Multivariate Analysis Applied in Bayesian Metareasoning

CARLOS EDUARDO BOGNAR; OSAMU SAOTOME
Department of Computer Science
University IMES
Av. Goias, 3400, São Caetano do Sul, São Paulo
BRAZIL
carlos@uscs.edu.br http://www.uscs.edu.br

Abstract: - As Bayesian networks are applied to more complex and realistic real-world applications, the development of more efficient inference approaches is increasingly important. This paper presents a method for metareasoning in Bayesian networks adopting prediction models to select algorithms for the inference tasks, when multiple schemes are used to calculate the propagation of evidence. The proposed method is based on multiple characterizations of Bayesian networks and prediction models to select the algorithm that will provide the best performance in future inferences. Logistic regression analysis is applied to determine when exact algorithms may be used for specific tasks. The prediction models of approximate inference algorithms are created by multiple regression analysis, based on experimental results using Variable Elimination, Gibbs Sampling and Stratified Simulation algorithms. These algorithms belong to exact method, stochastic and deterministic sampling methods, respectively. Experimental analyses compare some alternative models and show better results when multivariate analysis is applied.

Key-Words: Bayesian, Metareasoning, Logistic Regression, Multiple Regression, Inferences, Prediction Models

1 Introduction

Bayesian networks (BNs) have become a popular representation for reasoning under uncertainty, as they integrate a graphical representation of causal relationships with Bayesian foundation, providing both a compact method to represent probability distributions and a powerful tool for uncertainty management.

BNs are directed acyclic graphs in which every node is associated with a random variable X_i and edges represent conditional dependencies between variables. In this paper every variable is categorical (has a finite number of values), and the terms "node" and "variable" are used interchangeably.

The nodes in network are connected by directed arcs, which may be thought of as causal or influence links. The parent of node X_i are the nodes with direct edges pointing to X_i, indicated by $pa(X_i)$.

Each node has associated with it a probability distribution, which, for each combination of the variables of parent nodes (called a conditioning case), gives a probability of each value of the node variable. The probability distribution for a node with no predecessors is the prior distribution. The relationship between any set of state variables can be specified by a joint probability distribution.

Every variable in *BN* is assumed to be independent of its *nonparents nondescendants* given its parents, implying the following joint probability distribution [45]:

$$\Pr(X_1,\ldots,X_n)=\prod_{i=1}^{n}\Pr(X_i \mid pa(X_i)) \quad \textbf{(1)}$$

That is, a *BN* represents a unique joint distribution that factorizes as Expression (1). Every variable is thus associated with a single conditional distribution $Pr(X_i|pa(X_i))$.

Figure 1 shows an example of *BN*, in which the associated probability distributions are *Pr(A)*, *Pr(B)*, *Pr(C|A, B)*, *Pr(D|A)*, *Pr(E|C)*, *Pr(F|E)*, *Pr(G|D, E)*.

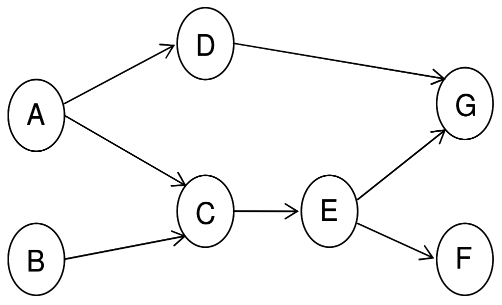

Figure 1. A example of Bayesian network.

BNs have been used in a variety of applications including medical diagnosis [1, 2], technical support troubleshooters [32], decision-theoretic systems to interpret live telemetry [31], genetic research [22], speech recognition systems [52], data compression methods [16], and diagnostic systems in industrial plants [47]. Such approaches involve determining the structure of the network; supplying the prior probabilities for root nodes and conditional probabilities for other nodes; adding or retracting evidence about nodes; repeating the inference algorithm for each change in evidence.

Given a *BN*, the computation of a posterior probability distribution is usually called an *inference*. That is, we select a set of *query* variables X_Q and a set of *observed* variables X_E, and we must compute

$$\Pr(X_Q \mid X_E) = \frac{\sum_{X_i \notin \{X_Q, X_E\}} \prod \Pr(X_i \mid pa(X_i))}{\sum_{X_i \notin \{X_E\}} \prod \Pr(X_i \mid pa(X_i))} \quad \textbf{(2)}$$

Evidence can be specified about the state of any of the nodes in the network: root nodes, leaf nodes or intermediate nodes. Evidence of specific values of some nodes is incorporated into decision-making by belief updating of other nodes of interest, the query nodes. We assume that X_Q and X_E are disjoint, and we note that in Expression (2) the values of variables in X_E are observed and therefore fixed.

Some algorithms can compute the expression (2) exactly, named *exact algorithms*, and can be classified in two groups: algorithms based on conditioning, and algorithms based on clustering — with a “third group” represented by Pearl’s propagation algorithm for polytrees, the only polynomial exact inference algorithm for *BN* [48].

