Using Evolutionary Programming and Minimum Description Length Principle for Data Mining of Bayesian Networks

Man Leung Wong[*] Wai Lam[†] Kwong Sak Leung[‡]

Correspondence Author:

Man Leung Wong
Department of Computing and Decision Sciences
Lingnan University
Tuen Mun
Hong Kong

mlwong@ln.edu.hk

Tel no: +852-2616-8093
Fax no: +852-2892-2442

[*]M.L. Wong is with Dept. of Computing and Decision Sciences, Lingnan University, Tuen Mun, Hong Kong

[†]W. Lam is with Dept. of Systems Engg. & Engg. Management, The Chinese University of Hong Kong, Shatin, Hong Kong

[‡]K.S. Leung is with Dept. of Computer Sci. and Engg., The Chinese University of Hong Kong, Shatin, Hong Kong

Abstract

We have developed a new approach (MDLEP) to learning Bayesian network structures based on the Minimum Description Length (MDL) principle and Evolutionary Programming (EP). It employs a MDL metric, which is founded on information theory, and integrates a knowledge-guided genetic operator for the optimization in the search process. In contrast, existing techniques based on genetic algorithms (GA) only adopt classical genetic operators. We conduct a series of experiments to demonstrate the performance of our approach and to compare it with that of the GA approach developed in a recent work. The empirical results illustrate that our approach is superior both in terms of quality of the solutions and computational time for most data sets we have tested. Lastly, our MDLEP approach does not need to impose the restriction of having a complete variable ordering as input.

1 Introduction

Data mining aims at extracting automatically implicit and nontrivial knowledge from data. Recently, researchers have begun to work on methods for learning Bayesian networks from data [7, 3, 17]. It has been shown that this problem is believed to be computationally intractable [2]. Herskovits and Cooper [9] proposed a system known as KUTATO based on the entropy technique. Later, they developed a Bayesian metric which can measure the fitness of a network structure to the data based on Bayesian approach [3]. A search algorithm called K2 was proposed to find the most desirable network. One limitation of this search method is that they require as input a total ordering among the variables. Heckerman *et. al.* [7] and Spirtes *et. al.* [16] proposed various approaches to learn network structures without the variable ordering restriction. More recently, Larrañaga *et. al.* [13, 14] have done some work on using genetic algorithms for learning Bayesian networks.

We develop a new approach (MDLEP) to learning multiple-connected network structures based on the Minimum Description Length (MDL) principle and Evolutionary Programming (EP). Specifically, our approach employs a MDL-based Bayesian network learning technique founded on information theory and integrates an optimization process which is based on evolutionary programming. An important characteristic of our approach is that, in addition to ordinary genetic operators, we design a knowledge-guided operator which incorporates a MDL learning scheme. Essentially we make use of the network structure discovery knowledge for developing the genetic operators. In contrast, previous work based on genetic algorithms (GA) do not consider such knowledge in the operators. For instance, Larrañaga *et. al.* [13] used a chromosome to represent a particular variable ordering. The purpose of the genetic operators is to evolve different variable orderings. For each new variable ordering formed by the operators, it is then passed to K2, an existing Bayesian network search algorithm proposed in [3], to obtain a network. In another recent work done by Larrañaga *et. al.* [14], they adopted a classical GA approach and used a chromosome to represent a particular network. Thus the purpose of the genetic operators is to evolve different networks. For each network formed by the operators, it is evaluated using an existing metric mentioned in [3] to measure its merits. However, only simple and standard genetic operators are used. Furthermore, our approach does not need to impose the restriction of having a complete variable ordering as input. Lastly, our approach can be applied to Bayesian metric as described in [7] although we currently employ the MDL metric. It has been shown that the MDL metric possesses similar theoretical properties found in Bayesian methods [15].

We conduct a series of experiments to demonstrate the performance of our MDLEP approach and to compare it with that of the classical GA approach described in [14]. The empirical results illustrate that our approach is superior both in terms of quality of the solutions and computational time for most data sets we have tested.

