Learning Bayesian Networks from Incomplete Data using Evolutionary Algorithms

James W. Myers
George Mason University
Fairfax, VA 22030

Kathryn B. Laskey
George Mason University
Fairfax, VA 22030

Kenneth A. DeJong
George Mason University
Fairfax, VA 22030

Abstract

This paper describes an evolutionary algorithm approach to learning Bayesian networks from incomplete data. This problem is characterized by a huge solution space with a highly multimodal landscape. State-of-the-art approaches all involve using deterministic approaches such as the expectation-maximization algorithm. These approaches are guaranteed to find local maxima, but do not explore the landscape for other modes. Our approach evolves the structure of the network and the missing data. We use a factorial design to choose a good set of parameters for selection, crossover, and mutation. We show that our algorithm produces accurate results for a classification problem with missing data.

1 INTRODUCTION

Bayesian networks are quickly becoming the tool of choice of many AI researchers for problems involving reasoning under uncertainty. They have been implemented in applications in areas such as medical diagnostics, classification systems, software agents for personal assistants, multisensor fusion, and legal analysis of trials [Heckerman, Geiger et al. 1995]. Until recently, the standard approach to constructing belief networks was a labor-intensive process of eliciting knowledge from experts. Methods for capturing available data to construct a Bayesian network or to refine an expert-provided network promise to greatly improve both the efficiency of knowledge engineering and the accuracy of the models. For this reason, learning Bayesian networks from data has become an increasingly active area of research. Most of the research to date has relied on the assumption that data are complete; that is, the values of all variables are known for all cases in the database. This assumption is not very realistic, since most real world situations involve incomplete information.

Learning a Bayesian network can be decomposed into the problem of learning the graph structure and learning the parameters. The first attempts at treating incomplete data involved learning the parameters of a fixed network structure [Lauritzen 1995]. Very recently, researchers have begun to tackle the problem of learning the structure of the network from incomplete data. A major stumbling block in this research is that when information is missing, closed form expressions do not exist for the scoring metric used to evaluate network structures. This has led many researchers down the path of estimating the score using parametric approaches such as the expectation-maximization (EM) algorithm [Dempster, Laird et al. 1977], [Friedman 1998]. The EM algorithm is a proven approach for dealing with incomplete information when building statistical models [Little and Rubin 1987]. EM and related algorithms show promise. However, it has been noted [Friedman 1998] that the search landscape is large and multimodal, and deterministic search algorithms are prone to find local optima. Multiple restarts have been suggested as a way to deal with this problem.

An obvious choice to combat the problem of "getting stuck" on local maxima is to use a stochastic search method. This paper explores the use of evolutionary algorithms for learning Bayesian networks from incomplete data. Our approach is unique in that it evolves both the solution space of network structures and the values of the missing data. Network structures are especially amenable for evolutionary algorithms since the substructures of the network behave as building blocks so we can evolve higher fit structures by exchanging substructures of parents with higher fitness. By evolving samples of missing data, we are in effect approximating a

maximum likelihood approach to scoring the network. In addition, we can impute the values of the missing data allowing us to use the closed form of the scoring metric.

We'll begin by briefly describing Bayesian networks and the learning problem. Next we will discuss the scoring metric, fitness function, and the landscape. In this section we'll make the point for using a stochastic search methods such as evolutionary algorithms. Section 4 will describe the design choices we made, to include results from an experiment using a factorial design and some results of an empirical study. We'll close in section 5 with a summary of our approach and experiments and discuss our future plans.

2 BAYESIAN NETWORKS

Bayesian networks are graphical models that encode probabilistic relationships among variables for problems of uncertain reasoning. They are composed of a structure and parameters. The structure is a directed acyclic graph that encodes a set of conditional independence relationships among variables. The nodes of the graph correspond directly to the variables and the directed arcs represent dependence of variables to their parents. The lack of directed arcs among variables represent a conditional independence relationship. Take, for example, the network in Figure 1. The lack of arcs between symptoms S1, S2, and S3 indicate that they are conditionally independent given C. In other words, knowledge of S1 is irrelevant to that of S2 given we already know the value of C. If C is not known, then knowledge of S1 is relevant to inferences about the value of S2.