- The *cutset conditioning* algorithm, also known as the *loop cutset* algorithm, exploits the fact that edges out of a node are “broken” if the node is observed. The algorithm selects a set of nodes (the *loop cutset*) that, once observed, “breaks” every cycle in a graph. Every instantiation of the cutset is then considered; for each one of them, Pearl’s propagation algorithm is employed. The result is an algorithm that uses a relatively small amount of memory, but takes exponential time on size of the loop cutset.
- In *clustering* algorithms, variables are grouped in potentially large clusters, a junction tree is built, and a propagation scheme on junction trees produces inferences. The Lauritzen-Spiegelhalter algorithm [37] and Shafer-Shenoy algorithm [50] are two different ways to organize this propagation. Many variants of clustering methods have appeared since these two basic algorithms were derived [27]. A few algorithms also proceed by “grouping” variables but are not directly related to the Lauritzen-Spiegelhalter or the Shafer-Shenoy algorithms: the family of *Variable Elimination (VE)* algorithms, Li and D’Ambrosio’s *SPI* algorithm [38], Shachter’s *arc-reversal / node-reduction* algorithm [49], and *differential* inference algorithms [15] are examples. *VE* is an algebraic scheme for exact inferences in *BN*. The idea of the algorithm is to multiply the sets of density in the sequence provided by the ordering process and attempts to eliminate variables as soon as possible to maintain intermediate products at a manageable size [12].

Although polynomial time inference algorithms exist for singly connected networks, the general problem of computing exact inferences in Bayesian networks is NP-hard [10].

In many real-world domains, approximate schemes may be used for inferences. Rather than the exact calculation of Expression (2), representative samples of variables can be generated to provide an approximation [44]. Approximate algorithms for *BN* inference can be divided in a few groups [28].

- *Stochastic approximations* are widely used in large and dense networks. Methods are generally divided into forward sampling and MCMC methods, and they display poor performance when probability values are extreme [26]. *Gibbs Sampling (GS)* is a stochastic simulation algorithm and uses the total conditional probability of each variable to generate the samples to obtain approximated probabilities.
- *Model simplifications* range from the removal of weak dependencies to cardinality reduction in probability distributions [8]. Simplifications may also affect secondary structures such as junction trees, as demonstrated by the mini-buckets framework [20].
- *Partial instantiation* algorithms approximate the summation in Expression (2) using only a number of terms. Examples are *bounded conditioning* [30], and *term computation* [13], Poole’s *conflict-based* [46] and Henrion’s *search-based* methods [29]. *Stratified Simulation (SS)* was initially suggested by Bouckaert [5], who presented several variants. These algorithms give more representative samples and, due to the possibility of efficient implementation, are faster than the previously known simulation algorithms [5]. The most commonly used version chooses a

deterministic sample consisting of equally spaced sample values.

- *Loopy propagation* uses Pearl's propagation algorithm in network with cycles, attempting to gradually improve the quality of inferences [42].

However, the problem of approximating the probability, given the evidence, when a constant error-bound is required, has also been proven to be NP-Hard [14]. Given the NP-hard complexity results, one of the major challenges in applying *BNs* in real-world applications is the design of efficient inference approaches for very large probabilistic models.

In the last decade, researchers have investigated a new tactic for knowledge processing [17][18][19]. Several algorithms are used to arrive at a solution with better quality than a single algorithm working alone. Each algorithm has strengths and weaknesses in each case.

When several algorithms may be applied to solve some inference instance, it is important to decide which algorithm should be used to achieve the best results. The algorithm selection problem is originally formulated by Rice [48]. Later it has been mainly applied to the selection of problem-solving method in scientific computing [34], specifically to the performance evaluation of numerical softwares.

The algorithm selection problem asks the following question: *which algorithm should be selected to solve this problem instance?*

Algorithm or problem-solving technique selection has been studied in various fields, including data compression [35], machine learning [7], planning [23] and Constraint Satisfaction Problem (CSP) [41]. In some works the knowledge representation of the "metareasoner" is often a set of rules, a statistical models or machine leaning approaches.

Researchers in AI have long realized that great efficient gains can be achieved by allocating a portion of costly computational resources to meta-level deliberation about the best way to solve a problem. In the particular field of *BN* inference, the quality of approximate inferences varies from each problem instance, because the input data manipulated contain characteristics that affect the performance of the algorithms, such as *BN* structure, parameters distribution and characteristics of the queries. If these features of the data can be associated together, they can be used to predict the performance of algorithms for that information, in metareasoning process.

To perform metareasoning, this paper proposes a method for *BN* characterization and associates these characteristics together, applying multivariate analysis. Based on experimental data, the proposed method creates the prediction models for *VE*, *GS* and *SS* algorithms. The models are used to predict the performance and select these algorithms in future inference tasks.

In section 2, we review some approaches used in metareasoning. In section 3, the proposed method is presented. Results comparing prediction models are presented in section 4. Section 5 shows the conclusions.

2 Metareasoning Approaches

In AI context, an agent acting in the world generally needs to spend some time and other resources on *deliberation*, to assess the quality of the various plans of action available to it. To find the absolutely optimal plan, an agent generally needs to perform a very large amount of deliberation: it has to consider all the relevant implications of all the relevant facts that it knows, and, if the agent is able to gather additional information, it also has to take all relevant information gathering actions.

Metareasoning may be defined as the process of reasoning about reasoning itself and it is composed of both meta-level control of computational activities and the introspective monitoring of reasoning to evaluate and to explain computation. Meta-level control is the ability of an agent to efficiently trade off its resources between object level actions (computations) and ground level actions to maximize the quality of its decisions. Figure 2, taken from Cox and Raja [11], illustrates the three different levels of doing (plans of action), reasoning (deliberation actions), and metareasoning.

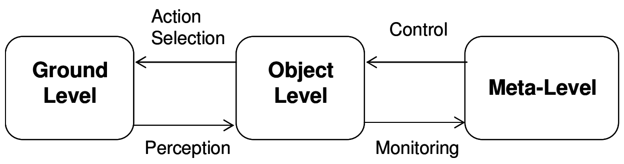

Figure 2. The three levels of doing, reasoning, and metareasoning.