2 Problem Definition and MDL metric

A Bayesian network is composed of a network structure and a set of parameters associated with the structure. In general, the structure consists of nodes which are connected by directed edges and form a directed acyclic graph. Each node represents a domain variable that can take on a finite set of values. Each edge represents a dependency between two nodes. A characteristic of such dependency is that it can be uncertain and is parameterized probabilistically. Formally, let $N = \{N_1, \ldots, N_n\}$ be the set of nodes representing the variables in a domain. Each N_i can instantiate from a finite set of values. In a Bayesian network concerned with N, there is a parent set Π_{N_i} for each node N_i. If N_i has no parent in the network structure, Π_{N_i} is an empty set. This structure captures the fact that the instantiation of node N_i depends on the instantiations of the nodes in Π_{N_i}. Since the dependency can be uncertain, there is a set of conditional probability parameters associated with each node. For node N_i, the probability parameters are in the form of $P(N_i \mid \Pi_{N_i})$.

The aim of learning Bayesian networks is to automatically construct such a network model from raw data. The learning problem can also be viewed as a kind of unsupervised learning where the targets to be learned are Bayesian networks. The input data are a collection of cases. Each case is a fully-instantiated set of domain variables corresponding to some observed real-world circumstance in the domain of interest. In the previous work of Lam and Bacchus on this problem [12, 11], we make use of the *Minimum Description Length* (MDL) principle as a means for balancing between simplicity and accuracy. Central to this approach is a cost metric for a candidate network structure. The cost metric is a function representing the *total description length* $D_t(B)$ of a candidate network structure B. Within the framework, a shorter length D_t corresponds to a better network. Ideally we would like to find a network structure which has the lowest D_t. We call such a network an *optimal* network. In situations where an optimal solution cannot be obtained due to limited computing resources, we wish to find a network with D_t as low as possible. The total description length D_t of a candidate network structure B can be decomposed into each individual variable. Let Π_{N_i} be the parent set of N_i. With overloading of the notation D_t, it can be expressed as:

$$D_t(B) = \sum_{N_i \in N} D_t(N_i, \Pi_{N_i})$$

More detailed description of the MDL approach can be found in [12].

3 The MDLEP Learning Approach

Evolutionary Programming (EP) uses the highest level of abstraction by emphasizing the adaptation of behavioral properties of various species [4, 5]. Genetic Algorithms (GAs) models evolution at genetic level [10, 6]. EP is a stochastic optimization strategy that emphasizes the behavioral linkage between parents and their offspring rather than seeking to emulate specific genetic operators as observed in nature. It is a useful method of optimization when other techniques such as gradient descent or direct, analytical methods are not possible. EP is suitable for difficult combinatoric and real-valued function optimization problems in which the fitness landscapes are rugged and have many locally optimal solutions. On the other hand, GAs cannot guarantee similarity between offspring and their parents because GAs emphasize on structural similarity.

There are three important differences between EP and the classical GA. Firstly, there is no constraint on the representation. The classical GA involves encoding the problem solutions as fixed-length binary strings [10, 6]. In EP, the representation follows from the problem. Thus the mutation operation does not demand and assume any particular encoding method. Secondly, the mutation operators simply change aspects of the parent according to a statistical distribution. Minor modifications in the behavior of the offspring occur more frequently than substantial variations in the behavior of the offspring. Furthermore, the severity of mutations is often reduced as the global optimum is approached. Thirdly, EP applies mutation operators only while the classical GA uses crossover, mutation, and other genetic operators. We describe our MDLEP approach to learning Bayesian network structures based on EP. The newly designed mutation operators used in our algorithm are also presented.

3.1 The Algorithm

The learning algorithm starts with an initial population of directed acyclic graphs (DAGs) called parents. Each parent is evaluated by using the MDL metric described in section 2. Next, each parent creates an offspring by performing a series of mutations to the parent. The probabilities of executing 1, 2, 3, 4, 5, or 6 times mutations are 0.2, 0.2, 0.2, 0.2, 0.1, and 0.1 respectively[1]. For each mutation, one of the four mutation operators: simple mutation, reversion, move, and knowledge-guided mutation; is selected for execution according to a uniform distribution. If mutations generate an invalid offspring that is cyclic, the algorithm deletes the edges of the offspring that invalidate the DAG conditions. The new offspring are then evaluated by using the MDL metric. The next generation of parents are selected from the current generation of parents and offspring. The algorithm performs this selection by requiring each DAG to compete against q DAGs randomly chosen from the population. If the MDL metric of the former is lower than or equal to the chosen opponent in each competition, the former receives one score. The algorithm retains the groups of DAGs with the highest scores as parents of the next generation. The algorithm repeats this process until the terminating condition is satisfied. The algorithm is summarized as follows:

1. Set t to 0.

2. Create an initial population, Pop(t), of PS random DAGs. The initial population size is PS.

3. Each DAG in the population Pop(t) is evaluated using the MDL metric.

4. While t is smaller than the maximum number of generations G

[1]These parameter values are selected to ensure that minor modifications of the offspring occur more frequently than substantial variations of the offspring.

