard to approximate within any constant bound even in singly connected diagrams of bounded treewidth and a single value node. These results appeared in

> Mauá, D. D., de Campos, C. P. and Zaffalon, M. [2012]. Solving limited memory influence diagrams, *Journal of Artificial Intelligence Research* **44**: 97–140.

4. We prove the existence of a fully polynomial-time approximation scheme for the strategy selection problem for diagrams of bounded treewidth over variable of bounded cardinality. This appeared in

> Mauá, D. D., de Campos, C. P. and Zaffalon, M. [2012]. The complexity of approximately solving influence diagrams, *Proceedings of the 28th Conference on Uncertainty in Artificial Intelligence (UAI)*, pp. 604–613.

5. Complementing previous results, we show that the strategy selection problem is NP-hard to solve even in singly connected diagrams over binary variables, unless there is a single value node, in which case we show the problem can be solved in polynomial time. This result has been submitted to a journal and is currently under review.

6. We study the parametrized complexity of the updating problem in credal networks. We show that under strong independence the problem is NP-hard even in trees. More generally, the problem is NP-hard even in polytrees over binary root variables and ternary non-root variables, whether we assume strong independence or epistemic irrelevance. These results appeared in

> Mauá, D. D., de Campos, C. P., Benavoli, A. and Antonucci, A. [2013]. On the complexity of strong and epistemic credal net-

works, *Proceedings of the 29th Conference on Uncertainty in Artificial Intelligence (UAI)*, pp. 391–400.[1]

7. We prove the existence of a fully polynomial-time approximation scheme for the updating problem in credal networks of bounded treewidth over variables of bounded cardinality, and assuming strong independence. We show empirically that the approximation outperforms a state-of-the-art approach for the problem. These results appeared in

   Mauá, D. D., de Campos, C. P. and Zaffalon, M. [2011]. A fully polynomial time approximation scheme for updating credal networks of bounded treewidth and number of variable states, *Proceedings of the Seventh International Symposium on Imprecise Probability: Theories and Applications*, pp. 277–286.

   and

   Mauá, D. D., de Campos, C. P. and Zaffalon, M. [2012]. Updating credal networks is approximable in polynomial time, *International Journal of Approximate Reasoning* **53**(8): 1183–1199.

In order to keep the length of this dissertation reasonable, work on closely related subjects performed during the course of the author's Ph.D. studies had to be omitted from this account, such as the evaluation of classifiers based on credal networks [Zaffalon et al., 2011, 2012], and the design of multilabel classifiers based on ensembles of Bayesian networks [Antonucci et al., 2013a].

1.2 Organization of the thesis

The rest of this document is organized in three chapters, each containing the results about one of the three classes of problems described. Chapter 2 contains the discussion on the problem of selecting maximum a posteriori configurations in discrete probabilistic graphical models. The complexity analysis of the strategy selection problem in limited memory influence diagrams is then presented in Chapter 3. The complexity results of updating credal networks are reported in Chapter 4. Finally, the overall conclusions of this work, together with directions on future work, are laid out in Chapter 5.

[1] This paper was awarded the Google Best Student Paper at UAI 2013.

Chapter 2

Maximum a posteriori inference

Finding a mode of a probability distribution over a large number of discrete variables, a task more commonly known as *maximum a posteriori* (MAP) inference, is a building block of many solutions to important applications such as image segmentation and categorization, 3D image reconstruction, natural language parsing, statistical machine translation, speech recognition, sentiment analysis, protein design, and multi-component fault diagnosis, to name but a few.

To allow for efficient manipulation and tractable inference, probability distributions need to be concisely encoded. A class of models that achieves this goal is the class of *probabilistic graphical models*. These are multivariate models where irrelevance assessments between sets of variables are concisely described by means of a graph whose nodes are identified with variables [Pearl, 1988; Koller and Friedman, 2009]. Graphical models are usually distinguished according to whether their underlying graphical structure is directed. *Bayesian networks* are models in which a directed acyclic graph is used to represent a set of *local Markov conditions*: a variable is independent of the variables associated to non-descendant non-parent nodes given the variables associated to its parents. *Markov random fields*, on the other hand, encode independence assessments by means of undirected graphs. In these models, the local Markov condition states that given (the variables associated to) its neighbors a variable is independent of its non-neighbors.

Although the type of graphical model chosen (i.e., directed or undirected) might greatly affect the complexity of specifying the requisite numerical parameters of the model, it has little effect on the complexity of maximum a posteriori inference in discrete models. Indeed, by including new variables and/or setting some evidence, every Bayesian network can be efficiently transformed into an equivalent Markov network that assigns the same probability value for all vari-

able assignments, and the converse is also true. Moreover, most algorithms can be applied equally to either class of graphical models. For these reasons, we shall not pursue the distinction between directed or undirected models in this chapter, and we shall adopt the language of *factor graphs* as a unifying visual representation of either type of model.

Probabilistic models often include latent variables, that is, variables that are not directly observable and yet are understood as important for modeling the phenomenon at hand. The inclusion of latent variables helps in eliciting the model from experts, and decreases the number of parameters required to capture reasonably well the dependencies in the model, which can prevent overfitting when learning models from data and lead to better results [Kwoh and Gillies, 1996; Binder et al., 1997; Friedman, 1998; Elidan et al., 2000; Zhang, 2004; Elidan and Friedman, 2005; Wang et al., 2013]. For instance, in determining the premium of a car insurance application, the driving skills and attitude of the applicant are important features which cannot be directly observed (by the analyst). Yet it is believed that these features strongly determine the likelihood of e.g. a driver being involved in a car accident and at the same time are strongly influenced by other personal factors such as the driver's age [Binder et al., 1997]. Neglecting these features from the model drastically increases the number of parameters required to model the correlations between variables, which makes the model more difficult to specify and less effective [Friedman, 1998].

The problem of finding a mode of a discrete probabilistic graphical model is notoriously hard, and the presence of latent variables increases its computational complexity. For instance, the decision version of the MAP inference problem is $\mathrm{NP}^{\mathrm{PP}}$-complete if the model contains (arbitrarily many) latent variables, while the same problem is "only" NP-complete if latent variables make up a bounded fraction of the variables [Shimony, 1994; Park and Darwiche, 2004]. When the underlying graph is a tree, the problem with (arbitrarily many) latent variables is NP-complete [Park and Darwiche, 2004], whereas the same problem can be solved in polynomial time in the absence of latent variables [Koller and Friedman, 2009]. Also finding a provably good approximation in trees of bounded degree is NP-hard in the presence of latent variables [Park and Darwiche, 2004] (however, a fully polynomial-time approximation scheme exists if the number of states per variable is assumed bounded, de Campos [2011]). Table 2.1 lists some of the known complexity results of the MAP inference problem in Bayesian networks (analogous results can be stated for Markov networks). Polytrees and loopy networks are defined in Chapter 3.

The theoretical difficulty of the problem motivates the search for approxi-

Table 2.1. Parametrized complexity of the MAP problem in Bayesian networks.

topology	treewidth	max. variable cardinality	complexity
"naive" tree	one	unbounded	NP-complete
tree	one	five	NP-complete
polytree	two	two	NP-complete
polytree	two	unbounded	NP-complete
loopy	unbounded	two	NP^{PP}-complete

mate solutions. Recently, there has been a growing interest in the problem, with the development of many new approximate algorithms. Most of these algorithms provide solutions that can be arbitrarily poor, which might prevent the user of such algorithms from fully understanding the effects of approximate inference in the bigger picture of the application, that is, to know whether better results could be obtained by improving the quality of inference, or if it is the model or methodology that are fundamentally flawed. This is the case of message-passing and beam-search algorithms [Park and Darwiche, 2003; Dechter and Rish, 2003; de Campos et al., 2003; Yuan et al., 2004; Huang et al., 2006; Yuan and Hansen, 2009; Liu and Ihler, 2011; Jiang et al., 2011]. Such worries have very recently been addressed by Meek and Wexler [2011] and Cheng et al. [2012], who designed algorithms that are able to provide bounds within which the (probability of the) true solution is to be found.[1] Moreover, these algorithms allow for some trade-off between the tightness of the bounds and the amount of computational resources (memory and time) used. Thus, loose bounds can justify an increase in processing time dedicated to inference if the final results turn out to be unsatisfactory, whereas tight bounds can reassure the quality of an efficient approximate inference algorithm. Moreover, bounds are necessary to account for alternative explanations in case conclusions are to be drawn from the result of the inference, as in scientific discovery (e.g., in finding associations between genes or proteins).

In the rest of this chapter, we briefly review some of the approaches to solve

[1] Some approximate methods such as the systematic search of Park and Darwiche [2003], the mini-bucket scheme of Dechter and Rish [2003] and the tree-reweighted variant of Liu and Ihler [2011] provide a "side guarantee", that is, they are either inner or outer approximations. We can obtain an algorithm that provides bounds for the mode probability by combining an inner and an outer approximation algorithm.

the problem and present a new algorithm for the computation of MAP inferences in discrete graphical models of bounded treewidth (Section 2.2). The algorithm provides an assignment to the variables of interest and bounds on its posterior probability. In other words, the algorithm returns a solution and an estimate of its error relative to the optimal solution (i.e., the MAP assignment). The restriction to bounded treewidth models is necessary to enable efficient evaluation of solutions, that is, to enable polynomial-time computation of the probability of a given assignment to the variables of interest. The algorithm is then extended into an *anytime* procedure, that is, an algorithm that is capable of monotonically improving the quality of its output as more time is granted (Section 2.3). We show asymptotic convergence and theoretical error bounds for any fixed number of steps of the algorithm. In Section 2.4, the performance of these algorithms is evaluated in experiments with real and synthetic models, and compared against the state-of-the-art systematic search algorithm of Park and Darwiche [2003]. Concluding remarks appear in Section 2.5.

Most of the content of this chapter is based on the work published in Reference [Mauá and de Campos, 2012].

2.1 Graphical models and the MAP assignment problem

Consider a set $\mathbf{X} = \{X_1, \ldots, X_n\}$ of categorical variables, and an indexing set $S \subseteq [n] \stackrel{\text{def}}{=} \{1, \ldots, n\}$. We call ϕ a *factor* if it is a mapping of the assignments $\mathbf{x}_S \stackrel{\text{def}}{=} \{x_i : i \in S\}$ of a subset $\mathbf{X}_S = \{X_i : i \in S\}$ of the variables into non-negative real numbers, in which case we call the corresponding variable-indexing set S its *scope*. A *(probabilistic) graphical model* over $\mathbf{X}$ is a collection $\Phi = \{\phi_1, \ldots, \phi_m\}$ of factors whose scopes $S_1, \ldots, S_m$ satisfy $S_1 \cup \cdots \cup S_m = [n]$. The model concisely defines a probability measure on the sigma-field of subsets E of joint assignments $\mathbf{x} \stackrel{\text{def}}{=} \{x_1, \ldots, x_n\}$ to $\mathbf{X}$ by

$$\mathrm{P}(E) \stackrel{\text{def}}{=} \sum_{\mathbf{x} \in E} \frac{1}{Z} \prod_{i \in [m]} \phi_i(\mathbf{x}_{S_i}), \tag{2.1}$$

where $Z = \sum_{\mathbf{x}} \prod_{i \in [m]} \phi_i(\mathbf{x}_{S_i})$ is a normalizing constant known as the *partition function*. We often write $\phi_i(\mathbf{x})$ to denote $\phi_i(\mathbf{x}_{S_i})$, leaving the domain implicit, if no ambiguity arises. If $\mathbf{X}_S$ is a subset of variables, and $\mathbf{x}_S$ a possible assignment of values, the notations $\mathrm{P}(\mathbf{X}_S = \mathbf{x}_S)$ and $\mathrm{P}(\mathbf{x}_S)$ are used to denote the probability of the event $E = \{\mathbf{x}' : \mathbf{x}'_S = \mathbf{x}_S\}$ induced by the joint assignments whose projection on S is $\mathbf{x}_S$.

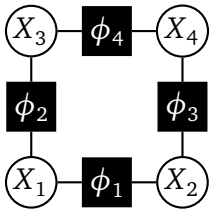


Figure 2.1. The factor graph of the graphical model in Example 2.1.

The *factor graph* of a graphical model Φ is a bipartite graph whose nodes are identified either with factors ϕ in Φ or with variables X_i in $\mathbf{X}$. The graph contains an edge $\{\phi, X_i\}$ if i is in the scope of ϕ. The following example helps in illustrating the concepts introduced thus far.

Example 2.1. *Consider a graphical model $\Phi = \{\phi_1, \phi_2, \phi_3, \phi_4\}$ over four binary variables X_1, X_2, X_3 and X_4, where the factors are specified by the tables below.*

X_1	X_2	$\phi_1(X_1, X_2)$
0	0	10
1	0	0.1
0	1	0.1
1	1	10

X_1	X_3	$\phi_2(X_1, X_3)$
0	0	5
1	0	0.2
0	1	0.2
1	1	5

X_2	X_4	$\phi_3(X_2, X_4)$
0	0	5
1	0	0.2
0	1	0.2
1	1	5

X_3	X_4	$\phi_4(X_3, X_4)$
0	0	0.5
1	0	20
0	1	1
1	1	2.5

The corresponding factor graph is shown in Figure 2.1. The partition function is

$$Z = \sum_{x_1, x_2, x_3, x_4} \phi_1(x_1, x_2)\phi_2(x_1, x_3)\phi_3(x_2, x_4)\phi_4(x_3, x_4) = 1224.384,$$

and $\mathrm{P}(X_1 = 1, X_2 = 1, X_3 = 1, X_4 = 1) = 10 \cdot 5 \cdot 5 \cdot 2.5 / Z = 0.5104607704.$ □

We say that a variable X_i in $\mathbf{X}$ is a *decision* variable if its value is to be determined.[2] The set D indexes the decision variables in $\mathbf{X}$. The variables not indexed

[2]The term "decision variable" can lead one to think that such a variable necessarily represents a quantity which can be controlled in the real world. This is however not the case, and such variables are often related to uncontrollable events.

by D are called *latent variables*, and are indexed by H (thus $D \cup H = [n]$ and $D \cap H = \emptyset$). The *MAP assignment problem* consists in finding[3]

$$\begin{aligned}\mathbf{d}^* &\in \operatorname*{argmax}_{\mathbf{d}} \mathrm{P}(\mathbf{X}_D = \mathbf{d}) \\ &= \operatorname*{argmax}_{\mathbf{d}} \sum_{\mathbf{x}} \prod_{i\in[m]} \phi_i(\mathbf{x}) \prod_{j\in D} \delta_{X_j=d_j}(\mathbf{x}), \end{aligned} \tag{2.2}$$

where $\delta_{X_j=\mathbf{d}_j} \stackrel{\text{def}}{=} \delta(\mathbf{x}_j - d_j)$ denotes the Kronecker delta function that is one if $\mathbf{x}_j = d_j$, and zero otherwise. Note that in our terminology, $\delta_{X_j=d_j}$ is a factor with scope $\{j\}$.

For any assignment $\mathbf{d} \stackrel{\text{def}}{=} \{d_j : j \in D\}$ to the decision variables, we can interpret the argument of the maximization in (2.2), that is, the expression

$$\sum_{\mathbf{x}} \prod_{i\in[m]} \phi_i(\mathbf{x}) \prod_{j\in D} \delta_{X_j=d_j}(\mathbf{x}),$$

as the probability measure defined by the augmented graphical model $\Phi_{\mathbf{d}} = \Phi \cup \bigcup_{j\in D}\{\delta_{X_j=\mathbf{d}_j}\}$. This probability measure assigns zero probability to joint assignments that are not consistent with $\mathbf{d}$, and thus the partition function of the augmented model is $Z_{\mathbf{d}} = \sum_{\mathbf{x}} \prod_{i\in[m]} \phi_i(\mathbf{x}) \prod_{j\in D} \delta_{X_j=\mathbf{d}_j}(\mathbf{x})$. In other words, there is a one-to-one correspondence between assignments $\mathbf{d}$ to the decision variables (which are the candidate solutions to the MAP assignment problem) and graphical models $\Phi_{\mathbf{d}} = \Phi \cup \bigcup_{j\in D}\{\delta_{X_j=\mathbf{d}_j}\}$, and the quality of each assignment $\mathbf{d}$ is given by the partition function $Z_{\mathbf{d}}$ of the corresponding graphical model $\Phi_{\mathbf{d}}$. This way, we can re-state the MAP assignment problem as a search over graphical models $\Phi_{\mathbf{d}}$ instead of a search over assignments. Assume without loss of generality that $D = \{1,\ldots,k\}$, for some integer $k < n$, and $H = \{k+1,\ldots,n\}$, and define $K_i = \{\phi_i\}$ for $i = 1,\ldots,m$, and $K_{j+m} = \{\delta_{X_j=x_j} : x_j \sim X_i\}$ for each decision $j \in D$.[4] Each combination of factors $\phi_1,\ldots,\phi_{m+k}$ from sets $K_1,\ldots,K_{m+k}$, respectively, specifies the graphical model $\Phi_{\mathbf{d}}$ corresponding to an assignment $\mathbf{d}$ to the decision variables. Let $\mathscr{M} = \{\{\phi_1,\ldots,\phi_{m+k}\} : \phi_i \in K_i\}$ denote all graphical models obtained in such a way. Finding a MAP assignment $\mathbf{d}^* \in \operatorname{argmax}_{\mathbf{d}} \mathrm{P}(\mathbf{X}_D = \mathbf{d})$ is equivalent to finding a graphical model $\Phi^* \in \operatorname{argmax}_{\Phi\in\mathscr{M}} \sum_{\mathbf{x}} \prod_{\phi\in\Phi} \phi(\mathbf{x})$. An assignment $\mathbf{d}^* = \{d_j^* : j \in D\}$ is a MAP assignment if and only if there is a model $\Phi^* = \{\phi_1^*,\ldots,\phi_{m+k}^*\}$ in $\mathscr{M}$ such that $\sum_{\mathbf{x}} \prod_{\phi\in\Phi^*} \phi(\mathbf{x}) = \max_{\Phi\in\mathscr{M}} \sum_{\mathbf{x}} \prod_{\phi\in\Phi} \phi(\mathbf{x})$

[3] The formulation of the MAP assignment problem with delta functions is unusual, but it is important for the results and algorithm we devise later on.

[4] The notation $x_i \sim X_i$ denotes that x_i is an element of the sample space of possible values of variable X_i.

and $\mathbf{d}^* = \text{argmax}_{\mathbf{d}} \prod_{j \in D} \phi^*_{j+k}(d_j)$ (or equivalently, if $d^*_j = \text{argmax}_{d_j} \phi^*_{j+k}(d_j)$ for every $j \in D$).

Example 2.2. *Consider the graphical model in Example 2.1, and suppose that* $D = \{2,3\}$ *and* $H = \{1,4\}$. *The probability of each assignment to the decision variables is shown in the table below. The rightmost column contains the unnormalized probabilities* $\text{P}'(x_2, x_3) = Z \cdot \text{P}(x_2, x_3)$.

X_2	X_3	$\text{P}(X_2, X_3)$	$\text{P}'(X_2, X_3)$
0	*0*	*0.11*	*135.054*
1	*0*	*0.2*	*251.25*
0	*1*	*0.01*	*12.75*
1	*1*	*0.67*	*825.33*

Thus, $\{X_2 = 1, X_3 = 1\}$ *is the single MAP assignment.*

To reformulate this MAP assignment problem as a search over graphical models, consider the factors $\delta_{X_2 = x_2}$ *and* $\delta_{X_3 = x_3}$ *for every MAP assignment* $\{x_2, x_3\}$. *Let* $K_1 = \{\phi_1\}$, $K_2 = \{\phi_2\}$, $K_3 = \{\phi_3\}$, $K_4 = \{\phi_4\}$, $K_5 = \{\delta_{X_2=1}, \delta_{X_2=0}\}$ *and* $K_6 = \{\delta_{X_3=1}, \delta_{X_3=0}\}$. *Each combination of factors* $\psi_1, \ldots, \psi_6 \in K_1, \ldots, K_6$ *corresponds to the graphical model* $\Phi_{\mathbf{d}}$ *induced by the assignment*

$$\mathbf{d} = \underset{x_1, x_2}{\text{argmax}} \, \psi_5(x_2) \psi_6(x_3) \, .$$

Let $\mathscr{M}$ *be the family of all such graphical models* $\Phi_{\mathbf{d}}$. *An assignment* $\{x^*_2, x^*_3\}$ *is a MAP assignment if and only if there is model* $\Phi^*_{\mathbf{d}} = \{\psi^*_1, \ldots, \psi^*_6\}$ *in the set*

$$\underset{\Phi \in \mathscr{M}}{\text{argmax}} \sum_{x_1, x_2, x_3, x_4} \prod_{i \in [6]} \psi_i(\mathbf{x})$$

such that $x^*_2 = \text{argmax}_{x_2} \psi^*_5(x_2)$ *and* $x^*_3 = \text{argmax}_{x_3} \psi^*_6(x_3)$. *Hence,*

$$\Phi^* = \{\phi_1, \ldots, \phi_4, \delta_{X_2=1}, \delta_{X_3=1}\}$$

is the solution equivalent to the MAP assignment $\{X_2 = 1, X_3 = 1\}$, *and* $Z_{\Phi^*} = 825.33$. □

2.2 Approximate algorithms for the MAP assignment problem

In this section, we review some of the approaches to evaluation and selection of approximate MAP assignments, and present a new algorithm that provides bounds on the quality of the solution.

2.2.1 Clique-tree computation

We start by reviewing clique-tree algorithms, which can be used among other things to compute (not necessarily efficiently) MAP assignments in graphical models or to evaluate the unnormalized probability of a given assignment. Many algorithms use clique trees to organize data and schedule computations, and this is also the case for the algorithm we develop later on.

Clique-tree algorithms first appeared as methods to compute marginal probabilities in Bayesian networks [Lauritzen and Spiegelhalter, 1988; Shenoy and Shafer, 1988]. Motivated by the fact that computations in tree-shaped graphical models are usually efficient and straightforward [Pearl, 1988], clique-tree algorithms re-cast any graphical model as a tree. The efficiency of the computations in a clique tree is a function of the resemblance of the original graph to a tree.

Formally, let T be a tree over $[m]$ such that each node i is associated to a set $C_i \subseteq [n]$ and $C_1 \cup \cdots \cup C_m = [n]$ for some positive integer n. We call T a *clique tree* if for $i = 1, \ldots, n$ the sub-graph obtained by removing from T all nodes j such that $i \notin C_j$ remains connected.[5] This condition is known as the *running intersection property*, and guarantees that for any path in the tree (i.e., a sequence of non-repeating adjacent nodes) either an integer associated to the current node (i.e, some $k \in C_i$) appears for the last time in the path or it also appears in the set associated to the adjacent node.

Let Φ be a graphical model whose factors $\phi_1, \ldots, \phi_k$ have scopes $S_1, \ldots, S_k$, respectively, and $S_1 \cup \cdots \cup S_k = [n]$. We say that $(T, \{C_1, \ldots, C_m\})$ is a clique tree for Φ if T is a clique tree over $[m]$, and for each $i = 1, \ldots, k$ there is $j \in [m]$ such that $S_i \subseteq C_j$. This last condition is called the *family preserving property*, and it guarantees that the scope of each factor is covered by some set associated to a node of the clique tree, allowing us to associate each factor in the model to exactly one node in the clique tree. In the following, we assume for ease of exposition and without loss of generality that if T is a clique tree for Φ then $m = k$ and $S_i \subseteq C_i$ for all i, which allows us to unambiguously associate each factor ϕ_i to the clique tree node i (for any C_i violating the assumption, we can include a new factor in Φ with domain $\mathbf{X}_{C_i}$ and image $\{1\}$; the running time of the computations on the clique trees for both models have the same asymptotic growth rate, and the new graphical model induces the same probability measure). This assumption is merely aesthetic, as it prevents us from dealing with

[5] In Chapter 3, we define the much similar concept of tree decomposition of a graph, which essentially refers to the same class of objects and could be used here instead. The distinction in terminology is that we see clique trees as objects not necessarily related to any graph, whereas the mention of a tree decomposition implies a reference graph.

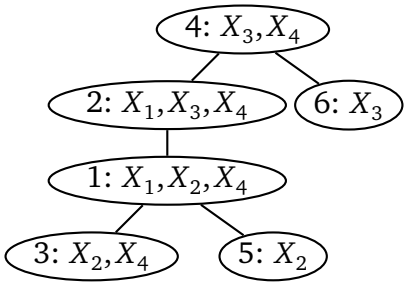


Figure 2.2. A clique tree for the family of graphical models $\mathscr{M}$ in Example 2.2.

two indexing schemes (one for the factors and other for the nodes of the clique tree) and avoids the subsequent overcomplicated notation. The results obtained here do not depend, by any means, on that assumption.

The *width* of a clique tree is the cardinality of the largest set C_i minus one. Since the complexity of algorithms that operate on clique trees is usually (at least) exponential in the tree width, one usually seeks to obtain a clique tree of low width. If it exists, a clique tree for a graphical model Φ of a given maximum width k (assumed constant) can be found in time linear in the scopes (but exponential in k) [Bodlaender, 1996]. In general, finding a minimum-width clique tree for a given graphical model is an NP-hard problem [Yannakakis, 1981], and one usually resorts to heuristics to obtain low-width trees, such as the minimum fill-in heuristic [Kjaerulff, 1990; van den Eijkhof et al., 2007].

Example 2.3. *The tree in Figure 2.2 is a clique tree for any graphical model $\Phi_{\mathbf{d}}$ in the family $\mathscr{M}$ in Example 2.2. The nodes $1,\ldots,4$ are associated, respectively, with the factors $\phi_1,\ldots,\phi_4$, whereas the nodes 5 and 6 are associated to factors $\delta_{X_2=x_2}$ and $\delta_{X_3=x_3}$, respectively, for some x_2, x_3. Note that $S_2 = \{1,3\} \subset C_2 = \{1,3,4\}$. The tree has width two, and one can show that there is no other suitable clique tree of smaller width.* □

The basic computation scheme with clique trees is the FACTOR-ELIMINATION procedure in Algorithm 1, which computes the partition function of a graphical model $\Phi = \{\phi_1,\ldots,\phi_m\}$ associated to a clique tree T over $[m]$.[6] Note that $C_{\mathrm{Pa}(r)} = \emptyset$ in line 5. In a nutshell, the algorithm roots the tree in an arbitrary node (line 1), and then propagates messages containing the factors μ_i from the leaves towards the root (lines 3–7). If r is the root node, we say that a node p of

[6]The algorithm is also known as the COLLECT algorithm [Shenoy and Shafer, 1988] and JUNCTION-TREE PROPAGATION algorithm [Jensen and Nielsen, 2007], and is closely related to variable elimination [Dechter, 1999], fusion in valuation algebras [Kohlas, 2003] and nonserial dynamic programming [Bertele and Brioschi, 1972].

Algorithm 1 FACTOR-ELIMINATION

Require: A clique tree T over a graphical model $\Phi = \{\phi_1, \ldots, \phi_m\}$
Ensure: $Z = \sum_{\mathbf{X}} \prod_{i \in [m]} \phi_i$
1: select a node r as root
2: label all nodes as inactive
3: **while** there is an inactive node i **do**
4: select an inactive node i with all children active
5: compute $\mu_i = \sum_{\mathbf{X}_{C_i \setminus C_{\mathrm{Pa}(i)}}} \phi_i \prod_{j \in \mathrm{Ch}(i)} \mu_j$
6: label i as active
7: **end while**
8: $Z = \mu_r$

the clique tree is the parent of a neighboring node i if p is closer to r. In this case, we also say that i is a child of p. The set of children and descendants of i are denoted respectively by $\mathrm{Ch}(i)$ and $\mathrm{De}(i)$. Note that every node but the root r has a single parent. The notation in line 5 representing the sum-marginal of a product of factors will be commonly used in the following discussion, and represents the factor μ_i over $\mathbf{X}_{C_i \cap C_{\mathrm{Pa}(i)}}$ such that

$$\mu_i(\mathbf{x}') = \sum_{\mathbf{x} \sim \mathbf{X}_{C_i} : \mathbf{x}_{C_i \cap C_{\mathrm{Pa}(i)}} = \mathbf{x}'} \phi_i(\mathbf{x}_{S_i}) \prod_{j \in \mathrm{Ch}(i)} \mu_j(\mathbf{x}_{C_i \cap C_j}), \tag{2.3}$$

for all $\mathbf{x}' \sim \mathbf{X}_{C_i \cap C_{\mathrm{Pa}(i)}}$. The propagation of messages halts when the root has received one message from each child, in which case the partition function is obtained by $Z = \mu_r$. The algorithm runs in $O(ms^{w+1})$ time, where $s = \max_i |\{x_i \sim X_i\}|$ is the maximum number of values a variable in the model can assume, and $w = \max_i |C_i| - 1$ is the width of the clique tree. Thus when the width w is bounded, the computations take polynomial time.

Let $h(i) \stackrel{\text{def}}{=} \bigcup_{j \in \mathrm{De}(i) \cup \{i\}} C_j \setminus C_{\mathrm{Pa}(i)}$ be the set of variable indexes that appear either in the set associated to node i or in any of its descendants. Due to the running intersection property, the factors containing only variables indexed by elements of $h(i)$ are not involved in any computations performed in the loop of FACTOR-ELIMINATION after node i has been processed (i.e., after it has been labeled as active). Consequently, it can be shown [Koller and Friedman, 2009] that for $i = 1, \ldots, m$ the factor μ_i satisfies

$$\mu_i = \sum_{\mathbf{X}_{h(i)}} \phi_i \prod_{j \in \mathrm{De}(i)} \phi_j . \tag{2.4}$$

Since $h(r)=[n]$ by definition of clique trees, the correctness of the computations follows from applying this result to the root: $Z=\mu_r=\sum_{\mathbf{X}}\prod_{i\in[m]}\phi_i$. Hence, we can evaluate the quality of a candidate solution $\mathbf{d}$ to the MAP assignment problem by building a clique tree T for the corresponding graphical model $\Phi_{\mathbf{d}}$ and then running FACTOR-ELIMINATION, which produces $Z_{\mathbf{d}}=\sum_{\mathbf{X}}\prod_{\phi\in\Phi_{\mathbf{d}}}\phi$. The same clique tree structure can be used to evaluate different candidates.

Example 2.4. *Consider the MAP problem in 2.2 and the clique tree T in Figure 2.2 as described in Example 2.3. We can evaluate the assignment $\mathbf{d}=\{X_2=1,X_3=0\}$ by running* FACTOR-ELIMINATION *with inputs T and $\Phi_{\mathbf{d}}=\{\phi_1,\ldots,\phi_4,\delta_{X_2=1},\delta_{X_3=0}\}$, which obtains $Z_{\mathbf{d}}=\sum_{X_1,X_2,X_3,X_4}\prod_{i=1}^{8}\phi_i\delta_{X_2=1}\delta_{X_3=0}\propto \mathrm{P}(X_2=1,X_2=0)$. The same clique tree could be used to evaluate e.g. the MAP assignment $\mathbf{d}=\{X_2=1,X_3=1\}$ by running* FACTOR-ELIMINATION *with factors $\Phi_{\mathbf{d}}=\{\phi_1,\ldots,\phi_4,\delta_{X_2=1},\delta_{X_3=1}\}$.* □

The algorithm can be straightforwardly modified to find a MAP assignment when there are no latent variables (i.e., when $H=\emptyset$) by substituting sums with maximizations in the computation of factors μ_i, obtaining a FACTOR-MAXIMIZATION version of the algorithm in which $\mu_i=\max_{\mathbf{X}_{C_i\setminus C_{\mathrm{Pa}(i)}}}\phi_i\prod_{j\in\mathrm{Ch}(i)}\mu_j$ (the correctness of the procedure follows from the commutativity of maximization and product on the real numbers, Koller and Friedman [2009]). Indeed, a common approximation to MAP inference is to augment the decision set to $D'=D\cup H$ and run FACTOR-MAXIMIZATION, which then computes a lower bound on the value of original MAP assignment problem. This idea naturally suggests a greedy approach to the computation of MAP assignments in the presence of latent variables (i.e., $H\neq\emptyset$), which consists in redefining the factors μ_i so that latent variables are summed out while decision variables are maximized. A FACTOR-MAX-ELIMINATION version of the algorithm computes messages

$$\mu_i=\max_{\mathbf{X}_{D_i}}\sum_{\mathbf{X}_{H_i}}\phi_i\prod_{j\in\mathrm{Ch}(i)}\mu_j,$$

where $D_i=(C_i\cap D)\setminus C_{\mathrm{Pa}(i)}$ and $H_i=(C_i\cap H)\setminus C_{\mathrm{Pa}(i)}$. This approach has been proposed by Park and Darwiche [2003] as a cheap way of obtaining upper bounds on subsets of MAP assignments, which allows for a branch-and-bound procedure. Recently, iterative variants of this procedure that propagate messages in all directions until some convergence criteria is met have been proposed and justified as approximations by variational inference [Liu and Ihler, 2011; Jiang et al., 2011]. All these approaches retain the efficiency of message-passing algorithms for inference in graphical models, but produce only rough estimates to the real value. An exception is the use of FACTOR-MAX-ELIMINATION in a tree whose

root node r contains all decision variables. In this case, the procedure performs maximizations only after all latent variables have been marginalized out, which guarantees the correctness of the procedure for the MAP problem. Enforcing the clique tree to contain a node over all decision variables results in an exponential complexity in the number of decision variables [Park and Darwiche, 2004], unless the factors in the root node are factorized [Meek and Wexler, 2011]. Thus, this approach is intractable in arbitrary large models. Yet, accuracy and runtime can be traded by "promoting" decision variables towards the root. Given any clique-tree rooted at r we promote a decision variable by including it in a node closer to the root (or in the root node itself), making sure that the running intersection property of tree decomposition is respected. Promoting variables can lead to tighter upper bounds, but also to a significant increase in running time [Park and Darwiche, 2004].

As mentioned in the previous paragraph, Park and Darwiche [2003] developed a branch-and-bound procedure to the MAP assignment problem that systematically searches over the space of assignments, running FACTOR-ELIMINATION to evaluate each candidate solution, FACTOR-MAX-ELIMINATION with decision variables promoted until a given threshold on the width of the clique tree is achieved to obtain upper bounds on partial assignments. This strategy greatly narrows the search space, and makes the approach very competitive in practice. Their algorithm also allows for anytime inference, as the search can be stopped at any time returning the best solution found so far, and a better solution can be found with more time. We compare the algorithm we devise later on against theirs. Their algorithm has exponential worst-case running time, which is not surprising as the problem is NP-hard and can be reduced from SAT [Darwiche, 2009].

2.2.2 Propagating sets

Recall from the previous section that we can compare the quality of different candidate solutions to the MAP assignment problem by running FACTOR-ELIMINATION with the same clique tree but different indicator factors. More generally, consider a collection of sets of factors $K_1, \ldots, K_m$ such that all factors in each set K_i have the same scope. Let $\Phi = \{\phi_1, \ldots, \phi_m\}$ be a graphical model obtained by selecting one factor ϕ_i from each set K_i, $i = 1, \ldots, m$, and let T be a clique tree for this model. Then T is also a clique tree for any other graphical model induced by $K_1, \ldots, K_m$. This is illustrated in the next example.

Example 2.5. *Consider the sets of factors $K_1, \ldots, K_6$ defined in Example 2.2, and the clique tree in Figure 2.2. For $i = 1, \ldots, 4$, we assign each singleton set K_i to*

the node i in the clique tree. We also assign sets $K_5 = \{\delta_{X_2=x_2} : x_2 \sim X_2\}$ and $K_6 = \{\delta_{X_3=x_3} : x_3 \sim X_3\}$ to nodes 5 and 6, respectively, which makes T a clique tree for all graphical models induced by these sets. Thus, the graphical models induced by the sets $K_1, \ldots, K_6$ are the graphical models $\Phi_{\mathbf{d}} = \{\phi_1, \ldots, \phi_4, \delta_{X_2=x_2}, \delta_{X_3=x_3}\}$ in Example 2.3 that are in one-to-one correspondence with (unnormalized) probabilities of MAP assignments. Moreover, T is a suitable clique tree for any such model. □

An arguably natural approach to approximately solve a MAP assignment problem, is to use some criteria to select a subset of candidate assignments, and evaluate the quality of each assignment in the subset running FACTOR-ELIMINATION on the same clique tree and the corresponding graphical model $\Phi_{\mathbf{d}}$. The FACTOR-SET-ELIMINATION procedure in Algorithm 2 implements such an approach, by selecting assignments while it propagates sets of factors over the clique tree. The algorithm takes a clique tree T over a collection of sets of factors $K_1, \ldots, K_m$, and a list of integers $k_1, \ldots, k_m$, and returns in time polynomial in the largest of those integers a set of numbers that correspond to the (unnormalized) probabilities of a subset of the assignments, thus performing search and evaluation concurrently.

The algorithm resembles FACTOR-ELIMINATION, but instead of propagating a single message factor μ_i per node, it propagates sets of factors

$$L_i \subseteq M_i \stackrel{\text{def}}{=} \left\{ \sum_{\mathbf{x}_{C_i \setminus C_{\text{Pa}(i)}}} \phi_i \prod_{j \in \text{Ch}(i)} \mu_j : \phi_i \in K_i, \mu_j \in L_j \right\} . \tag{2.5}$$

As we shall see later on, the integer k_i provided in the input determines the maximum cardinality of the set L_i, and thus the worst-case time complexity of the algorithm. A larger value of k_i allows more factors to be propagated at node i, which increases the computational burden and potentially the accuracy of the algorithm. The object σ first referred to in line 1 is a function from message factors μ_i into factors $\sigma(\mu_i)$, where the latter accounts for errors introduced when discarding elements from M_i so as to keep the cardinality of L_i within the given bound k_i. We shall discuss later on how the elements $\sigma(\mu_i)$ in lines 6 and 16 are obtained. The pruning operations in lines 8 and 18 return a subset $L_i \subseteq M_i$ of cardinality k_i and recompute the upper bounds $\sigma(\mu_i)$ to account for the discarded elements. So, if $k_i \geq |M_i|$, the pruning operation returns $L_i = M_i$. The algorithm outputs lower and upper bounds Z_l and Z_u, respectively, to the maximum partition function $Z^* = \max\{\sum_{\mathbf{x}} \prod_{i \in [m]} \phi_i : \phi_i \in K_i\}$ of a graphical model induced by the sets in the input. The following result shows the corre-

Algorithm 2 FACTOR-SET-ELIMINATION

Require: A clique tree T over the sets of factors $K_1, \ldots, K_m$ and positive integers $k_1, \ldots, k_m$
Ensure: $Z_l \leq Z^* \leq Z_u$

select a node r as root and let σ be an empty dictionary
for all leaf node i **do**
  let M_i be an empty set
  for all $\phi_i \in K_i$ **do**
    add $\mu_i = \sum_{\mathbf{X}_{C_i \setminus C_{\mathrm{Pa}(i)}}} \phi_i$ to M_i
    set $\sigma(\mu_i) \leftarrow \mu_i$
  end for
  $L_i = \mathrm{prune}(M_i, \sigma_i, k_i)$
end for
label leaves as active and internal nodes as inactive
while there is an inactive node **do**
  select an inactive node i whose children are all active
  let M_i be an empty set
  for all $\phi_i \in K_i$, $\mu_j \in L_j$, $j \in \mathrm{Ch}(i)$ **do**
    add $\mu_i = \sum_{\mathbf{X}_{C_i \setminus C_{\mathrm{Pa}(i)}}} \phi_i \prod_{j \in \mathrm{Ch}(i)} \mu_j$ to M_i
    set $\sigma(\mu_i) \leftarrow \sum_{\mathbf{X}_{C_i \setminus C_{\mathrm{Pa}(i)}}} \phi_i \prod_{j \in \mathrm{Ch}(i)} \sigma(\mu_j)$
  end for
  $L_i = \mathrm{prune}(M_i, \sigma, k_i)$
  label i as active
end while
$Z_l = \max\{\mu_r : \mu_r \in L_r\}$
$Z_u = \max\{\sigma(\mu_r) : \mu_r \in L_r\}$

spondence of factors $\mu_i \in L_i$ computed by this algorithm to those computed with FACTOR-ELIMINATION.

Theorem 2.1. *For $i = 1, \ldots, m$, any $\mu_i \in L_i$ satisfies $\mu_i = \sum_{\mathbf{X}_{h(i)}} \phi_i \prod_{j \in \mathrm{De}(i)} \phi_j$ for some combination of $\phi_i \in K_i$ and $\phi_j \in K_j$ for all $j \in \mathrm{De}(i)$.*

Proof. First, note that the definition of μ_i in FACTOR-SET-ELIMINATION is identical to the definition in FACTOR-ELIMINATION. Assume the pruning operations are not performed, that is, that $\mathrm{prune}(M_i, \sigma_i, k_i)$ returns M_i. Then it is not difficult to see that μ_i matches the computation in FACTOR-ELIMINATION for some graphical model induced by $K_1, \ldots, K_m$. But since the pruning operation returns a subset of M_i, this holds also for any $\mu_i \in L_i$. □

The following result follows immediately from the above theorem.

Corollary 2.1. *$Z_l = \sum_{\mathbf{X}} \prod_{i\in[m]} \phi_i$ for some combination of factors $(\phi_1, \ldots, \phi_m) \in K_1 \times \cdots \times K_m$.*

If the algorithm is run with factor sets $K_1, \ldots, K_m$ that induce graphical models corresponding to different assignments to decision variables as explained in Section 2.1, the numbers Z_l and Z_u returned are lower and upper bounds for the MAP assignment probability $Z^* = \max_{\mathbf{d}} \mathrm{P}(\mathbf{X}_D = \mathbf{d})$. In fact, if $k_i = |M_i|$ for all $i = 1, \ldots, m$, the algorithm is equivalent to an exhaustive search over the space of assignments, and thus returns $Z_l = Z^*$. Moreover, the value of Z_l is actually achieved by some assignment, and hence denotes the value of a feasible solution. The assignment corresponding to Z_l can be obtained by tracking back the indicator factors δ_i, $i \in D$ that were propagated to generate the number $\mu_r = Z_l$, as in the following example.