Conitzer [9] reported that optimal metareasoning is not computationally feasible in general, and it is important to consider approximation methods, heuristics and anytime algorithms that find close-to-optimal solutions to the metareasoning problem, or algorithms that find the optimal solution fast under certain conditions, as well as more involved meta-metareasoning approaches.

Artificial Intelligence researchers proposed Multiple Method (MM) approaches to perform inferences tasks in large Bayesian networks [25]. *MM* involves a number of different algorithms, each of which is capable of generating solutions having different characteristics [24]. When these algorithms are combined to work in Bayesian inferences, the main problem that arises is related to the quality of results. If possible, the exact schemes are preferred. When exact algorithms could not be applied, approximated approaches must be used and its results depend on some problem instances. In order to decide which algorithm will provide the highest quality results, when multiple methods are applied, a metareasoning process may be implemented.

In many real world situations, an algorithm selection is performed manually by certain experts who have a strong theoretical understanding of the computational complexities of various algorithms and are very familiar with their behaviors. The difficulty of automatic algorithm selection is largely due the uncertainty in the input problem space, the lack of understanding the working mechanism of algorithm space, and the uncertainty factors of implementations and run-time environments [28].

Zilberstein [51] initially investigates the main control problems that arise when a system is composed of several algorithms. This problem relates to optimal management of uncertainty and precision.

Jitnah and Nicholson [36] studied a metareasoning method based on *BN* characterization, such as number of nodes, instantiated nodes and the queried nodes. The measured characteristics were used to predict which algorithm would provide better performance in terms of execution time and accuracy of results on the given problem. Measures taken from the network were related to the number of nodes and arcs, the connectedness, path lengths, overall skewness of the conditional probability distributions and the maximum and average number of states, of parents and the Markov blanket size of any node. Based on these characteristics and utility curves, performance profiles were created applying Simple Regression Analysis (SRA).

Borghetti [4] also studied this problem and demonstrated that some individual *BN* characteristics may be useful to predict the quality of results. In his work, individual characteristics were analyzed and performance profiles were created.

Guo [28] proposed a method to perform metareasoning in Most Probable Explanation problems, based on certain parametric and structural characteristics of networks. The prediction models were used to select the algorithm that probably offered the best performance and were obtained by machine learning techniques.

In our proposed method, an alternative approach is used. Multiple characteristics of *BN* are considered and combined using multiple regression analysis to create the prediction models for inference algorithms in metareasoning problems. Once the prediction models have been created, the metareasoning process can selects the algorithm for some inference instance that may provide best quality of results (exact or approximate probabilities).

3 The Proposed Method

Herein a method is presented for using performance data gathered off-line to predict future behavior of algorithms on new instances. First, it is necessary to know whether exact methods may be used for the specific inference task. If not, the predicted relative error of inferences, when approximated methods are applied, allows selecting the algorithm that is most likely to have better performance in some inference instances. To predict the performance, mathematic relations must be developed, describing how the values of multiple characteristics (independent variables) influence the results.

Initially, a set of "*Relevant Characteristics*" (RC) are identified and samples must be randomly generated, exhibiting a variety of values over all of these *RC*. Consequently, performance data are gathered on networks when *RC* varies. These data are used to create the prediction models of algorithms, obtained by multivariate analysis. Finally, these models are used in the metareasoning process to select the appropriate algorithm for the task. The details are described as follows.

3.1 Selecting Relevant Characteristics

This process identifies a set of characteristics that may affect algorithm performance. To be "relevant", a characteristic must be related to different performance of algorithms when its domain is modified.

These characteristics must also be easily computed, because if the time to compute takes as long as executing the inference algorithm, it would be unwise to invest the time to compute it.

Each characteristic examined in this paper was proposed in previous publications [4, 28, 36], nonetheless, it should be pointed out that this method may be applied to analyze other relevant characteristics. The selected characteristics are:

- **Number of Nodes *(X_1)*.** This characteristic is easy to compute and represents the number of random variables in *BN*.

- **Connectedness *(X_2)*.** Connectedness is defined by Expression (3), where *L* denotes the number of edges and N denotes the number of nodes.

$$X_2 = \frac{2L}{(N(N-1))} \tag{3}$$

- **Maximum Number of States *(X_3)*.** Maximum number of states found on a node, considering all nodes.

- **Maximum CPT Size *(X_4)*.** A node's *CPT* size is the number of cells in its Conditional Probability Table, which is the product of the state space of its parents, multiplied its own state space.

- **Maximum Number of Parents *(X_5)*.** It corresponds to the maximum number of parents of a node, considering all nodes on *BN*.

- **Number of Roots *(X_6)*.** A root is a node without parents.

- **Average Skewness *(X_7)*.** Skewness is a metric proposed by Jitnah and Nicholson [36] and it is computed for each random variable, representing the asymmetry of its *CPT*. Borghetti [4] defined the Expression (4) to compute skewness:

$$SK(v) = \frac{\sum_{i=1}^{m} \left| \frac{1}{m} - v_i \right|}{\frac{m-2}{n_0} + 1} \tag{4}$$

where $v=(V_1,\ldots V_m)$ is the *CPT* for a node and n_0 represents the number of states of this node.

- **Maximum Skewness *(X_8)*.** The maximum skewness occurs when members of tables are either 0.0 or 1.0.

- **Number of Interactions *(X_9)*.** The number of simulation executed by each approximated method.