- Each DAG in Pop(t) produces one offspring by performing a number of mutation operations. If the offspring has cycles, delete the set of edges that violate the DAG condition. If choices of set of edges exist, we randomly pick one choice.
- The DAGs in Pop(t) and all new offspring are stored in the intermediate population Pop'(t). The size of Pop'(t) is 2*PS.
- Conduct a number of pairwise competitions over all DAGs in Pop'(t). Let B_i be the DAG being conditioned upon, q opponents are selected randomly from Pop'(t) with equal probability. Let $B_{ij}, 1 \leq j \leq q$, be the randomly selected opponent DAGs. The B_i gets one more score if $D_t(B_i) \leq D_t(B_{ij}), 1 \leq j \leq q$. Thus, the maximum score of a DAG is q.
- Select PS DAGs with the highest scores from Pop'(t) and store them in the new population Pop(t+1).
- Increase t by 1.

5. Return the DAG with lowest MDL metric found in any generation of a run as the result of the algorithm.

In our experiments, we set the value of q to be 5.

3.2 The Mutation Operators

The learning algorithm uses four mutation operators, simple mutation, reversion, move, and knowledge-guided mutation, to produce new offspring from existing DAGs. Let B be an existing DAG to be mutated, $N = \{N_1, \ldots, N_n\}$ be the set of nodes representing the variables in a domain, E be the set of edges in B. The operators as described below generate a new offspring by modifying E.

Simple Mutation - This operator first randomly selects an edge e_{ij} from nodes N_i to N_j, where $i \neq j$. If this edge is present in the network it is deleted from the network, otherwise it is added to the network.

Reversion - This new operator randomly selects an edge, says e_{ij}, from E, and modifies the direction of the edge. In other words, the set of edges, E', of the offspring is: $E' = (E - \{e_{ij}\}) \cup \{e_{ji}\}$

Move - This new operator modifies the parent set of a node, says N_i, if Π_{N_i} is not empty. Specifically, it deletes a node N_k where $N_k \in \Pi_{N_i}$, from the parent set of N_i randomly, and adds a new node N_j to Π_{N_i} if $N_j \notin (\Pi_{N_i} \cup \{N_i\})$.

Knowledge-Guided Mutation - This new operator is similar to the simple mutation operator. It removes an existing edge from a Bayesian network structure or adds an edge if there is no edge between the corresponding nodes. The main difference between these two operators is that knowledge-guided mutation considers the MDL metric of all possible edges and determines which edge should be removed or inserted. The MDL metric of an edge from N_j to N_i, where $i \neq j$, is computed by using $D_t(N_i, \{N_j\})$. Before the learning algorithm is executed, the MDL metric of all possible edges is computed and stored. When knowledge-guided mutation operator determines that an existing edge of the parental network structure B should be removed, it retrieves the stored MDL metric of all edges in E and those edges with higher MDL metric will be deleted with higher probabilities. On the other hand, if knowledge-guided mutation operator decides to add an edge to the parental network structure, it gets the stored MDL metric of the edges not in E, and the edges with lower MDL metric will have higher probabilities of being added. The motivation of this operator is described in the following paragraph.

Recall that the MDL metric of a network structure B is the sum of the MDL metric $D_t(N_i, \Pi_{N_i})$ of all nodes in the network. The objective of the learning algorithm is to find a network structure with minimal MDL metric. Thus, if we want to delete an existing edge from some node to N_i, it is better to select an edge with higher MDL metric because it is more likely to reduce the MDL metric $D_t(N_i, \Pi_{N_i})$ of node N_i.

Similarly, if we want to insert an edge from some node to N_i, it is better to select an edge with lower MDL metric because it is more likely to minimize the increment of the MDL metric of node N_i.