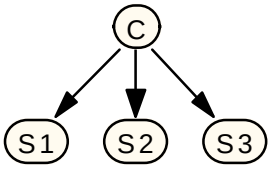

Figure 1 Bayesian Network for generic disease and symptoms

The parameters of the network are the local probability distributions attached to each variable. The structure and parameters taken together encode the joint probability of the variables. Let $\mathbf{U} = \{X_1,...,X_n\}$ represent a finite set of discrete random variables. We use upper-case letters (e.g. X, Y, Z) to represent random variables. Lower-case letters (e.g. x, y, z) are used to denote specific values taken on by the random variables. Bold upper-case letters (e.g. **X, Y, Z**) specifies sets of random variables and bold lower-case letters (e.g. **x, y, z**) indicates sets of specific values. The set of parents of X_i are given by $pa(X_i)$. The joint distribution represented by a Bayesian network over the set of variables **U** is

$$p(\mathbf{U}) = \prod_{i=1}^{n} p(X_i \mid pa(X_i)) \qquad (1)$$

where n is the number of variables in **U** and $p(X_i|pa(X_i))=p(X_i)$ when X_i has no parents.

The joint distribution for the set of variables **U** = {C, S1, S2, S3} from Figure 1 is specified as $p(\mathbf{U}) = p(C)p(S1|C)p(S2|C)p(S3|C)$. In addition to specifying the joint distribution of **U**, efficient inference algorithms allow any set of nodes to be queried given evidence on any other set of nodes. This means that a single model can be used for both prediction and classification. For example, suppose variable C in figure 1 is a list of possible diseases and variables S1, S2, and S3 are symptoms. Given observations of any set of the symptoms {S1=s1, S2=s2, and/or S3=s3}, we can determine the likelihood P(C|S1=s1,S2=s2,S3=s3) that any particular disease is present given the observed symptoms. Perhaps though, we want to predict the symptoms given a specific disease, by instantiating a disease in D, the network will return a probability distribution on the set of symptoms [Pearl 1988].

The most common approach to building Bayesian networks is to elicit knowledge from an expert. This works well for smaller networks, but when the number of variables becomes large, elicitation can become a tedious and time-consuming affair. There may also be situations where the expert is either unwilling or unavailable. Whether or not experts are available, if there are data it makes sense to use it in building a model.

The problem of learning a Bayesian network from data can be broken into two components: learning the structure, B_S, and learning the parameters, B_P. If the structure is known[1] then the problem reduces to learning the parameters. If the structure is unknown, the learner must first find the structure before learning the parameters (actually in many cases they are induced simultaneously). Learning the structure can itself be decomposed into searching for structures and evaluating structures. Until recently most research has concentrated on learning networks from complete datasets. By complete, we mean that all of the cases in the data contain values for all of the variables. It is clearly necessary to relax this assumption if these methods are to have wide applicability. Recently however, a few researchers have begun working on algorithms for learning networks from incomplete data. One major complicating factor for incomplete data is that the most common scoring metric used to evaluate structures, the Bayesian Dirichlet score, exists as a closed form expression for complete data but not for incomplete data. Most of the approaches for learning with incomplete data involve approximating the score and

[1] It is more accurate to say that the structure is "given" or "provided." We use the term "known" to be consistent with current usage.

using greedy algorithms to search the space of network structures.

The most common scoring approach to evaluating structures is by the posterior probability of the structure given the observations. That is, a structure is good to the extent it is probable given the available information. The posterior probability of a structure can be obtained by applying Bayes rule:

$$P(B_S|D) = \frac{P(D|B_S)P(B_S)}{P(D)} \quad (2)$$

where, $P(B_S|D)$ is the posterior distribution of the structure given the data, $P(D|B_S)$ is the likelihood function, $P(B_S)$ is the prior probability of the structure, and P(D) is the normalizing constant. Since P(D) is not dependent on the structure, it can be ignored when trying to find the best scoring function. In addition, without prior knowledge of structures, we can assume they have equal probability and use a noninfomative prior $P(B_S) \propto 1$. However, if we do have information on structures we can always use the prior information. See [Gelman, Carlin et al. 1996], [O'Hagan 1994], and [Press 1989] for a discussion of prior probabilities and assignments of priors.