- **Evidence Proportion *(X_{10})*.** Let the number of evidence nodes be *n_evid* and number of nodes is X_1. The evidence proportion is simply

$$X_{10} = \frac{n\_evid}{X_1} \tag{5}$$

Note that the number of interactions *(X_9)* and evidence proportion *(X_{10})* represent the characteristics of inferences tasks and is useful in prediction models.

3.2 Gathering Performance Data

After choosing which features are likely to affect the performance of algorithms, the next step must gather networks that exhibit a range of values for each characteristic.

It is important to have a method that generates *BNs* uniformly; that is, we would like to guarantee that averages taken with generated networks produce unbiased estimates.

The generation methods must also be flexible in the sense that constraints on generated networks can be added with relative ease. For example, it should be possible to add a constraint on the maximum number of parents for nodes, the average number of children, or the maximum number of states. Ad hoc methods are usually concocted for a particular set of constraints, and it is hard to imagine ways to add constraints to them.

Finally, the method must generate "realistic" networks, however hard it may be to define what a "real" Bayesian network is. A reasonable strategy is to look for properties that are commonly used to characterize *BNs*, and to allow some control over them.

In this experiment, the random generation of *BN* instances with controlled parameter values was based on the *Markov-chain* method, to "walk" randomly in the space of all possible networks that satisfy the constraints. This feature was implemented by *BNGenerator* [3], a program that generates random *BNs*, guaranteeing that the distribution of generated networks is asymptotically uniform.

BNGenerator was used to create 150 networks for the estimation sample. To generate *BNs* with specific constraints, an ergodic *Markov* chain was constructed with uniform limiting distribution, such that every state of the chain is a *DAG* satisfying the constraints. The *BNGenerator* accepts specification of number of nodes, maximum node degree, maximum number of edges, and maximum heuristic width. The software also performs uniformity test using χ^2 test.

The parameters specified in generation process were based on some real *BNs* and to simulate instances where exact algorithms could not be executed, the generation process considered higher values in its parameterization. Table (1) shows the domain values of generated networks.

Table 1. Domain values of generated *BNs*.

Domain	X_1	X_2	X_3	X_4	X_5	X_6	X_7	X_8
Min	2	0.01	2	8	2	1	0.10	0.10
Max	300	0.80	13	1048576	10	10	0.99	0.99

The next phase collects performance data for each algorithm. The *VE*, *GS* and *SS* algorithms were executed individually on each *BN* and provided a set of solutions. The evidence proportion *(X_{10})* was randomly defined from 0.0 to 0.3.

When exact method could be executed, *BN* characteristics were recorded. In instances where a *VE* algorithm could not be executed due to *memory-overflow* error, approximate methods were applied.

In each run of a specific approximate algorithm, the number of interactions was randomly defined from 1,000 to 100,000 and relative errors of inferences were collected. The relative errors were calculated for each state of each variable, from its approximate probabilities distribution and the exact. Therefore, the average relative errors, considering all states of all variables were calculated and recorded in a database.

Database samples used in multivariate analysis must consider two assumptions: (1) no outliers and (2) normal distribution of dependent variables.

An outlier is an observation that is numerically distant from the rest of the data. Statistics derived from databases that include outliers will often be misleading. However, a small outlier number is expected in normal distributions. The rejection of suspect observations must be based on an objective criterion, such as the *Dixon´s Q-Test* [21].

The *Kolmogorov-Smirnov* (KS) and *Lilliefors* tests were applied for normality testing [39]. The *two-sample KS* test is one of the most useful general nonparametric method and goodness of fit used to determine whether the underlying one-dimensional probability distribution differ from a hypothesized distribution (normal).

The results of *KS* test showed that the distributions of relative errors were not normal (2-tailed = 0.003 for *GS* and *SE*). In such situations mathematical transformations may be useful.

In statistics, the power transform is a family of transformations that map data from one space to another using power functions. This is a useful data (pre)processing technique used to reduce data variation, make the data more normal distribution-like, and improve the correlation between variables and for other data stabilization procedures.

The *Box-Cox* transformation [6] is one particular way of applying a power transform that has advantageous properties. The power transformation is defined as a continuously varying function, with respect to the power parameter λ, in a piece-wise function form that makes it continuous at the point of singularity (λ = 0). For all arguments y > 0, the analytic form of the power transform function is expressed as

$$Y^{\lambda} = \begin{cases} \left(Y^{\lambda} - 1\right)/\lambda & \text{if} \quad \lambda \neq 0 \\ \log y & \text{if} \quad \lambda = 0 \end{cases} \qquad \textbf{(6)}$$

The *Box-Cox* transformations were applied to database samples and the distributions of relative errors of approximations became normal (2-tailed = 0.526 for *GS* and 0.491 for *SE*).

3.3 Obtaining Prediction Models

In the next step, mathematical relations must be obtained and used in the decision process for selecting which algorithm will run under specific instances. Once the relation has been created, it can be used to predict which algorithm will perform better in a given instance.

In this approach, the relations combine the effects of multiple characteristics on the results of exact and approximate algorithms, which may be done by Logistic Regression (LR) and Multiple Regression Analysis (MRA).

LR is a form of regression used when the dependent is a dichotomy or *logit* variable, predicted on the basis of independents.

The proposed approach applies maximum likelihood to estimate the probability that the exact algorithm may be executed, based on *(X_1,… , X_5)*. Equation (7) shows the generic expression, where P(Y_{exact}) is the probability that *VE* may be executed and *(β_0,…, β_n)* are regression parameters.

$$\frac{P(Y_{exacte})}{1 - P(Y_{exact})} = e^{\beta_0 + \beta_1 X_1 + \ldots + \beta_n X_n} \qquad \textbf{(7)}$$

Table (2) summarizes the logistic model, showing the regression coefficients, the standard error (SE) and *Wald* statistics to test each coefficient. The coefficients are significant at level 0.01. Some statistics related to goodness of fit are also listed.