Example 2.6. *Consider the sets of factors $K_1, \ldots, K_6$ in Example 2.5, and the clique tree T in Figure 2.2. Let us simulate a run of* FACTOR-SET-ELIMINATION *on inputs $K_1, \ldots, K_6$, T and $k_1, \ldots, k_6$, with integers k_i set to some sufficiently high value so that no pruning takes effect (i.e., $L_i = M_i$ for all i). This also makes the upper bounds tight, that is, $\sigma(\mu_i) = \mu_i$ for any μ_i in M_i and L_i, $i = 1, \ldots, 6$.*

First, for $i = 1, \ldots, 6$, we associate to each node i the set K_i. Suppose node 4 is selected as root. Then, 3,5 and 6 are leaf nodes, while 1 and 2 are internal nodes. The first loop of the algorithm processes all the leaf nodes, obtaining the sets

$$M_3 = \{\phi_3\}, \quad M_5 = \{\phi_5 : \phi_5 \in K_5\} = \{\delta_{X_2=1}, \delta_{X_2=0}\},$$
$$M_6 = \{\phi_6 : \phi_6 \in K_3\} = \{\delta_{X_3=1}, \delta_{X_3=0}\}.$$

The second loop processes the remaining nodes. In the first iteration of the loop, the algorithm selects node 1 (as it is the only being inactive and having all children active at this stage), and computes

$$M_1 = \left\{ \sum_{X_2} \phi_1 \mu_2 \mu_3 : \mu_2 \in L_2, \mu_3 \in L_3 \right\} = \left\{ \sum_{X_2} \phi_1 \phi_3 \delta_{X_2=1}, \sum_{X_2} \phi_1 \phi_3 \delta_{X_2=0} \right\}.$$

Next, the algorithm selects node 2 and computes

$$M_2 = \left\{ \sum_{X_1} \phi_1 \mu_1 : \mu_1 \in L_1 \right\} = \left\{ \sum_{X_1} \phi_2 \sum_{X_2} \phi_1 \phi_3 \delta_{X_2=1}, \sum_{X_2} \phi_1 \phi_3 \delta_{X_2=0} \right\}.$$

Finally, node 4 is selected, which causes the computation of

$$
\begin{aligned}
M_4 &= \left\{ \sum_{X_3,X_4} \phi_4 \mu_2 \mu_3 : \mu_2 \in L_2, \mu_3 \in L_3 \right\} \\
&= \Bigg\{ \quad \sum_{X_3,X_4} \phi_4 \left(\sum_{X_1} \phi_2 \sum_{X_2} \phi_1 \phi_3 \delta_{X_2=1} \right) \delta_{X_3=1} , \\
&\qquad \sum_{X_3,X_4} \phi_4 \left(\sum_{X_1} \phi_2 \sum_{X_2} \phi_1 \phi_3 \delta_{X_2=0} \right) \delta_{X_3=1} , \\
&\qquad \sum_{X_3,X_4} \phi_4 \left(\sum_{X_1} \phi_2 \sum_{X_2} \phi_1 \phi_3 \delta_{X_2=1} \right) \delta_{X_3=0} , \\
&\qquad \sum_{X_3,X_4} \phi_4 \left(\sum_{X_1} \phi_2 \sum_{X_2} \phi_1 \phi_3 \delta_{X_2=0} \right) \delta_{X_3=0} \quad \Bigg\} .
\end{aligned}
$$

The algorithm returns the highest of the numbers $\{\mu_4 : \mu_4 \in L_4\}$*, which according to Corollary 2.1 and the computations in Example 2.2 is*

$$
Z_l = \sum_{X_1,X_2,X_3,X_4} \prod_{i \in [4]} \phi_1 \delta_{X_2=1} \delta_{X_3=1} = 825.33 .
$$

The MAP assignment can be obtained by labeling each factor with corresponding decision variable assignments (or an empty character if the factor does not correspond to a partial assignment of decision variables). To this end, label each factor ϕ_i *in* K_i*,* $i = 1, \ldots, 4$ *with an empty string, and label each factor* $\delta_{X_i=x_i}$ *in* K_i*,* $i = 5, 6$*, with the string "*$(X_i = x_i)$*". Then, label each* μ_i *in* M_i*,* $i = 1, \ldots, 6$ *with the concatenation of the labels of its constituting factors. An inductive argument suffices to show that e.g. the label of* $\mu_1^{(1)} = \sum_{X_2} \phi_1 \phi_2 \delta_{X_2=1}$ *is "*$(X_2 = 1)$*", the label of* $\mu_2^{(2)} = \sum_{X_1} \phi_2 \sum_{X_2} \phi_1 \phi_3 \delta_{X_2=0}$ *is "*$(X_2 = 0)$*", and the label of* $Z_l = \max\{\mu_4 : \mu_4 \in L_4\}$ *is "*$(X_2 = 1)(X_3 = 1)$*".*

Note that since no "pruning" was performed, the algorithm in the last step maximizes over as many numbers as assignments of decision variables, which is equivalent to an exhaustive search. □

The complexity of the algorithm is determined by the number of additions and multiplications needed to compute each factor μ_i in a set M_i plus the complexity of the pruning operation (which we describe in detail later on). Analogously to FACTOR-ELIMINATION, the complexity of computing each μ_i is $O(ms^{w+1})$. Let k be the maximum of $k_1, \ldots, k_m$ and $|K_1|, \ldots, |K_m|$. By design, each set M_i

contains $|K_i|\prod_{j\in \mathrm{Ch}(i)}|L_j| = |K_i|\prod_{j\in \mathrm{Ch}(i)} k_j \leq k^c$ elements, where c is the maximum number of neighbors of a node. Hence, the algorithm runs in $O(k^c ms^w)$. If the clique tree given as input contains a bounded number of neighbors for each node and bounded width, the algorithm runs in time polynomial in the inputs $k_1,\ldots,k_m$ and $K_1,\ldots,K_m$. Note that for any given graphical model of bounded treewidth we can obtain a suitable clique tree of bounded width and bounded number of neighbors per node (e.g., a binary clique tree, Shenoy [1997]).

2.2.3 Pruning messages

As in Example 2.6, without any pruning operation, the FACTOR-SET-ELIMINATION algorithm amounts to an efficient enumerative scheme, that evaluates all possible solutions while avoiding many redundant computations that would be performed by running FACTOR-ELIMINATION on each solution. As such, the algorithm is not applicable to any reasonably large problem. Thus, the algorithm's efficiency heavily depends on the pruning operations, which are responsible for limiting the cardinality of the propagated sets according to the input parameters $k_1,\ldots,k_m$. In the following, we discuss how to derive pruning operations that allow for a trade-off between the quality of the solutions obtained and the computation time as determined by those input parameters.

Pruning by convexification

Consider a set of factors M_i produced during FACTOR-SET-ELIMINATION which we wish to prune to produce a smaller set L_i whose cardinality is not greater than the allowed cardinality k_i. A factor $\mu_i^{(1)}$ in M_i is a *convex combination* of factors $\mu_i^{(2)}$ and $\mu_i^{(3)}$ if there is a real $0 \leq \lambda \leq 1$ such that $\mu_i^{(1)} = \lambda\mu_i^{(2)} + (1-\lambda)\mu_i^{(3)}$. A factor $\mu_i \in M_i$ is an *extreme* if it is not a convex combination of any two other elements in the set (extremes or not). As the following result shows, convex combinations can be safely discarded from the propagation, as they certainly are outperformed by some extrema.

Proposition 2.1. *Let $\mu_i^{(1)}$, $\mu_i^{(2)}$ and $\mu_i^{(3)}$ be three different factors in a set M_i such that $\mu_i^{(1)}$ is a convex combination of $\mu_i^{(2)}$ and $\mu_i^{(3)}$. Let also $\mu_r^{(1)}$ be a solution obtained by propagating $\mu_i^{(1)}$ up to the root. Then, there is a solution μ_r obtained by propagating either $\mu_i^{(2)}$ or $\mu_i^{(3)}$ up to the root that satisfies $\mu_r^{(1)} < \mu_r$.*

Proof. Let $\mu_j^{(1)} = \sum_{X_{C_j\setminus C_p}} \phi_j \mu_i^{(1)} \prod_{k\in \mathrm{Ch}(j)\setminus\{i\}} \mu_k$, $\mu_j^{(2)} = \sum_{X_{C_j\setminus C_p}} \phi_j \mu_i^{(2)} \prod_{k\in \mathrm{Ch}(j)\setminus\{i\}} \mu_k$ and $\mu_j^{(3)} = \sum_{X_{C_j\setminus C_p}} \phi_j \mu_i^{(3)} \prod_{k\in \mathrm{Ch}(j)\setminus\{i\}} \mu_k$ be factors in M_j, where $j = \mathrm{Pa}(i)$ and

$p = \text{Pa}(j)$. Then $\mu_j^{(1)}$ is a convex combination of $\mu_j^{(2)}$ and $\mu_j^{(3)}$. By induction in the nodes of the clique tree in the order they are processed, we find that any number $\mu_r^{(1)} \in M_r$ obtained by propagating $\mu_i^{(1)}$ up to the root is a convex combination of numbers $\mu_r^{(2)}$ and $\mu_r^{(3)}$ obtained by propagating $\mu_i^{(2)}$ and $\mu_i^{(3)}$, respectively, up to the root. Hence, $\mu_r^{(1)}$ is necessarily (strictly) less than $\max\{\mu_r^{(2)}, \mu_r^{(3)}\}$. □

As a consequence of the above result, the solutions induced by a convex combination $\mu_i^{(1)}$ are suboptimal, and can be discarded from M_i without compromising the accuracy of the algorithm.

Corollary 2.2. *Let $\mu_i^{(1)}$, $\mu_i^{(2)}$ and $\mu_i^{(3)}$ be three different factors in a set M_i such that $\mu_i^{(1)}$ is a convex combination of $\mu_i^{(2)}$ and $\mu_i^{(3)}$. Then any solution $\mu_r^{(1)}$ different from $\mu_r^{(2)}$ and $\mu_r^{(3)}$, where $\mu_r^{(\ell)}$ is obtained by propagating $\mu_i^{(\ell)}$ up to the root, $\ell = 1, 2, 3$, is not an optimal solution.*

Pruning by convex combination is analogous to the problem of obtaining the convex hull of a finite set of multidimensional points, a problem that has been largely studied in computational geometry. There are many algorithms that obtain the convex hull in time polynomial in the cardinality of the unprunned set and the dimensionality [Avis, 2000], though in practice these algorithms often suffer from numerical problems due to the use of arbitrary precision.

Pruning by dominance

Another condition that can be verified to prune factors without compromising accuracy is dominance. Let $\mu_i^{(1)}$ and $\mu_i^{(2)}$ be two factors in a set M_i. We say that $\mu_i^{(2)}$ dominates $\mu_i^{(1)}$, and write $\mu_i^{(2)} \geq \mu_i^{(1)}$, if $\mu_i^{(2)}(\mathbf{x}) \geq \mu_i^{(1)}(\mathbf{x})$ for all $\mathbf{x} \sim \mathbf{X}_{C_i \cap C_{\text{Pa}(i)}}$.The following result shows that dominated messages can be safely removed from M_i.

Proposition 2.2. *Let $\mu_i^{(1)}$ and $\mu_i^{(2)}$ be two different factors in a set M_i such that $\mu_i^{(2)} \geq \mu_i^{(1)}$. Let also $\mu_r^{(1)}$ be a solution obtained by propagating $\mu_i^{(1)}$ up to the root. Then there is a solution $\mu_r^{(2)}$ obtained by propagating $\mu_i^{(2)}$ up to the root that satisfies $\mu_r^{(2)} \geq \mu_r^{(1)}$.*

Proof. Let

$$\mu_j^{(1)} = \sum_{X_{C_j \setminus C_p}} \phi_j \mu_i^{(1)} \prod_{k \in \text{Ch}(j) \setminus \{i\}} \mu_k, \qquad \mu_j^{(2)} = \sum_{X_{C_j \setminus C_p}} \phi_j \mu_i^{(2)} \prod_{k \in \text{Ch}(j) \setminus \{i\}} \mu_k,$$

be factors in M_j, where $j = \text{Pa}(i)$ and $p = \text{Pa}(j)$. Since the factors contain only nonnegative values, it follows that $\mu_j^{(2)} \geq \mu_j^{(1)}$. By induction in the nodes of the

clique tree in the order they are processed, we find that any number $\mu_r^{(1)} \in M_r$ generated by propagating $\mu_i^{(1)}$ up to the root is dominated by at least one number $\mu_r^{(2)}$ obtained by propagating $\mu_i^{(2)}$. □

As with convex combination, dominated solutions are also suboptimal.

Corollary 2.3. *Let $\mu_i^{(1)}$ and $\mu_i^{(2)}$ be two different factors in a set M_i such that $\mu_i^{(2)} \geq \mu_i^{(1)}$. Then any solution $\mu_r^{(1)}$ is either not an optimal solution or it equals a solution $\mu_r^{(2)}$ obtained by propagating $\mu_i^{(2)}$ up to the root.*

Note that convexity and dominance are complementary in that one does not imply the other.

Pruning dominance can be easily implemented in time polynomial in the cardinality of the unprunned set and in the dimensionality of factors.

Pruning by clustering

The pruning operation prune(M_i, σ, k_i) in the FACTOR-SET-ELIMINATION first discards non-extreme and dominated factors from M_i. Albeit accurate, these operations are seldom enough to produce a set L_i of cardinality at most k_i. To be able to meet the cardinality constraint, we partition the remaining factors in M_i (after non-extreme and dominated elements have been removed) in k_i clusters $\Gamma_i^{(1)} \ldots, \Gamma_i^{(k_i)}$, and obtain L_i by selecting one *representative* factor $\underline{\mu}_i^{(\ell)}$ in each cluster $\Gamma_i^{(\ell)}$. These representatives are valid solutions in that they can be produced from combination of factors from the input sets. Hence, they provide attainable lower bounds for the optimal solution. To account for the (worst-case) errors introduced by the pruning operations we introduce upper-bound factors $\sigma(\mu_i)$ for each discarded factor $\mu_i \in \Gamma_i^{(\ell)} \setminus \{\underline{\mu}_i^{(\ell)}\}$. We discuss first how to obtain upper bounds for discarded factors.

Consider a set of factors $\mu_i^{(1)}, \ldots, \mu_i^{(k)}$ which we intend to discard, and let $\overline{\mu}_i$ be a factor such that $\overline{\mu}_i(\mathbf{x}) = \max\{\mu_i^{(1)}(\mathbf{x}), \ldots, \mu_i^{(k)}(\mathbf{x})\}$ for all $\mathbf{x} \in \mathbf{X}_{C_i \cap C_{\mathrm{Pa}(i)}}$. Then $\overline{\mu}_i \geq \mu_i^{(\ell)}$ for $\ell = 1, \ldots, k$, and it follows from Proposition 2.2 that any value $\overline{\mu}_r$ obtained by propagating $\overline{\mu}_i$ up to the root is greater than or equal to a solution $\mu_r^{(\ell)}$ obtained by propagating $\mu_i^{(\ell)}$ up to the root, for $\ell = 1, \ldots, k$. Thus, we can use the factor $\overline{\mu}_i$ as an *upper bound* for the factors we wish to discard, and propagate it to obtain upper bounds on the solutions that we did not compute (due to discarding factors).

Since the upper bounds need to be propagated, they participate in the overall running time just as much as the propagated solutions. Thus, we want to

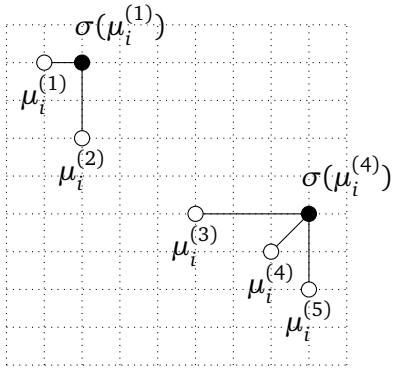


Figure 2.3. A clustering of factors $\Gamma_i^{(1)} = \{\mu_i^{(1)}, \mu_i^{(2)}\}$ and $\Gamma_i^{(2)} = \{\mu_i^{(3)}, \mu_i^{(4)}, \mu_i^{(5)}\}$ with representatives $\mu_i^{(1)}$ and $\mu_i^{(4)}$, respectively, and induced upper bounds $\sigma(\mu_i^{(1)})$ and $\sigma(\mu_i^{(4)})$.

propagate as few upper bounds as possible. We could for instance generate a single upper bound $\overline{\mu}$ that would dominate all factors discarded from M_i, but this would create too loose a bound. On the other extreme, we could use one upper bound for each discarded factor, but this would cause the propagation of an exponential number of upper bounds (therefore more than the limit k_i). Instead, we generate and propagate one upper bound for each cluster, which makes the overall complexity be still polynomial in k_i. Let $\underline{\mu}_i^{(\ell)}$ be the representative of a cluster $\Gamma_i^{(\ell)}$. To account for the removal of elements in the previous steps, we set the upper bound $\sigma(\underline{\mu}_i^{(\ell)})$ as $\max\{\sigma(\mu_i) : \mu_i \in \Gamma_i^{(\ell)}\}$. Figure 2.3 depicts one possible pruning by clustering of a set $M_i = \{\mu_i^{(1)}, \mu_i^{(2)}, \mu_i^{(3)}, \mu_i^{(4)}, \mu_i^{(5)}\}$ and $k_i = 2$.

Let $\mu_r^{(\ell)}$ be a solution obtained by propagating the representative $\underline{\mu}_i^{(\ell)}$ of cluster $\Gamma_i^{(\ell)}$, and let $\sigma(\mu_r^{(\ell)})$ be the corresponding propagated upper bound. Since the upper bound $\sigma(\underline{\mu}_i^{(\ell)})$ dominates all elements in the corresponding cluster (including the representative $\underline{\mu}_i^{(\ell)}$), it follows from Proposition 2.2 that $\sigma(\mu_r^{(\ell)}) \geq \mu_r^{(\ell)}$. And since every $\mu_r^{(\ell)}$ is a lower bound on the optimal solution it follows that

$$Z_l = \max\{\mu_r : \mu_r \in L_r\} \leq Z^* \leq \max\{\sigma(\mu_r) : \mu_r \in L_r\} = Z_u\,,$$

where Z^* is the optimal solution of the problem. The above argument guarantees the correctness of the algorithm.

There still remains to decide how to select good representatives. To this end, we define the following *error function* $\eta(\mu_i, \mu_i')$ that returns the error introduced

by "representing" a factor μ_i with another factor μ'_i as

$$\eta(\mu_i, \mu'_i) = \max_{\mathbf{x} \sim \mathbf{X}} \frac{\mu_i(\mathbf{x})}{\mu'_i(\mathbf{x})}, \tag{2.6}$$

where we assume that $0/0 \stackrel{\text{def}}{=} 1$ and $\epsilon/0 = \infty$, for any $\epsilon > 0$. The number $\eta(\mu_i, \mu'_i)$ matches the worst-case (multiplicative) error in the final solution if we locally discard a factor μ_i while selecting μ'_i as its representative (but never discard any other factor), that is $\mu_i(\mathbf{x}) \leq \mu'_i(\mathbf{x})\eta(\mu_i, \mu'_i)$, for any $\mathbf{x}$ such that $\mu'_i(\mathbf{x}) > 0$. The error function is asymmetric, and that it is greater than one if and only if μ_i is not dominated by μ'_i. When μ_i cannot be represented by a representative μ'_i with $\eta(\mu_i, \mu'_i) < \infty$, the representatives are effectively chosen arbitrarily, and in such cases the algorithm might be unable to provide theoretical guarantees on the error introduced. Nevertheless, we verified experimentally that even in this pathological case the actual error Z_l/Z_u produced by the algorithm is often bounded. For convenience, we assume in the rest of this discussion that $\eta(\mu_i, \mu'_i) < \infty$, reminding the reader that when this is not the case then the theoretical guarantee stated by Theorem 2.2 later on is not warranted.

We can extend the error function to measure the quality of representing a set M_i by a set L_i of representatives as follows:

$$\eta(M_i, L_i) = \max_{\mu_i \in M_i} \min_{\mu \in L_i} \eta(\mu_i, \mu). \tag{2.7}$$

The function above denotes the largest error within any two factors in a cluster, and thus provides an upper bound on the error introduced by the halting of the propagation of the elements in $M_i \setminus L_i$. The function bears some interesting properties with respect to dominance. Say that a factor in M_i is *maximal* if it is not dominated by any other element of the set. If M_i contains more than k_i maximal elements, then any subset $L_i \subset M_i$ of cardinality k_i satisfies $\eta(M_i, L_i) > 1$. On the other hand, let L_i be the set of maximal elements of M_i. Then $\eta(M_i, L_i) \leq 1$. Thus, we have that $\min\{\eta(M_i, V_i) : V_i \subseteq M_i, |V_i| \leq k_i\} \leq 1$ if and only if M_i has at most k_i maximal elements.

Ideally, we would like to find a set $L_i \subseteq M_i$ of at most k_i representatives that minimizes $\eta(M_i, L_i)$. However, this would add an extra complexity to the computations. Indeed, this problem has been shown to be NP-hard to solve exactly or to approximate up to a factor of $\log^* |M_i|$ [Chuzhoy et al., 2005].[7] Relaxing

[7] The function $\log^* n$ returns the least integer i such that applying $\log_2$ operator i times on n produces a number smaller than one. For instance, $\log^* 8 = 3$. Notice that $\log^* n > 2$ for any $n > 8$.

the problem to a (proper) distance (e.g., by defining $\eta(\phi,\psi) = \max_{\mathbf{x}} |\ln \phi(\mathbf{x}) - \ln \psi(\mathbf{x})|$) does not make the problem more tractable, as it has been shown by Feder and Greene [1988] that the corresponding problem is NP-hard to solve or approximate by any factor smaller than two (i.e., to find a set L_i whose error is provably $\eta(M_i, L_i) \leq 2 \min_{L'} \eta(M_i, L_i')$). Instead, we use a greedy approach that iteratively attempts to replace a factor in $M_i \setminus L_i$ with a factor in L_i such that $\eta(M_i, L_i)$ is decreased, until either a local optimum is reached or the number of iterations exceeds a pre-specified limit. The quality of the clusterings thus produced depends strongly on the initial candidate solution. A good heuristic is to pick points that are well spread in the space defined by set M_i. Algorithm 3 describes the K-MEDOIDS heuristic that implements this idea. For instance, this procedure guarantees a 2-approximation if the error function is a distance [Feder and Greene, 1988].[8]

Algorithm 3 K-MEDOIDS

Require: A positive integer k and a set K with at least k maximal factors
Ensure: A subset $L \subseteq K$ containing k elements
1: initialize $L \leftarrow \emptyset$
2: remove an arbitrary element ψ from K and add it to L
3: set $d(\phi) = \eta(\phi,\psi)$ for all $\phi \in K$
4: **while** $|L| \leq k$ **do**
5: let $\psi = \operatorname{argmax}_{\phi \in K} d(\phi)$
6: remove ψ from K, add it to L and set $d(\phi) = \min\{\eta(\phi,\psi), d(\phi)\}$
7: **end while**
8: return L

We have defined the pruning operation prune(M_i, σ, k_i) as first discarding non-extrema, then discarding dominated factors, and only lastly performing clustering. One might wonder whether performing clustering directly on the set M_i (without or before any other pruning) can potentially lead to better clustering as measured by $\eta(M_i, L_i)$. The following result shows that as far as dominance is concerned this is never the case.

Proposition 2.3. *Let V_i be a subset of M_i that contains at least one factor $\underline{\mu}_i$ dominated by some element μ_i^* in M_i. If $V_i^* = V_i \cup \{\mu_i^*\} \setminus \{\underline{\mu}_i\}$ then $\eta(M_i, V_i) \geq \eta(M_i, V_i^*)$.*

[8] Turning the error function into a distance makes the upper bounds obtained more conservative (since the symmetry property of distance functions lead one to considering also the error introduced on the "dominated" coordinates of discarded factors), but does not affect the correctness of the algorithm.

Proof. For any $\mu_i \in M_i$, we have that

$$\eta(\mu_i, \underline{\mu}_i) = \max_{\mathbf{x}:\underline{\mu}_i(\mathbf{x})>0} \frac{\mu_i(\mathbf{x})}{\underline{\mu}_i(\mathbf{x})} > \max_{\mathbf{x}:\mu_i^*(\mathbf{x})>0} \frac{\mu_i(\mathbf{x})}{\mu_i^*(\mathbf{x})} = \eta(\mu_i, \mu_i^*)\,.$$

For any $\mu_i \in M_i$, let $V_i(\mu_i) = \{\mu_i' \in M_i : \mu_i = \operatorname{argmin}_{\mu_i'' \in V_i} \eta(\mu_i', \mu_i'')\}$. Since $M_i = V_i(\mu_i) \cup (M_i \setminus V_i(\mu_i))$, it follows that

$$\begin{aligned}
\eta(M_i, V_i) &= \max\left\{ \max_{\mu_i \in V_i(\underline{\mu}_i)} \eta(\mu_i, \underline{\mu}_i), \max_{\mu_i \in M_i \setminus V_i(\underline{\mu}_i)} \min_{\mu_i' \in V_i \setminus \{\underline{\mu}_i\}} \eta(\mu_i, \mu_i') \right\} \\
&\geq \max\left\{ \max_{\mu_i \in V_i(\underline{\mu}_i)} \eta(\mu_i, \mu_i^*), \max_{\mu_i \in M_i \setminus V_i(\underline{\mu}_i)} \min_{\mu_i' \in V_i \setminus \{\underline{\mu}_i\}} \eta(\mu_i, \mu_i') \right\} \\
&\geq \max\left\{ \max_{\mu_i \in V_i(\underline{\mu}_i)} \min_{\mu_i' \in V_i^*} \eta(\mu_i, \mu_i'), \max_{\mu_i \in M_i \setminus V_i(\underline{\mu}_i)} \min_{\mu_i' \in V_i^*} \eta(\mu_i, \mu_i') \right\} = \eta(M_i, V_i^*)\,.
\end{aligned}$$

Note that if $\mu_i^* \in V_i$, the inequalities above are trivially satisfied with equality, since in this case $V_i(\underline{\mu}_i) = \emptyset$. □

According to the result above, if L_i contains a non-maximal element, we can replace that element by some maximal element not in L_i (if it exists) and obtain another set L_i' whose error with respect to M_i is not worse than the error of the initial set. Thus, removing dominated factors never degrades the quality of clustering (but speeds up computations, as we perform clustering on a smaller set). A similar result can be obtained for convex combinations.

Proposition 2.4. *Let V_i be a subset of M_i that contains at least one factor $\underline{\mu}_i$ which is a convex combination of factors $\mu_i^{(1)}$ and $\mu_i^{(2)}$ in M_i. If $V_i^* = V_i \cup \{\mu_i^{(1)}, \mu_i^{(2)}\} \setminus \{\underline{\mu}_i\}$ then $\eta(M_i, V_i) \geq \eta(M_i, V_i^*)$.*

Proof. Since $\underline{\mu}_i$ is a convex combination of $\mu_i^{(1)}$ and $\mu_i^{(2)}$, we have that $\underline{\mu}_i(\mathbf{x}) \leq \max\{\mu_i^{(1)}(\mathbf{x}), \mu_i^{(2)}(\mathbf{x})\}$ for all $\mathbf{x} \sim \mathbf{X}$. Thus, it follows for each $\mu_i \in M_i$ that

$$\eta(\mu_i, \underline{\mu}_i) \geq \max_{\mathbf{x}:\underline{\mu}_i(\mathbf{x})>0} \frac{\mu_i(\mathbf{x})}{\max\{\mu_i^{(1)}(\mathbf{x}), \mu_i^{(2)}(\mathbf{x})\}} = \min\{\eta(\mu_i, \mu_i^{(1)}), \eta(\mu_i, \mu_i^{(2)})\}\,.$$

Hence,

$$\begin{aligned}
\eta(M_i, V_i) &= \max\left\{ \max_{\mu_i \in V_i(\underline{\mu}_i)} \eta(\mu_i, \underline{\mu}_i), \max_{\mu_i \in M_i \setminus V_i(\underline{\mu}_i)} \min_{\mu_i' \in V_i \setminus \{\underline{\mu}_i\}} \eta(\mu_i, \mu_i') \right\} \\
&\geq \max\left\{ \max_{\mu_i \in V_i(\underline{\mu}_i)} \min\{\eta(\mu_i, \mu_i^{(1)}), \eta(\mu_i, \mu_i^{(2)})\}, \max_{\mu_i \in M_i \setminus V_i(\underline{\mu}_i)} \min_{\mu_i' \in V_i \setminus \{\underline{\mu}_i\}} \eta(\mu_i, \mu_i') \right\} \\
&\geq \max\left\{ \max_{\mu_i \in V_i(\underline{\mu}_i)} \min_{\mu_i' \in V_i^*} \eta(\mu_i, \mu_i'), \max_{\mu_i \in M_i \setminus V_i(\underline{\mu}_i)} \min_{\mu_i' \in V_i^*} \eta(\mu_i, \mu_i') \right\} = \eta(M_i, V_i^*)\,,
\end{aligned}$$

where $V_i(\mu_i)$ is defined as in the proof of Proposition 2.3. Note that if both $\mu_i^{(1)}$ and $\mu_i^{(2)}$ are in V_i, the inequalities above are trivially satisfied with equality, since in this case $V_i(\underline{\mu}_i) = \emptyset$. □

The result above shows that replacing a representative with the two extrema of a convex combination of it does not degrade the quality of the clustering. However, in doing so we increase the cardinality of the set of representatives by one. Thus, in principle, one might obtain a better clustering by including a convex combination representative (but not a convex combination and both of its extrema at the same time).

The most important characteristic of the error function and the clustering accuracy measure induced by set error function is that they guarantee that the overall accuracy of the algorithm improves monotonically if we improve the clusterings at any node of the clique tree:

Theorem 2.2. *The outputs Z_l and Z_u satisfy $Z_u \leq Z_l \prod_{i\in[m]} \eta(M_i, L_i)$.*

Proof. Consider some inactive node i whose children j are all active at some step of the algorithm, and assume by inductive hypothesis that for any $\mu_j \in L_j$ it holds that $\sigma(\mu_j) \leq \mu_j e_j$, where e_j is defined as $\eta(M_j, L_j)\prod_{k\in\mathrm{De}(j)} \eta(M_k, L_k)$. Then any $\mu_i \in M_i$ satisfies

$$\begin{aligned}\sigma(\mu_i) &= \sum_{\mathbf{x}_{C_i\setminus C_{\mathrm{Pa}(i)}}} \phi_i \prod_{j\in\mathrm{Ch}(i)} \sigma(\mu_j) \\ &\leq \prod_{k\in\mathrm{De}(i)} \eta(M_k, L_k) \left(\sum_{\mathbf{x}_{C_i\setminus C_{\mathrm{Pa}(i)}}} \phi_i \prod_{j\in\mathrm{Ch}(i)} \mu_j \right) = \mu_i \frac{e_i}{\eta(M_i, L_i)},\end{aligned}$$

where $\mu_j \in L_j$, $j \in \mathrm{Ch}(i)$. Let $\underline{\mu}_i$ be the representative of a cluster $\Gamma_i \subseteq M_i$, with $\sigma(\underline{\mu}_i) = \max\{\max_{\mathbf{x}} \mu_i(\mathbf{x}) : \mu_i \in \Gamma_i\}$. It follows from (2.7) that $\sigma(\underline{\mu}_i) \leq \eta(M_i, L_i)\underline{\mu}_i$. After the clustering, the new upper bound assigned to $\underline{\mu}_i$ is (by design) given by $\overline{\mu}_i = \max\{\sigma(\mu_i) : \mu_i \in \Gamma_i\}$, which satisfies $\overline{\mu}_i \leq \sigma(\underline{\mu}_i)e_i/\eta(M_i, L_i) \leq e_i\underline{\mu}_i$. □

The above result guarantees that the algorithm finds lower and upper bounds whose ratio is not greater than $\prod_{i\in[m]} \eta(M_i, L_i)$, which is an upper bound on the quality of the solution found. The quality of each cluster $\eta(M_i, L_i)$ can be improved by either increasing the maximum allowed number of elements k_i in the set, or improving the local clustering scheme. Furthermore, each set cannot have more than $|K_1|\cdots|K_m|$ elements. Thus, for a sufficiently high (but finite) value of k_i, $i = 1, \ldots, m$, the algorithm finishes with the correct answer (i.e., with

Algorithm 4 ANYTIME-INFERENCE

Require: A clique tree T over sets $K_1, \ldots, K_m$ and integer c

let $k_1^{(0)} = 1, \ldots, k_m^{(0)} = 1$, $Z_l^{(0)} = 0$ and $Z_u^{(0)} = 1$
set $t \leftarrow 0$
while $Z_l^{(t)} < Z_u^{(t)}$ and not interrupted **do**
  run FACTOR-SET-ELIMINATION with $k_1^{(t)}, \ldots, k_m^{(t)}$ and let (Z_l, Z_u) be its output
  set $Z_l^{(t+1)} = \max\{Z_l, Z_l^{(t)}\}$ and $Z_u^{(t+1)} = \min\{Z_u, Z_u^{(t)}\}$
  find the node i with highest $\eta(M_i, L_i)$
  set $k_i^{(t+1)} = k_i^{(t)} + c$, and $k_j^{(t+1)} = k_j^{(t)}$ for all $j \neq i$
  set $t \leftarrow t + 1$
end while
return $Z_l^{(t)}$ and $Z_u^{(t)}$

an optimal solution and $Z_l = Z_u$). These remarks lead naturally to the anytime inference algorithm we describe next.

2.3 Anytime inference

An anytime algorithm is a procedure that can be interrupted at any time with a meaningful solution whose quality is a monotonic function of runtime. Hence, anytime algorithms allow a trade-off between computation time and quality of solutions.

We can easily transform FACTOR-SET-ELIMINATION into an anytime algorithm that continuously improve the lower and upper bounds by increasing the maximum set cardinalities $k_1, \ldots, k_m$. The procedure is described in Algorithm 4. The anytime algorithm starts by running FACTOR-SET-ELIMINATION with all maximum set cardinalities $k_1, \ldots, k_m$ set to one. This produces an arbitrary (but feasible) lower bound $Z_l^{(0)}$, and an upper bound $Z_u^{(0)}$ that matches the value returned by FACTOR-MAX-ELIMINATION. Then, for each time step, the algorithm increases the maximum set cardinality k_i of the node i with lowest clustering accuracy $\eta(M_i, L_i)$ by a given constant c. In principle, even if we improve the clustering accuracy we might obtain a worse solution, as the function that evaluates clustering accuracy optimizes worst case. This can be circumvented by enlarging each set L_i incrementally.

2.4 Experiments

We performed experiments with three groups of graphical models, which range from simple to very challenging problems. The first group, which appears in the top five lines of Table 2.2, consists of benchmark Bayesian networks used in real applications.[9] In these networks, the MAP inference asks for optimum assignments of the root nodes given some evidence on every leaf. This creates hard MAP problems, as every variable in the network is relevant to the solution, and obtaining an exact solution by FACTOR-MAX-ELIMINATION would take time (at least) exponential in the number of decision variables (as the root node of the clique tree would contain all decision variables). The second group (lines 6–8 of the table) contains graphical models designed to solve multiprocessor scheduling (MS) problems with three processors and varying number of jobs (20, 50 and 100).[10] The underlying graph of these models consists of a chain of latent variables, each with a single root decision node as parent. These graphical models can be seen as inverted hidden Markov models, where the arcs point from observed to state variables, and the MAP assignment task is equivalent to finding the most likely joint observation. Besides the importance of the multiprocessor scheduling problem itself, this group allows us to evaluate the performance of the methods when the search space is large but the treewidth (of the underlying graph of the model) is low. Finally, the third group (last seven lines of the table) consists of grid-structured graphical models whose parameters were uniformly sampled. Each Grid-x-y-z model contains x rows, y columns and z layers. For $z=2$, variables are quaternary and the grid has two layers: one is the grid itself and the other is formed by decision variables that are linked to grid variables in a one-to-one correspondence; for $z=1$, the models are standard planar binary grids, with all border variables chosen as decision variables. These experiments allow us to better evaluate how the performance is affected by the treewidth and the size of the search space (note that grids have treewidth proportional to their smallest "side"). The factors of the graphical models of the second and third groups were generated by sampling independently and uniformly numbers between zero and one. The clique trees for our anytime algorithm using the minimum fill-in heuristic [van den Eijkhof et al., 2007].

We compare our anytime algorithm against SamIam's implementation of the systematic search algorithm of Park and Darwiche [2003], which we call SI. We

[9] At the time this manuscript was written, the networks were available at `http://www.cs.huji.ac.il/site/labs/compbio/Repository/`.

[10] The multiprocessor scheduling problem is to assign each of a given set of jobs to one of many available processors so as to minimize the overall workload [Garey and Johnson, 1979].

NETWORK	n	d	SI	AFSE	Z/Z^*
Insurance	27	2	0.2s	0.9s	1
Alarm	37	12	0.1s	0.2s	1
Barley	48	10	10s	$>$1h	0.1
Hailfinder	56	17	0.5s	1.4s	1
Pigs	441	145	6m	5m	1
MS-3-20	42	20	2.2s	0.1s	1
MS-3-50	102	50	$>$1h	0.4s	1
MS-3-100	202	100	$>$1h	11m	1
Grid-4-10-2	80	40	$>$1h	7m	0.96
Grid-4-25-2	200	100	$>$1h	22m	0.73
Grid-4-30-2	240	120	$>$5h	2.6h	0.55
Grid-6-6-1	36	20	1.1s	0.1s	1
Grid-10-10-1	100	36	1.8s	7s	1
Grid-16-16-1	256	60	48s	12m	1
Grid-18-18-1	324	68	–	2.9h	–

Table 2.2. Performance of Anytime Factor-Set-Elimination algorithm (AFSE) and SamIam (SI) on real and synthetic models.

chose SI because (i) it is a state-of-the-art algorithm, (ii) its implementation is publicly available, (iii) it is also an anytime procedure, and (iv) it returns feasible solutions.

Table 2.2 shows the results of the experiments, comparing the proposed method (named AFSE for short) and SI. The columns in the table refer, respectively, to the model names, total number of variables (n), number of decision variables (d), amount of time that SI and AFSE, respectively, spent to solve the instances, and the relative error of the solution obtained by the worst algorithm in that problem (in case one of the methods was unable to solve the instance in a reasonable amount of time and memory). The error Z/Z^* is calculated as the ratio of the worst value Z by an algorithm and the optimum Z^* (which of the two methods obtained such an error can be deduced by comparing the time each method spent; the error differs from one only if a method was not able to finish within a time limit of t, which is denoted by a "$>t$" in the table).

According to Table 2.2 AFSE greatly outperformed SI in the models in the second group and the two-layer grids of the third group, which indicates that AFSE performs best when the treewidth of the model is small, irrespective of the size of the search space. Even though SI was able to find the best solution

in the models of the second group (but it never converged, so the search have not stopped), it performed much worse in the two-layer grids, as can be seen in the error column of the table, which reaches 55% in Grid-4-30-2. This means that not only did the algorithm not finish within the long time limit but the best solution found was very poor. Such situations justify the use of methods that can provide anytime lower and upper bounds for the solution. Also, AFSE performed similarly to SI in (real) Bayesian networks, with the largest differences in the Barley and Pigs networks (the former favorable to SI, the latter favorable to AFSE). We see on the squared grids of the third group that SI can better handle the increase of treewidth, indeed a known characteristic of SI. The exception is Grid-18-18-1, where SI exhausted the 8 GB of memory granted without being able to produce a (candidate) solution. Finally, the time-accuracy trade-off of the algorithms can be seen in Figure 2.4, which shows the accuracy of AFSE and SI on models Grid-4-30-2 and Grid-4-25-2 as a function of time. Lower and upper bounds converge to the optimal solution, and while SI starts with a better lower bound, it gets stuck in the search and does not converge within the time limit.

2.5 Conclusion

In this chapter, we discussed the maximum a posteriori assignment problem in the presence of latent variables, which seeks for a configuration of a subset of the variables that maximizes the posterior probability. While these models are building blocks in many applications, most approaches find approximate solutions that can be arbitrarily poor.

To remedy this situation, we present a new anytime algorithm that improves its solution as more computation time is granted. The correctness and complexity of the algorithm are thoroughly analyzed, and bounds are obtained on the precision of the algorithm in any finite time.

The theoretical analysis is supported by experiments with real and synthetic graphical models, and a comparison against a competitive algorithm. In particular, the new algorithm compares favorably when the problems exhibit moderate treewidth but large search space. Unfortunately, as the treewidth increases, the bounds returned by the algorithm become too loose. This could be mitigated by decomposing the propagated factors into smaller domains, as in the work of Meek and Wexler [2011].

We note however that in these experiments the new algorithm is initialized with an arbitrary solution. We could improve the convergence of the algorithm

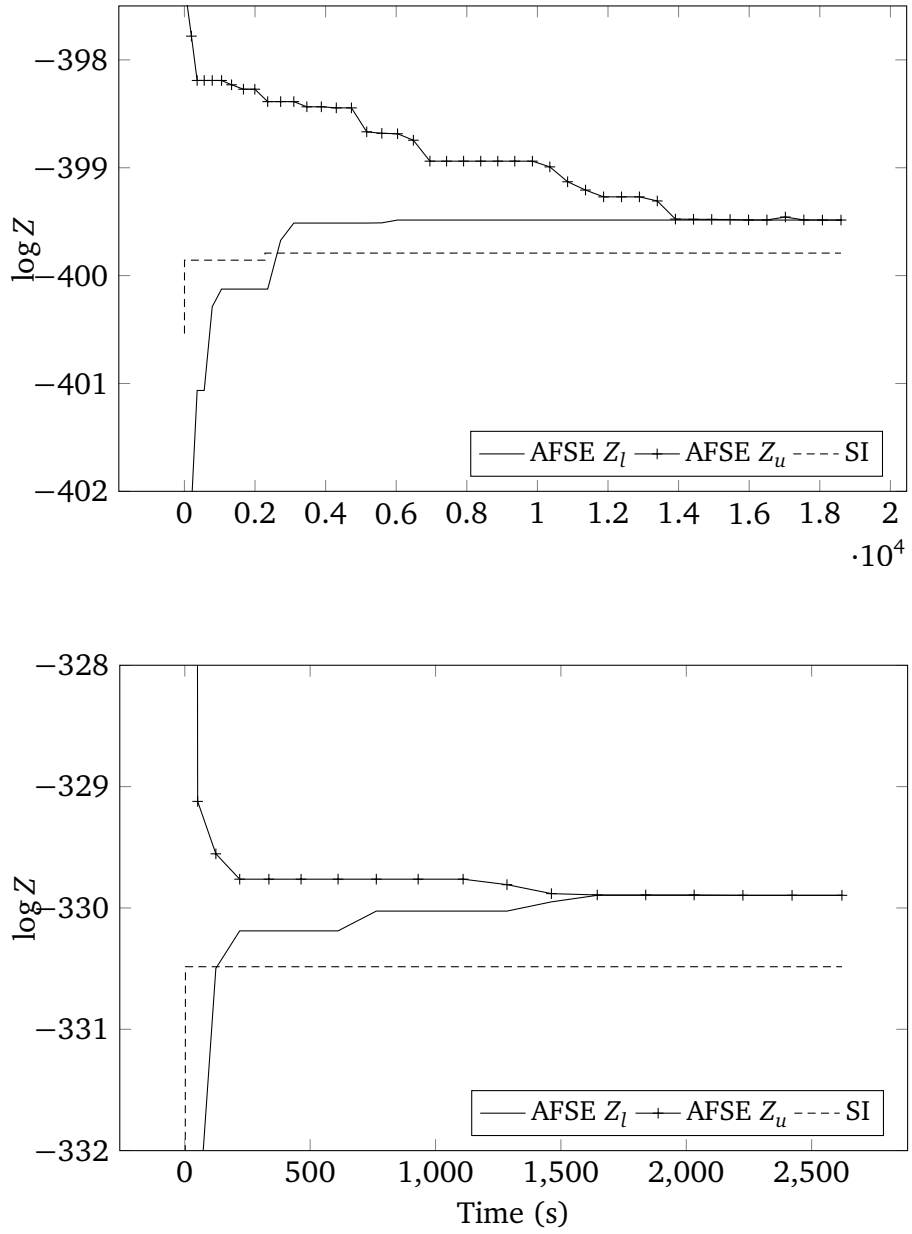


Figure 2.4. Quality of the solutions of AFSE and SI on the Grid-4-30-2 (top) and Grid-4-25-2 (bottom) models by running time.

by using common heuristics for finding assignments such as 1-neighborhood local search, sequential classification, and MPE initialization [Park and Darwiche, 2004].

Understanding how the numerical parameters of the input affect the complexity of the algorithm is an important question that remains open.

Chapter 3

Probabilistic planning in limited memory influence diagrams

Decision problems arise when one needs to select one among many possible actions in order to achieve some pre-defined goal. The problem is often made more difficult by the presence of uncertainty about the outcomes of the actions. Decision theory provides a principled framework for representing and solving decision problems under uncertainty. At the heart of the theory is the notion of (expected) utility: a number quantifying the decision maker's preference about a certain outcome.

The MAP assignment problem discussed in the previous chapter is a special type of decision problem: one needs to choose one among many possible assignments to action variables in order to maximize the posterior probability. In more general settings, actions can be taken conditionally on certain inputs or observations, and at different time steps; they can be taken by different agents, and their value might change over time. Examples include, but are not limited to, robot trajectory planning, home monitoring and assistance of elderly and disable people, and cooperative task solving in multi-agent systems [Kaelbling et al., 1998; Thrun et al., 2005; Hoey et al., 2013].

Influence diagrams [Howard and Matheson, 1984] are probabilistic graphical models especially designed for utility-based decision making under uncertainty. An influence diagram can be seen as a detailed description of a finite-horizon partially observable Markov decision process (POMDP), which is used to model many situations involving probabilistic reasoning [Tatman and Shachter, 1990; Kaelbling et al., 1998; Jensen and Nielsen, 2007]. Relative to the description of a POMDP as tables representing the transition and reward functions, an influence diagram provides a concise representation by exploiting functional

independencies between the variables.

Typically, influence diagrams are used to target situations involving a single, non-forgetful decision maker. This is implicit in the general assumptions of *regularity* and *no-forgetting* that most planning algorithms for influence diagrams make. Regularity refers to the assumption of a complete temporal ordering of the decisions, usually in the form of a single directed path connecting all decision variables in the diagram. No-forgetting requires that actions are taken conditionally on the whole history of actions and observations. Together, these assumptions imply that each agent being modeled is fully informed and counts on unlimited resources.