The problem is now reduced to finding the structure with the maximum likelihood $P(D|B_S)$. In other words, given a structure, structures are evaluated according to how probable it is that the data were generated from the structure. Cooper and Herskovitz and Heckerman et al. showed that when a Dirichlet prior is used for the parameters in the network, the likelihood $P(D|B_S)$ can be obtained in closed form:

$$P(D|B_S) = \prod_{i=1}^{n}\prod_{J=1}^{q_i}\frac{\Gamma\left(N'_{ij}\right)}{\Gamma\left(N'_{ij}+N_{ij}\right)}\prod_{k=1}^{r_i}\frac{\Gamma\left(N'_{ijk}+N_{ijk}\right)}{\Gamma\left(N'_{jik}\right)} \quad (3)$$

where n is the number of variables in the database, r_i is the number of possible states for variable X_i, q_i is the number of possible states for $pa(X_i)$, N_{ijk} are the sufficient statistics from the database (counts of occurrences of configurations of variables and their parents), N'_{ijk} are the hyperparameters (prior counts of occurences of variables and their parents) specified for the parameter prior (assuming an uninformative prior as in the prior for the structure we set the hyperparameters to 1), $N_{ij} = \sum_{k=1}^{r_i} N_{ijk}$, and $N'_{ij} = \sum_{k=1}^{r_i} N'_{ijk}$. For computational convenience, the number of parents allowed for a particular variable is limited. This scoring metric (3) is commonly referred to as the Bayesian Dirichlet metric. In practice, the logarithm of (3) is usually used to score networks [Cooper and Herskovits 1992], [Heckerman, Geiger et al. 1995].

With the Bayesian Dirichlet metric (3), we can now search over possible structures for the one that scores best. The search problem has been shown to be NP-hard [Chickering, Geiger et al. 1994], so most approaches use a greedy search by starting from an initial structure and adding, deleting, or reversing arcs in an attempt to find a higher scoring network structure. In order to compensate for climbing the nearest local maximum, the search is usually restarted with new randomly generated structures. This type of greedy algorithm must also make sure the modification doesn't produce an illegal structure by creating a directed cycle in the graph. One way to get around this is to specify a node ordering. Given a node ordering a variable can only take on parents from the set of variables that precede it in the order. This will always produce a legal structure but adds an additional burden of searching over n! node orderings (unless, of course, an expert can specify a node order in which case the search space is reduced tremendously).

The problem of learning Bayesian networks from incomplete data is much more difficult than for learning from complete data. First of all, the Bayesian Dirichlet metric (3) no longer exists in closed form when the data are incomplete. This is because the formula involves the sufficient statistics, which are not known when data are incomplete. Arguably the most common approach to dealing with incomplete data is to estimate the parameters using the Expectation-Maximization (EM) algorithm [Dempster, Laird et al. 1977]. The EM algorithm estimates the parameters by iteratively finding the expectation of the parameter and then finding the maximum likelihood estimate (MLE) using the parameter from the expectation step. Each iteration of the EM algorithm increases the MLE until a stable fixed point is reached. One of the first applications of EM to learning Bayesian networks was by Lauritzen [Lauritzen 1995]. He used EM to find the parameters of a given network structure from incomplete data. The algorithm worked well in most cases, but he noted that the problem of learning from incomplete data was multimodal and the algorithm would have to be randomly restarted because it would converge to the nearest local maximum.

Arguably the most significant advances in the area of learning from incomplete data have been the work of Friedman (see [Friedman 1998a] and [Friedman 1998b]). His approach is to interleave greedy search over structures with the EM algorithm to estimate parameters. He named this algorithm the Structural EM (SEM) algorithm. The concept is similar to those for the complete data problem, except the score of the network is found using the EM algorithm. For example, he begins with an initial structure. The structure is then passed to the EM algorithm and the MLE found by EM is returned as the score. The next structure is found by adding, deleting, or

reversing an arc and passed to the EM. Friedman's innovation was to save computation by using EM parameter estimates for the current structure to evaluate candidates for the next structure, and to run EM again only for the structure actually chosen. If the MLE for the best structure is lower than the current structure then the current structure becomes the best. This continues until no further improvements can be found. Friedman has also noted that SEM "gets stuck" on local maximum.

Additional analysis of the geometry of the search space for the incomplete data problem by [Geiger and Meek 1998] and [Settimi and Smith 1998] further confirm this is a very difficult problem. Given the huge search space, multiple dimensions, and extreme multi-modal landscape for this problem, it is no wonder deterministic algorithms continue to require multiple random restarts. We investigate the use of evolutionary algorithms for searching for "good" scoring structures from this very complex search space.