4 Empirical Results and Evaluation

We have conducted a number of experiments to evaluate the performance of our MDLEP approach. We also compare it with a classical GA approach. In each experiment, the learning algorithms attempt to learn a Bayesian network from a data set. The data sets are generated from known Bayesian network structures and conditional probability tables using probabilistic logic sampling technique. The learning algorithms take the data set only as input. They do not know the Bayesian networks that generate the data set in any ways during the learning process. After a Bayesian network structure is learned, it is evaluated by two measures. One undirected measure is the *structural difference* which is defined as: $\sum_{i=1}^{n} \phi_i$ where ϕ_i is the sum of the symmetric difference of each parent in the learned network and the known network. Another measure is the total description length D_t. The lower the measure, the better is the network.

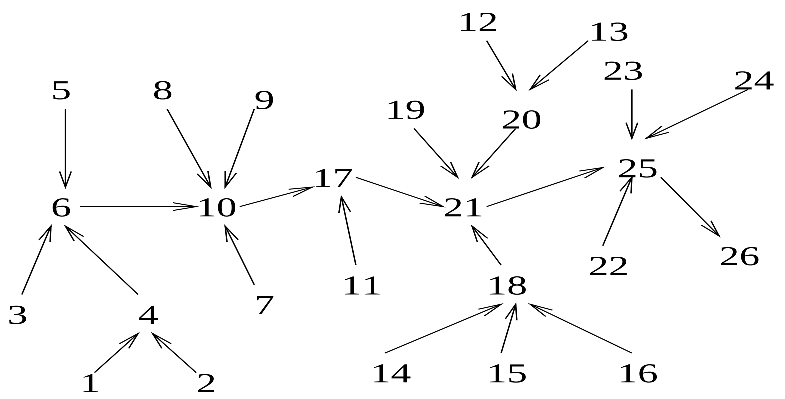

Figure 1: The PRINTD Network Structure

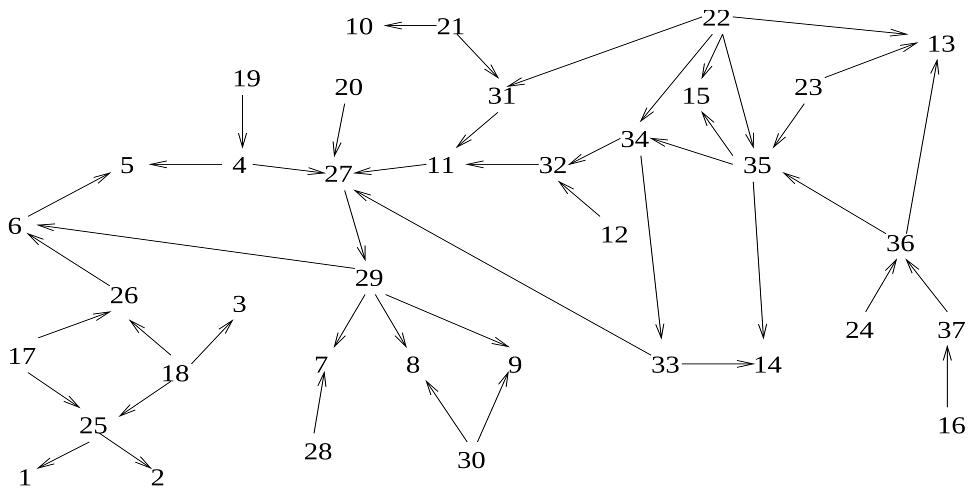

Figure 2: The ALARM Network Structure

The data set PRINTD is derived from a network structure shown in Figure 1. This network structure deals with troubleshooting a printing problem discussed in [8]. 5000 cases were generated. The data set ALARM is derived from a network structure shown in Figure 2. This structure is concerned with a medical domain of potential anesthesia diagnosis in the operating room [1]. We generated 500, 1000, 2000, 3000, 5000 cases from this structure. The MDL metric of the original network structures for the PRINTD data set is 106541.62. The MDL metric of the original network structures for the ALARM data sets of 500, 1000, 2000, 3000, and 5000 cases are 10533.33, 18533.45, 34287.88, 49595.82, and 81223.41 respectively.