In many real situations, bounded resources and physical constraints force decisions to be made based on limited information [Zhang et al., 1994; Lauritzen and Nilsson, 2001]. For instance, an agent acting according to a POMDP might be forced to disregard part of the available information in order to meet computational demands [Meuleau et al., 1999; Poupart and Boutilier, 2003]. Cooperative multi-agent settings offer yet another example: each agent might perceive only its surroundings and be unable to communicate with all other agents; hence, a policy specifying an agent's behavior must rely exclusively on local information [Hansen, 1998; Bernstein et al., 2005; Detwarasiti and Shachter, 2005; Wu et al., 2011]; it might be further constrained to a maximum size to be computationally tractable [Amato et al., 2010].

Limited memory influence diagrams (LIMIDs) generalize influence diagrams to allow for the explicit representation of bounded memory agents and cooperative and distributed decision making [Zhang et al., 1994; Lauritzen and Nilsson, 2001]. More precisely, LIMIDs relax the *regularity* and *no forgetting* assumptions of influence diagrams. A distinguishing feature of LIMIDs (in comparison with standard influence diagrams or typical encodings of POMDPs) is that the information available to any action variable is made explicit in the graphical structure. This is particularly important when one attempts to determine the theoretical computational complexity of planning with such models, as we do in this chapter.

Any influence diagram describing a fully observable Markov decision process has an optimal strategy (i.e., a collection of mappings from observations to actions for each decision stage maximizing expected utility) whose size grows linearly with the number of action variables. Moreover, this strategy can be obtained by dynamic programming in polynomial time in the number of possible observations in each decision stage [Tatman and Shachter, 1990; Jensen and Nielsen, 2007]. However, planning under the no-forgetting assumption in partially observable Markov decision processes suffers from the curse of history:

the size of an optimal strategy might grow exponentially large with the number of decision stages considered (e.g., the strategy includes a table prescribing a value for an action variable conditional on each one of the exponentially many assignments of the other action variables).

Lauritzen and Nilsson advocated the use of LIMIDs as a useful framework for explicitly modeling the computational constraints faced by planning algorithms in those cases [Lauritzen and Nilsson, 2001]. In this scenario, a LIMID is the outcome of removing arcs entering action nodes in an influence diagram that initially respected regularity and non-forgetting, until the size of a plan (represented as a collection of tables) is sufficiently small. In other words, a LIMID is a bounded-memory version of a decision problem initially represented as an influence diagram. Zhang, Qi, and Poole [1994] and more recently Lauritzen and Nilsson [2001] determined sufficient conditions under which even influence diagrams that violate no-forgetting can be solved exactly and efficiently by dynamic programming. Roughly speaking, the conditions state that the influence diagram can be efficiently translated into a fully observable Markov decision process by aggregating state variables and (re-)ordering actions. Any diagram of bounded treewidth meeting those conditions can be solved in polynomial time in the size of the diagram. As de Campos and Ji [2008] showed, even diagrams of bounded treewidth can fail to meet these conditions and be difficult to solve. Indeed, we show later on that singly connected LIMIDs of bounded treewidth with binary variables and a single value node can be solved in polynomial time, but removing any of these conditions can lead to intractable planning problems. Moreover, removing any but the assumption of bounded variable domain cardinality makes the problem hard to approximate.

A possible approach to solve a LIMID is to include arcs entering action variables so as to make the resulting diagram satisfy the sufficient conditions for efficient dynamic programming [Nilsson and Höhle, 2001; Jensen and Nielsen, 2007; Yuan et al., 2010]. In terms of the real problem, this implies observing quantities that were initially deemed unobservable by the semantics of the LIMID model, which might be undesirable or even unfeasible. Also, the value of the solution obtained by the augmented diagram in this way is an upper bound on the value of an optimal solution of the original diagram that can be arbitrarily loose.

A different approach to planning with LIMIDs is to use local search methods that sacrifice provably good accuracy for efficiency. Lauritzen and Nilsson [2001] and Detwarasiti and Shachter [2005] proposed combinatorial procedures that directly search the space of policies, while Liu and Ihler [2012] developed a message-passing algorithm that efficiently optimizes a surrogate objective

function. The inapproximability results we develop in this chapter together with some preliminary experiments we report suggest that these greedy approaches provide low quality solutions in a non-negligible fraction of diagrams representing partially observable processes.

Compared to other tasks in probabilistic reasoning and decision making, there has not been any in-depth study of the theoretical complexity of planning with LIMIDs, especially when the complexity is given as a function of structural parameters such as diagram treewidth, variable cardinality and number of value nodes. In the rest of this chapter, we give an in-depth discussion of the fixed-parameter complexity of exact and approximated planning in limited memory influence diagrams. We start, in Section 3.1, by formally describing the syntax and semantics of influence diagrams. We then state the main problem we address in this chapter, namely, the problem of finding a good strategy for a LIMID (Section 3.2). We move on to define what single and multi-stage diagrams are, and show their equivalence in the class of bounded treewidth diagrams (Section 3.3). Section 3.4 contains the results about the complexity of solving LIMIDs exactly, whereas Section 3.5 contains the results about the complexity of finding provably good approximations for the problem. The chapter ends with a recapitulation of the material covered and a final discussion (Section 3.6).

The material presented here is based on the contents appearing in the References [Mauá and de Campos, 2011], [Mauá et al., 2012] and [Mauá et al., 2012a].

3.1 Limited memory influence diagrams

To help introduce the notation and illustrate concepts, consider the following simple example of a decision problem.

Example 3.1. *Two agents R_1 and R_2 need to coordinate to accomplish a simple task. Each agent has two possible actions available, with different implications on the state of the world. Let X_{a_1} and X_{a_2} be discrete variables modeling the actions taken, respectively, by the agents R_1 and R_2. Suppose the agent R_1 acts first. An action x_{a_1} leads to an outcome x_{s_1} of finitely many possible outcomes with probability $\mathrm{P}(x_{s_1}|x_{a_1})$ (X_{s_1} is a discrete variable representing the state of the world after the action). This outcome is not observable by neither of the agents, and remains unknown. R_1 then informs R_2 of her action, based on which R_2 executes an action x_{a_2}, causing x_{s_2} to obtain with probability $\mathrm{P}(x_{s_2}|x_{s_1}, x_{a_2})$. The overall utility U is a deterministic function of the joint (unknown) configuration $\mathbf{x}_{a_1,s_1,a_2,s_2} = \{x_{a_1}, x_{s_1}, x_{a_2}, x_{s_2}\}$ of actions and outcomes assessing its desirability; the*

higher the value of the utility the more desirable the configuration for both agents. Assume that the utility of any configuration decomposes as a sum of intermediate rewards X_{v_i}, $i = 1, 2$, *that depend each only on the immediate outcome of agent i's action, that is,* $U = X_{v_1} + X_{v_2}$, *and that each* X_{v_i} *is a binary variable that evaluates to one if* $X_{s_i} = 0$ *and vanishes otherwise. Then the expected utility of the joint plan or strategy* $\Delta = \{(X_{a_1} = 1); (\textit{if } X_{a_1} = 1 \textit{ then } X_{a_2} = 1; \textit{if } X_{a_1} = 0 \textit{ then } X_{a_2} = 0)\}$ *is* $\mathrm{E}_\Delta(U) = \sum_{x_{v_1}} x_{v_1} \mathrm{P}(x_{v_1} | X_{a_1} = 1) + \sum_{x_{v_2}} x_{v_2} \mathrm{P}(x_{v_2} | X_{a_2} = 1) = \mathrm{P}(X_{s_1} = 1 | X_{a_1} = 1) + \mathrm{P}(X_{s_2} = 1 | X_{a_2} = 1)$. □

As in the above example, the quantities and events of interest in a discrete decision problem can be represented by a set of discrete variables $\mathbf{X} = \{X_i : i \in N\}$. These include *state variables*, which represent the unknown quantities over which the agent has no control (e.g. uncertain outcomes of an action), *action variables*, that enumerate the alternative courses of action at a given point, and *value variables*, that assess the decision maker's preference regarding a certain partial state of the world. The sets of state, action and value variables are denoted, respectively, by $\mathbf{X}_S = \{X_i : i \in S\}$, $\mathbf{X}_A = \{X_j : j \in A\}$, and $\mathbf{X}_V = \{X_k : k \in V\}$, with S, A and V forming a partition of N. We assume that there are only finitely many variables, and that each variable assumes finitely many values. Also, value variables take on real values.

Influence diagrams are graphical representations of structured decision problems [Howard and Matheson, 1984]. An influence diagram represents both the agents' architecture (i.e., what information is available to the agent at each decision stage of the problem) and the environment by means of a directed acyclic graph (DAG) $G = (N, E)$ where each node i is associated with a variable X_i in $\mathbf{X}$. The nodes in N are partitioned into sets of state, action and value nodes, according to the type of variable with which they are associated. An arc from a node i into an action node j in the graph indicates that by the time the corresponding action X_j is made the value of the variable associated with node i will be known. Hence, a different action $X_j = x_j$ can be planned for each possible value of that variable, as in the following example.

Example 3.2. *The problem in Example 3.1 can be represented by the influence diagram whose structure is depicted as the DAG in Figure 3.1 (as usual, state, action and value nodes are represented by ovals, squares and diamonds, respectively). The arc connecting* a_1 *to* a_2 *represents that by the time an action* a_2 *is selected the decision maker knows which action* a_1 *was selected, thus a different action* a_2 *can be chosen conditional on each action* a_1. □

The *regularity* condition states that the all variables can be linearly ordered in a way that the decision represented by an action variable depends only on the

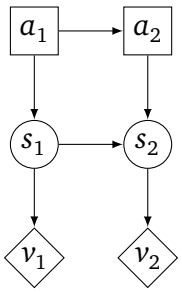


Figure 3.1. Influence diagram representation for the problem in Example 3.1.

variables that precede it in the order. Regularity is usually made stronger by assuming that the graph of the problem contains a single directed path connecting all action variables. The action variables are then linearly ordered according to the topological order.

The *no-forgetting* condition states that decisions and observations are permanently "remembered". Graphically, it entails that if i and j are two action nodes such that i is a parent of j, then all parents of i are also parents of j.[1] When the diagram satisfies regularity, then no-forgetting requires an action node i to be a parent of every action node j such that $j > i$ (i.e., j succeeds i in the linear ordering defined).

An influence diagram is said to have limited memory if the no-forgetting condition is not met. The least memory intensive agent architecture is arguably the one in which actions are taken independently of each other. Graphically, it is represented by a LIMID whose action nodes have no parents.

An arc from a node j into a state node i indicates that the probability of X_i depends on the variable associated to node j. Similarly, an arc entering a value node k from a node i indicates that the variable X_k is a (deterministic) function of the variable associated with node i. We assume that value nodes are leaves in the graph, and that the overall utility U decomposes additively in terms of the value variables, that is, that $U = \sum_{k \in V} X_k$.

A *strategy* $\Delta = \{\delta_j : j \in A\}$ is a multiset of local decision rules, or *policies* one for each action variable X_j in $\mathbf{X}_A$. Each policy δ_j is a mapping of assignments $\mathbf{x}_{\mathrm{Pa}(j)}$ to the variables $\mathbf{X}_{\mathrm{Pa}(j)}$ into values $x_j = \delta_j(\mathbf{x}_{\mathrm{Pa}(j)})$ of X_j. We assume that policies are encoded as tables associating every assignment of the parents to a value of the decision variable. *A policy for an action variable with no parents is simply an assignment of a value to that variable*. Associated with each action vari-

[1] We assume here that when no-forgetting is satisfied the "remembered" arcs are explicitly represented in the diagram.

able X_j and policy δ_j, there is a degenerate conditional probability distribution $\mathrm{P}(X_j|\mathbf{x}_{\mathrm{Pa}(j)}) = I(x_j = \delta_j(\mathbf{x}_{\mathrm{Pa}(j)}))$ (or $\mathrm{P}(X_j) = I(x_j = \delta_j)$ in case j has no parents), where $I(\cdot)$ denotes the indicator function that evaluates to one when the argument is true, and equals zero otherwise. With this correspondence between policies and (conditional) probability distributions, we define a joint probability distribution over state and action variables for a given strategy Δ as

$$\mathrm{P}_\Delta(\mathbf{x}_{S\cup A}) \stackrel{\mathrm{def}}{=} \prod_{i\in S} \mathrm{P}(x_i|\mathbf{x}_{\mathrm{Pa}(i)}) \prod_{j\in A} I(x_j = \delta_j(\mathbf{x}_{\mathrm{Pa}(j)})). \tag{3.1}$$

The identity above describes the joint probability distribution over variables $\mathbf{X}_{S\cup A}$ induced by the Bayesian network whose structure is the DAG of the influence diagram without value nodes, and whose parameters are $\mathrm{P}(X_i|\mathbf{X}_{\mathrm{Pa}(i)})$ for every $i \in S \cup A$. The expected utility of a strategy Δ is therefore

$$\mathrm{E}_\Delta(U) = \sum_{k\in V} \mathrm{E}_\Delta(X_k) = \sum_{k\in v} \sum_{\mathbf{x}_{\mathrm{Pa}(k)}} \mathrm{P}_\Delta(\mathbf{x}_{\mathrm{Pa}(k)}) \mathrm{E}(X_k|\mathbf{x}_{\mathrm{Pa}(k)}), \tag{3.2}$$

where $\mathrm{P}_\Delta(\mathbf{x}_{\mathrm{Pa}(k)}) = \sum_{\mathbf{x}_{(S\cup A)\setminus \mathrm{Pa}(k)}} \mathrm{P}_\Delta(\mathbf{x})$.

The conditional probabilities associated to state variables in Equation (3.1) and the conditional expected values of the value variables in Equation (3.2) can be specified independently of the strategy. An influence diagram is fully specified by its DAG G annotated with node types, its set of variables $\mathbf{X}$, the conditional expected values of value variables $\mathrm{E}(X_k|\mathbf{x}_{\mathrm{Pa}(k)})$ associated to each value variable X_k, and the conditional probability values $\mathrm{P}(x_i|\mathbf{x}_{\mathrm{Pa}(i)})$ associated to each state variable X_i. When proving hardness results, we implicitly assume that these probabilities and expected values are specified by *rational* numbers.

3.2 Solving LIMIDs

Planning in (limited memory) influence diagrams refers to maximizing expected utility on the space of strategies. As with other optimization problems, the problem comes in three variants, one of *deciding* whether any strategy obtains an expected utility greater than a given threshold, one of *evaluating* (i.e., calculating) the maximum expected utility, and one of *selecting* an optimal strategy. Provided that evaluating any given strategy is polynomial-time computable and that strategies are shortly encoded relative to the size of the encoding of the diagram, these three variants are equivalent from a complexity point of view.[2] Indeed, under such assumptions, the first two variants are trivially solved by the strategy

[2]This equivalence is satisfied for all NP optimization problems whose decision version is NP-complete [Paz and Moran, 1981; Ausiello et al., 1995]. To meet the requirements of NP opti-

selection variant. Moreover, to be able to achieve polynomial-time speed, an algorithm to evaluate strategies needs to output numbers whose encoding takes size at most polynomial in the size of the input. Hence, the tractability of strategy evaluation implies the existence of a polynomial poly(b), where b is the number of bits encoding the influence diagram, such that the expected utility of any strategy is bounded in absolute value by $2^{\text{poly}(b)}$ (otherwise we would need super-polynomially many bits to write the output), and such that the expected utilities of any two strategies are either equal or differ by at least $1/2^{\text{poly}(b)}$ (otherwise they would be indistinguishable by an algorithm running in polynomial-time). Consequently, the evaluation version can be efficiently computed by performing a binary search using at most 4poly(b) calls of a polynomial-time algorithm that solves the decision version. Also, given a polynomial-time algorithm for the decision version, we can select an optimal strategy by iteratively deciding whether there is a strategy that achieves the maximum expected utility of the initial diagram in a new LIMID where the current node is replaced by a state variable with degenerate conditional probability (we need to test all such possible functions). Hence, when no confusion arises, we shall often speak loosely of "solving" a LIMID without explicit mention to which variant of the problem we address. The capability of efficiently evaluating and shortly encoding strategies is deeply connected with the shape of the diagram, and particularly with its treewidth, both of which we review next.

We say that a LIMID is *polytree-shaped* or *singly connected* if the undirected graph we obtain by dropping arc directions is a tree. If a diagram is not a polytree, it is called *multiply connected* or *loopy*. The treewidth of an influence diagram is a measure of the resemblance of its *moral graph* to a tree, the moral graph being the (undirected) graph we obtain by linking any two nodes with a common child, removing value nodes, and ignoring arc directions.

A *tree decomposition* of (the moral graph of) an influence diagram is a tree where each node is associated to a subset of the state and action variables in the diagram.[3] The decomposition satisfies the *family preserving* and *running intersection* properties, which state that the family of each action and state node, and the parent set of each value node, is contained in at least one set associated to a node of the tree, and that the graph obtained by dropping nodes that do not contain any given state or action node is still a tree. If T is a tree decomposition with m nodes, we denote by $C_1, \ldots, C_m$ the sets of nodes of the diagram associated to nodes $1, \ldots, m$ of the tree, respectively. Thus, the family preserving

mization problems, we must require that any strategy can be evaluated in polynomial time, and that optimal strategies can be encoded in time and space polynomial in the input size.

[3]This definition is essentially equivalent to the notion of a clique tree, defined in Chapter 2.

property implies that $C_1 \cup \cdots \cup C_m = S \cup A$. The width of a tree decomposition is the maximum cardinality of a set C_i associated to a node i minus one. The *treewidth* of an influence diagram is the minimum width of a tree decomposition of it. Intuitively, the treewidth of a diagram measures the resemblance of its underlying moral graph to a tree. The treewidth of a tree-shaped diagram is one, and is minimal. The treewidth of a singly connected diagram is the maximum in-degree of a node. The following facts concerning tree decompositions will be useful in our future discussion about the complexity of solving LIMIDs.

For a fixed integer k, Bodlaender [1996] showed that one can in time linear in the size of a graph either obtain a tree decomposition of width at most k or know that such a decomposition does not exist. Hence, for any diagram of bounded treewidth we can obtain in linear time an optimal tree decomposition, that is, a tree decomposition of minimum width.[4]

Any tree decomposition can be turned into a *binary tree decomposition* (i.e., one in which each node has at most three neighbors) of same treewidth in linear time by inserting new nodes such that the number of nodes in the binary tree does not exceed twice the number of nodes in the original tree-decomposition, and the treewidth remains the same, which implies that we can obtain a binary tree decomposition of minimum treewidth in linear time for any influence diagram of bounded treewidth [Shenoy, 1997].

Any tree decomposition of a diagram contains supersets of the cliques of the moral graph associated to its nodes. Furthermore, there is a tree decomposition whose associated node sets are exactly the cliques of the moral graph if and only if the moral graph is triangulated, that is, if any cycle with more than three nodes contains a chord (a link between non-consecutive nodes in the cycle) [Jensen and Nielsen, 2007, Theorems 4.3 and 4.4, page 122]. A node in a(n undirected) graph is called *simplicial* if its neighbors form a clique. If G is a graph of treewidth k and i is a simplicial node in G of degree d, then the graph G' obtained by removing i and all its incident edges has treewidth k' such that $k = \max\{d, k'\}$ [Bodlaender et al., 2001]. This is a useful result for analyzing the treewidth of graphs obtained by augmenting a graph of known treewidth with simplicial nodes.

Given an influence diagram whose variables take on at most v values, and a suitable tree decomposition of width k, we can evaluate the expected utility of any strategy Δ using tree-decomposition algorithms in time and space $O(v^k)$ [Koller and Friedman, 2009, Chapter 23]. Hence, given an influence

[4] Starting with $k=1$, we can run Bodlaender's algorithm with an increasing value of k until a tree decomposition is found. If the treewidth is bounded by B, the procedure takes $O(Bf(B))= O(f(B))$ time, where $f(B)$ is the time to build a tree-decomposition of treewidth at most B.

diagram of bounded treewidth and a strategy, we can obtain an optimal tree-decomposition in linear time as discussed, and therefore compute the expected utility of the strategy in polynomial time. De Campos and Ji [2008] showed that deciding whether the maximum expected utility exceeds a given threshold is NP^{PP}-complete when the diagram has bounded in-degree, and NP-complete if it has bounded treewidth. In fact, we show later on that the problem is already NP-hard in even much simpler models, and even if we allow approximate solutions.

Not all the arcs and variables in a LIMID are relevant to the computation of optimal strategies, and the complexity of the problem can be drastically reduced by removing nodes and arcs that do not affect the expected utility of an optimal strategy [Shachter, 1998; Fagiuoli and Zaffalon, 1998a; Shachter, 1999; Nielsen and Jensen, 1999; Lauritzen and Nilsson, 2001]. A state or action node is called *barren* if it has no children. Barren nodes have no influence on any value node and thus no impact on the selection of an optimal strategy [Jensen and Nielsen, 2007]. Further irrelevances can be found by applying the concept of d-separation [Pearl, 1988] and non-requisiteness [Fagiuoli and Zaffalon, 1998a; Lauritzen and Nilsson, 2001].

The notion of d-separation is based on the concept of active and inactive (or blocked) trails in a directed graph. A triple of nodes i, k, j is said to be *active* with respect to a set of nodes Z either if i and j are both parents of k and either k or some of its descendants is in Z, or if i and j are not both parents of k and k is not in Z. A *trail* is a sequence of nodes containing an arc for any two consecutive nodes. Notice that a trail does not need to "follow" the direction of the arcs. A trail is *blocked* by a set of nodes Z if it contains a triple of consecutive nodes which is not active with respect to Z. Two sets of nodes X and Y are d-separated by a set of nodes Z if all trails from a node i in X to a node j in Y are blocked by Z. Intuitively, if X and Y are d-separated by Z, then any i in X is irrelevant to any j in Y and vice-versa once we know the state of the variables in Z.

A parent node k of an action node j is *nonrequisite* to j if it is d-separated from all the value nodes that descend from j given the remaining parents of j and j itself. The arc from k to j is then said to be a *nonrequisite arc*. A nonrequisite arc from node k into node j indicates that an optimal policy for X_j is invariant with respect to values of X_k, and thus removing the arc from k into j leaves the maximum expected utility unchanged [Fagiuoli and Zaffalon, 1998a; Lauritzen and Nilsson, 2001].[5] A variable that is nonrequisite to all its

[5]Note however that removing an arc entering an action node alters the structure of the solution of the problem.

children can be safely removed from the diagram without affecting the expected utility of an optimal strategy. Thus, the exponential growth of policies induced by requiring no-forgetting can be avoided if all memory arcs (that is, arcs from parents of a parent action node into an action node) are nonrequisite. This is the case, for instance, when state variables form a chain in the graph.

By removing a barren node, other nodes might become barren too. Similarly, by removing a nonrequisite arc, we might create new barren nodes and/or nonrequisite arcs. We say that a LIMID is *minimal* if it contains no nonrequisite arcs or barren nodes. Given a LIMID we can obtain its corresponding minimal form in polynomial time by repeatedly removing nonrequisite arcs and barren nodes [Lauritzen and Nilsson, 2001]. In singly connected LIMIDs, all arcs entering action nodes are by definition nonrequisite.

3.3 Single- and multi-stage influence diagrams

We say that a LIMID whose action nodes have no parents is *single stage*; otherwise, we say it is *multi stage*. A single-stage LIMID represents a situation in which all decisions are either made simultaneously or independently, in sharp contrast to multi-stage decision problems, where decisions the decisions of one stage depend on the outcomes of the previous stages. As it turns out, any multi-stage LIMID of bounded treewidth can be efficiently transformed into an equivalent single-stage diagram. Equivalence means that there is a bijection from strategies of the transformed diagram into the original diagram that can be computed in polynomial time, and that preserves expected utility of strategies.

Transformation 3.1. *Consider a multi-stage LIMID $\mathscr{L}$ and a decision node d with at least one parent, and let $\pi^{(1)},\ldots,\pi^{(m)}$ denote the possible (joint) assignments to the variables $\mathbf{X}_{\mathrm{Pa}(d)}$. Obtain a new diagram $\mathscr{L}'$ as follows. Remove X_d and add m state variables (and their corresponding nodes) $X_{i_1},\ldots,X_{i_m}$ and m action variables (and the corresponding nodes) $X_{j_1},\ldots,X_{j_m}$, all taking values in the domain of X_d. Add an arc from every parent of d to each of $i_1,\ldots,i_m$, an arc from every i_k to i_{k+1}, with $k<m$, and an arc from every j_k to i_k, $k=1,\ldots,m$. Also, add an arc from i_m to each child of d. Specify the conditional probability of X_{i_1} as*

$$\mathrm{P}(x_{i_1}|x_{j_1},\mathbf{x}_{\mathrm{Pa}(d)})=\begin{cases}1, & \text{if } \mathbf{x}_{\mathrm{Pa}(d)}=\pi^{(1)} \text{ and } x_{i_1}=x_{j_1},\\ 0, & \text{if } \mathbf{x}_{\mathrm{Pa}(d)}=\pi^{(1)} \text{ and } x_{i_1}\neq x_{j_1},\\ 1/m & \text{if } \mathbf{x}_{\mathrm{Pa}(d)}\neq\pi^{(1)}.\end{cases}$$

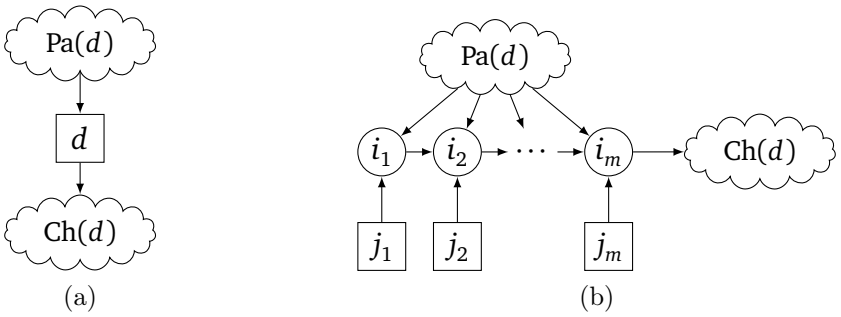


Figure 3.2. A piece of a diagram before (a) and after (b) Transformation 3.1.

For $k = 2, \ldots, m$, specify the conditional probability of each node X_{i_k} as

$$\mathrm{P}(x_{i_k} | x_{i_{k-1}}, x_{j_k}, \mathbf{x}_{\mathrm{Pa}(d)}) = \begin{cases} 1, & \text{if } \mathbf{x}_{\mathrm{Pa}(d)} \neq \pi^{(k)} \text{ and } x_{i_k} = x_{i_{k-1}}, \\ 0, & \text{if } \mathbf{x}_{\mathrm{Pa}(d)} \neq \pi^{(k)} \text{ and } x_{i_k} \neq x_{i_{k-1}}, \\ 1, & \text{if } \mathbf{x}_{\mathrm{Pa}(d)} = \pi^{(k)} \text{ and } x_{i_k} = x_{j_k}, \\ 0, & \text{if } \mathbf{x}_{\mathrm{Pa}(d)} = \pi^{(k)} \text{ and } x_{i_k} \neq x_{j_k}. \end{cases}$$

Finally, the conditional probability functions $\mathrm{P}(X_i | \mathbf{X}_{\mathrm{Pa}(i)})$ of each child i of d have X_d substituted by X_{i_m} in their domain, but otherwise remain the same. □

Figure 3.2 depicts a decision node with possibly many parents and many children (on the left) and the new sub-diagram generated by applying Transformation 3.1 (on the right).

Lemma 3.1. *Let $\mathscr{L}'$ be the result of applying Transformation 3.1 in a diagram $\mathscr{L}$ of treewidth k. Then the treewidth of $\mathscr{L}'$ is at most $k+2$, and this bound is tight for some diagram $\mathscr{L}$.*

Proof. Let $\mathscr{L}'$ be the result of applying the transformation in a diagram $\mathscr{L}$ of treewidth k. Moreover, let M and M' be the moral graphs of $\mathscr{L}$ and $\mathscr{L}'$, respectively. We can obtain M from M' by sequentially eliminating nodes $j_1, \ldots, j_m$ and $i_1, \ldots, i_{m-1}$, in this order, and replacing i_m with d. Let $M_1, \ldots, M_{2m}$ be the graphs obtained after applying each of these operations. Thus, M_1 is the graph obtained by removing j_1 from M', and M_{2m} equals M. Let also $k_1, \ldots, k_{2m}$ be the treewidth of the graphs $M_1, \ldots, M_{2m}$, respectively, and k_0 be the treewidth of M'. The node j_1 is simplicial and has degree $|\mathrm{Pa}(d)|+1$ in M'. Since M_1 contains the clique $\mathrm{Fa}(i_m) = \{i_m, i_{m-1}, j_m\} \cup \mathrm{Pa}(d)$ (where $\mathrm{Fa}(i_m)$ is taken with respect to M_1 and $\mathrm{Pa}(d)$ is taken with respect to M), it follows that $k_1 \geq |\mathrm{Pa}(i_m)| = |\mathrm{Pa}(d)|+2$,

which implies $k_0 = \max\{|\mathrm{Pa}(d)| + 1, k_1\} = k_1$. Assume that $k_\ell = k_0$, for some $1 \leq \ell < m-1$. The node $j_{\ell+1}$ is simplicial and has degree $|\mathrm{Pa}(d)| + 2$ in M_ℓ. The treewidth $k_{\ell+1} \geq |\mathrm{Pa}(d)| + 2$ because $M_{\ell+1}$ contains the clique $\mathrm{Fa}(i_m) = \{i_m, i_{m-1}, j_m\} \cup \mathrm{Pa}(d)$. Hence, $k_\ell = \max\{|\mathrm{Pa}(d)| + 2, k_{\ell+1}\} = k_{\ell+1}$, and by induction we have that

$$k_0 = k_{m-1}\,.$$

The node j_m is simplicial and has degree $|\mathrm{Pa}(d)|+2$ in M_{m-1}. Since M_m contains the clique $\mathrm{Fa}(i_m) = \{i_m, i_{m-1}\} \cup \mathrm{Pa}(d)$, it follows that $k_m \geq |\mathrm{Pa}(d)| + 1$, and thus $k_{m-1} = \max\{|\mathrm{Pa}(d)| + 2, k_m\} \leq k_m + 1$. Hence,

$$k_{m-1} \leq k_m + 1\,.$$

A similar reasoning applies for k_ℓ with $m < \ell < 2m$. M_{m+1} (i.e., the graph obtained by removing i_1) contains a clique of size $|\{i_m, i_{m-1}\} \cup \mathrm{Pa}(d)| = |\mathrm{Pa}(d)| + 2$, and the node i_1 is simplicial and has degree $|\mathrm{Pa}(d)| + 1$ in M_m. Hence, $k_m = \max\{|\mathrm{Pa}(d)|+1, k_{m+1}\} = k_{m+1}$. Assume $k_\ell = k_m$ for $m < \ell < 2m-2$. Then $i_{\ell-m+1}$ is simplicial and has degree $|\mathrm{Pa}(d)| + 1$ in M_ℓ. Since $M_{\ell+1}$ contains the clique $\{i_m, i_{m-1}\} \cup \mathrm{Pa}(d)$, it follows that $k_\ell = \max\{|\mathrm{Pa}(d)| + 1, k_{\ell+1}\} = k_{\ell+1}$. Thus, by induction

$$k_m = k_{2m-2}\,.$$

Finally, the graph M_{2m-1} (obtained by removing i_{m-1} from M_{2m-2}) contains the clique $\{i_m\} \cup \mathrm{Pa}(d)$ (so that $k_{2m-1} \geq |\mathrm{Pa}(d)|$), and i_{m-1} is simplicial and has degree $|\mathrm{Pa}(d)| + 1$ in M_{2m-2}. Thus, $k_{2m-2} = \max\{|\mathrm{Pa}(d)| + 1, k_{2m-1}\} \leq k_{2m-1} + 1$. Since the replacement of i_m with d used to generate $M_{2m} = M$ from M_{2m-1} does not change the treewidth (i.e., $k \stackrel{\text{def}}{=} k_{2m} = k_{2m-1}$), we have that

$$k_0 = k_{m-1} \leq k_m + 1 = k_{2m-2} + 1 \leq k_{2m-1} + 2 = k + 2\,.$$

Furthermore, one can show that the bound is tight if $\mathscr{L}$ is a diagram containing one state node S, one action node A and one value node V linked as a chain, that is, $S \to A \to V$. Then the treewidth of the transformed diagram is three while the treewidth of original graph is one. □

The result above can be generalized to multiple applications of Transformation 3.1. Applying the transformation on two different action variables affect different parts of the original diagram, and hence transforming a multi-stage diagram in a single-stage diagram does not increase the treewidth by more than two. The following result shows that the transformation can be made efficiently.

Lemma 3.2. *Transformation 3.1 can be performed in polynomial time on a diagram of bounded treewidth.*

Proof. Let $\mathscr{L}'$ be the result of applying the transformation on action node d in $\mathscr{L}$, and let v be the number of distinct states X_d can assume. As before, let m be the number of assignments to $\mathbf{X}_{\mathrm{Pa}(d)}$. The digraph of transformed diagram $\mathscr{L}'$ contains m additional state variables, and m additional action nodes (plus the corresponding arcs), so it can be obtained in time linear in the size of the digraph of $\mathscr{L}$ and in m. The new state and action variables all assume d states, so specifying the new variable takes time $O(md)$. Finally, the transformed diagram also specifies $O(m^2v^3)$ probability values $\mathrm{P}(x_{i_k}|x_{i_{k-1}}, x_{j_k}, \mathbf{x}_d)$. Let u be the maximum number of states a variable in the family of d can assume, and k be the treewidth of $\mathscr{L}$. Then $m \leq u^{|\mathrm{Pa}(d)|} \leq u^k$. Hence, if $\mathscr{L}$ has bounded treewidth (i.e., if we consider k a constant) then m is a polynomially bounded by u, which is part of the specification of $\mathscr{L}$. Since all steps of the transformation takes time at most polynomial in m, and polynomials are closed under composition, the transformation takes time polynomial in the size of $\mathscr{L}$. □

The bottleneck of the computational performance of the transformation is the potentially high value of m. It is still possible to apply the transformation in polynomial time if the in-degree of action nodes is considered bounded (which is a necessary but not sufficient condition for having bounded treewidth). If the in-degree of action nodes is not bounded then the output of any planning algorithm, that is, an optimal strategy, might take space exponential in the input. Thus, assuming that m is bounded is reasonable. To show the equivalence of single- and multi-stage diagrams, it remains to prove that the transformation preserves the expected utility of the strategies.

Proposition 3.1. *Let $\mathscr{L}'$ be the result of applying Transformation 3.1 on a decision variable X_d in a LIMID $\mathscr{L}$. The following assertions are true.*

- *For each strategy Δ' for $\mathscr{L}'$ we can obtain a strategy Δ for $\mathscr{L}$ in time polynomial in the size of Δ' such that $\mathrm{E}_{\Delta}(U) = \mathrm{E}_{\Delta'}(U)$;*
- *For each strategy Δ for $\mathscr{L}$ we can obtain a strategy Δ' for $\mathscr{L}'$ in time polynomial in the size of Δ such that $\mathrm{E}_{\Delta'}(U) = \mathrm{E}_{\Delta}(U)$.*

Proof. Suppose that $\mathbf{X}_{\mathrm{Pa}(d)} = \pi^{(k)}$ for some $k > 1$. By the chain rule of probability and the stochastic independencies implied by the graphical structure of the diagram, we have that

$$\mathrm{P}(x_{i_k}, \ldots, x_{i_m}|x_{j_k}, \ldots, x_{j_m}, \pi^{(k)}) = \mathrm{P}(x_{i_k}|x_{i_{k-1}}, x_{j_k}, \pi^{(k)}) \prod_{\ell > k} \mathrm{P}(x_{i_\ell}|x_{i_{\ell-1}}, x_{j_\ell}, \pi^{(k)}).$$

By definition, each state variable X_{i_ℓ} with $\ell \neq k$ is stochastically independent of its parent action variable X_{j_ℓ}, and the state variable X_{i_k} is stochastically independent of $X_{i_{k-1}}$. In other words, we have for any $\ell \neq k$ that

$$\mathrm{P}(x_{i_\ell}|x_{i_{\ell-1}}, x_{j_\ell}, \pi^{(k)}) = \mathrm{P}(x_{i_\ell}|x_{i_{\ell-1}}, \pi^{(k)}),$$

and

$$\mathrm{P}(x_{i_k}|x_{i_{k-1}}, x_{j_k}, \pi^{(k)}) = \mathrm{P}(x_{i_k}|x_{j_k}, \pi^{(k)}).$$

We can graphically represent this situation (of conditioning on $\mathbf{X}_{\mathrm{Pa}(d)} = \pi^{(k)}$) in the semantics of LIMIDs by removing the arc from i_{k-1} to i_k together with all the arcs from j_ℓ to i_ℓ for $\ell \neq k$ in the diagram of Figure 3.2(b), which results in the diagram in Figure 3.3. Using these context-specific independencies we get to

$$\mathrm{P}(x_{i_k}, \ldots, x_{i_m}|x_{j_k}, \ldots, x_{j_m}, \pi^{(k)}) = \mathrm{P}(x_{i_k}|x_{j_k}, \pi^{(k)}) \prod_{\ell>k} \mathrm{P}(x_{i_\ell}|x_{i_{\ell-1}}, \pi^{(k)}).$$

Hence, $\mathrm{P}(x_{i_k}, \ldots, x_{i_m}|x_{j_k}, \ldots, x_{j_m}, \pi^{(k)}) = \mathrm{P}(x_{i_k}, \ldots, x_{i_m}|x_{j_k}, \pi^{(k)})$, and

$$\begin{aligned}\mathrm{P}(X_{i_m}=x_{i_m}|\pi^{(k)}) &= \sum_{x_{j_k}} \sum_{x_{i_k},\ldots,x_{i_{m-1}}} \mathrm{P}(x_{i_k}, \ldots, x_{i_{m-1}}, x_{i_m}|x_{j_k}, \pi^{(k)})\mathrm{P}(x_{j_k}) \\ &= \sum_{x_{j_k}} \mathrm{P}(x_{j_k}) \sum_{x_{i_k},\ldots,x_{i_{m-1}}} \mathrm{P}(x_{i_k}|x_{j_k}, \pi^{(k)}) \prod_{\ell>k} \mathrm{P}(x_{i_\ell}|x_{i_{\ell-1}}, \pi^{(k)}).\end{aligned}$$

By design, each function $\mathrm{P}(x_{i_\ell}|x_{i_{\ell-1}}, \pi^{(k)})$ with $\ell \neq k$ equals the indicator function $I(x_{i_\ell}=x_{i_{\ell-1}})$, and $\mathrm{P}(x_{i_k}|x_{j_k}) = I(x_{i_k}=x_{j_k})$. We thus have that

$$\mathrm{P}(X_{i_m}=x_{i_m}|\pi^{(k)}) = \sum_{x_{j_k}} \mathrm{P}(x_{j_k}) \underbrace{\sum_{x_{i_k},\ldots,x_{i_{m-1}}} I(x_{i_k}=x_{j_k}) \prod_{\ell>k} I(x_{i_\ell}=x_{i_{\ell-1}})}_{=I(x_{j_k}=x_{i_m})} = \mathrm{P}(X_{j_k}=x_{i_m}).$$

The equation above says that $\mathbf{X}_{\mathrm{Pa}(d)}$ acts as a "selector" for the conditional probability distribution $\mathrm{P}(X_{i_m}=x_{i_m}|\pi^{(k)})$, which matches the probability distribution of the action variable X_{j_k}. Although we have assumed that $k > 1$, the same reasoning can be used to show that the result holds also for the case $k = 1$.

We will now show that for every strategy Δ' for $\mathscr{L}'$ we can efficiently obtain a strategy Δ for $\mathscr{L}$ that achieves the same expected utility and vice-versa. To this end, consider a strategy $\Delta' = \{\delta_{j_1}, \ldots, \delta_{j_m}, \ldots\}$ for $\mathscr{L}'$, and obtain a strategy $\Delta = \{\delta_d, \ldots\}$ for $\mathscr{L}$ such that δ_d is a policy for X_d satisfying $\delta_d(\pi^{(k)}) = \delta_{j_k}$ for all k. For each Δ' there is exactly one such Δ, and it follows that

$$\mathrm{P}_{\Delta'}(X_{i_m}=x_{i_m}|\pi^{(k)}) = I(x_{i_m}=\delta_{j_k}) = I(x_{i_m}=\delta_d(\pi^{(k)})) = \mathrm{P}_{\Delta}(X_d=x_{i_m}|\pi^{(k)}).$$

Conversely, let Δ be a strategy for $\mathscr{L}$ containing a policy δ_d for X_d, and obtain a strategy $\Delta' = (\Delta \setminus \delta_d) \cup \{\delta_{j_1}, \ldots, \delta_{j_m}\}$ such that $\delta_{j_k} = \delta_d(\pi^{(k)})$ for $k = 1, \ldots, m$. For each Δ there is exactly one such Δ', and it follows that

$$\mathrm{P}_{\Delta}(X_d = x_d | \pi^{(k)}) = I(x_d = \delta_d(\pi^{(k)})) = I(x_d = \delta_{j_k}) = \mathrm{P}_{\Delta'}(X_{i_m} = x_d | \pi^{(k)}) .$$

Consider a pair of strategies Δ and Δ' for $\mathscr{L}$ and $\mathscr{L}'$, respectively, that prescribe the same policies for all overlapping variables, and such that $\delta_{j_k} = \delta_d(\pi^{(k)})$ for all k. Let $N = S \cup A \cup V$ denote the nodes of $\mathscr{L}$, and $N' = S' \cup A' \cup V'$ denote the nodes of $\mathscr{L}'$. Thus, $S' = S \cup \{i_1, \ldots, i_m\}$, $A' = (A \setminus \{d\}) \cup \{j_1, \ldots, j_m\}$, $V' = V$, and we have that

$$\begin{aligned}
\mathrm{E}_{\Delta}(U) &= \sum_{\mathbf{x}_N} \mathrm{P}_{\Delta}(X_d = x_d | \mathbf{x}_{\mathrm{Pa}(d)}) \prod_{i \in N \setminus \{d\}} \mathrm{P}_{\Delta}(x_i | \mathbf{x}_{\mathrm{Pa}(i)}) \sum_{k \in V} x_k \\
&= \sum_{\mathbf{x}_{N \setminus \{d\}}} \sum_{x_d} \mathrm{P}_{\Delta'}(X_{i_m} = x_d | \mathbf{x}_{\mathrm{Pa}(d)}) \prod_{i \in N \setminus \{d\}} \mathrm{P}_{\Delta'}(x_i | \mathbf{x}_{\mathrm{Pa}(i)}) \sum_{k \in V'} x_k \\
&= \sum_{\mathbf{x}_{N \setminus \{d\}}} \sum_{x_{i_m}} \mathrm{P}_{\Delta'}(X_{i_m} = x_{i_m} | \mathbf{x}_{\mathrm{Pa}(d)}) \prod_{i \in N \setminus \{d\}} \mathrm{P}_{\Delta'}(x_i | \mathbf{x}_{\mathrm{Pa}(i)}) \sum_{k \in V'} x_k \\
&= \sum_{\mathbf{x}_{N \setminus \{d\}}} \sum_{\substack{x_{i_1}, \ldots, x_{i_m} \\ x_{j_1}, \ldots, x_{j_m}}} \mathrm{P}_{\Delta'}(x_{i_1}, x_{j_1}, \ldots, x_{i_m}, x_{j_m} | \mathbf{x}_{\mathrm{Pa}(d)}) \prod_{i \in N \setminus \{d\}} \mathrm{P}_{\Delta'}(x_i | \mathbf{x}_{\mathrm{Pa}(i)}) \sum_{k \in V'} x_k \\
&= \sum_{\mathbf{x}_{N'}} \prod_{k \in [m]} \mathrm{P}_{\Delta'}(x_{i_k} | x_{i_{k-1}}, x_{j_k}, \mathbf{x}_{\mathrm{Pa}(d)}) \mathrm{P}_{\Delta'}(x_{j_k}) \prod_{i \in N \setminus \{d\}} \mathrm{P}_{\Delta'}(x_i | \mathbf{x}_{\mathrm{Pa}(i)}) \sum_{k \in V'} x_k \\
&= \sum_{\mathbf{x}_{N \setminus \{d\}}} \sum_{x_{i_1}, \ldots, x_{i_{m-1}}} \prod_{i \in N'} \mathrm{P}_{\Delta'}(x_i | \mathbf{x}_{\mathrm{Pa}(i)}) \sum_{k \in V'} x_k \\
&= \mathrm{E}_{\Delta'}(U) .
\end{aligned}$$

Thus, for every strategy Δ' we can obtain in time polynomial in its size a strategy Δ such that $\mathrm{E}_{\Delta'}(U) = \mathrm{E}_{\Delta}(U)$, and vice-versa. □

Since the class of polynomials is closed under function composition, we can repeatedly apply Proposition 3.1 until no non-root decision node remains. Each of these transformations is polynomial-time computable, and we can obtain a strategy for the original diagram in polynomial time. Hence, we can show that the following result is true.