Likelihood is the probability that observed values of the dependent may be predicted from observed values of independents. The (-2LL) is its *log* and is the basis for tests in logistic regression. Additionally, *Cox and Snell* R^2 [40], *Nagelkerke* R^2 [43], *Hosmer and Lemeshow* [33] and percent of correct classifications are listed.

Table 2. Logistic regression model (estimation sample).

Coefficients					(-2LL)	Cox & Snell	Nagelkerke	Hosmer and Lemeshow		%Correct
Variable	*(B)*	*SE*	*Wald*	*Signif.*		*(R^2)*	*(R^2)*	*Chi-Quad*	*Signif.*	*Classif.*
Constant	*14,346*	*5,495*	*6,816*	*0,010*	*18,722*	*0,667*	*0,897*	*2,685*	*0,953*	*95,8%*
X_1	*-0,154*	*0,059*	*6,853*	*0,010*						
X_2	*17,264*	*16,951*	*1,037*	*0,308*						
X_3	*-0,813*	*0,339*	*5,75*	*0,016*						
X_4	*0,000*	*0,000*	*1,624*	*0,203*						
X_5	*-1,218*	*0,832*	*2,144*	*0,043*						

When P(Y_{exact})>0.5, *VE* algorithm must be selected by metareasoning process. Otherwise, approximate methods are applied, based on prediction models calculated by Multiple Regression Analysis (MRA).

The general purpose of *MRA* is to estimate the strength of a modeled relationship between one or more predicted variables and the predictors. For this propose, *MRA* was used to predict the relative error provided by an approximate algorithm, considering the values of all predictors $(X_1\ldots, X_{10})$.

Equation (8) shows a mathematical relation obtained by *MRA*, where *Y* denotes the relative error, *a* is the intercept and b_i denotes the regression parameter, which reflects the partial effect of the associated independent variable X_i to prediction of the dependent.

$$Y = a + b_1 X_1 + \dots + b_n X_n \tag{8}$$

Table (3) summarizes the model and shows regression coefficients β_i for each independent variable and its standard errors (SE).

Table 3. Multiple regression models.

Predictors	**Gibbs Sampling**				**Stratified Simulation**			
	β	***Standard Error***	***(R^2)***	***SEE***	***β***	***Standard Error***	***(R^2)***	***SEE***
(Constant)	*19,669*	*3,894*	*0,735*	*1,32275*	*-0,173*	*0,300*	*0,769*	*0,10181*
X_1	*-0,080*	*0,009*			*0,001*	*0,001*		
X_2	*-4,760*	*1,130*			*-0,245*	*0,087*		
X_3	*0,108*	*0,083*			*0,015*	*0,006*		
X_4	*-5,20E-05*	*0,001*			*6,32E-06*	*0,001*		
X_5	*0,182*	*0,162*			*-0,034*	*0,012*		
X_6	*-0,507*	*0,085*			*-0,008*	*0,007*		
X_7	*-4,523*	*2,461*			*0,865*	*0,189*		
X_8	*-10,707*	*5,011*			*-0,212*	*0,386*		
X_9	*6,90E-05*	*0,001*			*-2,60E-07*	*0,001*		
X_{10}	*2,288*	*1,799*			*0,841*	*0,138*		

The *R-squared* (R^2), also called multiple correlation, is the percent of variance in the dependent explained jointly by the independents. It can also be interpreted as the number of errors made when using the regression model to guess the value of the dependent, in ratio to the total errors made using only the dependent's mean as a basis for estimating all cases.

4 Experimental Results

To assure that the regression models are appropriate for metareasoning problems, this section presents some experimental results.

To allow for comparing the proposed approach with different metareasoning strategies [5, 36], individual characteristics were also mapped in mathematical equations, applying simple regression analysis (linear and non-linear) to estimate prediction models for *GS* and *SE*.

In this phase, 110 networks were randomly generated, over a variety of values for all characteristics and metrics of these characteristics were collected.

Initially, the values of network characterization and logistic regression models were used to predict when the *VE* algorithm should be used, considering this test sample. Table (4) shows the main statistics, including the percent of variation in estimation and test samples.

Table 4. Logistic model applied to a new sample.

(-2LL)	**Cox & Snell**	**Nagelkerke**	**Hosmer and Lemeshow**		**%Correct**
	(R^2)	*(R^2)*	*Chi-Quad*	*Signif.*	*Classification*
18.722	*0.667*	*0.897*	*2.685*	*0.953*	*95.8%*
16.998	*0.607*	*0.863*	*2.628*	*0.956*	*91.8%*
-10.1%	***-9.9%***	***-3.9%***	***-2.2%***	***0.3%***	***-4.2%***

For the test sample, the logistic model could correctly classify **91.8%** of all observations. That is, in only 9 of 110 observations, the predictions of *VE* execution were incorrect. In Guo´s approach [28], **60,0%** of total observations were correctly classified by machine learning techniques.

In 40 test sample observations, a *VE* algorithm could not be executed because *out-of-memory* error. In these cases, the metareasoning process selected the *GS* or *SS* algorithms to perform inference tasks, based on multiple regression models and *BN* characterizations. The relative errors of approximate inferences were calculated for each node on *BN* and the average relative errors were calculated considering the entire networks.