4.1 Comparison Between MDLEP and GA

We employ our MDLEP learning algorithm to solve the ALARM problem with 500 cases and the PRINTD problem with 5000 cases. The population size PS is 50 and the maximum number of generations is 5000. Forty trials of these experiments were performed. We also implemented a classical genetic algorithm (GA) similar to the work done by [14]. In the GA approach, the MDL metric is used for the objective function.

A Bayesian network structure with n nodes is represented by an $n \times n$ connectivity matrix C, where its elements, c_{ij} is 1 if j is a parent of i and 0 otherwise. An individual of the population is represented as a string: $c_{11}c_{21} \cdots c_{n1}c_{12}c_{22} \cdots c_{n2} \cdots c_{1n}c_{2n} \cdots c_{nn}$ The one point crossover and mutation operations of classical GA are used [6, 10]. The population size PS is 50 and the maximum number of generations G is 5000. The crossover probability p_c is 0.9 and the mutation rate p_m is 0.01[2]. Elitist selection without local optimization is used in the experiments. Forty trials of these experiments are performed. For the ALARM problem with 500 cases, we record the MDL metric of the best Bayesian network of each successive population and calculate the average values of the 40 trials for increasing generations. The MDL metric for our MDLEP learning algorithm and the GA are delineated in Figure 3. The structural differences between the best Bayesian network structure found in each trial and the original ALARM network for the two algorithms are summarized in Table 1. The last two columns of the table contain:

- the average of the structural differences of the 40 trials (ASD), and
- the smallest structural difference of the 40 trials (SSD)

We have also collected a number of statistics presented in Table 2. The rows of the table are:

- The average of the MDL metric of the 40 trials (AOM).
- The smallest MDL metric of the 40 trials (SMM).

[2]We have performed a number of experiments of using the GA with different combinations of parameter values to solve the PRINTD problem with 5000 cases. These combinations are ($P_c = 0.9$, $P_m = 0.01$), ($P_c = 0.9$, $P_m = 0.1$), ($P_c = 0.5$, $P_m = 0.01$), and ($P_c = 0.5$, $P_m = 0.1$). The results of these experiments indicate that ($P_c = 0.9$, $P_m = 0.01$) is the best combination.

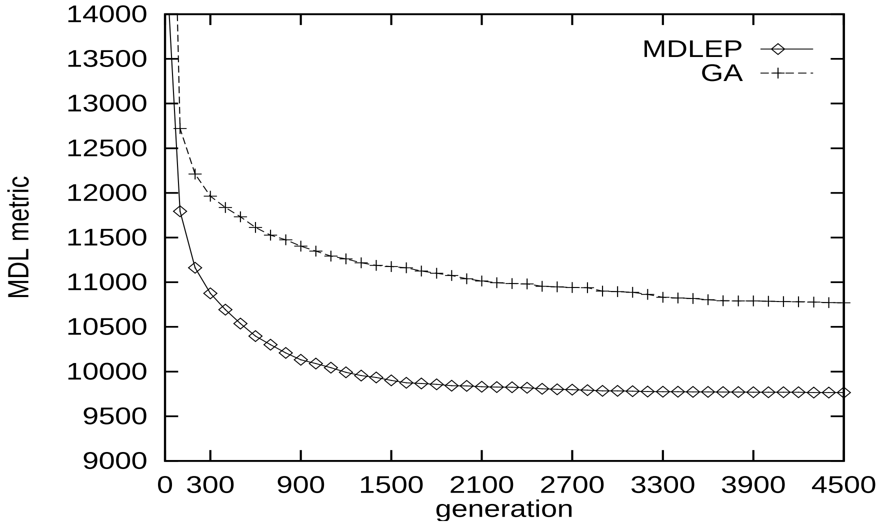

Figure 3: The MDL metric for the ALARM problem with 500 cases

- The average number of generations (ANG) performed before the best Bayesian network structure is found.
- The average number of MDL metric evaluations per Bayesian network structure (AME).
- The average size of the parent sets (APS).
- The average number of invalid Bayesian network structures produced in each generation (AIB).
- The average number of edges deleted in each invalid Bayesian network (AED).