Theorem 3.1. *Given a multi-stage LIMID $\mathscr{L}$ of bounded treewidth, we can obtain in time polynomial in the size of $\mathscr{L}$ a single-stage LIMID $\mathscr{L}'$ of bounded treewidth, and such that for each strategy Δ' for $\mathscr{L}'$ there is a polynomial-time computable strategy Δ for $\mathscr{L}$ for which $\mathrm{E}_{\Delta}(U) = \mathrm{E}_{\Delta'}(U)$.*

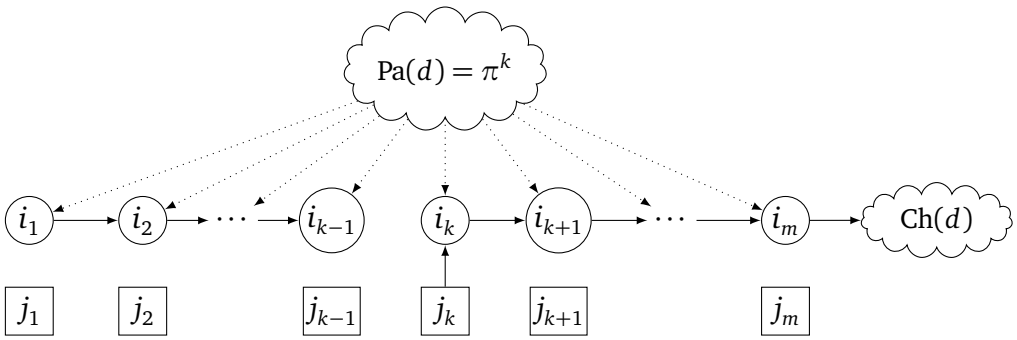


Figure 3.3. Illustration of the reasoning in the proof of Proposition 3.1.

Proof. It follows directly from Lemmas 3.1, 3.2, and 3.1. □

Thus, single-stage diagrams of bounded treewidth are not easier to solve than multi-stage diagrams of bounded treewidth, since if this was the case one could use the above theorem to reduce the latter to the former. The converse is also true, since we can transform a single-stage diagram into a multi-stage diagram such that the expected utility of strategies is unchanged by adding a root state variable with uniform probability as a parent of (some or all) action nodes. Consequently, the class of single-stage LIMIDs of bounded treewidth is theoretically as hard to solve as the class of multi-stage LIMIDs. Since these transformations preserve the value of the solutions (i.e., the expected utility of strategies), the equivalence extends to approximate solutions as well, that is, approximately solving single-stage LIMIDs of bounded treewidth within provably good bounds is as hard as approximately solving multi-stage LIMIDs.

3.4 The complexity of solving LIMIDs

In this section we analyze the fixed-parameter complexity of solving LIMIDs with respect to the topology of the underlying DAG (i.e., whether it is singly or multiply connected), the cardinality of the variable domains, and the number of values nodes. We show that solving LIMIDs of bounded treewidth is NP-hard, even if we bound the number of values each variable can assume, and admit only singly connected diagrams. A remarkable exception is the case of singly connected diagrams with binary state variables and a single value node, which we show can be solved in polynomial-time. We start by the negative results.

Theorem 3.2. *Given a singly connected LIMID of treewidth two over binary state and action variables, deciding whether there is a strategy with expected utility greater than a given rational number k is NP-complete.*

Proof. Since the diagram has bounded treewidth, for any fixed strategy Δ we can compute its expected utility in polynomial time. Thus, $\mathrm{E}_\Delta(U) > k$ can be decided in polynomial time, and the problem is in NP. Hardness is shown by a reduction from the *partition problem*, which can be stated as follows.

> Given positive even integer numbers $z_1, \ldots, z_n$, is it possible to partition them into two sets of equal sum?

This problem is well-known to be NP-complete [Garey and Johnson, 1979].[6] As usual, we assume that the instances of the partition problem are "reasonably" and "concisely" encoded as bit-strings of length $b = 2(\sum_{i=1}^{n} \lceil \log_2 z_i \rceil + n - 1)$.[7] Any partition of the numbers into two sets can be represented as an n-dimensional binary vector $(\delta_1, \ldots, \delta_n) \in \{0,1\}^n$. Let $z \stackrel{\text{def}}{=} \frac{1}{2}\sum_{i=1}^{n} z_i$. The partition problem is thus equivalent to deciding whether there is a binary vector $(\delta_1, \ldots, \delta_n)$ such that $\sum_{i=1}^{n} z_i \delta_i = \sum_{i=1}^{n} z_i(1-\delta_i) = z$. A binary vector $(\delta_1, \ldots, \delta_n)$ satisfying that equality is said to be a *yes-solution* of the problem, otherwise it is called a *no-solution*. In either case, the vector is called a solution and the quantity $\sum_{i=1}^{n} z_i \delta_i$ is called its *value*. Since the input numbers are even, also the value of a solution is an even number. Moreover, since only yes-solutions have value z, and z is an integer number, the value of any no-solution $(\delta_1, \ldots, \delta_n)$ satisfies $|z - \sum_{i=1}^{n} z_i \delta_i| \geq 1$. We will exploit this integer gap between yes- and no-solutions later on in the reduction.

Our reduction maps any instance of the partition problem into a LIMID whose graph structure is shown in Figure 3.4. The action variables $X_{d_1}, \ldots, X_{d_n}$ are binary variables and represent the assignments of numbers of the partition problem into one of the two partitions. Thus, a strategy $\Delta = \{\delta_1, \ldots, \delta_n\}$ represents a partition of the input numbers into two sets. The value variables $X_{v_1}, \ldots, X_{v_n}$ are set so that $X_{v_i} \stackrel{\text{def}}{=} -z_i X_{d_i}/z$. Hence, $\sum_i \mathrm{E}_\Delta(X_{v_i}) = -\sum_{i=1}^{n} z_i \delta_i / z$, which equals minus one if and only if the strategy Δ is a yes-solution to the partition problem. The probabilities of the state variables X_{s_i}, $i = 1, \ldots, n$, are

[6] The standard definition of the problem does not require numbers to be even. This however does not alter the complexity of the problem, as an instance with even numbers admits a yes-solution if and only if the instance obtained by halving each number admits a yes-solution.

[7] The usual encoding of an instance of partition problem is a binary string $s_1 01 s_2 01 \cdots 01 s_n$, where each substring s_i is the binary representation of the number z_i with every digit duplicated. For example, the encoding of the problem $z_1 = 2$ and $z_2 = 3$ is 1100011111.

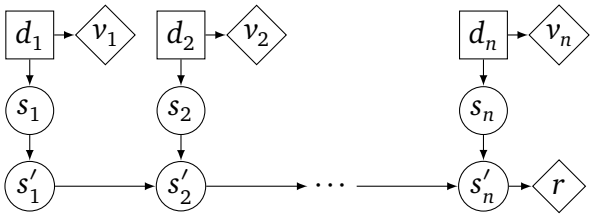


Figure 3.4. LIMID used to solve the partition problem in the proof of Theorem 3.2.

specified as rational numbers such that $\mathrm{P}(X_{s_i}=1|X_{d_i}=0) \stackrel{\text{def}}{=} 1$ and

$$\exp(-z_i/z) < \mathrm{P}(X_{s_i}=1|X_{d_i}=1) < \exp(-z_i/z) + 2^{-2-10b},$$

where b is the number of bits encoding the instance of the partition problem.[8] The state variables $X_{s'_1},\ldots,X_{s'_n}$ are deterministic variables that satisfy $X_{s'_1} \stackrel{\text{def}}{=} X_{s_1}$ and $X_{s'_i} \stackrel{\text{def}}{=} X_{s'_{i-1}} \wedge X_{s_i} = X_{s'_{i-1}} X_{s_i}$, $i=2,\ldots,n$ (where $\wedge$ denotes logical conjunction). A simple inductive argument on $i=1,\ldots,n$ shows that $X_{s'_n} = X_{s_1}X_{s_2}\cdots X_{s_n}$, so that for any strategy Δ we have that

$$\begin{aligned}\mathrm{E}_\Delta(X_{s'_n}) &= \sum_{x_{s_1},\ldots,x_{s_n}} \mathrm{P}_\Delta(X_{s'_n}=1|x_{s_1},\ldots,x_{s_n})\mathrm{P}_\Delta(x_{s_1},\ldots,x_{s_n}) \\ &= \sum_{x_{s_1},\ldots,x_{s_n}} x_{s_1}x_{s_2}\cdots x_{s_n}\prod_{i=1}^{n}\mathrm{P}(x_{s_i}|X_{d_i}=\delta_i) = \prod_{i=1}^{n}\mathrm{P}(X_{s_i}=1|X_{d_i}=1)^{\delta_i}. \quad (3.3)\end{aligned}$$

Finally, the diagram is fully specified by defining $X_r \stackrel{\text{def}}{=} 2 + 2^{-9b} - qX_{s'_n}$, where q is a rational number such that $e < q < e + 2^{-2-9b}$. It follows that

$$\mathrm{E}_\Delta(U) = \mathrm{E}_\Delta(X_r) + \sum_{i=1}^{n}\mathrm{E}_\Delta(X_{v_i}) = 2 + 2^{-9b} - q\mathrm{E}_\Delta(X_{s'_n}) - \frac{1}{z}\sum_{i=1}^{n} z_i\delta_i. \quad (3.4)$$

Consider the continuous function over $s \in [0,1]$

$$f(s) = 2 + 2^{-9b} - \exp(1-s) - s. \quad (3.5)$$

It follows from (3.4) that

$$f\left(\sum_{i=1}^{n} z_i\delta_i/z\right) - \mathrm{E}_\Delta(U) = q\mathrm{E}_\Delta(X_{s'_n}) - \exp\left(1 - \sum_{i=1}^{n} z_i\delta_i/z\right).$$

[8] We can obtain each number $\mathrm{P}(X_{s_i}=1|X_{d_i}=1)$ in time polynomial in b by the truncated Taylor expansion of $\exp(-z_i/z)$.

By (3.3) and the definition of $\mathrm{P}(X_{s_i}=1|X_{d_i}=1)$, we have that

$$\mathrm{E}_\Delta(X_{s'_n}) = \prod_{i=1}^{n}[\exp(-z_i/z)+\epsilon_i]^{\delta_i} > \prod_{i=1}^{n}[\exp(-z_i/z)]^{\delta_i} = \exp\left(-\sum_{i=1}^{n} z_i\delta_i/z\right),$$

where $0<\epsilon_i<2^{-2-10b}$. It follows that

$$q\mathrm{E}_\Delta(X_{s'_n}) > e\mathrm{E}_\Delta(X_{s'_n}) > \exp\left(1-\sum_{i=1}^{n} z_i\delta_i/z\right),$$

whence $q\mathrm{E}_\Delta(X_{s'_n}) - \exp(1-\sum_{i=1}^{n} z_i\delta_i/z) > 0$.

According to the Multivariate Binomial Theorem, we have that

$$\begin{aligned}\mathrm{E}_\Delta(X_{s'_n}) &= \prod_{i=1}^{n}[\exp(-z_i/z)+\epsilon_i]^{\delta_i} = \sum_{k\in\mathscr{C}}\prod_{i=1}^{n}\exp(-z_ik_i/z)\epsilon_i^{\delta_i-k_i}\\ &= \exp\left(-\sum_{i=1}^{n} z_i\delta_i/z\right) + \sum_{k\in\mathscr{C},k\neq\delta}\prod_{i=1}^{n}\exp(-z_ik_i/z)\epsilon_i^{\delta_i-k_i},\end{aligned}$$

where $\mathscr{C} = \{(k_1,\ldots,k_n)\in\{0,1\}^n : k_i\leq\delta_i, i=1,\ldots,n\}$. Each term inside the sum on the right-hand side of the equation above contains at least one factor equal to ϵ_i for some $i=1\ldots,n$. Since the sum contains at most 2^n terms, $n\leq b$, $0<\exp(-z_ik_i/z)\leq 1$, and $\epsilon_i<2^{-2-10b}<1$, it follows that

$$\mathrm{E}_\Delta(S'_n) - \exp\left(-\sum_{i=1}^{n} z_i\delta_i/z\right) < \sum_{k\in\{0,1\}^n}\max_i \epsilon_i \leq 2^{-2-9b}.$$

Consequently, we have that

$$\begin{aligned}q\mathrm{E}_\Delta(S'_n) - \exp\left(1-\sum_{i=1}^{n} z_i\delta_i/z\right) =\\ \underbrace{q\mathrm{E}_\Delta(S'_n) - q\exp\left(-\sum_{i=1}^{n} z_i\delta_i/z\right)}_{<q2^{-2-9b}} + \overbrace{[q-e]}^{\leq 2^{-2-9b}}\underbrace{\exp\left(-\sum_{i=1}^{n} z_i\delta_i/z\right)}_{\leq 1},\end{aligned}$$

which, since $q<3$, is strictly smaller than 2^{-9b}. By combining these inequalities we find for any strategy Δ that

$$0 < f\left(\sum_{i=1}^{n} z_i\delta_i/z\right) - \mathrm{E}_\Delta(U) < 2^{-9b}. \tag{3.6}$$

Thus, for any strategy Δ it follows that $\mathrm{E}_\Delta(U) > f(\sum_{i=1}^{n} z_i\delta_i/z) - 2^{-9b}$. Recall that the partition problem admits a yes-solution if and only if $\sum_i z_i\delta_i/z = 1$. Hence, if a yes-solution exists then $\max_\delta \mathrm{E}_\Delta(U) > f(1) - 2^{-9b} = 0$.

To show that any no-solution has non-positive expected utility, consider the function f in Equation (3.5). Its first and second derivatives are, respectively, $f'(s) = \exp(1-s) - 1$ and $f''(s) = -\exp(1-s)$. Thus, the function is strictly concave, increases for $s < 1$, decreases for $s > 1$, and has a maximum at $s = 1$.

If the partition problem does not admit a yes-solution then any strategy Δ satisfies $|z - \sum_i z_i\delta_i| \geq 1$, from which it follows that either $\sum_i z_i\delta_i/z \geq 1 + 1/z$ or $\sum_i z_i\delta_i/z \leq 1 - 1/z$. We have from Inequality 3.6, the analysis of f, and the reasoning above, that if a yes-solution does not exist then $\max_\delta \mathrm{E}_\Delta(U) < \max\{f(1+1/z), f(1-1/z)\}$. Consider the difference $f(1+1/z) - f(1-1/z) = -\exp(-1/z) + \exp(1/z) - 2/z$. By analyzing its first and second derivatives, one can show that the difference is a strictly convex function of z whose infimum is zero. Hence, the difference is positive and it suffices for the result to show that $f(1+1/z)$ is negative for any positive integer z. To this end, consider the second-order Taylor series approximation of $\exp(-s)$ around zero given by $T_2(s) \stackrel{\text{def}}{=} 1 - s + s^2/2$, whose Lagrange-form residual is

$$R_2(s) \stackrel{\text{def}}{=} \exp(-s) - T_2(s) = -s^3\exp(-\xi)/6 > -s^3/6\,,$$

where ξ is a number between 0 and s. Hence, $-\exp(-s) < s^3/6 - T_2(s)$, from which it follows that

$$\begin{aligned} f(1+1/z) &= 2 + 2^{-9b} - \exp(-1/z) - 1 - \frac{1}{z} \\ &< 2 + 2^{-9b} + \left[\frac{1}{6z^3} - T_2(1/z)\right] - 1 - \frac{1}{z} \\ &= 2^{-9b} - \frac{1}{2z^2} + \frac{1}{6z^3} \leq 2^{-9b} - \frac{1}{3z^3}\,, \end{aligned}$$

which, as $2z \leq 2^b$ in the usual encoding of the partition problem, is negative for any positive z.

Thus, for any instance of the partition problem we can build in polynomial time a singly connected influence diagram of bounded treewidth with only binary variables and such that the partition problem has a yes-solution if and only if the optimum strategy has non-negative expected utility. Hence, deciding if the maximum expected utility exceeds a rational solves the partition problem. □

The result above required an unbounded number of value nodes. The next result shows that if we relax the assumption of binary variables and allow vari-

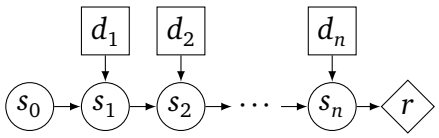


Figure 3.5. LIMID used to solve the partition problem in the Proof of Theorem 3.3.

ables to take on at most three values, we obtain the same hardness result even in the case of a single value node.

Theorem 3.3. *Given a singly connected LIMID of treewidth two over binary action variables, ternary state variables and a single value variable, deciding whether there is a strategy with expected utility greater than a given rational k is NP-complete.*

Proof. Membership in NP follows from the same argument used in the proof of Theorem 3.2. Hardness is once again showed using a reduction from the partition problem with $n > 3$. As before, define $z \stackrel{\text{def}}{=} \frac{1}{2}\sum_{i=1}^{n} z_i$.

Consider the following LIMID with topology as in Figure 3.5. There are n binary action variables $X_{d_1},\ldots,X_{d_n}$ associated to root nodes, and a chain of $n+1$ ternary state variables $X_{s_0},X_{s_1},\ldots,X_{s_n}$ taking on states 1, 2 and 3. Also, there is a single value variable X_r with X_{s_n} as single parent. We specify the conditional expected utility of X_r by $\mathrm{E}(X_r|X_{s_n}) = I(X_r=3)$. The probability distribution of X_{s_0} is uniformly specified, that is, $P(x_{s_0}) = 1/3$. For $i = 1,\ldots,n$, let t_i be a rational such that

$$2^{-z_i/z} \leq t_i < 2^{-z_i/z} + 2^{-(6b+3)},$$

where b is the number of bits in the encoding of the partition problem (we can compute t_i by the truncated Taylor expansion of $2^{-z_i/z}$ in time polynomial in b). The conditional probabilities of state variable X_{s_i} given X_{d_i} and $X_{s_{i-1}}$ are specified as

$$\mathrm{P}(X_{s_i}=1|x_{d_i},x_{s_{i-1}}) = I(x_{s_i}=x_{s_{i-1}})t_i^{x_{d_i}},$$
$$\mathrm{P}(X_{s_i}=2|x_{d_i},x_{s_{i-1}}) = I(x_{s_i}=x_{s_{i-1}})t_i^{1-x_{d_i}},$$

and $\mathrm{P}(X_{s_i}=3|x_{d_i},x_{s_{i-1}}) = 1-\mathrm{P}(X_{s_i}=1|x_{d_i},x_{s_{i-1}})-\mathrm{P}(X_{s_i}=2|x_{d_i},x_{s_{i-1}})$. Note that $\mathrm{P}(x_{s_i}|x_{d_i},X_{s_{i-1}}=3) = I(x_{s_i}=3)$.

Given a strategy $\Delta = \{\delta_{d_1},\ldots,\delta_{d_n}\}$, let $I \stackrel{\text{def}}{=} \{i \in [n] : \delta_{d_i} = 1\}$ and $I^c \stackrel{\text{def}}{=}$

$[n] \setminus I$. We have that

$$\begin{aligned}
\mathrm{E}_{\Delta}(X_r) &= \sum_{x_{s_n}} \mathrm{E}(X_r|x_{s_n}) \sum_{x_{S\setminus\{s_n\}}} \mathrm{P}(x_{s_0}) \prod_{i=1}^{n} \mathrm{P}(x_{s_i}|X_{d_i}=\delta_{d_i}, x_{s_{i-1}}) \\
&= \sum_{x_{S\setminus\{s_n\}}} \mathrm{P}(X_{s_n}=3|X_{d_n}=\delta_{d_n}, x_{s_{n-1}}) \mathrm{P}(x_{s_0}) \prod_{i=1}^{n-1} \mathrm{P}(x_{s_i}|X_{d_i}=\delta_{d_i}, x_{s_{i-1}}) \\
&= \sum_{x_{S\setminus\{s_n\}}} \left[1 - \mathrm{P}(X_{s_n}=1|X_{d_n}=\delta_{d_n}, x_{s_{n-1}}) - \mathrm{P}(X_{s_n}=2|X_{d_n}=\delta_{d_n}, x_{s_{n-1}})\right] \cdot \\
&\qquad \cdot \mathrm{P}(x_{s_0}) \prod_{i=1}^{n-1} \mathrm{P}(x_{s_i}|X_{d_i}=\delta_{d_i}, x_{s_{i-1}}) \\
&= 1 - \frac{1}{3}\left(\prod_{i\in I} t_i + \prod_{i\in I^c} t_i\right).
\end{aligned}$$

The last equality follows from the definition of the functions $\mathrm{P}(x_{s_i}|X_{d_i}=\delta_{d_i}, x_{s_{i-1}})$, which are nonzero only if $x_{s_i} = x_{s_{i-1}}$, and therefore the sum over $x_{s\setminus\{s_n\}}$ is non zero only when $x_{s_1} = \cdots = x_{s_n}$.

According to Lemma 23 in [Mauá et al., 2012] the numbers t_i, $i = 1, \ldots, n$, satisfy $2^{-z_i/z} \leq t_i < 2^{-z_i/z+2^{-6b}}$. Thus,[9]

$$\prod_{i\in S} t_i < 2^{2^{-6b}n - \sum_{i\in S} z_i/z} \leq 2^{2^{-5b} - \sum_{i\in S} z_i/z},$$

for any $S \subseteq [n]$. In particular, if Δ is a yes-solution to the partition problem then $\sum_{i\in I} z_i/z = \sum_{i\in I^c} z_i/z = 1$, and

$$\mathrm{E}_{\Delta}(X_r) > 1 - \frac{1}{3}\left(2^{-1+2^{-5b}} + 2^{-1+2^{-5b}}\right) = 1 - \frac{2^{2^{-5b}}}{3}. \tag{3.7}$$

Let q be equal to $2^{2^{-5b}}$ encoded with $5b+3$ bits of precision (and rounded up), that is, $2^{2^{-5b}} \leq q < 2^{2^{-5b}} + 2^{-(5b+3)}$, which implies (by Lemma 24 in [Mauá et al., 2012]) that

$$2^{2^{-5b}} \leq q < 2^{2^{-5b}+2^{-5b}} = 2^{2^{1-5b}} < 2^{2^{-4b}}. \tag{3.8}$$

The reduction is carried out by verifying whether $\max_{\Delta} \mathrm{E}_{\Delta}(X_r) > 1 - q/3$. We already know that if a yes-solution exists then there is a strategy which obtains

[9] Since the number of bits used to encode the partition problem must be greater than or equal to n, we have that $n/2^b \leq n/b \leq 1$, and hence $2^{-(j+1)b}n < 2^{-jb}$, for any $j > 0$.

an expected utility greater than $1 - q/3$, because of (3.7) and the fact that q is rounded up. Let us then consider the case where a yes-solution does not exist. We want to show that in this case $\max_\Delta \mathrm{E}_\Delta(X_r) \leq 1 - 2^{2^{-4b}}/3$, which by (3.8) implies $\max_\Delta \mathrm{E}_\Delta(X_r) < 1 - q/3$. Since there is no yes-solution, any strategy induces a partition (I, I^c) such that, for some integer $-z \leq c \leq z$ different from zero, we have that $\sum_{i\in I} z_i = z - c$ and $\sum_{i\in I^c} z_i = z + c$, because the original numbers z_i are positive integers that add up to $2z$. It follows that

$$\prod_{i\in I} t_i + \prod_{i\in I^c} t_i \geq 2^{c/z-1} + 2^{-c/z-1}\,.$$

The right-hand side of the equality above is a function on $c \in \{-z, \ldots, z\} \setminus \{0\}$, which is symmetric with respect to the y-axis and monotonically increasing for $c > 0$. Therefore, it obtains its minimum at $|c| = 1$. Hence,

$$\prod_{i\in I} t_i + \prod_{i\in I^c} t_i \geq 2^{1/z-1} + 2^{-1/z-1}\,.$$

Since $n > 3$ implies $z \geq 2$, it follows from Lemma 24 in Mauá et al. [2012] that

$$2^{1/z-1} + 2^{-1/z-1} \geq 2^{1/z^4}\,.$$

Each number z_i is encoded with at least $\log_2 z_i$ bits, and therefore $b \geq \log_2(z_1) + \cdots + \log_2(z_n) = \log_2(z_1 \cdots z_n)$. The latter is greater than or equal to $\log_2(z_1 + \cdots + z_n)$, and hence is also greater than $\log_2 z$. Thus, we have that $z \leq 2^b$, which implies $z^4 \leq 2^{4b}$ and therefore $1/z^4 \geq 2^{-4b}$ and $2^{1/z^4} \geq 2^{2^{-4b}}$. Hence,

$$2^{1/z-1} + 2^{-1/z-1} \geq 2^{2^{-4b}}\,.$$

Thus, if a yes-solution does not exist we have that

$$\max_\Delta \mathrm{E}_\Delta(X_r) = 1 - \frac{1}{3}\left(\prod_{i\in I} t_i + \prod_{i\in I^c} t_i\right) \leq 1 - \frac{2^{2^{-4b}}}{3} < 1 - q/3\,.$$

Therefore, the partition problem can be decided by verifying whether

$$\max_\Delta \mathrm{E}_\Delta(X_r) > 1 - q/3\,,$$

which concludes the proof. □

According to the previous result, trading the number of values nodes with the cardinality of variables does not affect the complexity of the problem. The next result shows that by allowing diagrams to be multiply connected in exchange of admitting a single value nodes also does not affect the hardness result.

Theorem 3.4. *For any LIMID $\mathscr{L}$, there is a LIMID $\mathscr{L}'$ containing a single value variable such that any strategy Δ for $\mathscr{L}$ is also a strategy for $\mathscr{L}'$ and obtains the same expected utility. Moreover, the treewidth of $\mathscr{L}'$ is at most the treewidth of $\mathscr{L}$ plus three, and the maximum variable domain cardinality is unchanged.*

Proof. Consider, without loss of generality, a binary tree decomposition T of $\mathscr{L}$ such that for every value node v_i in V there is a leaf node in the tree whose associated node set is $\mathrm{Pa}(v_i)$.[10] Assume additionally that the tree is rooted at a node r and that the leaf nodes $\ell_1, \ell_2, \ldots, \ell_q$ associated to the sets $\mathrm{Pa}(v_1)$,...,$\mathrm{Pa}(v_q)$, respectively, where $q = |V|$ denotes the number of value variables in $\mathscr{L}$, are ordered in such a way that they agree with an *in-order tree traversal* of the tree decomposition, that is, in a depth-first tree traversal of T rooted at r, the node ℓ_1 precedes ℓ_2, which precedes ℓ_3, and so on. Let $e_1 > e_0$ be upper and lower bounds, respectively, on all the value variables, that is $e_0 \leq X_{v_i} \leq e_1$ for all v_i.

Now consider a diagram $\mathscr{L}'$ width DAG G', which contains a single value variable X_v instead of the q value variables of $\mathscr{L}$ and an augmented set of state variables $\mathbf{X}_{S'} = \mathbf{X}_S \cup \{X_{w_1}, \ldots, X_{w_q}, X_{o_1}, \ldots, X_{o_q}\}$, where $X_{w_1}, \ldots, X_{w_q}, X_{o_1}, \ldots, X_{o_q}$ are binary variables. Furthermore, the subgraph of G' obtained by considering only nodes in S and A is identical to the DAG G of $\mathscr{L}$ with the state nodes $w_1, \ldots, w_q$ replacing the value nodes $v_1, \ldots, v_q$, respectively. Also, the nodes $o_1, \ldots, o_q$ are arranged in a chain such that w_1 is the parent of o_1, w_2 and o_1 are the parents of o_2 and so forth, as in Figure 3.6(b). The probability values associated with each variable X_{w_i}, $i = 1, \ldots, q$, is given by

$$\mathrm{P}(X_{w_i} = 1 | \mathbf{x}_{\mathrm{Pa}(v_i)}) = \frac{\mathrm{E}(X_{v_i} | \mathbf{x}_{\mathrm{Pa}(v_i)}) - e_0}{e_1 - e_0}.$$

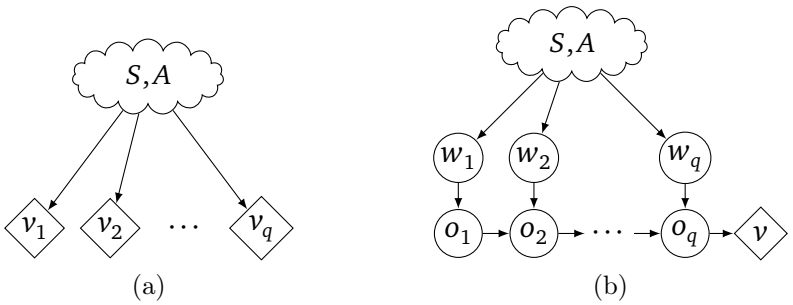
The probability distribution $\mathrm{P}(X_{o_i} | X_{o_{i-1}}, X_{w_i})$ of each variable X_{o_i}, $i = 1, \ldots, q$,[11] is such that

$$\begin{aligned} &\mathrm{P}(X_{o_i} = 1 | X_{o_{i-1}} = 1, X_{w_i} = 1) = 1, \quad &&\mathrm{P}(X_{o_i} = 1 | X_{o_{i-1}} = 1, X_{w_i} = 0) = (i-1)/i, \\ &\mathrm{P}(X_{o_i} = 1 | X_{o_{i-1}} = 0, X_{w_i} = 1) = 1/i, \quad &&\mathrm{P}(X_{o_i} = 1 | X_{o_{i-1}} = 0, X_{w_i} = 0) = 0, \end{aligned}$$

and $\mathrm{P}(X_{o_1} = 1 | X_{w_1} = 1) = 1$ and $\mathrm{P}(X_{o_1} = 1 | X_{w_1} = 0) = 0$. Finally, the single value node v has O_q as its sole parent; its conditional expected utility function $\mathrm{E}(X_v | X_{o_q})$ is defined as $\mathrm{E}(X_v | X_{o_q} = 1) = qe_1$ and $\mathrm{E}(X_v | X_{o_q} = 0) = qe_0$.

[10] Any binary tree decomposition can be transformed to meet this requirement by repeatedly selecting a node i associated to a superset of the parents of a value node v_i not meeting the requirement, and then adding two nodes j and k such that the children of i become children of j, and k is a child of i; the node j is associated to the set of variables associated to i, while k is associated to $\mathrm{Pa}(v_i)$. Note that the treewidth is unaltered by these operations.

[11] We assume $o_0 = \emptyset$ for notational convenience



(a) (b)

Figure 3.6. (a) Influence diagram with multiple value nodes. (b) Its equivalent influence diagram containing a single value node as used in the proof of Theorem 3.4.

To show that the treewidth of $\mathscr{L}'$ does not exceed the treewidth of $\mathscr{L}$ in more than three, we build a tree decomposition T' for $\mathscr{L}'$ based on the tree decomposition T as follows. The nodes of T' are the same nodes of T. For each node ℓ in T, let C_ℓ and C'_ℓ denote, respectively, the associated node set in T and T'. We start with $C'_\ell = C_\ell$ for all ℓ, and iteratively insert nodes in node sets of T'. We take each node ℓ_i of T' and include w_i, o_i and o_{i-1} in C'_i (o_{i-1} is included for $i > 1$), which is enough to cover their families and to satisfy the family preserving property. Note that at this stage T' does not satisfy the running intersection property, because o_{i-1} appears in both ℓ_{i-1} and ℓ_i but not necessarily in every (set associated to a) node in between. Then, we walk around the tree $\mathscr{T}'$ in a Euler tour tree traversal where each edge is visited exactly twice, and we include o_{i-1} in each node of T' that appears between ℓ_{i-1} and ℓ_i during the walk.[12] By doing so we guarantee that the running intersection property is also satisfied in T'. Since the Euler tour tree traversal visits each leaf once and each internal node at most three times, the procedure inserts three new variables in the sets associated to $\ell_1, \ldots, \ell_q$ and at most three variables o_i in the sets associated to non-leaf nodes, and therefore does not increase the treewidth of the decomposition by more than three.

It remains to show that the diagrams are equivalent with respect to expected

[12] An Euler tour tree traversal of T rooted at r is a list of $2m-1$ symbols produced by calling $ET(r)$, where $ET(i)$ is a recursive function that takes a node i with left children j and right children k (if either exists) and prints out i, calls $ET(j)$ (if j exists), prints out i again, calls $ET(k)$ (if k exists), and then prints out i once more. It is equivalent to the list produced by visiting the nodes in in-, pre- and post-orders at the same time, or more intuitively, by a walk around the tree starting at the root node.

utility of strategies. Let Δ be a strategy for $\mathscr{L}$. Then Δ is also a strategy for $\mathscr{L}'$ (because the two diagrams share the same action variables and graph over $S \cup A$). First, we need to show that for $i = 1, \ldots, q$ it follows that

$$\mathrm{P}(X_{o_i} = 1 | \mathbf{x}_{S \cup A}) = \frac{1}{i} \sum_{j=1}^{i} \mathrm{P}(X_{w_j} = 1 | \mathbf{x}_{S \cup A}),$$

which we do by induction on i. The basis ($i = 1$) follows trivially, because $\mathrm{P}(X_{o_1} = 1 | X_{w_1} = 1) = 1$ by definition, so according to the graph structure we have that $\mathrm{P}(X_{o_1} = 1 | \mathbf{x}_{S \cup A}) = \mathrm{P}(X_{w_1} = 1 | \mathbf{x}_{S \cup A})$. Now assume by hypothesis of the induction that the above result is valid for every $1 \leq i \leq k < q$. Then

$$\begin{aligned}
\mathrm{P}(X_{o_{k+1}} = 1 | \mathbf{x}_{S \cup A}) = & \underbrace{\mathrm{P}(X_{o_{k+1}} = 1 | X_{o_k} = 1, X_{w_{k+1}} = 1)}_{=1} \mathrm{P}(X_{o_k} = 1 | \mathbf{x}_{S \cup A}) \mathrm{P}(X_{w_{k+1}} = 1 | \mathbf{x}_{S \cup A}) \\
& + \underbrace{\mathrm{P}(X_{o_{k+1}} = 1 | X_{o_k} = 1, X_{w_{k+1}} = 0)}_{k/(k+1)} \mathrm{P}(X_{o_k} = 1 | \mathbf{x}_{S \cup A}) \mathrm{P}(X_{w_{k+1}} = 0 | \mathbf{x}_{S \cup A}) \\
& + \underbrace{\mathrm{P}(X_{o_{k+1}} = 1 | X_{o_k} = 0, X_{w_{k+1}} = 1)}_{=1/(k+1)} \mathrm{P}(X_{o_k} = 0 | \mathbf{x}_{S \cup A}) \mathrm{P}(X_{w_{k+1}} = 1 | \mathbf{x}_{S \cup A}) \\
& + \underbrace{\mathrm{P}(X_{o_{k+1}} = 1 | X_{o_k} = 0, X_{w_{k+1}} = 0)}_{=0} \mathrm{P}(X_{o_k} = 0 | \mathbf{x}_{S \cup A}) \mathrm{P}(X_{w_{k+1}} = 0 | \mathbf{x}_{S \cup A}),
\end{aligned}$$

which using $1 = 1/(k+1) + k/(k+1)$ is

$$\begin{aligned}
= & \left(\frac{1}{k+1} + \frac{k}{k+1} \right) \mathrm{P}(X_{o_k} = 1 | \mathbf{x}_{S \cup A}) \mathrm{P}(X_{w_{k+1}} = 1 | \mathbf{x}_{S \cup A}) \\
& + \frac{k}{k+1} \mathrm{P}(X_{o_k} = 1 | \mathbf{x}_{S \cup A}) \mathrm{P}(X_{w_{k+1}} = 0 | \mathbf{x}_{S \cup A}) \\
& + \frac{1}{k+1} \mathrm{P}(X_{o_k} = 0 | \mathbf{x}_{S \cup A}) \mathrm{P}(X_{w_{k+1}} = 1 | \mathbf{x}_{S \cup A}) \\
= & \frac{k}{k+1} \mathrm{P}(X_{o_k} = 1 | \mathbf{x}_{S \cup A}) + \frac{1}{k+1} \mathrm{P}(X_{w_{k+1}} = 1 | \mathbf{x}_{S \cup A}) \\
= & \frac{k}{(k+1)} \frac{1}{k} \sum_{j=1}^{k} \mathrm{P}(X_{w_j} = 1 | \mathbf{x}_{S \cup A}) + \frac{1}{k+1} \mathrm{P}(X_{w_{k+1}} = 1 | \mathbf{x}_{S \cup A}) \\
= & \frac{1}{k+1} \sum_{j=1}^{k+1} \mathrm{P}(X_{w_j} = 1 | \mathbf{x}_{S \cup A}),
\end{aligned}$$

which shows that the result holds also for $i = k + 1$.

We can now show that for any strategy Δ its expected utility $\mathrm{E}_\Delta(X_v)$ w.r.t. $\mathscr{L}'$ equals the associated expected utility $\mathrm{E}_\Delta(U)$ w.r.t. $\mathscr{L}$. Let $S'' = S' \setminus \{o_q\}$. By definition,

$$
\begin{aligned}
\mathrm{E}'_\Delta(X_v) &= \sum_{\mathbf{x}_{S'\cup A}} \mathrm{P}_\Delta(\mathbf{x}_{S'\cup A})\mathrm{E}(X_v|\mathbf{x}_{\mathrm{Pa}(v)}) \\
&= \sum_{\mathbf{x}_{S'\cup A}} \mathrm{P}_\Delta(\mathbf{x}_{S'\cup A})\mathrm{E}(X_v|x_{o_q}) \\
&= \sum_{\mathbf{x}_{S''\cup A}} \mathrm{P}_\Delta(\mathbf{x}_{S''\cup A}) \left(qe_1\mathrm{P}(X_{o_q}=1|\mathbf{x}_{S''\cup A}) + qe_0\mathrm{P}(X_{o_q}=0|\mathbf{x}_{S''\cup A}) \right) \\
&= qe_0 + \sum_{\mathbf{x}_{S''\cup A}} \mathrm{P}_\Delta(\mathbf{x}_{S''\cup A})q(e_1-e_0)\mathrm{P}(X_{o_q}=1|\mathbf{x}_{S''\cup A}) \\
&= qe_0 + q(e_1-e_0)\sum_{\mathbf{x}_{S\cup A}} \mathrm{P}_\Delta(\mathbf{x}_{S\cup A})\mathrm{P}(X_{o_q}=1|\mathbf{x}_{S\cup A}) \\
&= qe_0 + q(e_1-e_0)\sum_{\mathbf{x}_{S\cup A}} \mathrm{P}_\Delta(\mathbf{x}_{S\cup A})\frac{1}{q}\sum_{i=1}^{q}\mathrm{P}(X_{w_i}=1|\mathbf{x}_{S\cup A}) \\
&= qe_0 + q\frac{e_1-e_0}{q}\sum_{\mathbf{x}_{S\cup A}} \mathrm{P}_\Delta(\mathbf{x}_{S\cup A})\sum_{i=1}^{q}\mathrm{P}(X_{w_i}=1|\mathbf{x}_{\mathrm{Pa}(v_i)}) \\
&= qe_0 + (e_1-e_0)\sum_{\mathbf{x}_{S\cup A}} \mathrm{P}_\Delta(\mathbf{x}_{S\cup A})\sum_{j=1}^{q}\frac{\mathrm{E}(X_{v_i}|\mathbf{x}_{\mathrm{Pa}(v_i)})-e_0}{e_1-e_0} \\
&= qe_0 + \frac{e_1-e_0}{e_1-e_0}\sum_{\mathbf{x}_{S\cup A}} \mathrm{P}_\Delta(\mathbf{x}_{S\cup A})\sum_{i=1}^{q}\left(\mathrm{E}(X_{v_i}|\mathbf{x}_{\mathrm{Pa}(v_i)})-e_0\right) \\
&= qe_0 - qe_0 + \sum_{\mathbf{x}_{S\cup A}} \mathrm{P}_\Delta(\mathbf{x}_{S\cup A})\sum_{i=1}^{q}\mathrm{E}(X_{v_i}|x_{\mathrm{Pa}(v_i)}),
\end{aligned}
$$

which is equal to $\sum_{\mathbf{x}_{S\cup A}} \mathrm{P}_\Delta(\mathbf{x}_{S\cup A})\mathrm{E}(U|\mathbf{x}_{S\cup A}) = \mathrm{E}_\Delta(U)$. □

The hardness of solving multiply connected LIMIDs over binary variables follows directly from the result above.

Corollary 3.1. *Given a LIMID of bounded treewidth over binary state and action variables and containing a single value node, deciding whether there is a strategy with expected utility greater than a given rational number k is NP-complete.*

Proof. Membership in NP holds since the diagram has bounded treewidth. By Theorem 3.4, we can reduce any singly connected LIMID of treewidth two (hence bounded) over binary variables (potentially with with *multiple* value

nodes) into a multiply connected LIMID of bounded treewidth over binary variables and a *single* value node. The NP-hardness of the latter class of diagrams follows from the NP-hardness of the former class of diagrams proved in Theorem 3.2. □

We finally show the only positive result of this section.

Theorem 3.5. *Singly connected LIMIDs with binary state and action variables and a single value node can be solved in polynomial time.*

Proof. Consider a singly connected LIMID $\mathscr{L}$ with binary variables and a single value node, and assume w.l.o.g that the diagram is minimal. Since the diagram is singly connected, there is a single path between any node and the value node; since it is minimal, every node has a single child (otherwise we would have either barren nodes or a cycle), and action nodes have no parents (otherwise we would have nonrequisite arcs). Since all action nodes are parentless, the diagram is single-stage, and a strategy $\Delta = \{\delta_j : j \in A\}$ is a multiset of values for the action variables. Cooper [1988] showed that any influence diagram with a single value node can have its utility function normalized without loss of generality. Accordingly, we assume that the value variable $X_v = U$ associated with the (unique) value node v assumes values in a (finite) subset of $[0,1]$. We will prove the result by adapting the 2U algorithm of Fagiuoli and Zaffalon [1998b], which solves updating tasks in credal networks, to solve LIMIDs of the type assumed. This is an immediate consequence of the decision-theoretic view of credal networks given by Antonucci and Zaffalon [2008], which correlates LIMIDs and credal networks and will be discussed in the next chapter.