3 EVOLUTIONARY ALGORITHM

The very large, multi-dimensional, multi-modal landscape immediately suggests the use of evolutionary algorithms. Closer inspection of the equation for the joint distribution (1) of the variables in a Bayesian network suggest the network can be broken down into local structures that can be considered genes. Each local structure is a component of the joint distribution and has its own conditional probability, $P(X_i|pa(X_i))$. The conditional probabilities correspond to a node and its parents in the structure of the network. Further, the Bayesian Dirichlet metric (3) suggests a fitness function that can be broken into parts corresponding to a node and its parents and is additive in log form. This means the fitness function can be computed component-wise for each gene and added to produce the fitness of the network. Of course, this assumes the Bayesian Dirichlet is closed which is the case for complete data. For incomplete data, we either have to find a means to convert the incomplete problem into a complete problem or estimate the parameters.

3.1 PREVIOUS WORK

Evolutionary algorithms have been used by Larranga, et al., for learning Bayesian networks from complete data [Larranaga, Murga et al. 1996]. Their approach compared four evolutionary algorithmic approaches: steady state, hybrid steady state, elitist, and hybrid elitist. The steady state approaches created a single new individual each generation that replaced the worst individual from the previous generation if it had better fitness. With the elitist approaches, the λ best individuals from the parents and offspring were selected to survive to the next generation. The hybrid approach they refer to is an artifact of the fact that as the number of parents of a node increase, the computational complexity increases factorially. The hybrid approach selects the k best parent nodes from the parent row of each node, where $k<m$ and m is the maximum number of parents allowed. The remaining parameters of the algorithm are as follows: rank selection to select parents for reproduction, 1-point crossover, and mutation. The network structure was represented as a connection matrix where $c_{ij}=1$ if (j is a parent node of i) and ($i > j$), otherwise $c_{ij}=0$. The inequality $i > j$, upper-triangulation of the connection matrix, guarantees a node ordering thus assuring a legal structure (i.e. directed acyclic graph). They ran their algorithm on a dataset generated from a known network, the ALARM network, used to diagnosed potential anesthesia problems in operating rooms [Beinlich, Suermondt et al. 1989]. The tests were run using a node ordering derived from the original network. The results were networks that in many tests had higher Bayesian Dirichlet scores than the original network. In another set of experiments, Larranga, et al., lifted the node order restriction on the connection matrix. Building structures from the full connection matrix may result in illegal structures, i.e. cyclic directed graphs. When illegal structures were encountered they used a repair operator to remove any offensive arcs. The results from this experiment were similar to the ones above except the node ordering produced overall better results [Larranaga, Poza et al. 1996].

Larranga, et al., approach showed that evolutionary algorithms can find very good network structures for the complete data problem. However, since there are currently several greedy algorithms that produce similar results, their research did not prove an advantage for using evolutionary algorithms over a deterministic greedy search. However, they did pave the way for additional research in this area.

3.2 ALGORITHM

Recall from Section 2 that the search space for the incomplete data learning problem is very multi-modal. The state-of-the-art approaches are all based upon deterministic greedy search algorithms. These algorithms all suffer the fate of "getting stuck" at the nearest local optimum. Our approach is to use evolutionary algorithms to increase the exploration of the search space. Also recall that the closed form expression (3) for the Bayesian Dirichlet score applies only for complete data. In order to avoid using computationally costly parametric estimation approaches, we impute samples of the missing data into the database. This reduces the problem to a complete data problem and allows us to use the logarithm of (3) as the fitness function.

By imputing samples into the data, the search space becomes more complex. We now must search over the missing data and network structures. We take the unique approach that evolves both the missing values (samples)

and the structures simultaneously. This approach requires that we define a representation for both the missing data and the structures. The missing data representation is straightforward. We represent each cell from the dataset that has a missing value as a gene. The gene takes on sampled values from the set of values of the corresponding variable. The chromosome is a string of missing values.