From Figure 3, we can see that MDLEP induces much better Bayesian network structures than the GA. The values of ASD and SSD for MDLEP (Table 1) are 24.6 and 17 respectively. On the other hand, the values of ASD and SSD for the GA are 56.9 and 45 respectively. An one-tailed paired t-test is used to determine if ASD

											ASD	SSD
MDLEP	27	21	29	24	26	32	21	19	27	27		
GA	63	68	61	61	53	54	53	53	65	52		
MDLEP	27	24	27	24	25	24	25	25	24	25		
GA	54	54	45	58	53	56	68	59	57	55		
MDLEP	27	24	18	30	29	27	27	22	24	17		
GA	57	62	63	56	58	56	54	61	53	54		
MDLEP	18	25	29	25	24	25	23	24	18	25	24.6(3.4)	17
GA	56	58	49	55	61	59	57	56	52	57	56.9(4.7)	45

Table 1: The structural differences between the best Bayesian network structure found and the original ALARM network. The numbers in brackets are standard deviations

	AOM	SMM	ANG	AME	APS	AIB	AED
MDLEP	9761.90	9689.55	4205.47	3.58	1.04	19.91	1.27
	(36.56)		(562.60)	(0.01)	(0.02)	(0.48)	(0.02)
GA	10720.63	10436.42	4495.42	10.37	2.13	49.40	5.68
	(140.23)		(598.13)	(0.36)	(0.04)	(0.14)	(0.23)

Table 2: Comparison between MDLEP and the GA for the ALARM problem with 500 cases. The numbers in brackets are standard deviations

for MDLEP is significantly smaller than that for the GA at 95% confidence level. The t-statistics is 35.07, thus the Bayesian network structures generated by MDLEP are qualitatively better than those obtained by the GA. From Table 2, we find that MDLEP evolves good Bayesian network structures at an average generation of 4205. The values of AOM and SMM for MDLEP are 9761.90 and 9689.55 respectively. The values of AOM and SMM for the GA are higher than those of MDLEP. They are respectively 10720.63 and 10436.42. The GA obtains the solutions at an average generation of 4495.42. To determine if ANG for MDLEP is significantly smaller than that for the GA at 95% confidence level, an one-tailed paired t-test is performed and the t-statistics is 2.23. From this information, we can conclude that MDLEP finds better network structures at earlier generations than the GA. MDLEP is about 6.82 times faster than the GA when they are executed on a Sun UltraSparc machine. The values of AME, APS, AIB, and AED for MDLEP are smaller than those of the GA, thus MDLEP is better and faster than the GA.

For the PRINTD problem with 5000 cases, we record the MDL metric of the best Bayesian network of each successive population and calculate the average values of the 40 trials for increasing generations. The values for MDLEP and the GA are delineated in Figure 4. The structural differences between the best Bayesian network structure found in each trial and the original PRINTD network for the two algorithms are summarized in Table 3. A number of statistics are presented in Table 4.