To allow for comparing with different metareasoning strategies, simple linear regression analysis (SRA) and non-linear regression (NLR) were applied to obtain alternatives prediction models. For *SRA* and *NLR*, only independent variables with significant correlations were considered.

Table (5) shows the correlations, where "(*)" represents significance at the 0.01 level (2-tailed).

Table 5. Estimate correlations.

	X1	X2	X3	X4	X5	X6	X7	X8	X9	X10
Rel. Error (GS)	.205(*)	-0.0967	0.0233	0.02612	0.1298	-0.092	.203(*)	0.14413	-.465(*)	0.02255
Rel. Error (SS)	.418(*)	-.407(*)	0.1606	0.03057	0.1104	-0.181	.353(*)	0.11809	-.342(*)	.423(*)

Based on correlation results, alternative prediction models were obtained by *SRA* and *NLR* (power function), considering *(X_1, X_7 and X_9)*. All the prediction models were applied in metareasoning process for test sample (the 40 remaining observations). The results are shown in table (6), where "*" are incorrect classifications.

Table 6. Comparing different prediction models.

#	Observed Relative Error			Predicted (Meta-Inference)				
	(1=GS)	(2=SS)	Selected	(MRA)	(SRA $_{X1,X7,X9}$)	(NLR $_{X1}$)	(NLR $_{X7}$)	(NLR $_{X9}$)
1	0.0073	0.0016	2	1*	1*	1	2	1*
2	0.1181	0.0121	2	2	1*	2	1*	1*
3	0.0531	0.0729	1	1	1	2*	1	1
4	0.0471	0.1089	1	2*	1	2*	2*	1
5	0.0865	0.0004	2	1*	1*	2	2	1*
6	0.0370	0.0036	2	2	1*	2	1*	1*
7	0.0891	0.0144	2	2	1*	2	1*	1*
8	0.0174	0.0004	2	2	1*	2	1*	1*
9	0.0147	0.0529	1	1	1	2*	1	1
10	0.0260	0.0081	2	2	1*	2	1*	1*
11	0.0193	0.1089	1	1	1	2*	2*	1
12	0.1332	0.2601	1	1	1	2*	1	1
13	0.0230	0.0004	2	2	1*	2	1*	1*
14	0.0263	0.2809	1	1	1	2*	2*	1
15	0.0294	0.0625	1	1	1	2*	2*	1
16	0.0192	0.0676	1	2*	1	1	1	1
17	0.0392	0.1764	1	2*	1	2*	1	1
18	0.0320	0.0049	2	2	1*	2	1*	1*
19	0.0324	0.0320	2	2	1*	1*	1*	1*
20	0.0676	0.0505	2	2	1*	1*	1*	1*
21	0.0121	0.0196	1	1	1	1	1	1
22	0.0286	0.0529	1	1	1	1	1	1
24	0.0439	0.1296	1	1	1	1	1	1
25	0.0174	0.1521	1	1	1	1	1	1
26	0.0475	0.0784	1	2*	1	1	1	1
27	0.0144	0.0167	1	1	1	1	1	1
28	0.0309	0.0841	1	2*	1	1	1	1
29	0.0477	0.0729	1	2*	1	1	1	1
30	0.0501	0.0961	1	1	1	1	1	1
31	0.0841	0.0298	2	2	1*	1*	1*	1*
32	0.0484	0.0411	2	2	1*	1*	1*	1*
33	0.0092	0.0032	2	2	1*	1*	1*	1*
34	0.0215	0.1936	1	1	1	1	1	1
35	0.0164	0.2916	1	1	1	1	1	1
36	0.0389	0.2916	1	1	1	1	1	1
37	0.0204	0.0576	1	1	1	1	1	1
38	0.0092	0.1089	1	1	1	1	1	1
39	0.0180	0.0900	1	1	1	1	1	1
40	0.0505	0.1024	1	1	1	1	1	1
	% Correct Classification			77.5%	62.5%	65.0%	60.0%	62.5%

(*) Represents classification error

In table (6), the last row represents the percentage of correct classification made by metareasoning process, considering each prediction model. It is possible to see that better results were obtained when multiple regression models were applied. The percentage of correct classifications were 77.5% for multiple regression analysis, 62.5% for simple regression analysis using X_1, X_7 and X_9 as predictors, 65.0% for non-linear regression (power function) using X_1, 60.0% for non-linear regression (exponential function) using X_7 and 62.5% for non-linear regression (power function) using X_9 as predictor.

5 Conclusion

A number of exact and approximate *BN* inference algorithms exist. Each algorithm exploits different characteristics of the problem instances and works better for some classes of problem. This paper presents a method for Bayesian metareasoning to select an algorithm to perform the inference task, based on multiple characterizations of the networks. We have attempted to provide a relatively broad description of some factors involved in using this method, while keeping the exposition as simple and didactic as possible. Our goal was to propose a method that can add flexibility to probabilistic reasoning, getting into issues of metareasoning techniques.

We certainly make no claims that metareasoning is the *only* way to perform inferences in large and dense Bayesian networks. This approach is useful when there are bounded resources, such as execution time and memory restrictions. Given the large number of factors involved in such inferences, it is likely that no optimal algorithm exists, whatever is meant by optimal; we should instead focus on metareasoning strategies to select the appropriate algorithm for the task.

The experimental results show that multivariate analysis may be applied in metareasoning processes and is useful to predict the accuracy of selected approximate algorithms.