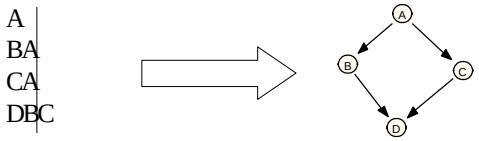

Figure 2 Structure Chromosome

The structure B_S can be represented as an adjacency list, see Figure 2, where each row represents a variable $\mathbf{V}_i$ and the members of each row, with the exception of the first member, are the parents of $\mathbf{V}_i$, pa($\mathbf{V}_i$). The first member of each row, i.e. the first column of the adjacency list, is the variable $\mathbf{V}_i$. Although we show it in the picture for clarity, the internal representation encodes the parents only, with the variable being encoded by position. The adjacency list can be thought of as a chromosome, where each row is a gene and the pa($\mathbf{V}_i$) are the alleles. This representation is convenient because from (3), the logarithm of the scoring metric is the summation of scores for each variable. Because of this, each gene can be scored separately and added to generate the fitness score for the entire structure.

The allele values that each gene can take on can become enormous. The values can range from no parents to n-1 parents, where n is the number of variables in the dataset. Thus an allele can take on $\sum_{i=1}^{m} \binom{n-1}{i}$ possible values where m is the maximum set of parents a variable can have and n is the number of variables in the dataset. As an example of the large size of allele values take n=11 and m=4, the number of possible values for a given allele is 376, while if n=41 and m=4, the number grows to 102,091.

In addition to the large combination of allele values per gene, the genes are highly correlated. This is because the alleles are combinations of other genes as parents. Many combinations can lead to illegal structures; in other words, structures that are not directed acyclic graphs. This problem is alleviated by arbitrarily assigning illegal structures a very low score. The reason for allowing illegal structures is the chromosome may contain very good genes and if selected as parents the genes can be reconstituted as building blocks for even better structures through recombination or mutation.

To help decide which evolutionary algorithm parameters to use we performed a factorial design where the factors were selection scheme, probabilities for parameterized uniform crossover for both missing data and structure, and probability of mutation for structure. The population size was kept small, 20-40, because of the computation time required scoring the fitness function. The selection schemes considered for the factorial design are fitness proportional, rank proportional, and binary tournament selection.

The genetic operators require some explanation. For the missing data chromosomes we chose uniform parameterized crossover [Syswerda 1989], [DeJong and Spears 1990]. Early tests indicated that parameterized uniform crossover worked better for these chromosomes. This may be due to the small population size. Mutation for the missing data chromosome differs somewhat from the canonical genetic algorithm. Instead of flipping a binary bit, we randomly select from the remaining possible values of the corresponding variable. We also chose to use uniform parameterized crossover for the structure chromosome. Figure 3 demonstrates how crossover keeps local structure for a node intact.

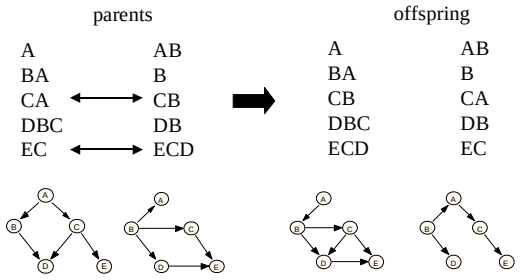

Figure 3 Crossover for Bayesian Network Structure

The mutation operator for the structure chromosome is tailored to the representation we used and its mapping to a directed graph phenotype. Recall the gene of the structure chromosome represents the gene's parent nodes in the graph. We include two basic modifications to a gene: add a node and delete a node. These operators have the effect in the phenotype of adding and deleting arcs, respectively. We also include a third basic modification, reversal of an arc, which is implemented genotypically by deleting the parent-child arc and adding the child-parent arc.

The algorithm is depicted graphically in Figure 4.