For each node i, let $A(i) \stackrel{\text{def}}{=} A \cap (\{i\} \cup \text{An}(i))$ denote its ancestor action nodes including i itself if i is an action node (and not including it otherwise), that is, $A(i)$ denote the action nodes $j \in A$ from which there is a directed path to i in the graph of $\mathscr{L}$ plus i itself in case i is an action node. Since $\mathscr{L}$ is singly connected, the ancestors of any two parents of a variable form disjoint sets. Hence, it follows for any strategy Δ that

$$\text{E}_\Delta(U) = \sum_{\mathbf{x}_{\text{Pa}(v)}} \text{E}(X_v|\mathbf{X}_{\text{Pa}(v)} = \mathbf{x}_{\text{Pa}(v)}) \prod_{i \in \text{Pa}(v)} \text{P}(X_i = x_i|\mathbf{X}_{A(i)} = \delta_{A(i)}),$$

where $\delta_{A(i)} \in \Delta$ denotes the policies associated with nodes in $A(i)$. Let Δ^* be an optimal strategy. We will show that the terms $\text{P}(X_i = x_i|\mathbf{X}_{A(i)} = \delta^*_{A(i)})$ in the equation above satisfy

$$\text{P}(X_i = x_i|\mathbf{X}_{A(i)} = \delta^*_{A(i)}) = \max_{\delta_{A(i)}} \text{P}(X_i = x_i|\mathbf{X}_{A(i)} = \delta_{A(i)}), \tag{3.9}$$

for some value $x_i \in \{0,1\}$. To this end, consider a state node i in $\mathrm{Pa}(v)$. Since X_i is binary, we have that $\min_{\delta_{A(i)}} \mathrm{P}(X_i=1|\mathbf{X}_{A(i)}=\delta_{A(i)}) = 1 - \max_{\delta_{A(i)}} \mathrm{P}(X_i=0|\mathbf{X}_{A(i)}=\delta_{A(i)})$. Suppose, to show a contradiction, that Equation (3.9) is false for some X_i. Hence,

$$\min_{\delta_{A(i)}} \mathrm{P}(X_i=0|\mathbf{X}_{A(i)}=\delta_{A(i)}) < \mathrm{P}(X_i=0|\mathbf{X}_{A(i)}=\delta^*_{A(i)}) < \max_{\delta_{A(i)}} \mathrm{P}(X_i=0|\mathbf{X}_{A(i)}=\delta_{A(i)}),$$

and

$$\min_{\delta_{A(i)}} \mathrm{P}(X_i=1|\mathbf{X}_{A(i)}=\delta_{A(i)}) < \mathrm{P}(X_i=1|\mathbf{X}_{A(i)}=\delta^*_{A(i)}) < \max_{\delta_{A(i)}} \mathrm{P}(X_i=1|\mathbf{X}_{A(i)}=\delta_{A(i)}).$$

The expected utility of Δ^* can be written as

$$\mathrm{E}_{\Delta^*}(U) = \mathrm{E}_{\Delta^*}(X_v|X_i=0) + \left[\mathrm{E}_{\Delta^*}(X_v|X_i=1) - \mathrm{E}_{\Delta^*}(X_v|X_i=0)\right] \mathrm{P}(X_i=1|\mathbf{X}_{A(i)}=\delta^*_{A(i)}),$$

where

$$\mathrm{E}_{\Delta^*}(X_v|X_i=x_i) = \sum_{x_{\mathrm{Pa}(v)\setminus\{i\}}} \mathrm{E}(X_v|\mathbf{X}_{\mathrm{Pa}(v)}=\mathbf{x}_{\mathrm{Pa}(v)}) \prod_{j\in\mathrm{Pa}(v)\setminus\{i\}} \mathrm{P}(X_j=x_j|\mathbf{X}_{A(j)}=\delta^*_{A(j)}).$$

Note that $\mathrm{E}_{\Delta^*}(X_v|X_i=x_i)$ is constant with respect to policies in $\delta^*_{A(i)}$, and that $\mathrm{E}_{\Delta^*}(X_v|X_i=1) = \mathrm{E}_{\Delta^*}(X_v|X_i=0)$ implies that $\mathrm{E}_{\Delta}(U)$ is constant with respect to $\delta_{A(i)}$ (so that Equation (3.9) is trivially satisfied). If $\mathrm{E}_{\Delta^*}(X_v|X_i=1) > \mathrm{E}_{\Delta^*}(X_v|X_i=0)$ then

$$\begin{aligned}\left[\mathrm{E}_{\Delta^*}(X_v|X_i=1) - \mathrm{E}_{\Delta^*}(X_v|X_i=0)\right] \mathrm{P}(X_i=1|\mathbf{X}_{A(i)}=\delta^*_{A(i)}) \\ < \left[\mathrm{E}_{\Delta^*}(X_v|X_i=1) - \mathrm{E}_{\Delta^*}(X_v|X_i=0)\right] \max_{\delta_{A(i)}} \mathrm{P}(X_i=1|\mathbf{X}_{A(i)}=\delta_{A(i)}),\end{aligned}$$

which implies that the strategy $\Delta = \Delta^* \setminus \{\delta^*_{A(i)}\} \cup \{\delta'_{A(i)}\}$ that we obtain from Δ^* by replacing $\delta^*_{A(i)}$ with

$$\delta'_{A(i)} = \operatorname*{argmax}_{\delta_{A(i)}} P(X_i=1|\mathbf{X}_{A(i)}=\delta_{A(i)})$$

is better than Δ^* itself (a contradiction). If, on the other hand, $\mathrm{E}_{\Delta^*}(X_v|X_i=1) < \mathrm{E}_{\Delta^*}(X_v|X_i=0)$ then

$$\begin{aligned}\left[\mathrm{E}_{\Delta^*}(X_v|X_i=1) - \mathrm{E}_{\Delta^*}(X_v|X_i=0)\right] \mathrm{P}(X_i=1|\mathbf{X}_{A(i)}=\delta^*_{A(i)}) \\ < \left[\mathrm{E}_{\Delta^*}(X_v|X_i=1) - \mathrm{E}_{\Delta^*}(X_v|X_i=0)\right] \min_{\delta_{A(i)}} \mathrm{P}(X_i=1|\mathbf{X}_{A(i)}=\delta_{A(i)}),\end{aligned}$$

and the strategy Δ obtained from Δ^* by substituting $\delta^*_{A(i)}$ with

$$\delta'_{A(i)} = \operatorname*{argmin}_{\delta_{A(i)}} P(X_i = 1 | \mathbf{X}_{A(i)} = \delta_{A(i)}) = \operatorname*{argmax}_{\delta_{A(i)}} P(X_i = 0 | \mathbf{X}_{A(i)} = \delta_{A(i)})$$

is a better strategy (another contradiction). Therefore, Equation (3.9) is true.

Let $\mathscr{L}_i^{x_i}$ be the diagram obtained from $\mathscr{L}$ by removing all variables that are not an ancestor of i or i, dropping their associated probabilities and expected values, reassigning i as a value node, and specifying $\mathrm{E}(X_i|\mathbf{X}_{\mathrm{Pa}(i)}) = \mathrm{P}(X_i = x_i|\mathbf{X}_{\mathrm{Pa}(i)})$. The diagram $\mathscr{L}_i^{x_i}$ can be obtained in time polynomial in the size of $\mathscr{L}$. Suppose an optimal strategy $\delta^{x_i}_{A(i)}$ for each $\mathscr{L}_i^{x_i}$, $i = 1, \ldots, |\mathrm{Pa}(v)|$ could be obtained in time polynomial in the size of $\mathscr{L}$. Then, by Equation (3.9) an optimal strategy Δ^* for $\mathscr{L}$ can be obtained by exhaustively evaluating the $2^{|\mathrm{Pa}(v)|}$ strategies $\Delta = \{\delta^{x_i}_{A(i)} : i \in \mathrm{Pa}(v)\}$ that we obtain by different combinations of $x_i \in \{0, 1\}$. Since $\mathscr{L}$ encodes $\mathrm{E}(X_v|\mathbf{X}_{\mathrm{Pa}(v)})$, which requires $2^{|\mathrm{Pa}(v)|}$ values, the exhaustive search takes time polynomial in $\mathscr{L}$. Since each $\mathscr{L}_i^{x_i}$ is singly connected, contains only binary variables, and a single value node, its optimal strategy can be obtained similarly, by enumeration of the $2^{|\mathrm{Pa}(i)|}$ strategies $\Delta_i = \{\delta^{x_j}_{A(j)} : j \in \mathrm{Pa}(i)\}$. And since the specification of $\mathscr{L}$ requires $\mathrm{P}(X_i|\mathbf{X}_{\mathrm{Pa}(i)})$, this step too takes time polynomial in $\mathscr{L}$. Finally, if i is a root note, than i is an action node an and optimal policy for $\mathscr{L}_i^{x_i}$ is simply $\delta_i = x_i$, which is obtained in constant time. By applying this reasoning inductively on the nodes of $\mathscr{L}$ in any topological order, we obtain an optimal strategy Δ^* for $\mathscr{L}$ using $O(\sum_i 2^{|\mathrm{Pa}(i)|})$, which is polynomial in the size of $\mathscr{L}$. □

Any singly connected LIMID can be efficiently transformed into an equivalent singly connected LIMID with binary action variables.[13] Thus, the above result remains valid if we lift the constraint on binary action variable. **The results in this section fully characterize the fixed-parameter complexity of solving LIMIDs with respect to topology, variable cardinality and number of value nodes: singly connected diagrams over binary state variables with a single value node are polynomial-time solvable, and relaxing any of these conditions creates diagrams which are potentially NP-hard to solve.** This is summarized in Table 3.1. In the next section, we analyze the complexity of the problem if the requirement of exactness is relaxed.

[13] We can replace any requisite non-binary action variable taking on v values with $\lceil \log_2 v \rceil$ binary action variables, obtain an arbitrary surjection from joint configurations of the new variables into values of the original variable, and redefine the probability distribution of its child variable so that it equals the original distribution when configurations of new variables are mapped back onto values of the original variable. One can verify that this transformation preserves the value of the maximum expected utility.

Table 3.1. Parametrized complexity of solving LIMIDs.

topology	number of value nodes	max. variable cardinality	complexity
polytree	one	two	P
polytree	unbounded	two	NP-complete
polytree	one	three	NP-complete
polytree	one	unbounded	NP-complete
loopy	one	two	NP-complete
loopy	unbounded	bounded	NP-complete

3.5 The complexity of approximately solving LIMIDs

The results in the previous section relate to the difficulty of provably obtaining an optimal solution, that is, a strategy whose expected utility is maximal. In practice, one is usually satisfied with approximate solutions, that is, with strategies that achieve a high but not necessarily maximum utility, provided that it can be obtained efficiently. In this section, we investigate the fixed-parameter complexity of approximately planning with LIMIDs. First, we define what we intend by approximately solving a LIMID.

The most usual measure of the quality of an approximate solution in complexity theory is the so-called *performance ratio*, which in the context of solving LIMIDs is defined for any strategy Δ with positive expected utility as

$$\mathrm{pf}(\Delta) = \frac{\mathrm{E}_{\Delta}(U)}{\max_{\Delta'} \mathrm{E}_{\Delta'}(U)}. \tag{3.10}$$

Note that the performance ratio is a number between zero and one, and it equals one if and only if the strategy is optimal. To apply the above measure to evaluating approximate algorithms, we need to constrain the inputs to LIMIDs where $\min_{\Delta'} \mathrm{E}_{\Delta'}(U) > 0$, so that the function is well-defined and behave properly (i.e., so that the higher the performance ratio of a solution the better it is). Since for any positive number β we have that

$$\frac{\mathrm{E}_{\Delta}(\beta U)}{\max_{\Delta'} \mathrm{E}_{\Delta'}(\beta U)} = \frac{\mathrm{E}_{\Delta}(U)}{\max_{\Delta'} \mathrm{E}_{\Delta'}(U)},$$

the performance ratio of any strategy remains unchanged if we scale the utilities by a positive number. This, in conjunction with the constraint that $\min_{\Delta'} \mathrm{E}_{\Delta'}(U) >$

0, allows us in the following discussion to focus on LIMIDs whose value variables take values in $[0, 1]$, without any loss of generality.[14]

A closely related measure of the quality of a strategy is the *relative error*, given by

$$\mathrm{re}(\Delta) = \frac{\max_{\Delta'} \mathrm{E}_{\Delta'}(U) - \mathrm{E}_{\Delta}(U)}{\max_{\Delta'} \mathrm{E}_{\Delta'}(U)}. \tag{3.11}$$

We have that $\mathrm{re}(\Delta) = 1 - \mathrm{pf}(\Delta)$, and therefore $\mathrm{re}(\Delta) \in [0, 1]$ and is also invariant to positive scalings of the utilities.

The following result shows that it is unlikely that an efficient procedure can be devised that constructs strategies whose relative error is bounded by a constant, even in structurally simple diagrams.

Theorem 3.6. *Unless P equals NP, there is* no *polynomial-time algorithm that for any given LIMID finds a strategy whose relative error is at most ε, for any fixed $\varepsilon < 1$, even if we admit only singly connected diagrams of bounded treewidth and a single value node.*

Proof. Suppose there exists a polynomial-time algorithm for solving singly connected LIMIDs of bounded treewidth, which outputs a strategy whose relative error is at most $\varepsilon < 1$. We will show that we could use such an algorithm to decide an NP-complete problem, implying that P equals NP. In particular, we will use a reduction from the CNF-SAT problem [Garey and Johnson, 1979] into singly connected LIMIDs of bounded treewidth.

Before stating CNF-SAT, we need to introduce some concepts regarding CNF Boolean formulas. A clause is a disjunction of literals, each literal being either a Boolean variable or its negation. We say that a clause is satisfied if, given an assignment of truth-values to its variables, at least one of the literals evaluates to 1. We can decide if a truth-value assignment satisfies a clause in time linear in the number of variables. The CNF-SAT problem is stated as follows.

> Given a set of clauses $C_1, \ldots, C_m$ over (subsets of) Boolean variables $Z_1, \ldots, Z_n$, is there an assignment of truth-values to the variables that satisfies all the clauses?

Let q be a positive integer defined as $q = (m + 1)\lceil -\ln(1 - \varepsilon)\rceil$. Note that q is a polynomial in m, since ε is considered fixed. We reduce an arbitrary instance of CNF-SAT into the LIMID whose graph structure is shown in Figure 3.7, and whose variables and parameters are specified as follows. For each Boolean

[14] The restriction to LIMIDs with positive maximum expected utility is necessary, as the performance ratio is not invariant to a translation of the utility values.

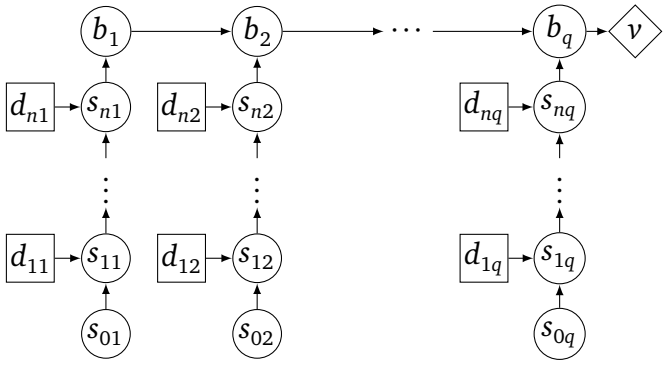


Figure 3.7. LIMID used in the proof of Theorem 3.6.

variable Z_i there are q action nodes $d_{i1},\ldots,d_{iq}$ whose associated variables are binary and take values in $\{0,1\}$, and q state nodes $s_{i1},\ldots,s_{iq}$ whose associated variables take values in $\{0,1,\ldots,m\}$. Additionally, there are q state nodes $s_{01},\ldots,s_{0q}$ whose associated variables also take values in $\{1,2,\ldots,m\}$. Finally, there are q state nodes $b_1,\ldots,b_q$ associated with binary variables, and a value node v with b_q as parent. The DAG of the LIMID thus consists of q replicas of a singly connected diagram over $d_{1j},\ldots,d_{nj},s_{0j},\ldots,s_{nj},b_j$. In any replica $j\in\{1,\ldots,q\}$, the node s_{0j} act as a clause selector, that is, $X_{s_{0j}}=k$ denotes that clause C_k is being "processed", and by summing out $X_{s_{0j}}$ we process all clauses. The action node d_{ij}, $i=1,\ldots,n$, represents an assignment of truth-value for the Boolean variable Z_i. Let us specify the numerical parameters. For all j, we assign uniform probabilities to the variable associated with s_{0j}, that is, $\mathrm{P}(x_{s_{0j}})=1/m$. The conditional probabilities of the variable associated with s_{ij} are defined as follows:

$$\mathrm{P}\left(x_{s_{ij}}\middle|x_{d_{ij}},x_{s_{(i-1)j}}\right)=\begin{cases}1, & \text{if } x_{s_{ij}}=x_{s_{(i-1)j}}=0,\\ 1, & \text{if } x_{s_{ij}}=0, x_{s_{(i-1)j}}=k\geq 1, Z_i=x_{d_{ij}} \text{ satisfies } C_k,\\ 1, & \text{if } x_{s_{ij}}=x_{s_{(i-1)j}}=k\geq 1, Z_i=x_{d_{ij}} \text{ does not satisfy } C_k,\\ 0, & \text{otherwise.}\end{cases}$$

Note that for $i=1$ the first condition in the definition of the probability function above is never met since $X_{s_{0j}}$ takes values in $\{1,\ldots,m\}$. The probabilities of the variables associated with nodes $b^1,\ldots,b^q$ are chosen so as to make the expected utility of any strategy equal the product of the expected utilities of each replica:

$$\mathrm{P}\left(x_{b_j}\middle|x_{s_{nj}},x_{b_{j-1}}\right)=\begin{cases}1, & \text{if } x_{b_j}=x_{b_{j-1}} \text{ and } x_{s_{nj}}=0,\\ 1, & \text{if } x_{b_j}=0 \text{ and } x_{s_{nj}}\neq 0,\\ 0, & \text{otherwise.}\end{cases}$$

The specification of $P(x_{b_1}|x_{s_{n1}})$ is identical to the above by assuming that $x_{b_0} = 1$. The single value variable X_v is defined as $X_v = X_{b_q}$. The LIMID contains $q(2n+2)+1$ variables, each requiring the specification of at most $2(m+1)^2$ numerical parameters taking values in $\{0, 1/m, 1\}$. So t, the number of numerical parameters in the diagram, is at most $q(m+1)^2(4n+4)+2$, which is a polynomial poly(n,m) in m and n, since q itself is a polynomial in m.

It remains to show that a polynomial-time algorithm that finds approximate strategies with provably good relative error can solve the CNF-SAT problem. It follows from the specification of the LIMID for any strategy Δ that

$$\mathrm{E}_\Delta(X_v) = \frac{1}{m^q} \prod_{j=1}^{q} \mathrm{SAT}_j(\Delta),$$

where $\mathrm{SAT}_j(\Delta)$ denotes the number of clauses satisfied by the truth-value assignment of $Z_1 = \delta_{d_{1j}}, \ldots, Z_n = \delta_{d_{nj}}$, $\delta_{d_{ij}} \in \Delta$.

If the CNF-SAT problem is satisfiable then there is an optimum strategy Δ such that $\mathrm{SAT}_j(\Delta) = m$ for all j, and therefore $\max_\Delta \mathrm{E}_\Delta(X_v) = 1$. On the other hand, if the problem is not satisfiable, then we have for all j and strategy Δ that $\mathrm{SAT}_j(\Delta) \leq m-1$, and hence $\max_\Delta \mathrm{E}_\Delta(X_v) \leq (m-1)^q/m^q$. It follows from the inequality $m + 1/2 > 1/\ln(1+1/m)$ [Mauá et al., 2012, Lemma 25] that

$$q \ln(1+1/m) > -\ln(1-\varepsilon),$$

which is equivalent to

$$\left(\frac{m}{m+1}\right)^q < 1-\epsilon\,.$$

If the CNF-SAT instance is satisfiable, any provably good approximate algorithm for solving LIMIDs returns a strategy Δ which for some constant ε satisfies

$$\mathrm{E}_\Delta(X_v) \geq (1-\varepsilon) \max_{\Delta'} \mathrm{E}_{\Delta'}(X_v) > \left(\frac{m}{m+1}\right)^q > \left(\frac{m-1}{m}\right)^q,$$

where the rightmost inequality follows from $m/(m+1) > (m-1)/m$. On the other hand, if the CNF-SAT instance is not satisfiable, then for any Δ:

$$\mathrm{E}_\Delta(X_v) \leq \max_{\Delta'} \mathrm{E}_{\Delta'}(X_v) \leq \left(\frac{m-1}{m}\right)^q.$$

Hence, we can use the approximate algorithm to obtain a strategy Δ with relative error at most ε and decide CNF-SAT by verifying whether $\mathrm{E}_\Delta(X_v) > (m-1)^q/m^q$, which can be done efficiently since the diagram has bounded treewidth, and the threshold $(m-1)^q/m^q$ can be computed in time polynomial in m. □

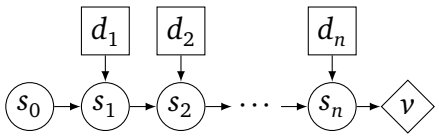


Figure 3.8. A chain structured LIMID with n action nodes.

The above result shows that approximately solving LIMIDs within any *constant* performance ratio is impossible, unless P equals NP. In the spirit of the proof of inapproximability of maximum a posterior inference in Bayesian networks in Park and Darwiche [2004, Theorem8], the result can be strengthened to show that even strategies whose relative error is at most $1-2^{\gamma t}$ are NP-hard to obtain, where t is the number of numerical parameters in the input and γ is an arbitrary fixed constant between zero and one. In other words, no polynomial-time algorithm exists that can provably obtain solutions with sub-exponential relative error (unless P equals NP).

According to Theorem 3.6, polynomial-time algorithms for solving LIMIDs like Single Policy Update (SPU) [Lauritzen and Nilsson, 2001] or Mini-Bucket Elimination [Dechter, 2000] cannot guarantee that the relative error of the strategies they return will be bounded for *all* diagrams, even if only singly connected diagrams of bounded treewidth and a single value node are considered. This result is however a worst-case scenario, and it is possible that these algorithms perform near optimally in almost all input instances. To refute such a claim, we evaluated the accuracy of SPU on randomly generated singly connected LIMIDs with topology as in Figure 3.8. Each diagram was generated by independently sampling each conditional distribution associated to a chance node from a symmetric Dirichlet distribution with parameter $1/m$, where m is the number of variable states. This has the effect of favoring low entropy distributions, which are arguably more realistic. The expected utility of the value variable is an identity function on some value of X_{s_n} selected arbitrarily.

The plots in Figure 3.9 show the relative error of the strategies returned by SPU on each experiment, organized according to the cardinality of variable domains and number of action nodes. The maximum expected utility of each diagram was computed using the Multiple Policy Update (MPU) algorithm [Mauá and de Campos, 2011; Mauá et al., 2012], which despite having exponential worst-case time complexity finished in at most three seconds in each of these experiments.[15] Each circle in the plots depicts the relative error of SPU on a given

[15]The MPU algorithm operates very similarly to the FACTOR-SET-ELIMINATION algorithm described in Chapter 2, except that it discards only dominated factors and it propagates pairs

LIMID. The solid line indicates the third quartile of each fixed configuration, that is, 75% of the vertically aligned experiments appear below the line. The diagrams in the left-hand side plot were obtained with the number of states per variable fixed at 15, while the diagrams on the right had the number of action nodes fixed at ten. We see from the third quartile line on the right-hand side plot that in 25% of the chain diagrams of 20 states and ten decision variables, SPU returned a strategy whose relative error was greater than 10%. Also, there were cases where SPU obtained up to 70% relative error, and a non-negligible amount of cases where SPU returned strategies with relative error greater than, say, 40%. On the other hand, we see that in the majority of the cases the solution returned by SPU achieved a relative error of less than 10%, and that the performance of SPU improved with less nodes and smaller variable cardinalities. The latter remark seems to suggest that the problem is considerably easier if the variable domain cardinalities are bounded by a small constant. We show in the following that indeed there exists a procedure for solving LIMIDs of bounded treewidth and bounded variable domain cardinality that takes a LIMID $\mathscr{L}$ and a constant ε between zero and one, and outputs in time polynomial in the size of $\mathscr{L}$ (measured as the number of symbols in a reasonable encoding of the diagram) and in $1/\varepsilon$ a strategy with relative error at most ε, for any given ε. Such an algorithm is known as a *fully polynomial-time approximation scheme* (FPTAS) for approximately solving LIMIDs.

Theorem 3.7. *There is a fully polynomial-time approximation scheme for solving LIMIDs of bounded treewidth and bounded variable domain cardinality.*

Proof. We prove the result constructively, that is, by designing an FPTAS for the problem. The algorithm we devise is a generalization of the MPU algorithm [Mauá and de Campos, 2011; Mauá et al., 2012], and it relies on the FACTOR-SET-ELIMINATION algorithm described in Chapter 2 (Algorithm 2) to compute provably good strategies by propagating functions over a tree decomposition of the diagram.

Consider a LIMID $\mathscr{L}$ of treewidth bounded by w and whose variables have cardinality bounded by v, and assume without loss of generality that $\mathscr{L}$ is single stage, and contains a single value variable X_v taking values in $[0,1]$. For each action node j in $\mathscr{L}$, let K_j denote the set of indicator functions corresponding to policies for X_j. Hence, K_j is a set containing D_j functions $I(x_j = \tilde{x}_j)$, one for each policy state of $\mathbf{X}_j$, where D_j is the cardinality of X_j's domain. For each state node

of functions. This makes it an exact solver for LIMIDs which can directly handle multiple value nodes.

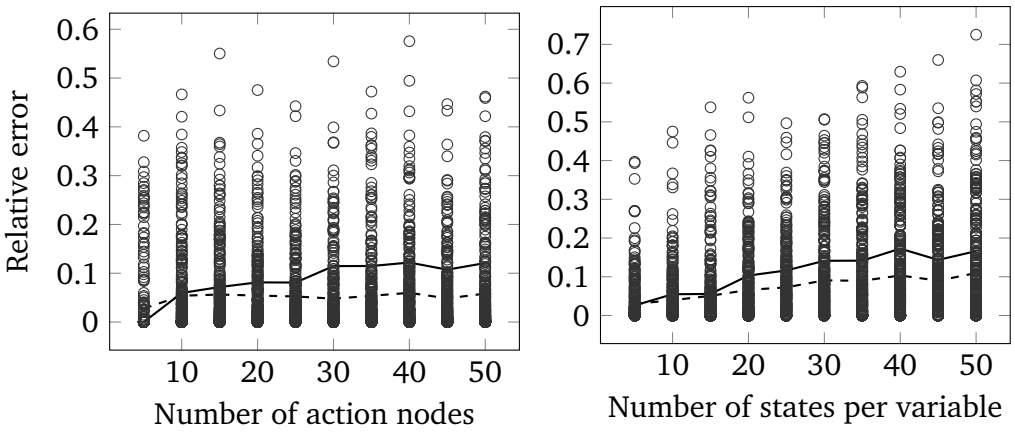


Figure 3.9. Accuracy of SPU on randomly generated chain diagrams. Each circle depicts an experiment, and the dashed and solid lines depict, respectively, the average and third quartile (i.e., 25% of the instances are above the line).

i, let K_i be the singleton containing the conditional probability distribution of X_i given $\mathbf{X}_{\mathrm{Pa}(i)}$. Finally, let K_v be the singleton containing the conditional expected utility function $\mathrm{E}(X_v|\mathbf{X}_{\mathrm{Pa}(v)})$. Note that each set K_i, $i \in N$, is a set of nonnegative real-valued functions of a subset of $\mathbf{X}$ (called factors in Chapter 2). Let be a minimal T binary tree-decomposition for $\mathscr{L}$ whose nodes are associated to sets $C_1, \ldots, C_m$ (T can be obtained in linear time as the treewidth of the diagram is assumed bounded, Bodlaender [1996]). Without loss of generality, assume that $C_i \supseteq \mathrm{Fa}(i)$. For every C_i with $i \notin N$, let K_i be a singleton containing the identity function over $\mathbf{X}_{C_i}$. Also, for every $i \in N$ with $C_i \supset \mathrm{Pa}(i)$, redefine the functions in K_i so that they have domain $\mathbf{X}_{C_i}$ and return the same values when restricted to $\mathbf{X}_{\mathrm{Fa}(i)}$ (i.e., replace every function ψ of $\mathbf{X}_{\mathrm{Fa}(i)}$ in K_i with a function ϕ of $\mathbf{X}_{C_i}$ such that $\phi(\mathbf{x}_{C_i}) = \psi(\mathbf{x}_{\mathrm{Fa}(i)})$). Consider a sequence of functions $\phi_1, \ldots, \phi_m$ such that $\phi_i \in K_i$, $i = 1, \ldots, m$. By design, we have that

$$\sum_{\mathbf{x}_{S \cup A}} \prod_{i=1}^{m} \phi_i(\mathbf{x}_{C_i}) = \mathrm{E}_{\Delta}(X_v),$$

for some strategy Δ, and for every strategy Δ there is a sequence of functions $\phi_1, \ldots, \phi_m$ taken, respectively, from $K_1, \ldots, K_m$, that satisfies the equation above. Thus, running the FACTOR-SET-ELIMINATION on sets $K_1, \ldots, K_m$ and tree decomposition T thus defined produces a strategy Δ and numbers Z_l and Z_u such that $\mathrm{E}_{\Delta}(X_v) = Z_l \leq \max_{\Delta'} \mathrm{E}_{\Delta'}(X_v) \leq Z_u$. Recall that FACTOR-SET-ELIMINATION runs in time polynomial in the constants $k_1, \ldots, k_m$ that set the maximum cardinality of

propagated sets L_i, and that

$$\frac{Z_l}{\max_{\Delta'} \mathrm{E}_{\Delta'}(X_v)} \geq \frac{Z_l}{Z_u} \geq \frac{1}{\prod_{i=1}^{m} \eta(M_i, L_i)},$$

where $\eta(M_i, L_i) = \max_{\phi \in M_i} \min_{\psi \in L_i} \max_{\mathbf{x}_{C_i}} \phi(\mathbf{x}_{C_i})/\psi(\mathbf{x}_{C_i})$. Thus, to obtain a strategy Δ with $\mathrm{pf}(\Delta) \geq 1 - \varepsilon$, it suffices to run FACTOR-SET-ELIMINATION with constants $k_1, \ldots, k_m$ such that

$$\prod_{i=1}^{m} \eta(M_i, L_i) \leq \left(1 + \frac{\varepsilon/(1-\varepsilon)}{2m}\right)^m \leq 1 + \frac{\varepsilon}{1-\varepsilon} = \frac{1}{1-\varepsilon},$$

where we used the known inequality $(1 + x/2k)^k \leq 1 + k$, valid for every positive integer k and real x in $[0, 1]$ (the inequality assumes that $\varepsilon \leq 1/2$; if this is not the case we use a constant error bound $1/2$, which obtains the desired relative error and is still polynomial in $1/\varepsilon$). Moreover, if $k_1, \ldots, k_m$ are polynomially bounded in the size of the input and in $1/\varepsilon$, the algorithm finishes in polynomial time in these quantities. This can be done by a suitable initialization of the greedy clustering scheme performed to obtain the sets $L_1, \ldots, L_m$. Recall that in Chapter 2 we obtained the set L_i, $i = 1, \ldots, m$, by performing a greedy search on an initial set of representatives that was obtained by the K-MEDOIDS procedure (Algorithm 3). To guarantee the bound $\eta(M_i, L_i)$, we instead select an initial set of representatives as follows. For an arbitrary node i in T, let $t = \min_{\phi \in M_i} \min_{\mathbf{x}_{C_i}} \phi(\mathbf{x}_{C_i})$ be the smallest positive number in the image of a function in M_i and q be the number of assignments to $\mathbf{X}_{C_i}$ (i.e., the cardinality of the domain of the functions). Consider the partition of the q-dimensional hypercube $[0, 1]^q$ into hyper-rectangles $[l_1, u_1] \times \cdots \times [l_q, u_q]$ such that for each $j = 1, \ldots, q$ either $[l_j, u_j] = [0, t)$ or $[l_j, u_j] = [t, \alpha^{-k}]$ or $[l_j, u_j] = (\alpha^k, \alpha^{k-1}]$, where k is some nonnegative integer, and

$$\alpha = 1 + \frac{\varepsilon/(1-\varepsilon)}{2m}$$

is the upper bound on $\eta(M_i, L_i)$. There are $(1 - \lfloor \log_\alpha t \rfloor)^q$ such hyper-rectangles, and each contains zero or more functions ϕ in M_i. Figure 3.10 illustrates such a partition for the case of $q = 2$ and set M_i. Let $k_i = (1 - \lfloor \log_\alpha t \rfloor)^q$ and construct an initial set L_i of representatives by selecting at most one function ψ in each hyper-rectangle. This can be done in time linear in k_i and q. Since each function ϕ in M_i is obtained by a polynomial number of multiplications and additions of the numerical parameters in the input, the number t is greater than 2^{-bn}, where

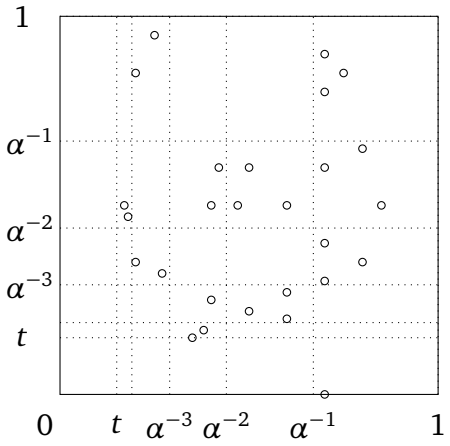


Figure 3.10. A partition of functions according to hyper-rectangles $[l_1, u_1] \times [l_2, u_2]$ such that $l_j/u_j = \alpha$, $j = 1, 2$. t is the smallest value returned by a function in the set.

b is the size of the bit-string encoding the LIMID. Hence,

$$1 - \lfloor \log_\alpha t \rfloor \leq -\log_\alpha t \leq \frac{bn}{\ln \alpha} \leq \frac{bmn}{1 - 1/\alpha} = \frac{2bnm^2}{\varepsilon},$$

where we have used the inequality $\ln(x) \geq 1 - 1/x$, valid for any $x \geq 1$. Thus, k_i is $O([bnm^2/\varepsilon]^q)$, which is polynomial in b and $1/\varepsilon$, since q is at most v^w, which is assumed constant, and there is always a minimal tree decomposition with a polynomial number of nodes in the number of variables n. □

The asymptotic analysis in the proof of Theorem 3.7 guarantees only that the worst-case running time of the FPTAS is bounded by a polynomial with a very high exponent and huge constants, much too high to enable practical application of the algorithm. For example, for a LIMID of treewidth five and maximum variable cardinality four, the polynomial has exponent $4^5 = 1024$. Moreover, if the smallest value in a function in a set L_i is $t = 0.05$ and ε and m are such that $\alpha = 1.1$, the bound on the maximum cardinality of L_i is $k_i \leq (1 - \log_\alpha t)^{4^5} > 32^{4^5} = 2^{5120}$. Hence, computing such a set would be prohibitively expensive for any current computer. This worst-case analysis however does not prevent us from applying the same reduction used in the proof and heuristically apply FACTOR-SET-ELIMINATION to solve LIMIDs.

To test the applicability of such an approach, we measured the running time of the procedure described with relative error upper bound of $\varepsilon = 0.1$ on a set of 1620 LIMIDs randomly generated as follows. Each LIMID is parametrized by

the number of action nodes a, the number of state nodes s, the maximum cardinality of the domain of the family of a state variable ω_S, and the maximum cardinality of the domain of the family of a action variable ω_A. Given a configuration of the parameters, we obtain a LIMID as follows. First, we specify $c + d + v$ variables taking values in sets containing from 2 to 4 elements (the actual cardinality is randomly chosen). Then we create the DAG structure as follows. We start with an empty graph and connect each action node to a value node. This guarantees that all actions are relevant for the computation of the maximum expected utility. Then we repeatedly insert arcs in a way that neither makes the domain of the family of a variable greater than the given bounds nor makes the treewidth more than 10, until no such arcs can be added.[16] The generated diagram contains action and state nodes with at most $\log_2 \omega_A - 1$ and $\log_2 \omega_S - 1$ parents, respectively. Once the DAG is obtained, we randomly sample the probability mass functions and utility functions associated to state and value variables, respectively.

We generated 33 different classes of diagrams by selecting different parameter configurations in the ranges $5 \leq a \leq 50$, $8 \leq s \leq 50$, $8 \leq \omega_A \leq 64$ and $16 \leq \omega_S \leq 64$ (we set $v = a + 2$). The procedure was able to finish with a solution within a time limit of 12 hours (of CPU usage) in approximately 96% of the cases. To attempt to test the performance of the procedure on difficult instances, we kept only the class of diagrams in which at most 10% could be solved exactly within the same time limit of 12 hours by the CR algorithm of de Campos and Ji [2008], which recasts the problem of solving LIMIDs as a mixed-integer linear program.[17] The results (average running time in seconds and percentage of solved instances) for each class of "hard" instances are shown in Table 3.2. We see that the FPTAS was able to solve a large number of instances containing up to 150 variables and 10^{64} strategies in which CR could not solve, but that it failed to solve any instances with $\omega_A = 64$. The latter quantity, together with the number of action nodes, determines the number of strategies in the diagram, and hence the size of the search space. To better visualize the correlation between size of the search space (which corresponds to the difficult of a brute-force approach) and performance, we plot the running time of the FPTAS in every instance (i.e.,

[16] Since current algorithms for checking whether the treewidth of a graph exceeds a fixed k are too slow for $k \geq 5$ Bodlaender [1996], we resort to the minimum fill-in heuristic that resulted in diagrams whose actual treewidth ranged from 5 to 10.

[17] We used the CR implementation available at `http://www.idsia.ch/~cassio/id2mip/` and CPLEX (`http://www.ilog.com`) as mixed-linear integer programming solver. An instance was consider "solved" by CR if the upper and lower bounds provided by CPLEX within the allowed time limit differed by less than 0.0001.

Table 3.2. Runtime performance of the FPTAS on "hard" instances.

N	a	s	v	ω_A	ω_S	% SOLVED	RUNTIME
30	50	48	52	8	16	100	0.5 ± 0.09
30	30	38	32	16	16	100	2 ± 10
30	10	28	12	16	64	96	47 ± 142
30	30	88	32	12	16	100	230 ± 1027
60	10	28	12	32	32	93	905 ± 2847
60	20	8	22	32	32	78	938 ± 1417
30	20	8	22	32	64	76	1592 ± 3402
30	50	48	52	12	16	96	1753 ± 7405
30	10	28	12	32	64	86	2440 ± 7606
30	10	28	12	64	64	0	—
30	20	8	22	64	64	0	—

not only those in the "hard" classes) against number of strategies. The result is shown in Figure 3.11, where we also show for each instance the maximum cardinality of a set L_i generated during the procedure. These plots suggest that the procedure complexity grows sub-exponentially with the number of strategies in diagrams, at least in medium-sized diagrams as the ones we generated (the plots in Figure 3.11 are in log-scale, so line-like appearance indicates sub-exponential behavior).

3.6 Conclusion

Influence diagrams are widely used models for decision making under uncertainty. Typically, they represent situations involving a single non-forgetful agent. This leads to computational difficulties, as an agent acting on a partially observable domain with perfect recall has to store an ever growing amount of information regarding previous actions and observations. Limited memory influence diagrams (LIMIDs) remedy this situation by explicitly modeling which actions and observations are to be remembered at each decision step.

This chapter provided a deep analysis of the theoretical complexity of planning with LIMIDs (i.e., of finding mappings from observations to actions that attempt to maximize expected utility). The outcome of this study shows that optimal polynomial-time algorithms are unlikely to exist even for diagrams representing the most simple decision scenarios. In particular, we have shown that

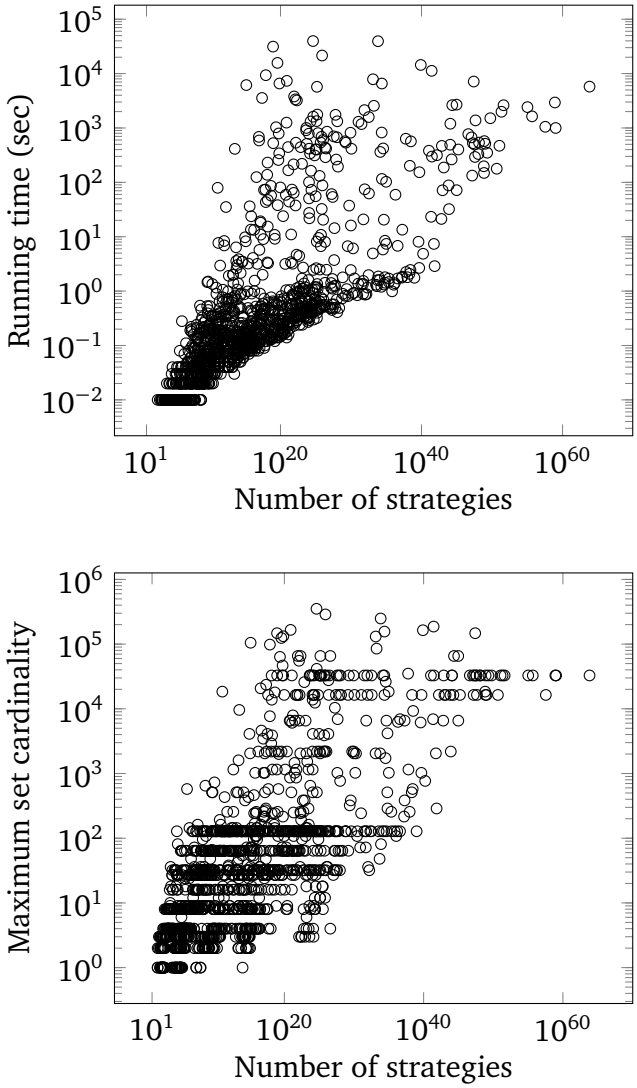


Figure 3.11. Performance of the FPTAS on all instances.

singly connected LIMIDs of bounded treewidth with binary variables and a single value node are "easy" to solve, but relaxing any of these conditions can lead to intractable problems (i.e., NP-hard).

The situation is slightly improved if we consider approximate algorithms with provably good outputs. In this case, we showed that bounded treewidth LIMIDs whose variables take on a bounded number of values can be approximately solved in polynomial time for any given accuracy. On the other hand, relaxing any of these conditions leads to inapproximability. In more technical terminology, we have shown that the bounded treewidth, bounded variable cardinality LIMIDs admit a fully polynomial-time approximation scheme, but there are no approximate algorithm with sub-exponential approximation factor for diagrams of unbounded variable cardinality (unless P equals NP).

The theoretical results we developed were corroborated by empirical analysis. We showed that local search procedures are unable to achieve a good approximation factor in a large number of problem instances (say 25%) involving diagrams with chain-like structure and a single value node, especially when the variable cardinality was increased above thirty. On the other hand, the approximation algorithm we devised, in spite of being too slow to run on large instances, was able to solve a significantly number of medium-sized diagrams of bounded treewidth and bounded variable cardinality in feasible time. Thus, variable cardinality seems to play a major role in the difficulty of planning with influence diagrams. To our knowledge, this correlation has not been noticed in the literature before, despite being of practical relevance for people interested in applying influence diagrams in real world problems.

Chapter 4

Updating credal networks

Bayesian networks are probabilistic graphical models where irrelevance assessments between sets of variables are represented by a directed acyclic graph (DAG) whose nodes are identified with variables [Pearl, 1988]. They have been successfully applied to a wide range of data and knowledge-based applications, from computer vision tasks to decision support systems [Jensen and Nielsen, 2007; Koller and Friedman, 2009]. In addition to its DAG, the specification of a Bayesian network requires the specification of a conditional probability distribution for every variable and every configuration of its parents. When information is costly to acquire, specifying these conditional probabilities can be a daunting task, whether they are estimated from data or elicited from experts. This causes the inferences drawn with the model to contain imprecisions and arbitrarinesses [Kwisthout and van der Gaag, 2008].

An arguably more principled approach to coping with the imprecision in the numerical parameters is by incorporating it into the formalism. One way of doing so is by means of closed and convex sets of probability distributions, which are called *credal sets* [Levi, 1980]. Other approaches include random sets [Kendall, 1974], Dempster-Shafer theory [Shafer, 1976; Shenoy and Shafer, 1988], possibility theory [Zadeh, 1978] and coherent lower previsions [Walley, 1991], the last one being largely equivalent to credal sets (there is a one-to-one correspondence between credal sets and coherent lower previsions). Bayesian networks whose numerical parameters are specified by conditional credal sets are known as *credal networks* [Cano et al., 1994; Cozman, 2000, 2005]. Credal networks have been successfully applied to robust pattern recognition [Zaffalon et al., 2003; Zaffalon, 2005; de Campos et al., 2009; Antonucci et al., 2009; Corani et al., 2010; Antonucci et al., 2011; de Campos and Ji, 2011], and knowledge-based systems, where it has been argued that allowing parameters

to be imprecisely specified facilitates elicitation from experts [Walley, 2000; Antonucci et al., 2007; Salvetti et al., 2008; Antonucci et al., 2009; Piatti et al., 2010; Antonucci et al., 2013c].

A Bayesian network provides a concise representation of the (single) joint probability distribution that is consistent with the network parameters and satisfies (at least) the set of stochastic independences encoded in the graph. Analogously, a credal network provides a concise representation of the credal set of joint distributions that are consistent with the local credal sets and satisfy (at least) the irrelevances encoded in the graph. The precise characterization of this joint credal set depends however on the concept of irrelevance adopted.