The benefits of this approach are that it is easy to understand and also provides a very clear indication as to which algorithm should be used based on prediction models. The metareasoning process rapidly computes using simple mathematical equations to predict the results.

While this research shows that predicting performance based on multiple domain characteristics is plausible, there are limitations to this approach. This method requires a suite of domain instances to be run for each algorithm so that the performance data can be generated and prediction models calculated. Furthermore, if any of the algorithms are modified, the entire test suite must be re-run. The development of new *BN* generators, specifically for this purpose, may improve the results.

The method is experimental in nature. No number of experiments is adequate to prove a theory; however, statistical analysis of large samples can show trends. At a higher level, multivariate analysis may be used in predicting the performance of any algorithm that has a performance profile that varies with changes in the characteristics of its input. With the capability of predicting the performance of multiple algorithms, the method presented may determine which algorithm will perform the best for a specific problem.

Future studies will be necessary to develop new efficient metareasoning approaches. Other multivariate techniques may be studied and applied to select the best algorithm for each instance, such as multivariate analysis of variance and covariance and discriminant analysis. However, integrating various kinds of inference algorithms into one unified system has been shown to be plausible and multiple regression analysis may be used for this goal.

References:

[1] Andreassen, S., Horvorka, R., Benn, J., A Model-Based Approach to Insulin Adjustment, *Proceeding of the 3rd Conference on Artificial Intelligence in Medicine*, 1991, pp. 239-248.

[2]Beinlich, I., Suermondt, H.J., Chavez, R.M., Cooper, G.F., The ALARM Monitoring System: A Case Study with Two Probabilistic Inference Techniques for Belief Networks, *Proceeding of the 2nd European Conference on Artificial Intelligence in Medicine*, 1989, pp. 247-256.

[3] http://www.pmr.poli.usp.br/ltd/soft/BNGenerator

[4] Borghetti, B.J., *Inference Algorithm Performance and Selection under Constrained Re-sources*, MS Thesis, AFIT/GCS/ENG/96D-05, 1996.

[5] Bouckaert, R., A Stratified Simulation Scheme for Inference in Bayesian Belief Networks, *Proceeding of the 10th Conference on Uncertainty in Artificial Intelligence*, 1994, pp. 110-117.

[6]Box, G. E., Cox, D. R., An Analysis of Transformation, *Journal of the Royal Statistical Society* 26, 1964, pp. 211252.

[7] Brodley, C., Addressing the Selective Superiority Problem: Automatic Algorithm/Model Class Selection, *Proceeding of the 10th International Conference on Machine Learning*, 1993, pp. 17-24.

[8] Cano, A., Moral, S., Using Probability Trees to Compute Marginals with Imprecise Probabilities, *International Journal of Approximate Reasoning,* 29, 2002, pp. 1-46.

[9] Conitzer, V., Metareasoning as a Formal Computational Problem, *Proceeding of the 23rd Conference on Artificial Intelligence*, 2008, pp. 29-33.

[10]Cooper,G.F., The Computational Complexity of Probabilistic Inference using Bayesian Belief Networks, *Artificial Intelligence* 42, 1990, pp. 393-405.

[11] Cox, M. T., and Raja, A., Metareasoning: A Manifesto, *Technical report BBN TM-2028*, BBN Technologies, 2007.

[12]Cozman, F. G., Generalizing Variable Elimination in Bayesian Networks, *Technical report*, University of São Paulo, 2000.

[13]D'Ambrosio, B., Incremental Probabilistic Inference, *Proceeding of the 9th Conference on Uncertainty in Artificial Intelligence*, 1993, pp. 301-308.

[14]Dagum, P., Luby, M., Approximating Probabilistic Inference in Bayesian Belief Networks is HP-Hard, *Artificial Intelligence*, 60, 1993, pp. 141-153 .

[15]Darwiche, A., Any-Space Probabilistic Inference, *Proceeding of the 16th Conference on Uncertainty in Artificial Intelligence*, 2000, pp. 133-142.

[16]Davies, S., Fast Factored Density Estimation and Compression with Bayesian Networks. *PHD Thesis*, School of Computer Science, Carnegie Mellon University, 2002.

[17] Dean, T., Solving Time-Dependent Planning Problems, *Technical report CS-89-03*, Brown University, 1989.

[18] Dean, T., Deliberation Scheduling for Time-Critical Sequential Decision Making, In *Proc. of the 9th Conference on Uncertainty in Artificial Intelligence*, 1993, pp. 181-188.

[19] Dean, T., Planning under Time Constraints in Stochastic Domains, *Artificial Intelligence,* 76, 1995, pp. 195-199.

[20] Dechter, R., Mini-buckets: A General Scheme for Generating Approximations in Automated Reasoning in Probabilistic Inference, *Proceeding of the 15th International Joint Conference on Artificial Intelligence*, 1997, pp. 1297-1302.

[21] Dixon, W.J., *Introduction to Statistical Analysis,* McGraw Hill, Boston,1983.

[22] Dwarkadas, S., Schaffer, A., Cottingham, R.W., Cox, A.L., Keleher, P., Zwaenepoel, W., Parallelization of General Linkage Analysis Problems, *Human Heredity,* 44, 1994, pp. 127-141.

[23] Fink, E., How to Solve it Automatically: Selection among Problem Solving Methods, In Simons, R. G., Veloso, M. M.and Smith, S., Editors, *Proceeding of the 4th International Conference on Artificial Intelligence Planning Systems*, 1998, pp. 128-136.

[24] Garvey, A.J., Lesser, V.R., Design-to-Time Real-Time Scheduling, *IEEE Transactions on*

Systems, Man and Cybernetics, 23, 1993, pp. 1491-1502.