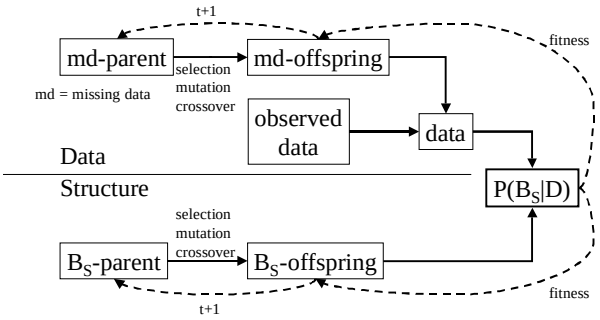

Figure 4 EA for Learning Bayesian Networks from Incomplete Data

4 ANALYSIS AND RESULT

Our approach was to first run a set of experiments to find a "good" set of algorithm parameters and then perform experiments to demonstrate the EA learns Bayesian networks from incomplete data that has strong predictive power. The first set of experiments was based on a factorial design with the algorithm parameters as main effects. We began with a known seven variable network and randomly generated 1000 samples for the training set and another 1000 samples for the test set. We then randomly selected a percentage of the cells in the dataset as missing. Each experiment was repeated 10 times. The parameters were selected based on their Bayesian Dirichlet score (posterior probability of the structure given the data).

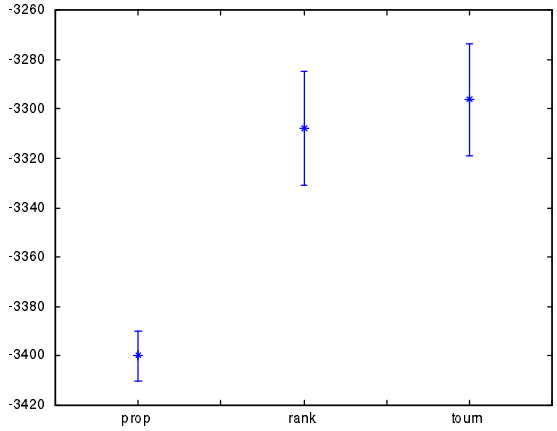

Figure 5 Main Effects for Selection Scheme

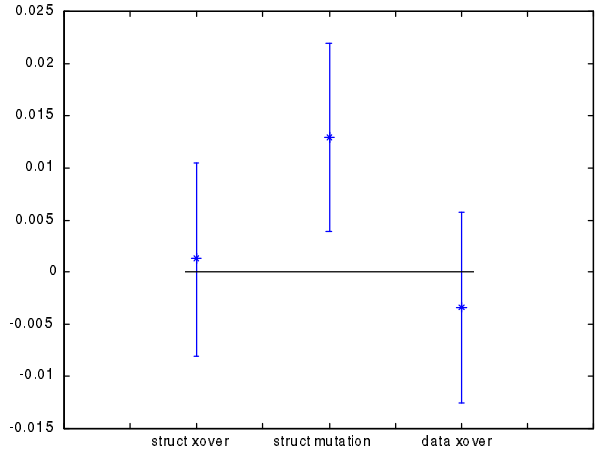

Figure 6 Main Effects of Genetic Operators

The main effects plots of Figure 5 show the 95 percent credible intervals for the selection schemes. For those unfamiliar with Bayesian Analysis, see Press for a description of credible intervals [Press 1989]. The plots indicate there is a strong correlation between rank selection and tournament selection and the response variable, the Bayesian Dirichlet metric. This is consistent with Goldberg and Deb's analysis of selection schemes for function optimization [Goldberg and Deb 1991]. Since tournament selection performed slightly better than rank selection for these tests, we chose to implement tournament selection for the empirical analysis presented later. Tournament selection should also be considered for future experiments because it is much easier to implement and more efficient computationally than rank selection [Goldberg and Deb 1991].

From Figure 6, we see that there is a slight advantage to using a mutation rate of 0.05. Since the credible intervals for both the crossover parameters intersect the origin we can make a claim that one value is better than the other. We chose to use the more common uniform crossover probability of 0.5.

The objective of the second set of experiments was to demonstrate that stochastic search finds "good" predictive networks. For this set of experiments we used a Bayesian network known as ASIA. The ASIA network was initially presented by Lauritzen and Spiegelhalter [Lauritzen and Spiegelhalter 1988]. It is a small (nine variable) fictitious model of medical knowledge concerning the relationships between visits to Asia, tuberculosis, smoking, lung cancer, and bronchitis. Our approach, as above, was to generate 1000 samples each from the original network for training and test. We used the set of "good" parameters found in the previous experiments. The algorithm was run with 0%, 5%, 15%, and 30% missing data. The experiment was run 10 times for each level of missing data. The stopping criterion for the algorithm was arbitrarily set at 500 generations. Using the "best" network from each run we calculated the log loss.