The two most commonly used irrelevance concepts in the literature are *strong independence* and *epistemic irrelevance*. Two variables X and Y are *strongly independent* if the joint credal set of X and Y can be regarded as originating from a number of precise probability distributions under each of which the two variables are stochastically independent. Strong independence is thus closely related to the sensitivity analysis interpretation of credal sets, which regards an imprecisely specified model as arising out of partial ignorance of an ideal precisely specified one [Walley, 1991; Kwisthout and van der Gaag, 2008; Antonucci and Piatti, 2009; Zaffalon and Miranda, 2009]. If a piece of uncertain knowledge is considered poorly represented by any precise probability distribution, then any irrelevance concept that is based on precise probability models might not be very suited for the task. Arguably, a proper notion of irrelevance between events in such a case should be a property of the conditional credal sets. Epistemic irrelevance is one such possible notion. A variable X is *epistemically irrelevant* to a variable Y if the marginal credal set of Y according to our model is the same whether we observe the value of X or not [Walley, 1991]. Intuitively, variable X is epistemically irrelevant to Y if our beliefs about the value of latter is unaltered by the disclosure of the value of former. Unlike strong independence, epistemic irrelevance is an asymmetric concept and cannot in general be characterized by the properties of the elements of the credal set alone [de Cooman et al., 2010]. Moreover, strong independence implies epistemic irrelevance, whereas the converse might not hold.

If on the one hand the flexibility provided by credal sets arguably facilitates model building, on the other, it imposes a great burden on the computation of inferences. For example, whereas computing the posterior probability of a variable is polynomial-time computable in polytree-shaped Bayesian networks, the analogous task in credal polytrees, that is, computing upper and lower bounds on the posterior probability of a given variable is an NP-hard task [de Campos and Cozman, 2005]. There are however exceptional cases, such as the case of

credal polytrees with binary variables, which can be solved in polynomial time. Like in Bayesian networks, the theoretical and practical tractability of inferences in credal networks depends strongly on the network topology and the cardinality of the variable domains. Credal networks however include another dimension in the parametrized complexity of inference, given by the type of irrelevance concept adopted, which in the Bayesian case is usually fixed. For instance, computing probability bounds in credal trees under the concept of epistemic irrelevance can be performed in polynomial time [de Cooman et al., 2010], whereas we show here that the same task is NP-hard under strong independence.

From an algorithmic perspective, drawing inferences from credal networks under strong independence shares a great deal of similarity with planning in limited memory influence diagrams and performing maximum a posteriori inference in Bayesian networks. In fact, one of the common approaches to updating probability bounds in credal networks under strong independence is to reduce the problem to the computation of a MAP inference in a properly designed Bayesian network [Cano et al., 1994; Cozman, 2000]. The converse also holds, that is, any MAP assignment task can be reduced to the computation of probability bounds in a credal network under strong independence [de Campos and Cozman, 2005]. Antonucci and Zaffalon [2008] showed the correspondence between inference in credal networks and planning with LIMIDs. Particularly, they showed that the computation of probability bounds in credal networks under strong independence can be given a decision-theoretic interpretation, in which one selects extreme distributions of the local conditional credal sets in order to maximize (or minimize) a posterior probability, and thus be reduced to the computation of an optimal strategy in a LIMID.

Although these reductions and equivalences between these seemingly different problems allow us to derive many complexity results and algorithms for one task from the others, they are often inconclusive with respect to the parametrized complexity, as the reductions often produce models with topology or variable cardinalities others than that of the original model. For instance, the known reductions map an inference in a credal tree into a MAP inference in a Bayesian polytree, or into solving a multiply connected LIMID. Furthermore, all these equivalences assume strong independence of the credal network, and it is not clear how results concerning epistemic irrelevance could be derived from them.

In the rest of this chapter, we define credal networks (Section 4.1), formalize the updating problem (Section 4.2), and investigate the parametrized theoretical computational complexity of inferences in credal networks (Section 4.3), both under strong independence and epistemic irrelevance. We show that a par-

ticular type of inference in imprecise hidden Markov models (i.e., HMMs with uncertainty quantified by local credal sets) is invariant to the choice of either irrelevance concept, being thus polynomial-time computable (as this is known to be the case under epistemic irrelevance. We also show that even in credal trees inferences are NP-hard if we assume strong independence, and that this is the same complexity of inference in credal polytrees for both irrelevance concepts, even if we assume that all variables are ternary. We prove that the so-called linear vacuous models, that is, credal networks that can be seen as a mixture of a vacuous prior and a precise likelihood, lead to the same inferences whether we assume epistemic irrelevance or strong independence. Finally, we show the existence of a fully polynomial-time approximation scheme for inferences in credal networks of bounded treewidth and bounded variable domain cardinality under strong independence.

Most of the material presented here appeared in References [Mauá et al., 2011], [Mauá et al., 2012b] and [Mauá et al., 2013].

4.1 Credal networks

Consider a finite set of variables $\mathbf{X} = \{X_i : i \in N\}$, each variable taking values in a finite set. A *credal set* is a closed and convex set of probability distributions of the same set of variables [Levi, 1980]. The *vacuous credal set* of a set of variables $\mathbf{X}_S \subseteq \mathbf{X}$ is the largest credal set of probability distributions of those variables, and is denoted by $V(\mathbf{X}_S)$. A *conditional credal set* is a credal set of conditional probability distributions of the same set of variables, and all conditioned on the same event. The *lower expectation* $\mathrm{L}_M(f)$ of a real-valued function f of a subset of variables $\mathbf{X}_S$ under a fixed credal set M of probability distributions of $\mathbf{X}_S$ is the minimum expectation of f under any distribution in the credal set, that is,

$$\mathrm{L}_M(f) \stackrel{\text{def}}{=} \min_{p \in M} \mathrm{E}_p(f) \stackrel{\text{def}}{=} \min_{p \in M} \sum_{\mathbf{x}_S} f(\mathbf{x}_S) p(\mathbf{x}_S), \tag{4.1}$$

where $\mathrm{E}_p(f)$ denotes the expectation of f under a given probability distribution p of $\mathbf{X}_S$ in M. One can verify that lower expectations under the same credal set are positive homogeneous and superlinear, and avoid sure loss, that is, $\mathrm{L}_M(af) = a\mathrm{L}_M(f)$ if $a \geq 0$, $\mathrm{L}_M(f+g) \geq \mathrm{L}_M(f) + \mathrm{L}_M(g)$, and $\mathrm{L}_M(f) \geq \min f$ for any real-valued functions f and g and number a. Thus, lower expectations thus defined are equivalent to Walley's definition of coherent lower previsions [Walley, 1991, Chapter 3.3].

An *extreme distribution* of a credal set is an element of the set that cannot be written as a convex combination of other elements in the same set. We denote the set of extreme distributions of a credal set M by $\operatorname{ext} M$. A credal set is *finitely generated* if it contains a finite number of extreme distributions. A finite representation of a finitely generated credal set by means of its extreme distributions is called *vertex based*. Any finitely generated (conditional) credal set of a subset of variables $\mathbf{X}_S$ in $\mathbf{X}$ defines a (bounded) polytope in the probability simplex of distributions of $\mathbf{X}_S$, and can therefore be written as a finite set of linear inequalities in $p(\mathbf{x}_S)$ of the form [Cozman, 2000]

$$\sum_{\mathbf{x}_S} f(\mathbf{x}_S) p(\mathbf{x}_S) \leq 0\,, \tag{4.2}$$

where f is any real-valued function of $\mathbf{X}_S$. The converse is also true: any finite set of linear inequalities of the form above determines a (bounded) polytope in the probability simplex [Boyd and Vandenberghe, 2004, Chapter 2], and hence a finitely generated credal set. Thus, an alternative finite representation of a credal set is by means of a finite set of functions defining linear inequalities of the type above. Such a representation is called *constraint based*. Vertex- and constraint-based representations of the same credal set can have very different sizes. To see this, consider a single variable X taking values in $\{0, 1, \ldots, m\} = \{0\} \cup [m]$, and let $M = \{p \in V(X) : p(k) \leq 1/(m+1), k = 1, \ldots, m\}$. The set M is isomorphic to an m-dimensional hypercube, and therefore has 2^m extreme distributions,[1] whereas the same set can be represented in constraint-based form by m degenerate functions of X translated by $1/(m+1)$. Moving from a vertex-based to a constraint-based representation can also result in an exponential increase in the size of the input. Consider a variable X taking values in $[m]$ and let $M = \operatorname{co}\{e_1, \ldots, e_m, 1 - e_1, \ldots, 1 - e_m\}$, where co denotes the convex hull operator, and e_k the degenerate distribution placing all mass at $X = k$. It can be shown that M is affinely equivalent to the m-dimensional cross-polytope $\{f(X) : \sum_x |f(x)| \leq 1\}$, which is the dual of the m-dimensional hypercube and whose constraint-based representation requires 2^m inequalities, whereas its vertex-based representation needs only $2m$ distributions [Kalai and Ziegler, 2000, page 11]. Tessem [1992] and de Campos et al. [1994] studied the representation of credal sets defined by linear constraints of the form

[1] For any nonnegative integer k not greater than m and any (potentially empty) subset S of $[m]$ of cardinality k, any distribution that assigns mass $(m+1-k)/(m+1)$ to $p(0)$, mass $1/(m+1)$ to $p(j)$ such that j is in S, and zero mass elsewhere, is in M, since it satisfies all the constraints in M and is a valid distribution. There are 2^m such distributions, and each one cannot be written as a convex combination of any other distribution in the set, since each "touches" an inequality in at least one dimension.

$l_x \leq p(x) \leq u_x$, where l_x and u_x are real numbers, and showed that these credal sets can have exponentially many extreme distributions in the number of constraints. Wallner [2007] proved an attainable upper bound of $m!$ extreme distributions on credal sets more generally defined by a coherent lower probability function of an m-ary variable. More recently, Miranda and Destercke [2013] investigated the number of extreme distributions in credal sets defined by linear constraints of the form $p(x) \leq p(x')$ for $x \neq x'$, and proved an attainable upper bound of 2^{m-1} for the case of an m-ary variable. Importantly, both vacuous credal sets (of variables of any cardinality) and credal sets of binary variables can be succinctly represented in either vertex- or constraint-based form. The next example is supposed to clarify the terminology and concepts.

Example 4.1. *Consider* $\mathbf{X} = \{X_1, X_2\}$*, where* X_1 *takes values in* $\{0,1,2\}$*, and* X_2 *takes values in* $\{0,1\}$*. The conditional vacuous set for* X_1 *given* X_2 *is the probability simplex on the plane, drawn as a triangles with vertices* $p(1)$*,* $p(2)$ *and* $p(3)$ *in Figure 4.1. Let*

$$M(X_1|X_2{=}0) = \{p \in V(X_1) : p(k) \leq 1/3, k = 1,2\}$$

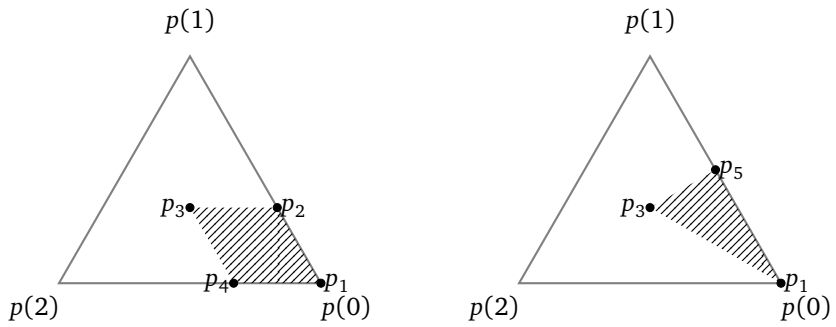
and

$$M(X_1|X_2{=}1) = \{p \in V(X_1) : p(0) \leq p(1) \leq p(2)\}$$

be conditional credal sets for X_1 *given* X_2*, and* $M(X_2)$ *be the singleton containing the distribution* p *of* X_2 *such that* $p(0) = p(1) = 1/2$*. The first two sets are depicted in Figure 4.1. Let us represent a generic function* f *on* $\{0,\ldots,m\}$ *by the* m*-tuple* $(f(0),\ldots,f(m))$*, and define*

$$\begin{aligned}
p_1 &= (1,0,0), & p_2 &= (2/3,1/3,0),\\
p_3 &= (1/3,1/3,1/3), & p_4 &= (2/3,0,1/3),\\
p_5 &= (1/2,1/2,0), & f_1 &= (-1,2,-1),\\
f_2 &= (-1,-1,2), & f_3 &= (1,-1,0),\\
f_4 &= (0,1,-1), & f_5 &= (1,-1).
\end{aligned}$$

Then the set $M(X_1|X_2 = 0)$ *can be represented in vertex- and constraint-based form, respectively, as* $M(X_1|X_2 = 0) = \text{co}\{p_1,p_2,p_3,p_4\}$ *and* $M(X_1|X_2 = 0) = \{p \in V(X_1) : \mathrm{E}_p(f_1) \leq 0, \mathrm{E}_p(f_2) \leq 0\}$*, while the set* $M(X_1|X_2 = 1)$ *is represented in vertex- and constraint-based forms as* $M(X_1|X_2 = 1) = \text{co}\{p_1,p_3,p_5\}$ *and* $M(X_1|X_2{=}1) = \{p \in V(X_1) : \mathrm{E}_p(f_3) \leq 0, \mathrm{E}_p(f_4) \leq 0\}$*, respectively. Similarly,* $M(X_2)$ *can be represented as* $M(X_2) = \{(1/2,1/2)\}$ *in vertex-based form, and as* $M(X_2) = \{p \in V(X_2) : \mathrm{E}_p(f_5) \leq 0, \mathrm{E}_p(f_5) \geq 0\}$ *in constraint-based form.* □



(a) $M(X_1|X_2{=}0)$ (b) $M(X_1|X_2{=}1)$

Figure 4.1. Barycentric coordinate-system visualization of the conditional credal sets in Example 4.1 (hatched regions) and their corresponding extreme distributions (black circles).

A separately specified *credal network* is a triple $(\mathbf{X}, G, M)$, where $G = (N, A)$ is a DAG, and M is a set of "local" credal sets $M(X_i|\mathbf{x}_{\mathrm{Pa}(i)})$ of conditional probability distributions of X_i, one set for each variable X_i in $\mathbf{X}$ and each configuration $\mathbf{x}_{\mathrm{Pa}(i)}$ of its parents. A node i and its associated variable X_i are said to be *precise* if the corresponding conditional credal sets $M(X_i|\mathbf{x}_{\mathrm{Pa}(I)})$ are all singletons, otherwise they are said to be *imprecise*. If all local credal sets are vacuous, the node is said to be vacuous. A Bayesian network is simply a credal network with all nodes precise. The following example contains a simply separately specified credal network.

Example 4.2. *Consider the credal network $\mathcal{N}$ over variables X_1, X_2 and X_3 that take values in $\{0,1\}$, and with DAG as shown in Figure 4.2. The local credal sets are*

$$M(X_1) = \{p \in V(X_1) : 0.5 \leq p(1) \leq 0.6\} = \mathrm{co}\{(0.4, 0.6), (0.5, 0.5)\},$$
$$M(X_2) = \{p \in V(X_2) : 0.5 \leq p(1) \leq 0.6\} = \mathrm{co}\{(0.4, 0.6), (0.5, 0.5)\},$$

and $M(X_3|X_1 = i, X_2 = j) = \{p^{ij}\}$ for any i and j, where p^{ij} is the probability distribution of X_3 such that $p^{ij}(1) = 0$ if $i = j$ and $p^{ij}(1) = 1$ otherwise. □

The local credal sets in a credal network are represented either in constraint- or vertex-based form. When represented by vertices, it is also common to allow for the specification of logical constraints on the extreme distributions of the local credal sets of a variable. This is usually accomplished by replacing the collection of local conditional credal sets with a collection of *conditional probability*

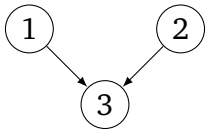


Figure 4.2. DAG of the credal network in Example 4.2.

potentials $p(X_i;\mathbf{X}_{\mathrm{Pa}(i)})$, one for each variable X_i, where a probability potential is simply a function of both X_i and $\mathbf{X}_{\mathrm{Pa}(i)}$ that returns a conditional probability distribution $p(X_i;\mathbf{x}_{\mathrm{Pa}(i)})$ for every assignment $\mathbf{x}_{\mathrm{Pa}(i)}$. A conditional probability potential $p(X_i;\mathbf{X}_{\mathrm{Pa}(i)})$ is a collection of statements that the simultaneous selection of extreme distributions $p(X_i;\mathbf{x}_{\mathrm{Pa}(i)})$ and $p(X_i;\mathbf{x}'_{\mathrm{Pa}(i)})$ from different local credal sets $M(X_i|\mathbf{x}_{\mathrm{Pa}(i)})$ and $M(X_i|\mathbf{x}'_{\mathrm{Pa}(i)})$, with $\mathbf{x}_{\mathrm{Pa}(i)} \neq \mathbf{x}'_{\mathrm{Pa}(i)}$, is valid or feasible, implying that missing combinations are inadmissible. A node whose local model is represented by probability potentials is said to be *extensively specified*. Credal networks whose nodes are extensively specified are called also extensively specified. By definition, when represented in vertex-based form the unconditional credal sets associated to root nodes of the network are both extensively and separately specified.[2] Thus, vertex-based credal networks whose non-root nodes are precise are both extensively and separately specified. In the literature, extensively specified local credal sets have only been considered when strong independence is assumed. We follow the literature and implicitly assume separate specification of local credal sets when discussing epistemic irrelevance. The following example illustrates a simple extensively specified credal network.

Example 4.3. *Consider the separately specified credal network $\mathcal{N}$ in Example 4.2, and redefine the local credal sets associated to X_3 as vacuous credal sets. Note that for any values of x_1 and x_2, the extreme distributions of the local credal sets $M(X_3|x_1,x_2)$ are the distributions p^{00} and p^{01} as defined in Example 4.2. Specify logical constraints on extreme distributions of $M(X_3|x_1,x_2)$ by the two potentials*

X_1	X_2	$p_1(X_3;X_1,X_2)$	$p_2(X_3;X_1,X_2)$
0	*0*	p^{00}	p^{01}
0	*1*	p^{00}	p^{01}
1	*0*	p^{00}	p^{01}
1	*1*	p^{00}	p^{01}

In other words, X_3 is a vacuous node with the extensive constraint that the same degenerate distribution ought to be selected on each of the local credal sets corresponding to different configurations of the parents. □

[2]We say that a node i is separately specified if there are no logical constraints on the distributions of any two conditional credal sets $M(X_i|x_j)$ and $M(X_i|x'_j)$, with $x_j \neq x'_j$.

The DAG G of a credal network specifies a set of conditional irrelevancies between sets of variables which generalize the Markov condition in Bayesian networks. More specifically, for any node i in G, the set $\mathbf{X}_{\mathrm{Nd}(i)\setminus\mathrm{Pa}(i)}$ of non-descendant non-parent variables of X_i is assumed irrelevant to X_i conditional on its parent variables $\mathbf{X}_{\mathrm{Pa}(i)}$. The precise definition of this statement requires the definition of an irrelevance concept. For instance, if stochastic independence is adopted as irrelevance concept, then the DAG G describes a set of Markov conditions as a Bayesian network. In the credal network formalism, the two most common irrelevance concepts used are *strong independence* and *epistemic irrelevance*.

Any joint probability distribution $p \in V(\mathbf{X})$ induces a probability measure P_p on the sigma-field of all subsets of assignments of $\mathbf{X}$. If K is a credal set of joint probability distributions of $\mathbf{X}$, we define, for any two subsets $\mathbf{X}_S$ and $\mathbf{X}_R$, and assignment $\mathbf{x}_R$, the conditional credal set of $\mathbf{X}_s$ given $\mathbf{X}_R = \mathbf{x}_R$ induced from K as

$$\mathrm{K}(\mathbf{X}_S|\mathbf{x}_R) = \left\{ p' \in V(\mathbf{X}_S) : p'(\mathbf{x}_S) = \mathrm{P}_p(\mathbf{x}_S|\mathbf{x}_R), \mathrm{P}_p(\mathbf{x}_R) > 0, p \in \mathrm{K}(\mathbf{X}) \right\}. \qquad (4.3)$$

In other words, $\mathrm{K}(\mathbf{X}_S|\mathbf{x}_R)$ is the set of conditional distributions obtained by applying the standard definition of conditional probability to every joint distribution p in $K(\mathbf{X})$ whenever that operation is well-defined (i.e., whenever it assigns positive probability to the event $\mathbf{x}_R$). We say that a set of variables $\mathbf{X}_R$ is *strongly independent* of a set of variables $\mathbf{X}_S$ given variables $\mathbf{X}_T$ if $\mathbf{X}_S$ and $\mathbf{X}_R$ are stochastically independent conditional on $\mathbf{X}_T$ under every extreme distribution $\mathrm{P} \in \operatorname{ext}\mathrm{K}(\mathbf{X})$, which implies for every $\mathbf{x}_R$ and $\mathbf{x}_T$ that $\mathrm{K}(\mathbf{X}_S|\mathbf{x}_R, \mathbf{x}_T) = \mathrm{K}(\mathbf{x}_S|\mathbf{x}_T)$. We say that a set of variables $\mathbf{X}_R$ is *epistemically irrelevant* to a set of variables $\mathbf{X}_S$ conditional on variables $\mathbf{X}_T$ if $K(\mathbf{X}_S|\mathbf{x}_R, \mathbf{x}_T) = \mathrm{K}(\mathbf{X}_S|\mathbf{x}_T)$ for all assignments of $\mathbf{x}_R$ and $\mathbf{x}_T$. Hence, strong independence implies epistemic irrelevance (and the converse is not necessarily true) [Cozman, 2000; de Cooman and Troffaes, 2004]. Variables $\mathbf{X}_S$ and $\mathbf{X}_R$ are *epistemically independent* conditional on $\mathbf{X}_T$ if, given any assignment $\mathbf{x}_T$, $\mathbf{X}_R$ and $\mathbf{X}_S$ are epistemically irrelevant to each other [Walley, 1991, Ch. 9].

The *strong extension* of a credal network $(\mathbf{X}, G, M)$ is the largest credal set K_S of distributions of $\mathbf{X}$ that satisfies the strong independence assessments in G (viz. that every variable is strongly independent of its non-descendant non-parents given its parents), and whose induced conditional credal sets $\mathrm{K}_S(X_i|\mathbf{x}_{\mathrm{Pa}(i)})$ equal the local credal set $M(X_i|\mathbf{x}_{\mathrm{Pa}(i)})$ for any X_i and $\mathbf{x}_{\mathrm{Pa}(i)}$. The strong extension can be equivalently defined as

$$\mathrm{K}_S = \operatorname{co}\left\{ p \in V(\mathbf{X}) : p(\mathbf{x}) = \prod_{i \in N} p_i^{\mathbf{x}_{\mathrm{Pa}(i)}}(x_i),\ p_i^{\mathbf{x}_{\mathrm{Pa}(i)}} \in \operatorname{ext} M(X_i|\mathbf{x}_{\mathrm{Pa}(i)}) \right\}. \qquad (4.4)$$

By generating a probability potential for every possible combination of extreme distributions of every local credal set associated to a variable, a vertex-based,

separately specified credal network can be reduced to an extensively specified credal network that induces the same strong extension. Such a transformation preserves the topology of the network, but takes time exponential in the number of parents. On the other hand, any extensively specified credal network can be efficiently reduced to a separately specified credal network that induces the same strong extension, although the reduction inserts new root nodes which causes trees to be mapped into polytrees [Antonucci and Zaffalon, 2006, 2008]. Also, any vertex-based (separately specified) credal network can be efficiently reduced to a constraint-based network that induces the same strong extension, but the reduction inserts loops (i.e., cycles in the subjacent undirected graph) in the network.[3] It is unclear whether constraint-based networks can be efficiently reduced to vertex-based form by inserting new variables, but we conjecture that this is true.

The *epistemic extension* of a credal network is the largest joint credal set $\mathrm{K}_E(\mathbf{x})$ that satisfies the epistemic irrelevance assessments in G (viz. the non-descendant non-parents are irrelevant to a variable given its parents), and whose induced local credal set sets agree with the local credal sets in M. Equivalently, the epistemic extension is the credal set K_E such that

$$\mathrm{K}_E(X_i|\mathbf{x}_{\mathrm{Nd}(i)}) = M(X_i|\mathbf{x}_{\mathrm{Pa}(i)}), \tag{4.5}$$

for every variable X_i and assignment $\mathbf{x}_{\mathrm{Nd}(i)}$ of $\mathbf{X}_{\mathrm{Nd}(i)}$. Equation 4.5 is equivalent to the constraints

$$\sum_{x_i} f(x_i)\mathrm{P}_p(x_i|\mathbf{x}_{\mathrm{Nd}(i)}) \geq \min \sum_{x_i} f(x_i)q(x_i|\mathbf{x}_{\mathrm{Pa}(i)}), \tag{4.6}$$

for all functions f of X_i, assignment $\mathbf{x}_{\mathrm{Nd}(i)}$, and distribution $p \in \mathrm{K}_E$ with positive $\mathrm{P}_p(\mathbf{x}_{\mathrm{Nd}(i)})$, where the minimization is performed over $q \in M(X_i|\mathbf{x}_{\mathrm{Pa}(i)})$. Note that these inequalities can be turned into linear inequalities of the form (4.2) by multiplying both sides by $\mathrm{P}_p(\mathbf{x}_{\mathrm{Nd}(i)})$ and rearranging terms.

Example 4.4. *Consider the network in Example 4.2, and represent a function f of a binary variable as the pair $(f(0), f(1))$. The strong extension K_S is the*

[3] Let X_i be a variable whose local credal set $M(X_i|\mathbf{x}_{\mathrm{Pa}(i)})$ is specified by the extreme distributions $p_1, \ldots, p_m$, for a given configuration of the parents. Insert a new vacuous variable X_α taking values in $[m]$, and with X_i as its child and $\mathbf{X}_{\mathrm{Pa}(i)}$ as its parents, and redefine $M(X_i|\mathbf{x}_{\mathrm{Pa}(i)})$ as the singleton that contains the conditional distribution $p(x_i|\mathbf{x}_{\mathrm{Pa}(i)}, x_\alpha = k) = p_k(x_i)$. One can verify that the strong extension of the new network after marginalizing X_α coincides with the original strong extension.

credal set whose extreme distributions are the four joint probability distributions $p \in V(X_1, X_2, X_3)$ *such that*

$$p(x_1, x_2, x_3) = p_1(x_1)p_2(x_2)p_3^{x_1x_2}(x_3) \quad [x_1, x_2, x_3 = 0, 1]\,,$$

where

$$\begin{aligned} &p_1 \in \{(0.4, 0.6), (0.5, 0.5)\}\,, && p_2 \in \{(0.4, 0.6), (0.5, 0.5)\}\,, \\ &p_3^{00} = p_3^{11} = (1, 0)\,, && p_3^{01} = p_3^{10} = (0, 1)\,. \end{aligned}$$

The epistemic extension K_E *is the set of joint probability distributions* $p \in V(X_1, X_2, X_3)$ *that satisfy the system of linear inequalities*

$$\begin{aligned} &0.5 = \min_{q \in M(X_1)} q(X_1 = 1) \leq \mathrm{P}_p(X_1 = 1 | x_2) \leq \max_{q \in M(X_1)} q(X_1 = 1) = 0.6 \quad [x_2 = 0, 1]\,, \\ &0.5 = \min_{q \in M(X_2)} q(X_2 = 1) \leq \mathrm{P}_p(X_2 = 1 | x_1) \leq \max_{q \in M(X_2)} q(X_2 = 1) = 0.6 \quad [x_1 = 0, 1]\,, \\ &\mathrm{P}_p(X_3 = 1 | X_1 = x, X_2 = x) = 0 \qquad\qquad [x = 0, 1]\,, \\ &\mathrm{P}_p(X_3 = 1 | X_1 = 0, X_2 = 1) = \mathrm{P}_p(X_3 = 1 | X_1 = 1, X_2 = 0) = 1\,. \end{aligned}$$

One can verify that the set K_E *has six extreme distributions, of which two are not in the strong extension.* □

The example above shows an interesting and well-known relation between epistemic and strong extensions, namely, that the latter is always contained in the former, and thus produces more precise results [Walley, 1991, Chapter 9.2].

4.2 Belief updating

A credal network can be seen as a complete (although imprecise) quantification of the decision maker's beliefs about certain scenarios involving local domains, specifically, about the likely value of any variable conditional on its parents. These beliefs are quantified by the local credal sets that specify the network. A typical application of probabilistic reasoning is to prescribe what the beliefs about scenarios other than those already specified ought to be if the decision maker is to behave rationally. This inferential task is generally called *belief updating*. In precise models (e.g. Bayesian networks), it is performed by applying Bayes' rule with likelihood and prior distributions induced by the single joint probability distribution that is consistent with the network constraints. The *generalized Bayes' rule* (GBR) extends this inference to credal networks. Let X_q be a

variable of interest, f be a function of X_q, and $\mathbf{X}_O$ be a set of evidence variables which we know to be $\tilde{\mathbf{x}}_O$. Then the GBR is the solution μ of the identity

$$\min_{p\in \mathrm{K}} \sum_{\mathbf{x}:\mathbf{x}_O=\tilde{\mathbf{x}}_O} \Big(f(x_q)-\mu\Big)\, p(\mathbf{x}) = 0\,, \tag{4.7}$$

where K is either the epistemic or strong extension of the network. Assuming that $\min_{p\in \mathrm{K}(\mathbf{X}_O)} p(\tilde{\mathbf{x}}_O) > 0$, it follows that

$$\mu = \mathrm{L}_{\mathrm{K}(X_q|\mathbf{X}_O=\tilde{\mathbf{x}}_O)}(f)\,, \tag{4.8}$$

that is, μ is the lower posterior expectation of f with respect to the (strong or epistemic) extension of the network. In particular, when K includes a single distribution, the GBR is equivalent to Bayes' rule. For the rest of this chapter, we assume that the lower probability of the evidence is positive whenever the GBR is applied. For a recent treatment of the zero probability case, see Ref. [de Bock and de Cooman, 2013].

Example 4.5. *Consider again the network in Example 4.2, and assume that $X_q = X_3$, $f = (1,0)$ and $\mathbf{X}_O$ is the empty set. Then applying the GBR is equivalent to finding the lower marginal probability $\mu = \min_{p\in \mathrm{K}(X_3)} p(0)$ induced by some network extension. Assuming strong independence (hence considering the strong extension K_S in Example 4.4), the outcome of the GBR is*

$$\begin{aligned}\mu &= \min \sum_{x_1,x_2} p_1(x_1)p_2(x_2)p_3^{x_1x_2}(0)\\ &= 1+\min\{2p_1(0)p_2(0)-p_1(0)-p_2(0)\}\\ &= 1-1/2 = 1/2\,,\end{aligned}$$

where the minimizations are performed over p_1 and p_2. The outcome of the GBR under epistemic irrelevance is the value of the solution of the linear program $\mu = \min\{p(0,0,0)+p(1,1,0) : p \in \mathrm{K}_E\} = 5/11 < 1/2$, where K_E is the epistemic extension defined in Example 4.4. □

The fact that the outcome of the GBR under epistemic irrelevance in the example above is smaller than under strong independence is a direct consequence of the fact that the strong extension is contained in the epistemic extension.

Computing the GBR is notoriously a hard task, whose complexity strongly depends on the topology of the DAG, the cardinality of the variable domains, and the irrelevance concept adopted. Cozman et al. [2004] showed that under strong independence, the problem is $\mathrm{NP}^{\mathrm{PP}}$-hard. De Campos and Cozman

Table 4.1. Parametrized complexity of the GBR.

MODEL	STRONG	EPISTEMIC
Imprecise HMM (query on last node)	P	P
Imprecise HMMs	Unknown	P
Credal trees	NP-hard	P
Credal polytrees with binary variables	P	Unknown
Credal polytrees with ternary variables	NP-hard	NP-hard
Bounded treewidth networks	NP-hard	NP-hard
Credal networks	NP^{PP}-hard	NP^{PP}-hard

[2005] studied the fixed-parameter complexity under strong independence and concluded that the problem is NP-hard even on polytree-shaped networks of bounded treewidth. We show here that the problem remains NP-hard in polytree-shaped credal networks if we constraint variables to take on at most three values. When instead epistemic irrelevance is assumed, no polynomial-time algorithm for the task is known. Indeed, we show later on this chapter that a polynomial-time algorithm for this case implies that P equals NP, and is therefore unlikely. Under strong independence, a long-known positive result is the 2U algorithm of Fagiuoli and Zaffalon [1998b], which solves the problem in polynomial time if the DAG is a polytree and all variables are binary. A positive result under epistemic irrelevance was more recently given by de Cooman et al. [2010], who developed a polynomial-time algorithm for GBR computations in credal trees. No analogous algorithm is known to exist under strong independence, and in fact we show here that also this problem variant is NP-hard. A credal hidden Markov model (HMM) is a particular type of tree-shaped credal network commonly used to represent time-dependent processes. Since an HMM is a tree, the GBR can be computed efficiently under epistemic irrelevance. It remains unknown whether the same task is NP-hard under strong independence. However, we show here that at least for a particular type of inference (viz. when there is no node succeeding the queried variable in the topological ordering), the GBR under strong independence can be computed efficiently in HMMs. These results are summarized in Table 4.1.

In terms of approximate results, de Campos and Cozman [2005] showed that computing provably good approximations under strong independence is NP-hard, even if we consider only credal polytrees of bounded treewidth. Recently, we showed the existence of a fully polynomial-time approximation scheme for

credal networks of bounded treewidth and bounded variable cardinality under strong independence [Mauá et al., 2012b]. It is still unknown whether an analogous result can be obtained under epistemic irrelevance. In spite of such negative results of the problem, several algorithms have been developed that either sacrifice accuracy or runtime performance. Under strong independence, the A/R algorithm of Tessem [1992] extends Pearl's belief propagation algorithm for Bayesian networks to allow probability intervals to be propagated, which led to an efficient but highly inaccurate algorithm. Cano et al. [1994] and Cano and Moral [1996] used greedy approaches to solve the problem that sample extreme distributions from the local credal sets using standard combinatorial optimization techniques. De Campos and Cozman [2004] reduced the computation of the GBR under strong independence to a multilinear program, which was then solved by a custom-made software. Their algorithm runs in exponential-time in the worst case, but often finds the solution in feasible time, and can be stopped with an approximate answer at any time. Da Rocha and Cozman [2005] improved on the A/R algorithm, giving rise to the A/R+ algorithm, and combined its improved version with local search methods to reach an exact solution (in exponential time in the worst case) by branch-and-bound. Cano et al. [2007] used probability trees and local search to arrive at a branch-and-bound procedure. De Campos and Cozman [2007b] showed that it is possible to efficiently reduce the computation of the GBR in bounded treewidth networks to a mixed integer linear program, which can then be solved using standard solvers. As with any mixed integer linear program, the solver can be run until a desired accuracy is achieved (which might take exponential time), and stopped at any moment to produce a solution within known error bounds (which can be arbitrarily loose for any sub-exponential time interval). The GL2U algorithm [Antonucci et al., 2010] generalizes 2U to multiply connected networks with non-binary variables, and runs in polynomial time on any network (i.e., its running time does not depend on the network treewidth). GL2U's efficiency comes at the expense of accuracy in the results, as the algorithm does not provide any guarantees on the quality of the solutions it returns. Recently, Antonucci et al. [2013b] devised an algorithm that computes the GBR by solving a sequence of linear programs obtained by fixing the local distributions of all but one variable in the network. They report significant improvements in accuracy with respect to GL2U and the iterated local search devised by da Rocha and Cozman [2005].

Not so many algorithms have been developed for epistemic irrelevance. We have already mentioned the exact polynomial-time algorithm for tree-shaped networks of de Cooman et al. [2010]. De Bock and de Cooman [2011] developed an algorithm to compute a particular type of the GBR inference with the

intent of finding the maximal assignments of the hidden variables in imprecise hidden Markov models under epistemic irrelevance (these models are special types of tree-shaped credal networks defined in Section 4.3.2). Perhaps the only practical algorithm for networks with structure more complex than trees is the one developed by de Campos and Cozman [2007a], which recasts the computation of the GBR under epistemic irrelevance as a multilinear program. Their method is able to cope with mixed epistemic and strong irrelevance assessments. On the other hand, the method comes with no performance guarantees, which is not surprising in the light of the NP-hardness results we show later on.

4.3 Parametrized complexity of the GBR

In this section, we study the complexity of computing the GBR in credal networks with different assumptions about the network topology and the variable domain cardinality. We start with a result that shows the equivalence between epistemic irrelevance and strong independence is a special class of networks. This equivalence is important as it allows us to use known results about the hardness of computing the GBR under strong independence to derive hardness of the GBR under epistemic irrelevance.

Proposition 4.1. *Consider a credal network whose root nodes are vacuous and non-root nodes are precise. Then the result of the GBR for an arbitrary function f of a variable X_q associated to a non-root node q and no evidence is the same whether we assume epistemic irrelevance or strong independence.*

Proof. Let $\mathbf{X}_R$ be the vacuous variables associated to root nodes (hence to vacuous local credal sets), and $\mathbf{X}_I$ denote the remaining variables (which are associated to singleton local credal sets). For every precise node i in I, let $p_i^{\mathbf{x}_{\mathrm{Pa}(i)}}(x_i)$ be the single distribution in the associated credal set. Consider an arbitrary distribution p in the epistemic extension K_E, and let $<$ be a topological ordering of the nodes. For every node i the set $\{j \in N : j < i\}$ is a subset of $\mathrm{Nd}(i)$, and it follows from the definition of epistemic extension that $\mathrm{P}_p(x_i|\mathbf{x}_{j<i}) = p_i^{\mathbf{x}_{\mathrm{Pa}(i)}}(x_i)$ for every precise node i and assignments x_i and $\mathbf{x}_{j<i}$. By the chain rule of probability we have that

$$(\forall \mathbf{x}) \quad \mathrm{P}_p(\mathbf{x}) = \mathrm{P}_p(\mathbf{x}_R) \prod_{i \in I} \mathrm{P}_p(x_i|\mathbf{x}_{j<i}) = q(\mathbf{x}_R) \prod_{i \in I} p_i^{\mathbf{x}_{\mathrm{Pa}(i)}}(x_i),$$

where q is any distribution of $\mathbf{X}_R$ (since these nodes are vacuous, any distribution satisfies the constraints in K_E for them). The result of the GBR under epistemic

irrelevance is thus given by

$$\begin{aligned}\mu = \min_{p \in \mathrm{K}_E} \mathrm{E}_p(f) &= \min_{q \in V(\mathbf{X}_R)} \sum_{\mathbf{x}} q(\mathbf{x}_R) \prod_{i \in I} p_i^{\mathbf{x}_{\mathrm{Pa}(i)}}(x_i) f(x_q) \\ &= \min_{q \in V(\mathbf{X}_R)} \sum_{\mathbf{x}_R} q(\mathbf{x}_R) \sum_{\mathbf{x}_I} \prod_{i \in I} p_i^{\mathbf{x}_{\mathrm{Pa}(i)}}(x_i) f(x_q) \\ &= \min_{q \in V(\mathbf{X}_R)} \sum_{\mathbf{x}_R} q(\mathbf{x}_R) \mathrm{E}_{p(\mathbf{X}_I|\mathbf{x}_R)}(f|\mathbf{x}_R),\end{aligned}$$

where $p(\mathbf{x}_I|\mathbf{x}_R) \stackrel{\text{def}}{=} \prod_{i \in I} p_i^{\mathbf{x}_{\mathrm{Pa}(i)}}(x_i)$. According to the last equality, μ is a convex combination of $\mathrm{E}_{p(\mathbf{X}_I|\mathbf{x}_R)}(f|\mathbf{x}_R)$ (which is a function of $\mathbf{x}_R$ only). Hence,

$$\mu \geq \min_{\mathbf{x}_R} g(\mathbf{x}_R) = \min_{\mathbf{x}_R} \sum_{\mathbf{x}_I} \prod_{i \in I} p_i^{\mathbf{x}_{\mathrm{Pa}(i)}}(x_i) f(x_q).$$

The rightmost minimization is exactly the value of the GBR under strong independence, and since the strong extension is contained in the epistemic extension, the inequality above is tight. □

The class of networks considered in the result above might seem restrictive at first sight. However, Antonucci and Zaffalon [2008] showed that the computation of the GBR for any credal network of bounded treewidth whose local credal sets are represented in vertex-based form can be reduced in polynomial time to the computation of the GBR in a credal network whose non-root nodes are all precise and whose imprecise variables are all vacuous. The hardness of the GBR under epistemic irrelevance in such credal networks follows immediately from the result above, since the same is true for under strong independence, and one can efficiently reduce one problem to another.

Corollary 4.1. *Computing the GBR under epistemic irrelevance is NP^{PP}-hard.*

Proof. Cozman et al. [2004] used a reduction from E-MAJSTAT to show that the computing the GBR under strong independence in a credal network whose root nodes are vacuous and non-root nodes are precise is NP$^{\mathrm{PP}}$-hard. Since according to Proposition 4.1, the result of the GBR is the same under epistemic irrelevance, the result holds. □

Note that the result holds irrespective of how the local credal sets are represented, since vacuous and precise nodes can be mapped from constraint-based to vertex-based form in polynomial time (and vice-versa).

Another direct consequence of Proposition 4.1 is the NP-hardness of the GBR in singly connected credal networks under epistemic irrelevance, as the same

task is NP-hard under strong independence, even if we admit imprecision only on root nodes. The proof of NP-hardness of the GBR under strong independence provided by de Campos and Cozman [2005] requires the variable domain cardinalities to be unbounded. The next result presents a stronger result in that we admit only networks where imprecise variables are binary and precise ones are at most ternary.

Theorem 4.1. *Computing the GBR is NP-hard whether we assume epistemic irrelevance or strong independence, even if the network is singly connected and has treewidth at most two, all imprecise variables are binary, and all precise variables are (at most) ternary.*

Proof. Consider a minimal singly connected LIMID over binary action variables, ternary state variables, and a single value variable X_r. By Theorem 3.3, deciding whether the maximum expected utility of any strategy exceeds a given threshold is an NP-complete task. As the LIMID is minimal, action nodes have no parents. Obtain a credal network from that LIMID by discarding the value node, replacing the action nodes with vacuous nodes, and leaving the rest unchanged (i.e., converting state variables into precise variables). The network can be obtained in time linear in the size of the diagram, as the precise nodes are represented in the same way, and the representation of vacuous binary nodes requires either two degenerate distributions in vertex-base form or two vacuous constraints in constraint-based form. Note that the credal network is polytree-shaped, contains only binary imprecise nodes and ternary precise nodes. Let f be the function of X_{s_n} that returns -1 if $X_{s_n}=1$ and zero otherwise. The outcome of the GBR is

$$\min_{p\in \mathrm{K}(X_q)} \mathrm{E}_p(f) = -\max_{p\in \mathrm{K}(X_q)} \mathrm{E}_p(-f) = \min_{\Delta} \mathrm{E}_\Delta(X_r),$$

where K is either the epistemic or the strong extension, and the rightmost expression is the maximum expected utility of the LIMID. Thus, computing the GBR decides the maximum expected utility problem. □

4.3.1 Credal trees

The previous complexity results showed that, from a theoretical standpoint, computing the GBR under epistemic irrelevance is just as difficult as under strong independence. When the DAG is a tree, de Cooman et al. [2010] showed that the GBR can be computed efficiently under epistemic irrelevance, and it was previously unknown whether a similar result could be obtained under strong independence. We shall show that in this case the equivalence on the tractability

under the two different irrelevance concepts is highly unlikely. To accomplish this, we first need to obtain a result concerning networks whose numerical parameters are more generally given by *(polynomial-time) computable* numbers, which might not all be encodable trivially as rational numbers. A number r is computable if there exists a Turing machine M_r that, for input b, runs in at most time $\text{poly}(b)$ (the notation $\text{poly}(b)$ denotes an arbitrary polynomial function of b) and outputs a rational number t such that $|r - t| < 2^{-b}$. Of special relevance are numbers of the form $2^{t_1}/(1 + 2^{t_2})$, with $|t_1|, |t_2|$ being rationals no greater than two, for which we can build a machine that outputs a rational t with the necessary precision in time $\text{poly}(b)$ as follows: compute the Taylor expansions of 2^{t_1} and 2^{t_2} around zero with sufficiently many terms (depending on the value of b), and then compute the fractional expression. The following lemma ensures that, assuming strong independence, the result of the GBR on any function computed with respect to a network specified with computable numbers can be approximated arbitrarily well by computing the GBR using a network specified only with rational numbers.