[25] Garvey, A.J., Lesser, V.R., A Survey of Research in Deliberative Real-Time Artificial Intelligence, *Real-Time Systems,* 6, 1994, pp. 317-347.

[26] Gilks, W.R., Richardson, S., Spiegelhalter, D.J., *Markov Chain Monte Carlo in Practice*, Chapman and Hall, 1996.

[27] Guo, H., Hsu, W., A Survey of Algorithms for Real-Time Bayesian Network Inference, *Proceeding of AAAI/KDD/UAI-2002 Joint Workshop on Real-Time Decision Support and Diagnosis Systems*, 2002, pp. 1-12.

[28] Guo, H., Algorithm Selection for Sorting and Probabilistic Inference: A Machine Learning-Based Approach. *PHD Thesis*, Kansas State University, 2003.

[29] Henrion, M., Search-Based Methods to Bound Diagnostic in Very Large Belief Nets, *Proceeding of the 7th Conference on Uncertainty in Artificial Intelligence*, 1991, pp. 14-150.

[30] Horvitz, E., Suermondt, H.J., Bounded Conditioning: Flexible Inference for Decisions under Scarce Resources, *Proceeding of the 5th Conference on Uncertainty in Artificial Intelligence*, 1989, pp. 182-193.

[31] Horvitz, E., Barry, M., Display of Information for Time-Critical Decision Making, *Proceeding of the 11th Conference on Uncertainty in Artificial Intelligence*, 1995, pp. 286-305.

[32] Horvitz, E., Breese, J., Heckerman, D., Hovel, D., Rommelse, K., The Lumiere Project: Bayesian User Modeling for Inferring the Goals and Needs of Software Users, *Proceeding of the 14th Conference on Uncertainty in Artificial Intelligence*, 1998, pp. 256-265.

[33] Hosmer, D.W., Lemeshow, S., *Applied Logistic Regression,* John Wiley & Sons, 1989.

[34] Houstis, N. E., Catlin, A. C., Rice, J. R., Verykios, V. S., PYTHIA-II: A Knowledge Database System for Managing Performance Data and Recommending Scientific Software, *TOMS,* 26 (2), 2000, pp. 227-253.

[35] Hsu, W. H., Zwarico, A. E., Automatic Synthesis of Compression Techniques for Heterogeneous Files, *Software: Practice and Experience ,* 25 (10), 1995, pp. 1097-1116.

[36] Jitnah, N., Nicholson, A.E., Belief Network Inference Algorithms: a Study of Performance Based on Domain Characterization, *Technical report TR-96-249*, Nonash University, 1996.

[37] Lauritzen, S.L., Speigehalter, D.J., Local Computations with Probabilities on Graphical Structures and their Application to Expert Systems, *Journal of Royal Statistics Society ,* 50 (2), 1988, pp. 157-224.

[38] Li, Z., D'Ambrosio, B., Efficient Inference in Bayesian Networks as a Combinatorial Optimization Problem, *International Journal of of Approximate Reasoning,* 11, 1994.

[39] Lilliefors, H., On the Kolmogorov-Smirnov Test for Normality with Mean and Variance Unknown, *Journal of the American Statistical Association* 62, 1967, pp399-402.

[40] Loynes, R.M., On Cox and Snell´s General Definition of Residuals, *Journal of the Royal Statistical Society* 31, 1969, pp. 103-106.

[41] Minton, S., Automatically Configuring Constraint Satisfaction Programs: A Case Study, *Constraints,* 1(1/2), 1996, pp. 7-43.

[42] Murphy, K.P., Weiss, Y., Jordan, M.I., Loopy Belief Propagation for Approximate Inference: An Empirical Study, *Proceeding of the 15th Conference on Uncertainty in Artificial Intelligence*, 1999, pp. 467-475.

[43] Nagelkerke, N.J., A Note on a General Definition of the Coefficient of Determination, *Biometrika,* 78, 1991, pp. 691-692.

[44] Pearl, J., Evidential Reasoning using Stochastic Simulation of Causal Models, *Artificial Intelligence,* 32, 1987, pp. 245-257.

[45] Pearl, J., *Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference*, Morgan Kaufmann, 1988.

[46] Poole, D., Probabilistic Conflicts in a Search Algorithm for Estimating Posterior Probabilities in Bayesian Networks, *Artificial Intelligence,* 88, 1996, pp. 69-100.

[47] Ramos, F.T., Mikami, F., Cozman, F.G., Implementação de Redes Bayesianas em Sistemas Embarcados, *Proceeding of the IBERAMIA/SBIA 2000 Workshops (Workshop on Probabilistic Reasoning in Artifficial Intelligence)*, 2000, pp. 65-69 (in Portuguese).

[48] Rice, J. R., The Algorithm Selection Problem, *Advances in Computers ,* 15, 1976, pp. 65-118.

[49] Shachter, R.D., Evaluating Influence Diagrams, *Operations Research,* 34(6), 1986, pp. 873-882.

[50] Shafer, G., Shenoy, P.P., Probability Propagation, *Annals of Mathematics and Artificial Intelligence 2,* 1990, pp. 327-352.

[51]Zilberstein, S., Algorithm Operational Rationality through Compilation of Anytime Algo-rithms, *PHD Thesis*, University of California at Berkeley, 1993.

[52] Zweig, G., Russell, S.J., Speech Recognition with Dynamic Bayesian Networks, *Proceeding of AAAI/IAAI*, 1998, pp. 173-180.