The log loss is a commonly used metric appropriate for probabilistic learning algorithms. It is a member of the family of proper scoring rules. Proper scoring rules have the characteristic that they are maximized when the learned probability distribution corresponds to the empirically observed probabilities. Mathematically, the log loss is the average over all test cases of the log of the joint probability the test case was generated by the model.

Figure 7 depicts the 95 percent credible intervals for the Bayesian Dirichlet for each level of missing data. The Bayesian Dirichlet is the log probability of the structure given the data so the higher the score the better. Figure 8 shows the 95 percent credible interval for the log loss score for each level of missing data. The lower the log loss score the better since it is a measure of predictive power. For comparison purposes the Bayesian Dirichlet of the original network is –2172.9 and the log loss score for the set of test data is 2.1953.

As can be seen from the figures, the EA finds good predictive networks at 0%, 5%, and 15% missing data. At 30% the predictive accuracy degrades sharply as can be expected.

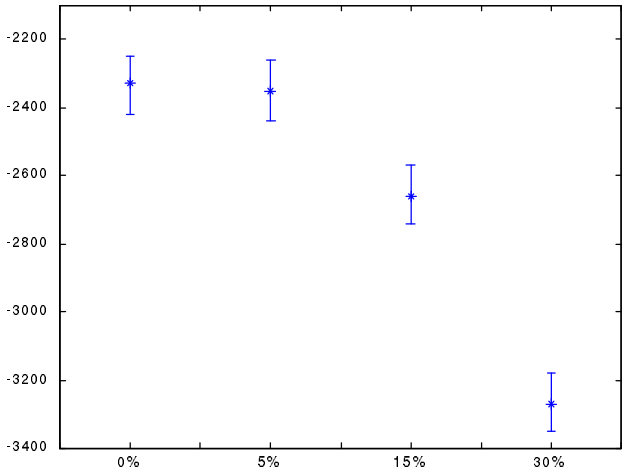

Figure 7 95 Percent Credible Interval for Bayesian Dirichlet

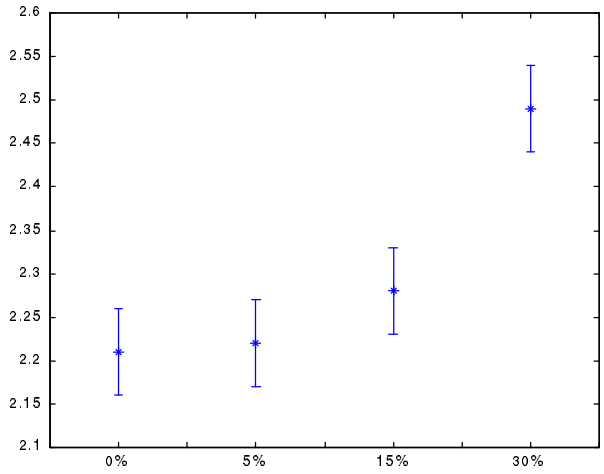

Figure 8 95 Percent Credible Interval for Log Loss

5 SUMMARY AND FUTURE WORK

In this paper we describe a novel evolutionary algorithm for learning Bayesian networks from incomplete data. This problem is extremely difficult for deterministic algorithms and is characterized by a large, multi-dimensional, multi-modal search space. Previous attempts to solve this problem with deterministic algorithms have led researchers to the conclusion that they must use random restarts. We believe stochastic algorithms will give better results. Our approach was to find a "good" set of parameters for the evolutionary algorithm by first conducting experiments based on a factorial design. Then we demonstrated that we could learn networks from incomplete data that perform well in terms of predictive accuracy.

Though the algorithm we used was unique in that we evolved two populations simultaneously, we did not explore more advanced techniques from evolutionary algorithm literature. An interesting experiment we intend to perform is to add speciation to increase exploration [Spears 1994]. We are also exploring the use of adaptive operators. Other important future work includes Markov Chain Monte Carlo (MCMC) algorithms and an Evolutionary Markov Chain Monte Carlo (EMCMC) approach that combines the benefits of evolutionary algorithms and MCMC algorithms. We believe the EMCMC will offer advantages researchers in both evolutionary algorithms and statistics will find very useful.

Acknowledgments

The research reported in this paper was sponsored by DARPA and the Air Force Research Laboratory under contract DACA76-93-0025 to Information Extraction and Transport, Inc., with subcontract to George Mason University. Many thanks to Tod Levitt for helpful and stimulating discussions. Our gratitude to the Genetic Algorithms Group and Decision Theoretic Group at George Mason University for enlightening dialogues and to Brent Boerlage for input on the log loss function.