Lemma 4.1. *Consider a vertex-based credal network $\mathcal{N}$ whose numerical parameters are specified with computable numbers encoded by their respective machines, and let b be the size of the encoding of the network. Given any rational number $\varepsilon \geq 2^{-\text{poly}(b)}$, we can construct in time* $\text{poly}(b)$ *a vertex-based credal network $\mathcal{N}'$ over the same variables whose numerical parameters are all rational numbers, and such that there is a polynomial-time computable bijection (p, p') that associates any extreme p of the strong extension of $\mathcal{N}$ with an extreme p' of the strong extension of $\mathcal{N}'$ satisfying*

$$\max_{\mathbf{x}_S} |\mathrm{P}_{p'}(\mathbf{x}_S) - \mathrm{P}_p(\mathbf{x}_S)| \leq \varepsilon\,,$$

for any subset $\mathbf{X}_S$ of the variables.

Proof. Take $\mathcal{N}'$ to be equal to $\mathcal{N}$ except that each computable number r used in the specification of $\mathcal{N}$ is replaced by a rational t such that $|t - r| < 2^{-(n+1)(v+1)}\varepsilon$, where n is the number of variables, and v is the maximum cardinality of the domain of any variable in $\mathcal{N}$. Because $\varepsilon \geq 2^{-\text{poly}(b)}$, we can run the Turing machine M_r used to represent r on input $\text{poly}(b) + (n + 1)(v + 1)$ to obtain t in time $O(\text{poly}(\text{poly}(b) + (n + 1)(v + 1))) = O(\text{poly}(b))$. Exactly one of the probability values in each distribution used to represent an extreme of a local credal set in $\mathcal{N}'$ is computed as one minus the sum of the other numbers to ensure that the distribution adds up exactly to one; its error is at most $(v - 1) \cdot 2^{-(n+1)(v+1)}\varepsilon < 2^{-n(v+1)}\varepsilon$.

Let $q_i(x_i|\mathbf{x}_{\mathrm{Pa}(i)})$ and $q'_i(x_i|\mathbf{x}_{\mathrm{Pa}(i)})$ denote, respectively, the parameters of $\mathcal{N}$ and $\mathcal{N}'$ (i.e. they are corresponding extreme distributions of the local credal sets $M(X_i|\mathbf{x}_{\mathrm{Pa}(i)})$ in the two networks), and consider an assignment $\mathbf{x}$ to all variables in $\mathcal{N}$ (or in $\mathcal{N}'$). Let also p be an extreme of the strong extension of $\mathcal{N}$. Then p factorizes as $p(\mathbf{x}) = \prod_{i\in N} q_i(x_i|\mathbf{x}_{\mathrm{Pa}(i)})$, for some combination of extreme distributions $q_i(\cdot|\mathbf{x}_{\mathrm{Pa}(i)})$ from $M(X_i|\mathbf{x}_{\mathrm{Pa}(i)})$, $i \in N$. Finally, let p' be an extreme distribution in the strong extension of $\mathcal{N}'$ that satisfies $p'(\mathbf{x}) = \prod_{i\in N} q'_i(x_i|\mathbf{x}_{\mathrm{Pa}(i)})$. By design, $|q'_i(x_i|\mathbf{x}_{\mathrm{Pa}(i)}) - q_i(x_i|\mathbf{x}_{\mathrm{Pa}(i)})| \leq 2^{-n(v+1)}\varepsilon$. It follows from the binomial expansion of the factorization of $p'(\mathbf{x})$ on any $\mathbf{x}$ that

$$\begin{aligned} p'(\mathbf{x}) = \prod_{i\in N} q'_i(x_i|\mathbf{x}_{\mathrm{Pa}(i)}) &\leq \prod_{i\in N} \left(2^{-n(v+1)}\varepsilon + q_i(x_i|\mathbf{x}_{\mathrm{Pa}(i)})\right) \\ &= \sum_{S\subseteq N} \prod_{i\in S} q_i(x_i|\mathbf{x}_{\mathrm{Pa}(i)})(2^{-n-vn}\varepsilon)^{n-|S|} \\ &\leq 2^n 2^{-n-vn}\varepsilon + \prod_{i\in N} q_i(x_i|\mathbf{x}_{\mathrm{Pa}(i)}) \\ &= p(\mathbf{x}) + 2^{-nv}\varepsilon\,. \end{aligned}$$

The second inequality follows from the fact that there is one term for $p(\mathbf{x})$ in the expansion and $2^n - 1$ terms that can be written as a product of $2^{-n(v+1)}\varepsilon$ by nonnegative numbers less than or equal to one. With a similar reasoning, we can show that

$$p'(\mathbf{x}) \geq \prod_{i=1}^{n} \left(q_i(x_i|\mathbf{x}_{\mathrm{Pa}(i)}) - 2^{-n(v+1)}\varepsilon\right) \geq p(\mathbf{x}) - 2^{-nv}\varepsilon\,.$$

Thus, $\max_{\mathbf{x}} |p'(\mathbf{x}) - p(\mathbf{x})| \leq 2^{-nv}\varepsilon$. Now consider a subset of the variables $\mathbf{X}_S$ and an assignment $\mathbf{x}_S$ to $\mathbf{X}_S$. Since $\mathrm{P}_{p'}(\mathbf{x}_S) \stackrel{\text{def}}{=} \sum_{\mathbf{x}':\mathbf{x}'_S=\mathbf{x}_S} p'(\mathbf{x}')$, each term $p'(\mathbf{x}')$ in that sum satisfies $p'(\mathbf{x}') \leq p(\mathbf{x}') + 2^{-nv}\varepsilon$, and there are less than $v^n \leq 2^{vn}$ terms being summed, we have that

$$\mathrm{P}_{p'}(\mathbf{x}_S) \leq \sum_{\mathbf{x}':\mathbf{x}'_S=\mathbf{x}_S} \left(p(\mathbf{x}) + 2^{-vn}\varepsilon\right) \leq \mathrm{P}_p(\mathbf{x}_S) + \varepsilon\,.$$

An analogous argument can be used to show that $\mathrm{P}_{p'}(\mathbf{x}_S) \geq \mathrm{P}_p(\mathbf{x}_S) - \varepsilon$. □

The above lemma has the following direct consequence on the computation of the GBR.

Corollary 4.2. *Consider a vertex-based credal network $\mathcal{N}$ whose numerical parameters are specified with computable numbers encoded by their respective machines,*

and let b be the size of the encoding of the network. Given any rational number ε with $2^{-\text{poly}(b)} \leq \varepsilon < 1$, evidence $\mathbf{X}_O = \mathbf{x}_O$ on some set of variables $\mathbf{X}_O$ in $\mathcal{N}$, and function f of a variable X_q in $\mathcal{N}$ such that $\max_{x_q} |f(x_q)| \leq 1$, we can construct in time $\text{poly}(b)$ *a vertex-based credal network $\mathcal{N}'$ over the same variables whose numerical parameters are all rational numbers, and such that*

$$|\mu' - \mu| \leq \varepsilon\,,$$

where μ and μ' are the outcomes of the GBR on $\mathcal{N}$ and $\mathcal{N}'$, respectively.

Proof. According to Lemma 4.1, there is a polynomial-time computable network $\mathcal{N}'$ whose parameters are rational numbers and a polynomial-time computable bijection (p, p') such that p and p' are, respectively, extreme distributions of the strong extension of $\mathcal{N}$ and $\mathcal{N}'$, and satisfy $|\mathrm{P}_{p'}(\mathbf{x}_S) - \mathrm{P}_p(\mathbf{x}_S)| \leq r^n\varepsilon^2/(2v)$ for all $\mathbf{X}_S \subseteq \mathbf{X}$ and $\mathbf{x}_S \sim \mathbf{X}_S$, where $n \stackrel{\text{def}}{=} |\mathbf{X}|$ is the number of variables, v is the cardinality of variable X_q ($v \geq 2$), and r is the smallest strictly positive rational number in the specification of $\mathcal{N}'$. It follows that

$$\mathrm{P}_p(x_q|\mathbf{x}_O) = \frac{\mathrm{P}_p(x_q, \mathbf{x}_O)}{\mathrm{P}_p(\mathbf{x}_O)} \geq \frac{\mathrm{P}_{p'}(x_q, \mathbf{x}_O) - r^n\varepsilon^2/(2v)}{\mathrm{P}_{p'}(\mathbf{x}_O) + r^n\epsilon^2/(2v)}\,,$$

where p' is the image of p according to the bijection. By Lemma 7 of [de Campos and Cozman, 2013], we have that

$$\frac{\mathrm{P}_{p'}(x_q, \mathbf{x}_O) - r^n\varepsilon^2/(2v)}{\mathrm{P}_{p'}(\mathbf{x}_O) + r^n\varepsilon^2/(2v)} \geq \mathrm{P}_{p'}(x_q|\mathbf{x}_O) - 2\frac{r^n\varepsilon^2/(2v)}{r^n\varepsilon/4} = \mathrm{P}_{p'}(x_q|\mathbf{x}_O) - \varepsilon/v\,.$$

The other side of the inequality is obtained by switching p and p' in the inequalities above. Hence, $|\mathrm{P}_{p'}(x_q|\mathbf{x}_O) - \mathrm{P}_p(x_q|\mathbf{x}_O)| \leq \varepsilon/v$. Let p be an extreme distribution in the strong extension of $\mathcal{N}$ for which $\mu = \sum_{x_q} f(x_q)\mathrm{P}_p(x_q|\mathbf{x}_0)$, that is, p is a distribution that attains $L_{\mathrm{K}_S(X_q|\mathbf{x}_O)}(f)$. Define $f^+(x_q) = \max\{f(x_q), 0\}$ and $f^-(x_q) = \min\{f(x_q), 0\}$. Thus,

$$\begin{aligned}
\mu &= \sum_{x_q} f^+(x_q)\mathrm{P}_p(x_q|\mathbf{x}_O) + \sum_{x_q} f^-(x_q)\mathrm{P}_p(x_q|\mathbf{x}_O) \\
&\geq \sum_{x_q} f^+(x_q)\left(\mathrm{P}_{p'}(x_q|\mathbf{x}_O) - \varepsilon/v\right) + \sum_{x_q} f^-(x_q)\left(\mathrm{P}_{p'}(x_q|\mathbf{x}_O) + \varepsilon/v\right) \\
&= -\frac{\varepsilon}{v}\sum_{x_q}|f(x_q)| + \sum_{x_q} f(x_q)\mathrm{P}_{p'}(x_q|\mathbf{x}_O) \\
&\geq -\varepsilon + \min_{p'' \in K'_S(X_q|\mathbf{x}_o)} \sum_{x_q} f(x_q)\mathrm{P}_{p''}(x_q|\mathbf{x}_O) = -\varepsilon + \mu'\,,
\end{aligned}$$

where p' in the first inequality is the image of p' according to the bijection, and K'_S in the last inequality is the strong extension of $\mathcal{N}'$. It follows that $\mu' - \mu \leq \varepsilon$. If p is instead the extreme distribution that attains $L_{K'_S(X_q|\mathbf{x}_O)}(f)$, then the same argument shows that $\mu - \mu' \leq \varepsilon$, whence the result follows. □

We can now use the result above to show the NP-hardness of the GBR in credal trees under strong independence.

Theorem 4.2. *Computing the GBR in credal trees under strong extension is NP-hard, even if only one variable is ternary and all the rest are binary.*

Proof. We show hardness by a reduction from the *partition problem*, which is the NP-hard problem of deciding, given positive integers $z_1, \ldots, z_n$, whether there is $S \subseteq N \stackrel{\text{def}}{=} \{1, \ldots, n\}$ such that $\sum_{i \in S} z_i = \sum_{i \notin S} z_i$, where the notation $i \notin S$ denotes that $i \in N \setminus S$. We define $v_i \stackrel{\text{def}}{=} z_i / z$, $i = 1, \ldots, n$, where $z \stackrel{\text{def}}{=} \sum_i z_i / 2$, and work w.l.o.g. with the partition problem using v_i instead of z_i. Let $v_S \stackrel{\text{def}}{=} \sum_{i \in S} v_i$. Then, it follows for any S that $v_S = 2 - \sum_{i \notin S} v_i$. Also, if an instance of the partition problem is a *yes-instance*, there is S for which $v_S = 1$, whereas if it is a *no-instance*, then for any S, it follows that $|v_S - 1| \geq 1/(2z)$. Consider the function

$$h(v_S) = \frac{2^{-(v_S - 1)} + 2^{v_S - 1}}{2}.$$

Seen as a function of a continuous variable $v_S \in [0, 2]$, the function above is strictly convex, symmetric around one, and achieves the minimum value of one at $v_S = 1$. Thus, if the partition problem is a yes-instance, then $\min_S h(v_S) = 1$, while if it is a no-instance, then $\min_S h(v_S) \geq 2^{-1/(2z)-1} + 2^{1/(2z)-1} \geq 2^{(2z)^{-4}} > 1 + (2z)^{-4}/2 = 1 + 1/(32z^4)$, where the second inequality is due to Lemma 24 in [Mauá et al., 2012], and the strict inequality follows from the first-order Taylor expansion of $2^{(2z)^{-4}}$.

Given an instance of the partition problem, we build a credal tree over variables $X_0, \ldots, X_{2n}$ with DAG as in Figure 4.3. The root variable X_0 takes values in $\{1, 2, 3\}$, and is precise and uniformly distributed (i.e., its local credal set contains only the distribution $p_0(x_0) = 1/3$). The remaining variables are all binary and take values in $\{0, 1\}$. For $i = 1, \ldots, n$, we specify the local conditional credal sets $M(X_i|x_0)$ as singletons $\{p_i^{x_0}\}$ such that

$$p_i^{x_0}(1) = \begin{cases} 2^{-v_i}/(1 + 2^{-v_i}), & \text{if } x_0 = 1, \\ 1/(1 + 2^{-v_i}), & \text{if } x_0 = 2, \\ 1/2, & \text{if } x_0 = 3. \end{cases}$$

Finally, for $i = 1+n, \ldots, 2n$ we specify for all x_{i-n} the local credal sets $M(X_i|x_{i-n}) = \{p \in V(X_i) : \epsilon \leq p(1) \leq 1\}$, where $\epsilon = 2^{-n-3}/(64z^4)$. Each of these local credal sets can be represented either in vertex-based form by two extreme distributions or by a single constraint.

Consider the computation of the GBR on the function f of X_0 that returns -1 if $X_0 = 3$ and zero otherwise, and with evidence $X_{n+1} = 1, \ldots, X_{2n} = 1$. By definition, any extreme distribution p in the strong extension K_s satisfies for $\mathbf{x}$ such that $x_{n+1}=1, \ldots, x_{2n}=1$ the equality

$$p(\mathbf{x}) = p_0(x_0) \prod_{i \in [n]} p_i^{x_0}(x_i) \delta_i^{x_i},$$

where each $\delta_i^{x_i}$ is in $[\epsilon, 1]$. Hence, the GBR returns the value of μ that solves

$$\min \sum_{x_0, \ldots, x_n} \left(f(x_0) - \mu\right) p_0(x_0) \prod_{i \in [n]} p_i^{x_0}(x_i) \delta_i^{x_i} = 0,$$

where the minimizations are performed on δ_i^0, δ_i^1, for $i = 1, \ldots, n$. Consider $j \in [n]$ and let

$$\beta_{x_j} \stackrel{\text{def}}{=} \sum_{x_0, \ldots, x_{j-1}, x_{j+1}, \ldots, x_n} \left(f(x_0) - \mu\right) p_0(x_0) p_j^{x_0}(x_j) \prod_{i \in [n], i \neq j} p_i^{x_0}(x_i) \delta_i^{x_i}.$$

Then,

$$\min \sum_{x_0, \ldots, x_n} \left(f(x_0) - \mu\right) p_0(x_0) \prod_{i \in [n]} p_i^{x_0}(x_i) \delta_i^{x_i} = \min \left(\delta_j^0 \beta_0 + \delta_j^1 \beta_1\right).$$

Since δ_j^0 and δ_j^1 are both positive, the minimization on the right-hand side above equals zero only if either β_0 and β_1 have different signs or both are zero. In the former case, we have that δ_j^0 and δ_j^1 are minimized at values such that $\delta_j^0 \neq \delta_j^1$, with $\delta_j^0 < \delta_j^1$ if and only if $\beta_0 > \beta_1$. In the latter case, any assignment to variables δ_j^0 and δ_j^1 minimizes the expression, and we can assume that $\delta_j^0 \neq \delta_j^1$. Since we selected j arbitrarily, the result holds for all j. Thus, the minimization is equivalent to selecting, for $i = 1, \ldots, n$, a value y_i in $\{0, 1\}$ such that $\delta_i^0 = \epsilon^{1-y_i}$ and $\delta_i^1 = \epsilon^{y_i}$. It follows that

$$\min \sum_{x_0, \ldots, x_n} \left(f(x_0) - \mu\right) p_0(x_0) \prod_{i \in [n]} p_i^{x_0}(x_i) \delta_i^{x_i} = \min_{\mathbf{y} \in \{0,1\}^n} \sum_{x_0, \ldots, x_n} \left(f(x_0) - \mu\right) p_0(x_0) \prod_{i \in [n]} p_i^{x_0}(x_i) \epsilon^{(1-x_i)(1-y_i)} \epsilon^{x_i y_i}.$$

By rearranging terms, we obtain

$$\min_{\mathbf{y}\in\{0,1\}^n} \sum_{x_0} \left(f(x_0)-\mu\right) p_0(x_0) \prod_{i\in[n]} \left(p_i^{x_0}(0)\epsilon^{1-y_i} + p_i^{x_0}(1)\epsilon^{y_i}\right),$$

which by design equals

$$\min_{\mathbf{y}\in\{0,1\}^n} -\left(\frac{\mu}{3}\prod_{i\in[n]}\frac{1}{1+2^{-\nu_i}}(\epsilon^{1-y_i}+2^{-\nu_i}\epsilon^{y_i}) + \frac{\mu}{3}\prod_{i\in[n]}\frac{1}{1+2^{-\nu_i}}(2^{-\nu_i}\epsilon^{1-y_i}+\epsilon^{y_i}) + \frac{1+\mu}{3}\prod_{i\in[n]}\frac{1}{2}(1+\epsilon)\right).$$

The binary vector $\mathbf{y}$ can be seen as the characteristic vector of a subset $S\subset[n]$. Define

$$b_S \stackrel{\text{def}}{=} \prod_{i\in S}(2^{-\nu_i}+\epsilon)\prod_{i\notin S}(1+2^{-\nu_i}\epsilon)$$

for every subset S of $[n]$. The optimization on $\mathbf{y}$ can be rewritten as the following optimization over subsets:

$$-\frac{1+\mu}{3}\left(\frac{1+\epsilon}{2}\right)^n + \min_{S\subseteq[n]} -\frac{\mu}{3}\left(b_S + b_{[n]\setminus S}\right)\prod_{i\in[n]}\frac{1}{1+2^{-\nu_i}}.$$

Solving the expression above for μ, we get to

$$\begin{aligned}\mu &= -\left(1+\left(\frac{2}{1+\epsilon}\right)^n \min_S \left(b_S + b_{[n]\setminus S}\right)\prod_{i\in[n]}\frac{1}{1+2^{-\nu_i}}\right)^{-1}\\ &= -\frac{1}{\min_S g(a_S)},\end{aligned}$$

where

$$g(a) \stackrel{\text{def}}{=} 1+(1+a)\left(\frac{2}{1+\epsilon}\right)^n\prod_{i\in[n]}\frac{1}{1+2^{-\nu_i}}$$

is defined for any real number a, and $a_S \stackrel{\text{def}}{=} b_S + b_{N\setminus S} - 1$ for any $S\subseteq N$. Note that $g(a_S) > 1+(1+a_S)2^{-n}$. It follows from the Binomial Theorem that

$$\begin{aligned}2^{-\nu_S} \le b_S &\le (2^{-\nu_S}+2^n\epsilon)(1+\epsilon)^n\\ &\le (2^{-\nu_S}+2^n\epsilon)(1+2n\epsilon)\\ &\le 2^{-\nu_S}+2^{n+2}\epsilon\end{aligned}$$

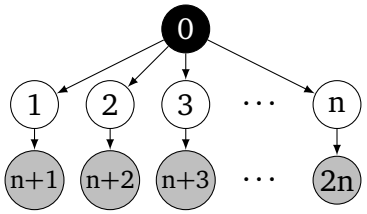


Figure 4.3. DAG of the credal tree used to prove Theorem 4.2.

where we use the inequality $(1+r/k)^k \leq 1+2r$ valid for $r \in [0,1]$ and positive integer k [Mauá et al., 2011, Lemma 37]. Thus,

$$h(v_S) - 1 \leq a_S \leq h(v_S) + 2^{n+3}\epsilon - 1\,.$$

Now if the partition problem is a yes-instance, then $a_S \leq 1/(64z^4)$, while if it is a no-instance, we have that $a_S > 1/(32z^4)$. Hence, there is a gap of at least $1/(64z^4)$ in the value of a_S between yes- and no-instances, and we can decide the partition problem by verifying whether $\mu \leq -1/g(\alpha)$, where $\alpha \stackrel{\text{def}}{=} 3/(128z^4)$. This proof shall be completed with the guarantee that we can approximate in polynomial time the irrational numbers used to specify the credal tree and $g(a)$ well enough so that $-1/g(\alpha)$ falls in the gap between the values of μ for yes- and no-instances. First, note that

$$g\left(\frac{1}{32z^4}\right) - g\left(\frac{1}{64z^4}\right) = \frac{1}{64z^4}\left(\frac{2}{1+\epsilon}\right)^n \prod_{i=1}^{n} \frac{1}{1+2^{-v_i}}\,,$$

which is greater than $2^{-n}/(64z^4)$. The gap in the value of μ is at least

$$\begin{aligned}\frac{1}{g(1/(64z^4))} - \frac{1}{g(1/(32z^4))} &= \frac{g(1/(32z^4)) - g(1/(64z^4))}{g(1/(64z^4))g(1/(32z^4))} \\ &> \frac{g(1/(32z^4)) - g(1/(64z^4))}{g(1/(32z^4))^2} \\ &> \frac{2^{-n}/(64z^4)}{(1+(1+\frac{1}{32z^4})2^{-n})^2} > \frac{2^{-n}}{4\cdot 64z^4}\,.\end{aligned}$$

So we apply Corollary 4.2 with $\varepsilon = \frac{1}{2}\frac{2^{-n}}{4\cdot 64z^4}$ and use the same rational numbers $p_i^2(1)$ as in the specification of the new network instead of the irrational values $1/(1+2^{-v_i})$ to approximate $g(\alpha)$, which guarantees that the gap will continue to exist. □

The credal network used in the reduction that proves the previous result is in a sense the simplest structure on which performing the GBR is hard, since the

problem would become polynomial-time solvable if the root node were replaced with a binary variable. It is also interesting as it describes a naive Bayes structure with a single layer of latent variables, a useful topology for robust classification problems on non-linearly separable feature spaces.

4.3.2 Imprecise hidden Markov models

An imprecise hidden Markov model (HMM) is a credal tree whose nodes can be partitioned into *hidden* and *manifest* nodes such that the hidden nodes form a chain (i.e., a sequence of nodes with one node linking to the next and to no other in the sequence), and each manifest node is a leaf with a single hidden node as parent. HMMs are widely used to represent discrete dynamic systems whose output at any given time step can be stochastically determined by the current state of the system, which is assumed to be only partially observable.

Since an HMM is simply a credal tree, the algorithm of de Cooman et al. [2010] can be used to efficiently compute the GBR in HMM under epistemic irrelevance, while 2U can be used in the case of strong independence if all variables all binary. For network with variables taking on more than two values, no polynomial-time is known for GBR inference under strong independence. In this section, we show that when the evidence variables topologically precede the queried variable, the result of the GBR is the same whether we consider epistemic irrelevance or strong independence. On these cases, we can run the algorithm for inference under epistemic irrelevance to obtain the GBR under strong independence in polynomial time. This is however not always true, that is, there are cases in which the result of the GBR depends on the irrelevance concept adopted, as the following example shows.

Example 4.6. *Consider an HMM of length two whose topology is depicted in Figure 4.4. All variables are binary and take values in* $\{0,1\}$. *Variables* X_1 *and* X_2 *are hidden, while variables* X_3 *and* X_4 *are manifest. The local credal sets are given by* $M(X_1) = M(X_2|0) = M(X_4|0) = \{p \in V(X_4) : p(1) = 1/4\}$, $M(X_2|1) = M(X_4|1) = \{p \in V(X_4) : p(1) = 3/4\}$, *and* $M(X_3|0) = \{p \in V(X_3) : 1/2 \leq p(1) \leq 3/4\}$ *and* $M(X_3|1) = \{p \in V(X_3) : 1/4 \leq p(1) \leq 1/2\}$. *Thus, variable* X_3 *is imprecise, and the remaining variables are precise. Consider the function* f *of* X_4 *that returns one at* $X_4 = 0$ *and zero elsewhere, and the evidence* $X_3 = 0$. *Under strong independence, the GBR is to solve for* μ *the equation*

$$\min \sum_{x_2} p_3^{x_2}(0) g_\mu(x_2) = \sum_{x_2} \min p_3^{x_2}(0) g_\mu(x_2) = 0\,,$$

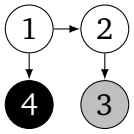


Figure 4.4. DAG of the HMM in Example 4.6.

where the minimizations are performed over $p_3^{x_2} \in M(X_3|x_2)$, $x_2 = 0, 1$, *and*

$$g_\mu(x_2) = \sum_{x_1,x_4} \big(f(x_4) - \mu\big)\, p_1(x_1) p_2^{x_1}(x_2) p_4^{x_1}(x_4)\,,$$

with $p_1 = p_2^0 = p_4^0 = (3/4, 1/4)$ *and* $p_2^1 = p_4^1 = (1/4, 3/4)$. *The values of* $p_3^{x_2}(0)$ *depend only on the signs of* $g_\mu(0)$ *and* $g_\mu(1)$, *which ought to be different for the expression to vanish. Solving for* μ *for each of the four possibilities, and taking the minimum value of* μ, *we find that* $\mu = \min\{p(0) : p \in \mathrm{K}_S(X_4|X_3{=}0)\} = 4/7$.

Under epistemic irrelevance, the GBR is equal to

$$\begin{aligned}\min \sum_{x_1,x_2,x_4} p_1(x_1) p_2^{x_1}(x_2) p_4^{x_1}(x_4) p_{x_1,x_2,x_3}(0) h_\mu(x_4) = \\ (1-\mu) \sum_{x_1,x_2} p_1(x_1) p_2^{x_1}(x_2) p_4^{x_1}(0) \min p_{x_1,x_2,0}(0) \\ - \mu \sum_{x_1,x_2} p_1(x_1) p_2^{x_1}(x_2) p_4^{x_1}(1) \max p_{x_1,x_2,1}(0) = 0\,,\end{aligned}$$

where $h_\mu(x_4) = f(x_4) - \mu$, p_1, $p_2^{x_1}$ *and* $p_4^{x_1}$ *are defined as before, and* $p_{x_1,x_2,x_4} \in M(X_3|x_2)$ *for every* x_1, x_2, x_4. *Solving the equation above for* μ *we get that* $\mu = 13/28$. □

The above example shows that the outcome of the GBR might depend on the irrelevance concept adopted, even in the simple case of HMMs with binary variables. It is currently unknown whether this type of inference is hard under strong independence. The following result shows that at least for a particular case, computation of the GBR in HMMs under strong independence is easy, as it reduces to the GBR under epistemic irrelevance in trees.

Theorem 4.3. *Consider a separately specified HMM over variables* $X_0, \ldots, X_n$. *The variables associated to odd numbers are manifest, and the remaining variables are hidden (see Figure 4.5). Consider also a function* f *of the hidden node* X_n, *and some evidence* $\tilde{x}_O$ *on a subset* O *of the manifest nodes. The outcome of the GBR is the same whether we assume epistemic irrelevance or strong independence.*

Proof. Consider the distribution p in the epistemic extension K_E that minimizes $\sum_{\mathbf{x}:\mathbf{x}_O=\tilde{\mathbf{x}}_O} f_\mu(x_n)p(\mathbf{x})$, where $f_\mu(x_q) = f(x_n) - \mu$ for some given μ. Let $<$ be any topological ordering of the nodes. By the chain rule of probability, we have for all $\mathbf{x}$ that p factorizes as $p(\mathbf{x}) = \mathrm{P}_p(x_0)\prod_{i\in[n]}\mathrm{P}_p(x_i|\mathbf{x}_{j<i})$. Assume that for some nonnegative i integer less than n it holds that

$$\sum_{\mathbf{x}:\mathbf{x}_O=\tilde{\mathbf{x}}_O} f_\mu(x_n)p(\mathbf{x}) \geq \sum_{\mathbf{x}:\mathbf{x}_O=\tilde{\mathbf{x}}_O} f_\mu(x_n)\prod_{j\leq i}\mathrm{P}_p(x_j|\mathbf{x}_{k<j})\prod_{j>i} p_j^{x_{\mathrm{Pa}(j)}}(x_j),$$

where each $p_j^{x_{\mathrm{Pa}(j)}}$ is recursively defined as the extreme distribution of the local credal set $M(X_i|\mathbf{x}_{\mathrm{Pa}(j)})$ that minimizes either

$$\sum_{x_j} p_j^{x_{\mathrm{Pa}(j)}}(x_j)\sum_{\mathbf{x}_{k>j}} f_\mu(x_n)\prod_{k>j} p_k^{x_{\mathrm{Pa}(k)}}(x_k),$$

if j is not in O, or

$$p_j^{x_{\mathrm{Pa}(j)}}(\tilde{x}_j)\sum_{\mathbf{x}_{k>j}} f_\mu(x_n)\prod_{k>j} p_k^{x_{\mathrm{Pa}(k)}}(x_k),$$

if j is in O, where $\tilde{x}_j$ is the value of X_j compatible with $\tilde{\mathbf{x}}_O$. We will show by induction in $i = n, \ldots, 0$ that the assumption is true. If i is not in O then

$$\begin{aligned}
&\sum_{\mathbf{x}:\mathbf{x}_O=\tilde{\mathbf{x}}_O} f_\mu(x_q)\prod_{j\leq i}\mathrm{P}_p(x_j|\mathbf{x}_{k<j})\prod_{j>i} p_j^{x_{\mathrm{Pa}(j)}}(x_j) = \\
&\qquad \sum_{\mathbf{x}_{j<i}:\mathbf{x}_O=\tilde{\mathbf{x}}_O}\prod_{j<i}\mathrm{P}_p(x_j|\mathbf{x}_{k<j})\sum_{x_i}\mathrm{P}_p(x_i|\mathbf{x}_{j<i})\sum_{\mathbf{x}_{j>i}} f_\mu(x_n)\prod_{j>i} p_j^{x_{\mathrm{Pa}(j)}}(x_j) \geq \\
&\qquad \sum_{\mathbf{x}_{j<i}:\mathbf{x}_O=\tilde{\mathbf{x}}_O}\prod_{j<i}\mathrm{P}_p(x_j|\mathbf{x}_{k<j})\min_{q\in M(X_i|x_{\mathrm{Pa}(i)})}\sum_{x_i} q(x_i)\sum_{\mathbf{x}_{j>i}} f_\mu(x_n)\prod_{j>i} p_j^{x_{\mathrm{Pa}(j)}}(x_j) = \\
&\qquad\qquad \sum_{\mathbf{x}:\mathbf{x}_O=\tilde{\mathbf{x}}_O} f_\mu(x_n)\prod_{j<i}\mathrm{P}_p(x_j|\mathbf{x}_{k<j})\prod_{j\geq i} p_j^{x_{\mathrm{Pa}(j)}}(x_j),
\end{aligned}$$

where the inequality follows from the definition of epistemic extension, which implies that $\sum_{x_i} h(x_i)\mathrm{P}_p(x_i|\mathbf{x}_{\mathrm{Nd}(i)}) \geq \min_{q\in M(X_i|x_{\mathrm{Pa}(i)})}\sum_{x_i} h(x_i)\mathrm{P}_p(x_i|\mathbf{x}_{\mathrm{Nd}(i)})$ for any function h' (note that $\mathrm{Nd}(i) \supseteq \{j < i\}$ and that $q_i^{x_{\mathrm{Pa}(i)}}$ is indeed a function of $x_{\mathrm{Pa}(i)}$ only, as the minimization is constant w.r.t. values $\mathbf{x}_{j<i:j\notin\mathrm{Pa}(i)}$). The case of a node i in O is analogous with the sum substituted by a single term. For $i = n$, it follows that

$$\begin{aligned}
\sum_{\mathbf{x}:\mathbf{x}_O=\tilde{\mathbf{x}}_O} f_\mu(x_q)p(\mathbf{x}) &= \sum_{\mathbf{x}_{j<n}:\mathbf{x}_O=\tilde{\mathbf{x}}_O}\prod_{j\leq n}\mathrm{P}_p(x_j|\mathbf{x}_{k<j})\sum_{x_q} f_\mu(x_n)\mathrm{P}_p(x_q|\mathbf{x}_{k<n}) \\
&\geq \sum_{\mathbf{x}:\mathbf{x}_O=\tilde{\mathbf{x}}_O} f_\mu(x_n)p_n^{x_{n-2}}(x_n)\prod_{j<n}\mathrm{P}_p(x_j|\mathbf{x}_{k<j}),
\end{aligned}$$

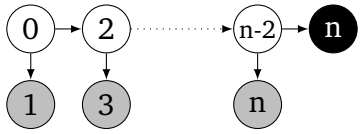


Figure 4.5. DAG of the HMM considered in Theorem 4.3.

so that the basis of the induction holds. For $i = 0$, we have that

$$\sum_{\mathbf{x}:\mathbf{x}_O=\tilde{\mathbf{x}}_O} f_\mu(x_q)p(\mathbf{x}) \geq \sum_{\mathbf{x}:\mathbf{x}_O=\tilde{\mathbf{x}}_O} f_\mu(x_n)p_0(x_0)\prod_{i\in[n]} p_j^{x_{\mathrm{Pa}(j)}}(x_j)\,,$$

which is the lower expectation of f_μ under strong independence. Thus, since the epistemic extension contains the strong extension, the inequality above is tight. In particular, the equality holds if μ is the outcome of the GBR under epistemic irrelevance, so that

$$\min_{p\in \mathrm{K}_S} \sum_{\mathbf{x}:\mathbf{x}_O=\tilde{\mathbf{x}}_O} f_\mu(x)p(\mathbf{x}) = \min_{p\in \mathrm{K}_E} \sum_{\mathbf{x}:\mathbf{x}_O=\tilde{\mathbf{x}}_O} f_\mu(x)p(\mathbf{x}) = 0\,,$$

where K_S denotes the strong extension. Thus, μ is also the outcome of the GBR under strong independence. □

The previous result shows that at least for the particular case where one seeks the lower posterior expectation of the "last" variable, the GBR can be computed in polynomial time. It is not known whether other type of inferences under strong independence have different complexity. Although restrictive, this type of inference is highly relevant, as it corresponds to predicting the future state of a partially observable dynamic system whose future state depends in some level only on its current (unknown) state.

4.3.3 Imprecise Markov chains

The simplest DAG structure forming a connected graph is that of a chain, that is, of a network in which each variable has at most one parent and one child. Credal chains are more usually known as (imprecise) Markov chains. As a chain is also a tree, computing the GBR under epistemic irrelevance can be done in polynomial time; this is also the case for chains of binary variables under strong independence. As the following result shows, a chain can be seen as an HMM where the values of the manifest variables are deterministically determined by the values of the hidden variables.

Corollary 4.3. *Consider a function f of the single leaf variable of a separately specified (imprecise) Markov chain, and some evidence $\tilde{\mathbf{x}}_O$ on arbitrary non-leaf variables. The result of the GBR on f with evidence $\tilde{\mathbf{x}}_O$ is the same whether we assume epistemic irrelevance or strong independence.*

Proof. The proof is analogous to the proof of Theorem 4.3. □

The result above implies that computing the GBR under strong independence in separately specified networks can be done in polynomial time, if the queried variable succeeds the evidence variables (or, equivalently, if there is no evidence). Recall that under strong independence the GBR can be computed efficiently in the case of credal polytrees with binary variables, whether the local credal sets are separately or extensively specified [Antonucci and Zaffalon, 2008]. The following result shows that this computational equivalence between separately and extensively specified networks under strong independence is unlikely to hold if we consider networks whose structure is simpler than polytrees.

Theorem 4.4. *Computing the GBR in extensively specified credal chains under strong independence is NP-hard, even if all variables are ternary.*

Proof. Consider an instance of the partition problem with integers $z_1,\ldots,z_n$ and let $v_i \stackrel{\text{def}}{=} z_i/z$ and $z \stackrel{\text{def}}{=} \sum_i z_i/2$. Build a credal chain of ternary variables $X_0,X_1,\ldots,X_n$, with X_{i-1} as parent of X_i, $i=1,\ldots,n$. The variable X_0 is precise and uniformly distributed. The local conditional credal sets $M(X_i|X_{i-1}=1)$, $i=1,\ldots,n$, have each two extreme distributions q_1 and q_2 such that

$$q_1(1)=2^{-v_i}, \qquad q_1(2)=0, \qquad q_1(3)=1-2^{-v_i},$$

and

$$q_2(1)=1, \qquad q_2(2)=0, \qquad q_2(3)=0.$$

The local conditional credal sets $M(X_i|X_{i-1}=2)$, $i=1,\ldots,n$, have each two extreme distributions q_3 and q_4 such that

$$q_3(1)=0, \qquad q_3(2)=2^{-v_i}, \qquad q_3(3)=1-2^{-v_i},$$

and

$$q_4(1)=0, \qquad q_4(2)=1, \qquad q_4(3)=0.$$

The local conditional credal sets $M(X_i|X_{i-1}=3)$ are singletons and contain the distribution that places all mass on $X_i=3$. Additionally, we include constraints

that forbid $\mathrm{P}_p(X_i|x_{i-1})$ to equal both q_1 and q_3 at the same time (for different values of x_{i-1}), and forbid $\mathrm{P}_p(X_i|x_{i-1})$ to equal both q_2 and q_4 at the same time. In other words, for each variable, the selection of q_1 forces the selection of q_4 (and vice-versa), and the selection of q_2 forces the selection of q_3.

Consider an extreme distribution p of the strong extension of the network. By design, p implies for $i = 1, \ldots, n$ that

$$\begin{aligned}\mathrm{P}_p(X_i{=}1) &= \sum_{x_{i-1}} \mathrm{P}_p(X_i{=}1|x_{i-1})\mathrm{P}_p(x_{i-1}) \\ &= q_{k_1}(1)\mathrm{P}_p(X_{i-1}{=}1) + q_{k_2}(1)\mathrm{P}_p(X_{i-1}{=}2) + q(1)\mathrm{P}_p(X_{i-1}{=}3),\end{aligned}$$

where $k_1 \in \{1,2\}$ and $k_2 \in \{3,4\}$ and q is the degenerate distribution assigning all mass on $X_i{=}3$ (thus $q(1) = 0$). It follows that

$$\mathrm{P}_p(X_i{=}1) = 2^{-y_i v_i}\mathrm{P}_p(X_{i-1}{=}1),$$

where $y_i \in \{0,1\}$. Similarly, we have that

$$\mathrm{P}_p(X_i{=}2) = \sum_{x_{i-1}} \mathrm{P}_p(X_i{=}2|x_{i-1})\mathrm{P}_p(x_{i-1}) = 2^{-(1-y_i)v_i}\mathrm{P}_p(X_{i-1}{=}2).$$

The term $1 - y_i$ in the exponent guarantees that the logical constraints on the local distributions are satisfied. In particular, we have that

$$\mathrm{P}_p(X_1{=}1) = 2^{-y_1 v_1}/3, \qquad \mathrm{P}_p(X_1{=}2) = 2^{-(1-y_1)v_1}/3.$$

Hence, it follows from induction on $i = 1, \ldots, n$ that

$$\mathrm{P}_p(X_n{=}1) = 2^{-\sum_{i\in[n]} v_i y_i}/3, \qquad \mathrm{P}_p(X_n{=}2) = 2^{-\sum_{i\in[n]} v_i(1-y_i)}/3.$$

Let f be a function of X_n that returns 0 at $X_n{=}3$ and 3 elsewhere. The result of the GBR is then

$$\begin{aligned}\mu &= \min_{p\in\mathrm{K}_S} 3\mathrm{P}_p(X_n{=}1) + 3\mathrm{P}_p(X_n{=}2) \\ &= \min_{\mathbf{y}\in\{0,1\}^n} 2^{-\sum_{i\in[n]} v_i y_i} + 2^{-\sum_{i\in[n]} v_i(1-y_i)} \\ &= \min_{S\subset[n]} \frac{2^{-(v_S-1)} + 2^{v_S-1}}{2} = \min_S h(v_S).\end{aligned}$$

If the partition problem is an yes-instance, then $\mu = 1$, whereas if it is a no-instance then $\mu \geq 1 + 1/(32z^4)$. According to Corollary 4.2, we can use that gap to reduce the problem to a credal network quantified by rational numbers that preserves the distinction of yes- and no-instances of the partition problem. □

4.3.4 Approximate results

De Campos and Cozman [2005] showed that approximating the value of the GBR under strong independence is NP-hard, even if we consider only credal polytrees of bounded treewidth. This is however not the case if variables are binary, as in this case we can run the 2U algorithm to obtain the exact value. We show here that there is a polynomial-time approximation algorithm for computing the value of GBR within any given relative error for any model whose variables take values in a small enough domain whose size is assumed constant (but not necessarily binary). To simplify the proof of the existence of the approximation algorithm, we need the following result.

Lemma 4.2. *Let μ be the outcome of the GBR under strong independence on a function f of a variable X_q in a credal network $\mathcal{N}$ of bounded treewidth and arbitrary evidence $\tilde{\mathbf{x}}_O$. We can obtain in polynomial time a credal network $\mathcal{N}'$ of bounded treewidth for which the GBR on the same f and with evidence on a single binary variable returns μ.*

Proof. Let $X_{o_1},\ldots,X_{o_m}$ denote the evidence variables in $\mathcal{N}$. Without loss of generality [Cozman, 1999; Antonucci and Zaffalon, 2008], assume that these variables are associated to leaf nodes in the DAG and take values in $\{0,1\}$ and that the evidence is $\tilde{\mathbf{x}}_O = \{X_{o_1}=1,\ldots,X_{o_m}=1\}$. Consider the deterministic variable $Z = X_{o_1}X_{o_2}\cdots X_{o_m}$, that is, Z evaluates to zero unless all evidence variables are set to one, in which case Z evaluates to one. since the variable Z determines a symmetric and decomposable function of binary variables, it can be succinctly represented by a sequence of binary deterministic variables $X_{e_1},\ldots,X_{e_m}$ satisfying $X_{e_i} = X_{e_{i-1}}X_{o_i}$, $i>1$, and $X_{e_1} = X_{o_1}$ [Koller and Friedman, 2009, Chapter 9.6.1.2]. One can easily verify that $X_{e_m} = Z$. Each of these variables can in turn be represented by a conditional probability distribution requiring only eight numbers. Consider the network $\mathcal{N}'$ obtained from $\mathcal{N}$ by inserting the precise variables X_{e_i}, $i=1,\ldots,m$, each having $X_{e_{i-1}}$ and X_{o_i} are parents (X_{e_1} is the single parent of X_{e_1}). By the same argument used to prove Theorem 3.4, we can show that treewidth of $\mathcal{N}'$ exceeds the treewidth of $\mathcal{N}$ in at most three. Specify the conditional distributions $p_{e_i}^{x_{e_{i-1}},x_{o_i}}(1)$ associated with each X_{e_i}, for $i=1,\ldots,m$, such that $p_{e_i}^{x_{e_{i-1}},x_{o_i}}(1) = x_{e_{i-1}}x_{o_i}$ (with $p_{e_1}^{x_{o_1}}(1)=x_{o_1}$). Let p be an extreme distribution of the strong extension of $\mathcal{N}'$, and define $E=\{e_1,\ldots,e_{m-1}\}$. We have that

$$\mathrm{P}_p(x_q,X_{e_m}=1) = \sum_{\mathbf{x}_O}\mathrm{P}_p(X_{e_m}=1|\mathbf{x}_O)\mathrm{P}_p(x_q,\mathbf{x}_O) = \mathrm{P}_p(x_q,\tilde{\mathbf{x}}_O).$$

Now the joint probability on the right factorizes as a product of extreme distri-

butions taken from the local credal sets associated to variables in $\mathcal{N}$. Thus, the right-hand side is attained by an extreme distribution p in the strong extension of $\mathcal{N}$. The converse is also true, that is, given an extreme distribution of $\mathcal{N}$, its product by distributions $p_{e_1}^{x_{o_1}}$ and $p_{e_i}^{x_{e_{i-1}},x_{o_i}}$ is an extreme distribution of the strong extension $\mathcal{N}'$ satisfying the identity above. Since the minimization in the GBR is solved by an extreme distribution of the strong extension, the result follows. □

Theorem 4.5. *There is a fully polynomial-time approximation scheme to compute the GBR under strong independence in credal networks of bounded treewidth whose variables take values in domains of bounded cardinality.*

Proof. The proof is very similar to the proof of Theorem 3.7, except that we cope with the existence of evidence in the query. Once more, we show the existence of an FTPAS constructively, and using the FACTOR-SET-ELIMINATION algorithm.

Consider a credal network $\mathcal{N}$ of treewidth bounded by w and whose variables have cardinality bounded by v, and let ε be the desired relative error of the output. Any local credal set $M(X_i|\mathbf{x}_{\mathrm{Pa}(i)})$ in $\mathcal{N}$ defined by m linear constraints in variables $p \in V(\mathbf{X}_{\mathrm{Fa}(i)})$ has at most m^{v^w} extreme distributions, and these can be found in time $O(m^{v^w})$, which is polynomial in the input, as v and w are assumed constant [Avis, 2000]. Thus, a constraint-based network can be transformed into vertex-based form in polynomial time. Moreover, any vertex-based network can be mapped into an equivalent network whose non-root nodes are all precise, where equivalence means equality of the outcome of the GBR on common variables [Antonucci and Zaffalon, 2008]. Given such equivalences, we assume in the following that $\mathcal{N}$ is specified in vertex-based form, with all non-root notes precise, and, in the light of Lemma 4.2, that the task is to compute the GBR on a function f of a variable X_q with a single binary evidence variable X_e. We also assume that f is a nonnegative function, so that the relative error of any solution (i.e., a number μ that attains a posterior expectation of f) is well-defined. Since the relative error measure is invariant to positive scalings of the solutions, the quality of a solution remains the same if we normalize the function f, which is equivalent to introducing a new binary precise variable x_i as a child of X_q whose conditional distribution is defined as $\mathrm{P}(X_i = 1|x_q) \stackrel{\mathrm{def}}{=} f(x_q)/\max_{x'_q} f(x'_q)$. Thus, we assume with no loss of generality that f is an indicator function so that computing the GBR amounts to calculating a posterior probability.

For each imprecise root node j in $\mathcal{N}$, let K_j denote the set of degenerate distributions of $V(X_j)$. Hence, K_j is a set containing $O(v)$ functions δ_j. For each precise node $i \neq e$, let K_i be the singleton containing the corresponding conditional probability distribution of X_i given $\mathbf{X}_{\mathrm{Pa}(i)}$. Finally, let K_e be the singleton containing the conditional distribution of X_e with zero mass assigned to

values $X_e \neq 1$. Note that each set K_i, $i \in N$, is a set of nonnegative real-valued functions of a subset of $\mathbf{X}$ (called factors in Chapter 2). Obtain a minimal binary tree-decomposition T for $\mathcal{N}$ whose nodes are associated to sets $C_1, \ldots, C_{m+1}$, and assume that there is a set $C_i = \{q\}$ for some $i \in [m+1]$ with a single child (such T can be obtained in linear time as the treewidth of the diagram is assumed bounded, Bodlaender [1996]). Without loss of generality, assume that $C_i \supseteq \mathrm{Fa}(i)$ and that $C_{m+1} = \{q, e\}$. For every C_i with $i \notin N$, let K_i be a singleton containing the identity function of $\mathbf{X}_{C_i}$. Also, for every $i \in N$ with $C_i \supset \mathrm{Pa}(i)$, redefine the functions in K_i so that they have domain $\mathbf{X}_{C_i}$ and return the same values when restricted to $\mathbf{X}_{\mathrm{Fa}(i)}$ (i.e., replace every function ψ of $\mathbf{X}_{\mathrm{Fa}(i)}$ in K_i with a function ϕ of $\mathbf{X}_{C_i}$ such that $\phi(\mathbf{x}_{C_i}) = \psi(\mathbf{x}_{\mathrm{Fa}(i)})$). Run FACTOR-SET-ELIMINATION on sets $K_1, \ldots, K_{m+1}$, tree decomposition T, and constants $k_1, \ldots, k_{m+1} = \Theta(bnm^2/\varepsilon)$, select $m+1$ as the root node, and initialize the clusterings in each pruning step with the same partitioning of the functions into hyper-rectangles as in the proof of Theorem 3.7. Then FACTOR-SET-ELIMINATION finishes in time polynomial in the input size (given as the number of bits b) and produces a set L_m during its execution. By construction, and according to Theorem 2.1, any function p in L_m satisfies

$$(\forall x_q) \quad p(x_q) = \sum_{\mathbf{x}' : x'_q = x_q, x'_e = 1} \prod_{i \in [n]} p_i^{\mathbf{x}'_{\mathrm{Pa}(i)}}(x'_i),$$

where each $p_i^{\mathbf{x}'_{\mathrm{Pa}(i)}}$ is an extreme distribution of $M(X_i | \mathbf{x}'_{\mathrm{Pa}(i)})$. Thus, p is an element of the credal set $\mathrm{K}_S(X_q, X_e = 1)$. Let μ^* be the (true) outcome of the GBR, and p^* be a joint distribution that solves the minimization in the GBR for that value of μ^*, that is, $\mu^* = \sum_{x_q} f(x_q) p^*_q(x_q) / \sum_{x'_q} p^*_q(x'_q)$, where $p^*_q(x_q) \stackrel{\mathrm{def}}{=} \sum_{\mathbf{x}' : x'_q = x_q, x'_e = 1} p^*(\mathbf{x}')$. Since p^*_q is an element of $\mathrm{K}_S(X_q, X_e = 1)$, and each L_i is an α-covering of M_i with $\alpha \stackrel{\mathrm{def}}{=} 1 + \varepsilon/(1-\varepsilon)/4m$, we have that $p\alpha^{-m} \leq p^*_q \leq \alpha^m p$ for some p in L_m, which implies that

$$\frac{\mu_p}{\mu^*} = \frac{p(x_q)}{\sum_{x'_q} p(x'_q)} \frac{\sum_{x'_q} p^*_q(x'_q)}{p^*(x_q)} \geq \frac{p(x_q)}{p^*_q(x_q)} \geq \frac{1}{\alpha^{2m}},$$

where x_q is the support of f (assumed unique), and

$$\mu_p \stackrel{\mathrm{def}}{=} \frac{p(x_q)}{\sum_{x'_q} p(x'_q)}.$$

Since p is a member of $\mathrm{K}_S(X_q, X_e = 1)$ and μ_p is a linear function of p, it follows that μ_p is a member of the posterior credal set $\mathrm{K}_S(X_q | X_e = 1)$ [Boyd and Vandenberghe, 2004, Chapter 2.3.3]. Thus, μ_p is a (feasible) solution to the GBR with

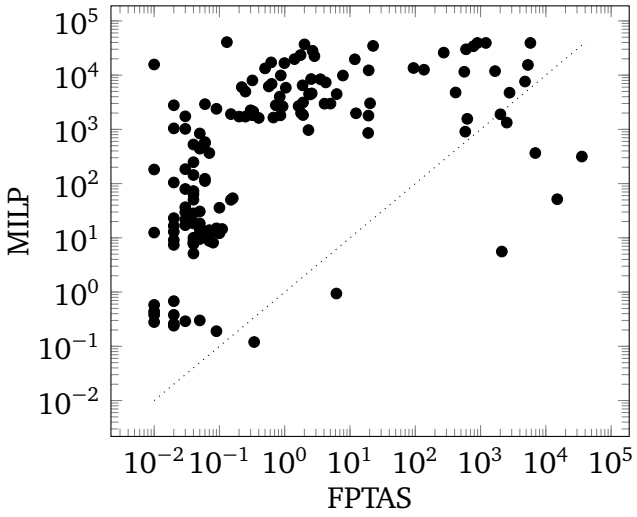
relative error at most ε, since

$$\frac{1}{\alpha^{2m}} = \left(1 + \frac{\varepsilon/(1-\varepsilon)}{4m}\right)^m \leq 1 + \frac{\varepsilon}{1-\varepsilon} = \frac{1}{1-\varepsilon},$$

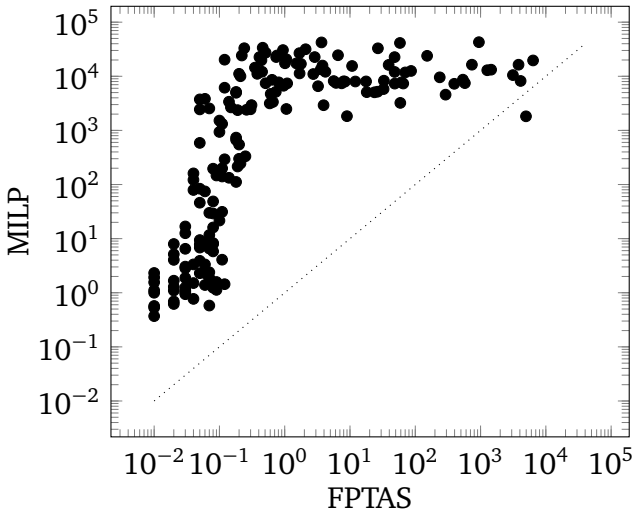
and the result follows. □

Like the FPTAS for solving LIMIDs we described in Chapter 3, the worst-case running time of the approximation algorithm described in the proof of the result above is a polynomial with very high exponent and constants, too high for practical computation on current computers. Nevertheless, we can use the same described procedure to compute the GBR in credal networks by a single call of FACTOR-SET-ELIMINATION. To verify the tractability of such an approach, we compared its performance against the mixed linear integer programming reformulation approach of de Campos and Cozman [2007b] on a collection of 1860 extensively specified credal networks randomly generated using the BNGen package [Ide et al., 2004]. The networks were generated containing treewidth no greater than four, 10 to 30 nodes, 2 to 4 states per variable, and 2 to 16 extreme distribution potentials in each local extensive credal set. For each network, we set some evidence to every leaf node (i.e., the set of evidence variables e correspond to set of leaves of the network) and arbitrarily chose a node with no parents as query. The FTPAS was run with a relative error of $\varepsilon = 0.1$. The tree decompositions were found using the fill-in heuristic.

For each network, we granted each algorithm 12 hours of CPU time and 2GB of RAM on a fast computer. Figure 4.6 compares the running times of both methods on a log-log scale according to the network topology (each point in the plot represents a network, and only networks which both methods were able to solve within the time and memory limits are shown). Since the MILP method implements an anytime procedure, we ran it in each problem until the difference between lower and upper bounds were smaller than 0.0001. Hence, we considered a problem unsolved by MILP if the CPLEX mixed linear integer solver was not able to meet such a requirement within the time and memory limits we set. The FPTAS and MILP methods solved, respectively, 805 and 357 out of the 1860 problem instances. Thus, while the FTPAS can be considered a state-of-the-art method for computing the GBR, it still fails to solve a large number of cases in feasible time. Interestingly, MILP performs relatively worse on loopy networks than in credal polytrees.



(a) Polytrees



(b) Loopy networks

Figure 4.6. Comparison of the running times (in sec) of the FPTAS and MILP approaches on randomly generated networks.

4.4 Conclusion

Credal networks generalize Bayesian networks to allow for the representation of uncertain knowledge in the form of credal sets, closed and convex sets of probability distributions. The use of credal sets arguably facilitates the constructions of complex models, but presents a challenge to the computation of inferences with the model.

In this chapter, we studied the theoretical complexity of inferences in credal networks, in what concerns the topology of the network, the semantics of the arcs (i.e., whether epistemic irrelevance or strong independence is assumed), and the cardinality of variable domains. In a nutshell, computing with credal networks is NP-hard except in the cases of tree-shaped models under epistemic irrelevance, and polytree-shaped models under strong independence. A notable exception is the computation of probability bounds on the value of the last variable in a imprecise hidden Markov models, in which case we have shown that inferences under epistemic irrelevance and strong independence coincide, which implies that the latter is polynomial-time computable. Also, we showed that updating extensively specified credal networks is NP-hard, even if the DAG is a tree. Finally, we proved that for the case of bounded treewidth networks with variables taking values in bounded domains, there exists a fully polynomial-time approximation scheme for updating under strong independence. Experiments on randomly generated networks showed that the FPTAS is not only a theoretical result but a competitive approach for computing inferences in such networks.

We left as an open question the complexity of generic inferences in imprecise HMMs under strong independence.

Chapter 5

Conclusions

This thesis addressed three hard computational problems that arise in tasks involving probabilistic reasoning, namely, the problems of

- maximum a posteriori (MAP) inference in probabilistic graphical models (Bayesian and Markov networks),
- planning with limited memory influence diagrams (LIMIDs), and
- belief updating in credal networks (credal belief updating).

These problems address somewhat very different situations. Roughly speaking, the MAP inference problem aims at finding the most probable explanation of a complex phenomenon represented as a graphical model. The problem of planning with (or solving) influence diagrams consists in selecting a course of actions that maximizes expected utility. Finally, belief updating in credal networks can be described as the problem of assessing the sensitivity of probabilistic inference in Bayesian networks to global changes in the model parameters.

At first sight, these three problems might seem connected only by means of their combinatorial or optimization nature, or yet their use of graphs as a concise representational device. Nevertheless, correspondences between instances of these problems have long been noticed in the literature. For instance, it has been shown that credal belief updating can be reduced to MAP inference in Bayesian networks [Cano et al., 1994] and vice-versa [de Campos and Cozman, 2005]. De Campos and Ji [2008] showed that planning with influence diagrams can be reduced to belief updating in credal networks, and the converse was also shown by Antonucci and Zaffalon [2008] to be true. These correspondences are depicted in Figure 5.1. The diagram makes it more clear the missing correspondences, namely, the direct reductions from MAP inference problems into solving

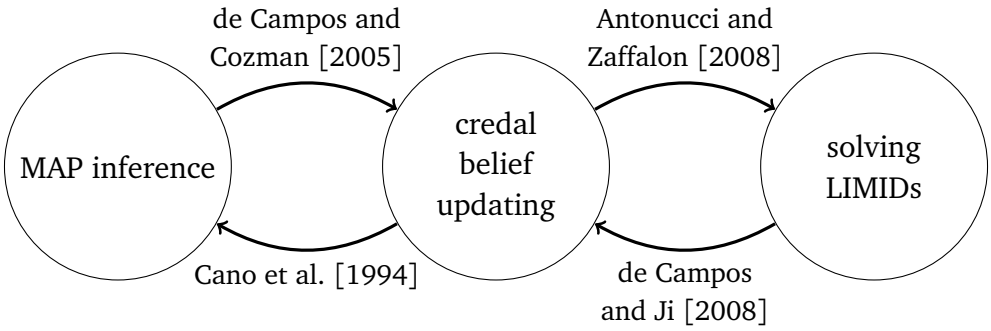


Figure 5.1. Correspondences between instances of the three class of problems considered.

LIMIDs and vice-versa. The latter can be obtained by applying the sequence of results derived in Chapter 3 and graphically shown in Figure 5.2. The former reduction remains as an item for future work.

The previously known correspondences between instances of these problems focused either on its practical side or on its semantic implications. Cano et al. [1994] reduced credal belief updating in order to be able use the available algorithms for MAP inference. De Campos and Ji [2008] reduced planning in influence diagrams to belief updating in credal networks so that the former problem could be solved using algorithms designed for the latter. Antonucci and Zaffalon [2008] reduced belief updating in credal networks to planning in influence diagrams in order to provide a decision-theoretic view of credal networks.

To our knowledge, the only work that exploited such correspondences in order to derive results regarding the theoretical computational complexity of these problems is the work of de Campos and Cozman [2005], which showed hardness of credal belief updating by a reduction from MAP inference. By following their approach, we showed in this work that a certain class of planning problems on LIMIDs can be solved in polynomial time, by reducing them into instances of credal belief updating that are known to be polynomial-time computable (Theorem 3.5). Using the converse reduction, we were able to show the NP-hardness of credal belief updating even in polytrees with binary imprecise variables and ternary precise ones (Theorem 4.1). Additionally, the reductions allowed us to use the previously result of fully polynomial-time approximability of MAP inference proved by de Campos [2011] to show that also planning in influence diagrams and credal belief updating admit fully polynomial-time approximations (Theorems 3.7 and 4.5). These are the first results concerning

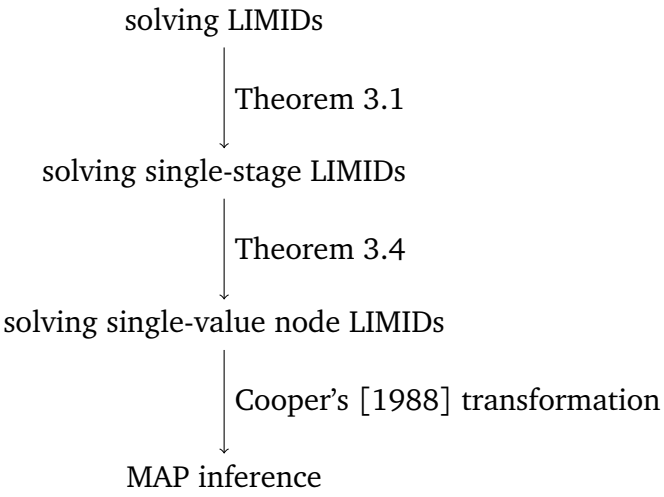


Figure 5.2. Steps in the reduction from planning with LIMIDs to MAP inference.

approximability of these problems of which we are aware.

The complexity of these three classes of problems is better understood by assuming certain structure to be present, or conversely, by looking into the complexity of subclasses of those problems. However, the reductions from one problem into another do not preserve all of the structure present in a problem. An example is the reduction from MAP inference into planning with influence diagrams, which maps tree-shaped Bayesian networks into multiply connected influence diagrams. Thus, many of the several NP-hardness results developed in Chapters 3 and 4, which regard subclasses of the problems of solving LIMIDs and credal belief updating, are necessary in that they could not be obtained immediately from similar hardness results in the other problem class. For instance, since polytree-shaped instances of MAP inference problems are reduced into loopy instances of LIMIDs, we cannot use the known NP-hardness result of MAP inference in polytree-shaped Bayesian networks to show NP-hardness of solving polytree-shaped LIMIDs, which warrants the proof of the latter that we provide in Chapter 3.

Although the result of fully polynomial-time approximability of MAP inference in models was obtained constructively (i.e., by designing a fully polynomial-time approximation scheme) by de Campos [2011] for the case of bounded treewidth graphs and bounded cardinality variables, the asymptotic analysis he provides hides huge constants which hinders it practical applicability. To overcome such inefficiency while still preserving some of the theoretical performance

guarantees, we developed in Chapter 2 a routine for approximate MAP inference based on the propagation of set-valued messages in a clique-tree. Our approximate algorithm allows for specification by the user of the trade-off between accuracy and speed. This trade-off is most likely necessary, as approximating any of the three problems we address here within a sub-exponential factor is known to be NP-hard, as shown by Park and Darwiche [2004] for the case of MAP inference, by de Campos and Cozman [2005] for the case of credal belief updating, and by Theorem 3.6 in this work for the case of solving LIMIDs. Moreover, benefiting from the aforementioned correspondences between instances of three problems considered in this thesis, the approximate algorithm can be easily extended to the other two problems. Empirical comparisons with state-of-the-art algorithms especially designed for each class of problems have shown that the approximate algorithm is competitive in any of the problems, and often finds optimal solutions in feasible time, which is remarkable in light of the theoretical complexity results derived (most of them in this work).

Although many previously open questions have been answered by this work, and in spite of the promising results obtained in the empirical evaluation of the approximate algorithm developed here, this work leaves many open avenues to pursue, both on the practical and theoretical sides. On the practical side, the approximate algorithm developed has complexity exponential on the treewidth of the underlying graph, what prevent us from using it in models of high treewidth, which arise in many real applications. Overcoming such a limitation is a non-trivial but important contribution of future work. On the theoretical side, there are subclasses of problem with still unknown theoretical complexity. A non exhaustive list includes MAP inference in Bayesian trees over binary variables, credal belief updating under strong independence in HMM-like trees, and approximate credal belief updating under epistemic irrelevance in bounded treewidth networks (e.g., polytrees with bounded in-degree) over binary variables.

Bibliography

C. Amato, D. S. Bernstein, and S. Zilberstein. Optimizing fixed-size stochastic controllers for POMDPs and decentralized POMDPs. *Autonomous Agents and Multi-Agent Systems*, 21(3):293–320, 2010.

A. Antonucci and A. Piatti. Modeling unreliable observations in Bayesian networks by credal networks. In *Proceedings of the 3rd International Conference on Scalable Uncertainty Management (SUM)*, volume 5785 of *Lecture Notes in Computer Science*, pages 28–39. Springer, 2009.

A. Antonucci and M. Zaffalon. Equivalence between Bayesian and credal nets on an updating problem. In *Proceedings of 3rd International Conference on Soft Methods in Probability and Statistics (SMPS)*, pages 223–230, 2006.

A. Antonucci and M. Zaffalon. Decision-theoretic specification of credal networks: a unified language for uncertain modeling with sets of Bayesian networks. *International Journal of Approximate Reasoning*, 49(2):345–361, 2008.

A. Antonucci, A. Piatti, and M. Zaffalon. Credal networks for operational risk measurement and management. In *International Conference on Knowledge-Based and Intelligent Information & Engineering Systems (KES)*, volume LNCS 4693, pages 604–611, 2007.

A. Antonucci, R. Brühlmann, A. Piatti, and M. Zaffalon. Credal networks for military identification problems. *International Journal of Approximate Reasoning*, 50:666–679, 2009.

A. Antonucci, R. de Rosa, and A. Giusti. Action recognition by imprecise hidden Markov models. In *Proceedings of the 2011 International Conference on Image Processing, Computer Vision and Pattern Recognition (IPCV)*, pages 474–478, 2011.

A. Antonucci, G. Corani, D. D. Mauá, and S. Gabaglio. An ensemble of Bayesian networks for multilabel classification. In *Proceedings of the 23rd International Joint Conference on Artificial Intelligence (IJCAI)*, 2013a.

A. Antonucci, C. P. de Campos, D. Huber, and M. Zaffalon. Approximating credal network inferences by linear programming. In *Proceedings of the 12th European Conference on Symbolic and Quantitative Approaches to Reasoning with Uncertainty (ECSQARU)*, volume 7958, pages 13–25, 2013b.

A. Antonucci, D. Huber, M. Zaffalon, P. Luginbühl, I. Chapman, and R. Ladouceur. CREDO: a military decision-support system based on credal networks. In *Proceedings of the 16th Conference on Information Fusion (FUSION 2013)*, 2013c.

Alessandro Antonucci, Yi Sun, C. P. de Campos, and Marco Zaffalon. Generalized loopy 2U: A new algorithm for approximate inference in credal networks. *International Journal of Approximate Reasoning*, 55, 2010.

G. Ausiello, P. Crescenzi, and M. Protasi. Approximate solution of NP optimization problems. *Theoretical Computer Science*, 150(1):1–55, 1995.

D. Avis. A revised implementation of the reverse search vertex enumeration algorithm. In G. Kalai and G. M.. Ziegler, editors, *Polytopes: Combinatorics and Computation*, volume 29 of *DMV Seminar*, pages 177–198. Birkhäuser Basel, 2000.

D. S. Bernstein, E. A. Hansen, and S. Zilberstein. Bounded policy iteration for decentralized POMDPs. In *Proceedings of the Nineteenth International Joint Conference on Artificial Intelligence (IJCAI)*, pages 1287–1292, 2005.

U. Bertele and F. Brioschi. *Nonserial Dynamic Programming*. Academic Press, Inc., 1972.

John Binder, Daphne Koller, Stuart Russell, and Keiji Kanazawa. Adaptive probabilistic networks with hidden variables. *Machine Learning*, 29(2–3):213–244, 1997.

H. Bodlaender, A. Koster, F. van den Eijkhof, and L. van der Gaag. Pre-processing for triangulation of probabilistic networks. In *Proceedings of the Seventeenth Conference Annual Conference on Uncertainty in Artificial Intelligence (UAI)*, pages 32–39, 2001.

H. L. Bodlaender. A linear-time algorithm for finding tree-decompositions of small treewidth. *SIAM Journal on Computing*, 25(6):1305–1317, 1996.

S. P. Boyd and L. Vandenberghe. *Convex Optimization*. Cambridge University Press, 2004.

A. Cano and S. Moral. A genetic algorithm to approximate convex sets of probabilities. In *Proceedings of the Fifth International Conference on Information Processing and Management of Uncertainty in Knowledge Based Systems (IPMU)*, pages 859–864, 1996.

A. Cano, J. E. Cano, and S. Moral. Convex sets of probabilities propagation by simulated annealing. In *Proceedings of the Fith International Conference on Information Processing and Management of Uncertainty in Knowledge Based Systems (IPMU)*, pages 4–8, 1994.

A. Cano, M. Gómez, S. Moral, and J. Abellán. Hill-climbing and branch-and-bound algorithms for exact and approximate inference in credal networks. *International Journal of Approximate Reasoning*, 44:261–280, 2007.

Cheng, Chen, Dong, Xu, and Ihler. Approximating the sum operation for marginal-map inference. In *Proceedings of the 26th AAAI Conference on Artificial Intelligence (AAAI)*, 2012.

J. Chuzhoy, S. Guha, E. Halperin, S. Khanna, G. Kortsarz, R. Krauthgamer, and J. Naor. Asymmetric k-center is log* n-hard to approximate. *Journal of the ACM*, 52:538–551, 2005.

G. F. Cooper. A method for using belief networks as influence diagrams. *Fourth Workshop on Uncertainty in Artificial Intelligence*, 1988.

G. Corani, A. Giusti, D. Migliore, and J. Schmidhuber. Robust texture recognition using credal classifiers. In F. Labrosse, R. Zwiggelaar, Y. Liu, and B. Tiddeman, editors, *Proceedings of the British Machine Vision Conference*, pages 78.1–78.10. BMVA Press, 2010.

F. G. Cozman. Calculation of posterior bounds given convex sets of prior probability measures and likelihood functions. *Journal of Computational and Graphical Statistics*, 8(4):824–838, 1999.

F. G. Cozman. Credal networks. *Artificial Intelligence*, 120(2):199–233, 2000.

F. G. Cozman. Graphical models for imprecise probabilities. *International Journal of Approximate Reasoning*, 39(2–3):167–184, 2005.

F. G. Cozman, C. P. de Campos, J. S. Ide, and J. C. F. da Rocha. Propositional and relational Bayesian networks associated with imprecise and qualitative probabilistic assessments. In *Proceedings of the 20th conference on Uncertainty in artificial intelligence (UAI)*, pages 104–111, 2004.

J. C. F. da Rocha and F. G. Cozman. Inference in credal networks: branch-and-bound methods and the A/R+ algorithm. *International Journal of Approximate Reasoning*, 39(2–3):279–296, 2005.

A. Darwiche. *Modeling and Reasoning with Bayesian Networks*. Cambridge University Press, 2009.

J. de Bock and G. de Cooman. State sequence prediction in imprecise hidden Markov models. In *Proceedings of the 7th International Symposium on Imprecise Probabilty: Theories and Applications (ISIPTA)*, 2011.

J. de Bock and G. de Cooman. Allowing for probability zero in credal networks under epistemic irrelevance. In *Proceedings of the 8th International Symposium on Imprecise Probabilty: Theories and Applications (ISIPTA)*, 2013.

C. de Campos and Q. Ji. Bayesian networks and the imprecise Dirichlet model applied to recognition problems. In W. Liu, editor, *Symbolic and Quantitative Approaches to Reasoning With Uncertainty*, volume 6717 of *Lecture Notes in Computer Science*, pages 158–169. Springer, Berlin / Heidelberg, 2011.

C. P. de Campos. New complexity results for MAP in Bayesian networks. In *Proceedings of the 22nd International Joint Conference on Artificial Intelligence (IJCAI)*, pages 2100–2106, 2011.

C. P. de Campos and F. G. Cozman. Inference in credal networks using multilinear programming. In *Second Starting AI Researcher Symposium*, pages 50–61, 2004.

C. P. de Campos and F. G. Cozman. The inferential complexity of Bayesian and credal networks. In *Proceedings of the 19th International Joint Conference on Artificial Intelligence (IJCAI)*, pages 1313–1318, 2005.

C. P. de Campos and F. G. Cozman. Computing lower and upper expectations under epistemic independence. *International Journal of Approximate Reasoning*, 44(3):244–260, 2007a.

C. P. de Campos and F. G. Cozman. Inference in credal networks through integer programming. In *Proceeding of the Fifth International Symposium on Imprecise Probability: Theories and Applications (ISIPTA)*, pages 145–154, 2007b.

C. P. de Campos and F. G. Cozman. Complexity of inferences in polytree-shaped semi-qualitative probabilistic networks. In *Proceedings of the AAAI Conference on Advances in Artificial Intelligence*, page 10, 2013.

C. P. de Campos and Q. Ji. Strategy selection in influence diagrams using imprecise probabilities. In *Proceedings of the 24th Conference in Uncertainty in Artificial Intelligence (UAI)*, pages 121–128, 2008.

C. P. de Campos, L. Zhang, Y. Tong, and Q. Ji. Semi-qualitative probabilistic networks in computer vision problems. *Journal of Statistical Theory and Practice*, 3(1):197–210, 2009.

L. M. de Campos, J. A. Gámez, and S. Moral. Partial abductive inference in Bayesian networks by using probability trees. In *5th International Conference on Enterprise Information Systems (ICEIS)*, pages 83–91, 2003.

L.M. de Campos, J.F. Huete, and S. Moral. Probability intervals: a tool for uncertain reasoning. *International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems*, 2:167–196, 1994.

G. de Cooman and M. C. M. Troffaes. Coherent lower previsions in systems modelling : products and aggregation rules. *Reliability engineering & system safety*, 85(1–3):113–134, 2004.

G. de Cooman, F. Hermans, A. Antonucci, and M. Zaffalon. Epistemic irrelevance in credal nets: the case of imprecise Markov trees. *International Journal of Approximate Reasoning*, 51(9):1029–1052, 2010.

R. Dechter. Bucket elimination: A unifying framework for reasoning. *Artificial Intelligence*, 113(1–2):41–85, 1999.

R. Dechter. An anytime approximation for optimizing policies under uncertainty. In *Workshop of Decision Theoretic Planning, AIPS*, 2000.

R. Dechter and I. Rish. Mini-buckets: A general scheme for bounded inference. *Journal of the ACM*, 50(2):107–153, 2003.

A. Detwarasiti and R. D. Shachter. Influence diagrams for team decision analysis. *Decision Analysis*, 2(4):207–228, 2005.

G. Elidan and N. Friedman. Learning hidden variable networks: The information bottleneck approach. *Journal of Machine Learing Research*, 6:81–127, 2005.

G. Elidan, N. Lotner, N. Friedman, and D. Koller. Discovering hidden variables: A structure-based approach. In *Advances in Neural Information Processing Systems 13 (NIPS)*, 2000.

E. Fagiuoli and M. Zaffalon. A note about redundancy in influence diagrams. *International Journal of Approximate Reasoning*, 19(3-4):351–365, 1998a.

E. Fagiuoli and M. Zaffalon. 2U: An exact interval propagation algorithm for polytrees with binary variables. *Artificial Intelligence*, 106(1):77–107, 1998b.

T. Feder and D. Greene. Optimal algorithms for approximate clustering. In *Proceedings of the 20th Annual ACM Symposium on Theory of Computing (STOC)*, pages 434–444, 1988.

N. Friedman. The bayesian structural EM algorithm. In *Proceedings of the Fourteenth Conference on Uncertainty in Artificial Intelligence (UAI)*, pages 129–138, 1998.

M. R. Garey and D. S. Johnson. *Computers and Intractability: A Guide to the Theory of NP-Completeness*. W. H. Freeman, 1979.

Eric A. Hansen. Solving POMDPs by searching in policy space. In *Proceedings of the Fourteenth Conference on Uncertainty in Artificial Intelligence (UAI)*, pages 211–219, 1998.

J. Hoey, C. Boutilier, P. Poupart, P. Olivier, A. Monk, and A. Mihailidis. People, sensors, decisions: Customizable and adaptive technologies for assistance in healthcare. *ACM Transactions on Interactive Intelligente Systems*, 2(4):20:1–20:36, 2013.

R. A. Howard and J. E. Matheson. Influence diagrams. In *Readings on the Principles and Applications of Decision Analysis*, pages 721–762. Strategic Decisions Group, 1984.

J. Huang, M. Chavira, and A. Darwiche. Solving MAP exactly by searching on compiled arithmetic circuits. In *Proceedings of the 21st National Conference on Artificial Intelligence (NCAI)*, pages 1143–1148, 2006.

J. S. Ide, F. G. Cozman, and F. T. Ramos. Generating random Bayesian networks with constraints on induced width. In *In Proceedings of the 16th Eureopean Conference on Artificial Intelligence (ECAI)*, pages 323–327, 2004.

F. V. Jensen and T. D. Nielsen. *Bayesian Networks and Decision Graphs*. Information Science and Statistics. Springer, 2^{nd} edition, 2007.

J. Jiang, P. Rai, and H. Daume III. Message-passing for approximate MAP inference with latent variables. In *Advances in Neural Information Processing Systems 24 (NIPS)*, pages 1197–1205, 2011.

L. P. Kaelbling, M. L. Littman, and A. R. Cassandra. Planning and acting in partially observable stochastic domains. *Artificial Intelligence*, 101(1–2):99–134, 1998.

G. Kalai and G. N. M. Ziegler. *Polytopes: Combinatorics and Computation*. DMV Seminar. Birkhäuser Verlag, 2000.

D. G. Kendall. Foundations of a theory of random sets. In E. Harding and D. G. Kendall, editors, *Stochastic Geometry*, pages 322–376. John Wiley & Sons, 1974.

U. Kjaerulff. Triangulation of graphs: Algorithms giving small total state space. Technical Report R-90-09, Depatment of Mathematics and Computer System, Aalborg University, 1990.

J. Kohlas. *Information Algebras: Generic Structures for Inference*. Springer-Verlag, 2003.

D. Koller and N. Friedman. *Probabilistic Graphical Models*. MIT press, 2009.

J. Kwisthout and L. C. van der Gaag. The computational complexity of sensitivity analysis and parameter tuning. In *Proceedings of the 24th Conference in Uncertainty in Artificial Intelligence (UAI)*, pages 349–356, 2008.

C.-K. Kwoh and D. F. Gillies. Using hidden nodes in Bayesian networks. *Artificial Intelligence*, 88(1–2):1–38, 1996.

S. L. Lauritzen and D. Nilsson. Representing and solving decision problems with limited information. *Management Science*, 47:1235–1251, 2001.

S. L. Lauritzen and D. J. Spiegelhalter. Local computations with probabilities on graphical structures and their application to expert systems. *Journal of the Royal Statistical Society (Series B)*, pages 157–225, 1988.

I. Levi. *The Enterprise of Knowledge*. MIT Press, 1980.

Q. Liu and A. Ihler. Variational algorithms for marginal MAP. In *Proceedings of the 27th Conference on Uncertainty in Artificial Intelligence (UAI)*, pages 453–462, 2011.

Q. Liu and A. Ihler. Belief propagation for structured decision making. In *Proceedings of the 28th Annual Conference on Uncertainty in Artificial Intelligence (UAI)*, pages 523–532, 2012.

D. D. Mauá and C. P. de Campos. Solving decision problems with limited information. In *Advances in Neural Information Processing Systems 24 (NIPS)*, pages 603–611, 2011.

D. D. Mauá and C. P. de Campos. Anytime marginal MAP inference. In *Proceedings of the 28th International Conference on Machine Learning (ICML)*, 2012.

D. D. Mauá, C. P. de Campos, and M. Zaffalon. Solving limited memory influence diagrams. *CoRR*, abs/1109.1754, 2011.

D. D. Mauá, C. P. de Campos, and M. Zaffalon. A fully polynomial time approximation scheme for updating credal networks of bounded treewidth and number of variable states. In *Proceedings of the Seventh International Symposium on Imprecise Probability: Theories and Applications*, pages 277–286, 2011.

D. D. Mauá, C. P. de Campos, and M. Zaffalon. Solving limited memory influence diagrams. *Journal of Artificial Intelligence Research*, 44:97–140, 2012.

D. D. Mauá, C. P. de Campos, and M. Zaffalon. The complexity of approximately solving influence diagrams. In *Proceedings of the 28th Conference on Uncertainty in Artificial Intelligence (UAI)*, pages 604–613, 2012a.

D. D. Mauá, C. P. de Campos, and Marco Zaffalon. Updating credal networks is approximable in polynomial time. *International Journal of Approximate Reasoning*, 53(8):1183–1199, 2012b.

D. D. Mauá, C. P. de Campos, A. Benavoli, and A. Antonucci. On the complexity of strong and epistemic credal networks. In *Proceedings of the 29th Conference on Uncertainty in Artificial Intelligence (UAI)*, pages 391–400, 2013.

C. Meek and Y. Wexler. Approximating max-sum-product problems using multiplicative error bounds. *Bayesian Statistics*, 9:439–472, 2011.

N. Meuleau, K.-E. Kim, L. P. Kaelbling, and A. R. Cassandra. Solving POMDPs by searching the space of finite policies. In *Proceedings of the 15th Conference on Uncertainty in Artificial Intelligence (UAI)*, pages 417–426, 1999.

E. Miranda and S. Destercke. Extreme points of the credal sets generated by elementary comparative probabilities. In *Proceedings of the 12th European Conference on Symbolic and Quantitative Approaches to Reasoning with Uncertainty (ECSQARU)*, volume 7958, pages 424–435, 2013.

T. D. Nielsen and F. V. Jensen. Well-defined decision scenarios. In *Proceedings of the 15th Conference on Uncertainty in Artificial Intelligence (UAI)*, pages 502–511, 1999.

D. Nilsson and M. Höhle. Computing bounds on expected utilities for optimal policies based on limited information. Research Report 94, Dina, 2001.

J. D. Park and A. Darwiche. Solving MAP exactly using systematic search. In *Proceedings of the 19th Conference Conference on Uncertainty in Artificial Intelligence (UAI)*, pages 459–468, 2003.

J. D. Park and A. Darwiche. Complexity results and approximation strategies for MAP explanations. *Journal of Artificial Intelligence Research*, 21:101–133, 2004.

A. Paz and S. Moran. Non deterministic polynomial optimization problems and their approximations. *Theoretical Computer Science*, 15(3):251–277, 1981.

J. Pearl. *Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference*. Morgan Kaufmann, 1988.

A. Piatti, A. Antonucci, and M. Zaffalon. *Building Knowledge-Based Systems by Credal Networks: A Tutorial*, page 0. Nova Science, 2010.

P. Poupart and C. Boutilier. Bounded finite state controllers. In *Advances in Neural Information Processing Systems 16 (NIPS)*, 2003.

A. Salvetti, A. Antonucci, and M. Zaffalon. Spatially distributed identification of debris flow source areas by credal networks. In M. Sanchez-Marrè, J. Béjar, J. Comas, A. Rizzoli, and G. Guariso, editors, *Transactions of the 4th Bieenial Meeting of the International Congress on Environmental Modelling and Software Integrating Sciences and Information Technology for Environmental Assessment and Decision Making (iEMSs)*, pages 380–387, Manno, Switzerland, 2008. iEMSs.

R. D. Shachter. Bayes-ball: The rational pastime (for determining irrelevance and requisite information in belief networks and influence diagrams). In *Proceedings of the Fourteenth Conference on Uncertainty in Artificial Intelligence (UAI)*, pages 480–487, 1998.

R. D. Shachter. Efficient value of information computation. In *Proceedings of the 15th Conference on Uncertainty in Artificial Intelligence (UAI)*, pages 594–601, 1999.

G. Shafer. *A Mathematical Theory of Evidence*. Princeton University Press, 1976.

P. P. Shenoy. Binary join trees for computing marginals in the Shenoy-Shafer architecture. *International Journal of Approximate Reasoning*, 17(2–3):239–263, 1997.

P. P. Shenoy and G. Shafer. Axioms for probability and belief-function propagation. In *Proceedings of the Fourth Annual Conference on Uncertainty in Artificial Intelligence (UAI)*, pages 169–198, 1988.

S. E. Shimony. Finding MAPs for belief networks is NP-hard. *Artificial Intelligence*, 68:399–410, 1994.

J. A. Tatman and R. D. Shachter. Dynamic programming and influence diagrams. *IEEE Transactions on Systems, Man and Cybernetics*, 20(2):365–379, 1990.

B. Tessem. Interval probability propagation. *International Journal of Approximate Reasoning*, 7(3–4):95–120, 1992.

S. Thrun, W. Burgard, and D. Fox. *Probabilistic Robotics*. Intelligent robotics and autonomous agents series. The MIT Press, 2005.

F. van den Eijkhof, H. L. Bodlaender, and M. C. A. Koster. Safe reduction rules for weighted treewidth. *Algorithmica*, 2(47):139–158, 2007.

P. Walley. *Statistical Reasoning with Imprecise Probabilities*. Chapman and Hall, 1991.

P. Walley. Towards a unified theory of imprecise probability. *International Journal of Approximate Reasoning*, 24(2–3):125–148, 2000.

A. Wallner. Extreme points of coherent probabilities in finite spaces. *International Journal of Approximate Reasoning*, 44(3):339–357, 2007.

Y. Wang, N. L. Zhang, T. Chen, and L. K. M. Poon. LTC: A latent tree approach to classification. *International Journal of Approximate Reasoning*, 54(4):560–572, 2013.

X. Wu, A. Kumar, and S. Zilberstein. Influence diagrams with memory states: Representation and algorithms. In *Proceedings of the Second International Conference on Algorithmic Decision Theory*, pages 306–319, 2011.

M. Yannakakis. Computing the minimum fill-in is NP-complete. *Journal on Algebraic and Discrete Methods*, 2(1):77–79, 1981.

C. Yuan and E.A. Hansen. Efficient computation of jointree bounds for systematic MAP search. In *Proceedings of the 21st International Joint Conference on Artificial Intelligence (IJCAI)*, pages 1982–1989, 2009.

C. Yuan, T.-C. Lu, and M. J. Druzdzel. Annealed MAP. In *Proceedings of the 20th Conference Conference on Uncertainty in Artificial Intelligence (UAI)*, pages 628–635, 2004.

C. Yuan, X. Wu, and E. A. Hansen. Solving multistage influence diagrams using branch-and-bound search. In *Proceedings of the 26th Conference on Uncertainty in Artificial Intelligence (UAI)*, pages 691–700, 2010.

L. A. Zadeh. Fuzzy sets as a basis for a theory of possibility. *Fuzzy Sets and Systems*, 1(1):3–28, 1978.

M. Zaffalon. Credible classification for environmental problems. *Environmental modelling and software*, 20(8):1003–1012, 2005.

M. Zaffalon and E. Miranda. Conservative inference rule for uncertain reasoning under incompleteness. *Journal of Artificial Intelligence Research*, 34:757–821, 2009.

M. Zaffalon, K. Wesnes, and O. Petrini. Reliable diagnoses of dementia by the naive credal classifier inferred from incomplete cognitive data. *Artificial Intelligence in Medicine*, 29(1–2):61–79, 2003.

M. Zaffalon, G. Corani, and D. D. Mauá. Utility-based accuracy measures to empirically evaluate credal classifiers. In *Proceedings of the Seventh International Symposium on Imprecise Probability: Theories and Applications (ISIPTA)*, pages 401–410, 2011.

M. Zaffalon, G. Corani, and D. D. Mauá. Evaluating credal classifiers by utility-discounted predictive accuracy. *International Journal of Approximate Reasoning*, 53(8):1282–1301, 2012.

N. L. Zhang. Hierarchical latent class models for cluster analysis. *Journal of Machine Learning Research*, 5:697–723, 2004.

N. L. Zhang, R. Qi, and D. Poole. A computational theory of decision networks. *International Journal of Approximate Reasoning*, 11(2):83–158, 1994.