References

Beinlich, I. A., H. J. Suermondt, et al. (1989). The ALARM Monitoring System: A Case Study with Two Probabilistic Inference Techniques for Belief Networks. Proceedings of the Second European Conference on Artificial Intelligence in Medicine.

Chickering, D. M., D. Geiger, et al. (1994). Learning Bayesian Networks is NP-Hard. Redmond, WA, Microsoft Research.

Cooper, G. F. and E. Herskovits (1992). "A Bayesian Method for the Induction of Probabilistic Networks from Data." Machine Learning 9: 309-347.

DeJong, K. A. and W. M. Spears (1990). An Analysis of the Interacting Roles of Population Size and Crossover in Genetic Algorithms. Proceedings of the First International Conference on Parallel Problem Solving from Nature, Dortmund, Germany.

Dempster, A. P., N. M. Laird, et al. (1977). "Maximum Likelihood Estimation from Incomplete Data via the EM Algorithm." Journal of the Royal Statistical Society V39: 1-38.

Friedman, N. (1998a). The Bayesian Structural EM Algorithm. Proceedings of the Fourteenth Conference on Uncertainty in Artificial Intelligence, Madison, WI, Morgan Kaufmann Publishers.

Friedman, N. (1998b). Learning Belief Networks in the Presence of Missing Values and Hidden Variables. Fourteenth International Conference on Machine Learning (ICML-97), Vanderbilt University, Morgan Kaufmann Publishers.

Geiger, D. and C. Meek (1998). Graphical Models and Exponential Families. Proceedings of the Fourteenth Conference on Uncertainty in Artificial Intelligence, Madison, WI, Morgan Kaufmann.

Gelman, A., J. B. Carlin, et al. (1996). Bayesian Data Analysis. London, Chapman & Hall.

Goldberg, D. E. and K. Deb (1991). A Comparative Analysis of Selection Schemes Used in Genetic Algorithms. Foundations of Genetic Algorithms. G. J. E. Rawlins. San Mateo, CA, Morgan Kaufmann Publishers. 1: 69-93.

Heckerman, D., D. Geiger, et al. (1995). "Learning Bayesian Networks: The Combination of Knowledge and Statistical Data." Machine Learning 20: 197-243.

Larranaga, P., R. Murga, et al. (1996). Structure Learning of Bayesian Networks by Hybrid Genetic Algorithms. Learning from Data; Artificial Intelligence and Statistics V. D. Fisher and H.-J. Lenz. New York, Springer: 450.

Larranaga, P., M. Poza, et al. (1996). "Structure Learning of Bayesian Networks by Genetic Algorithms: A Performance Analysis of Control Parameters." IEEE Journal on Pattern Analysis and Machine Intelligence 18(9): 912-926.

Lauritzen, S. L. (1995). "The EM algorithm for graphical association models with missing data." Computational Statistics & Data Analysis 19: 191-201.

Lauritzen, S. L. and D. J. Spiegelhalter (1988). "Local Computations with Probabilities on Graphical Structures and Their Application on Expert Systems." Journal of Royal Statistical Society 50(2): 157-224.

Little, R. and D. Rubin (1987). Statistical Analysis with Missing Data. New York, John WIley & Sons.

O'Hagan, A. (1994). Bayesian Inference. London, Edward Arnold.

Pearl, J. (1988). Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference. San Francisco, Morgan Kaufmann Publishers, Inc.

Press, S. J. (1989). Bayesian Statistics: Principles, Models, and Applications. New York, John Wiley & Sons.

Settimi, R. and J. Q. Smith (1998). On the Geometry of Bayesian Graphical Models with Hidden Variables. Proceedings of the Fourteenth Conference on Uncertainty in Artificial Intelligence, Madison, WI, Morgan Kaufmann.

Spears, W. M. (1994). Simple Subpopulation Schemes. Proceedings of the Third Annual Conference on Evolutionary Programming, San Diego, World Scientific.

Syswerda, G. (1989). Uniform Crossover in Genetic Algorithms. Proceedings of the 3rd International Conference on Genetic Algorithms, Morgan Kaufman.