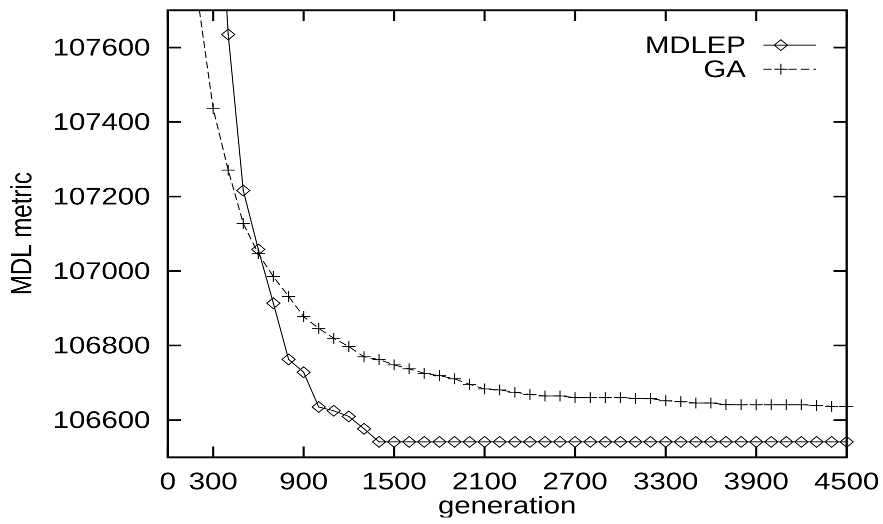

Figure 4: The MDL metric for the PRINTD problem with 5000 cases

We observe from Figure 4 that MDLEP induces better Bayesian network structures than the GA. Table 3 indicates that MDLEP evolves the original PRINTD network structure in all trials. On the other hand, the GA produces the original PRINTD in only 12 trials. From Table 4, we find that MDLEP evolves good Bayesian network structures at an average generation of 1032.32. The values of AOM and SMM for MDLEP are both 106541.62. The values of AOM and SMM for the GA are slightly higher than those of MDLEP. They are respectively 106622.09 and 106541.62. The GA obtains the solutions at an average generation of 3555.55. An one-tailed paired t-

											ASD	SSD
MDLEP	0	0	0	0	0	0	0	0	0	0		
GA	4	8	0	21	0	0	6	12	0	0		
MDLEP	0	0	0	0	0	0	0	0	0	0		
GA	6	0	3	3	15	3	0	0	5	4		
MDLEP	0	0	0	0	0	0	0	0	0	0		
GA	3	6	0	9	0	8	10	0	2	13		
MDLEP	0	0	0	0	0	0	0	0	0	0	0.0(0.0)	0
GA	7	6	2	0	3	9	1	6	8	10	4.82(4.91)	0

Table 3: The structural differences between the best Bayesian network structure found and the original PRINTD network. The numbers in brackets are standard deviations

	AOM	SMM	ANG	AME	APS	AIB	AED
MDLEP	106541.62	106541.62	1032.32	2.84	1.98	18.69	1.31
	(0.0)		(243.89)	(0.88)	(0.05)	(5.90)	(0.01)
GA	106622.09	106541.62	3555.55	6.36	2.19	38.64	2.81
	(82.06)		(1033.69)	(0.11)	(0.07)	(2.00)	(0.25)

Table 4: Comparison between MDLEP and the GA for the PRINTD problem with 5000 cases. The numbers in brackets are standard deviations

test is used to determine if ANG for MDLEP is significantly smaller than that for the GA at 95% confidence level. The t-statistics is 15.02, thus, MDLEP induces the original PRINTD network structure at much earlier generations than the GA. MDLEP is about 3.62 times faster than the GA when they are executed on a Sun UltraSparc machine. The values of AME, APS, AIB, and AED for MDLEP are smaller than those of the GA, thus MDLEP is better and faster than the GA.

4.2 Parameter of MDLEP

We study the effect of different values of q for the MDLEP algorithm on solving the ALARM problem with 500 cases and the PRINTD problem with 5000 cases. The population size PS is 50 and the maximum number of generations is 5000. The values of q are 3, 5, 7, 9, and 11. Ten trials of these experiments were performed. The structural differences between the best Bayesian network structure found in each trial and the original ALARM network for the MDLEP algorithm with different values of q are summarized in Table 5. For the PRINTD problem with 5000 cases, the structural differences are presented in Table 6. We find that there is not significant difference for various values of q by using a two-tailed paired t-test at 95% confidence level.

	Trial label										ASD	SSD
	1	2	3	4	5	6	7	8	9	10		
q=3	27	21	29	24	26	32	21	19	27	27	25.3(4.03)	19
q=5	27	24	27	24	25	24	25	25	24	25	25(1.15)	24
q=7	27	24	18	30	29	27	27	22	24	17	24.5(4.40)	17
q=9	18	25	29	25	24	25	23	24	18	25	23.6(3.34)	18
q=11	30	27	21	25	17	23	22	18	28	23	23.4(4.20)	17

Table 5: The structural differences of the best BAYESIAN network found by the MDLEP algorithm with different values of q for the ALARM problem with 500 cases. The numbers in brackets are standard deviations

	Trial label										ASD	SSD
	1	2	3	4	5	6	7	8	9	10		
q=3	0	0	0	0	0	0	0	0	0	0	0.0(0.0)	0
q=5	0	0	0	0	0	0	0	0	0	0	0.0(0.0)	0
q=7	0	0	0	0	0	0	0	0	0	0	0.0(0.0)	0
q=9	0	0	0	0	0	0	0	0	0	0	0.0(0.0)	0
q=11	0	0	0	0	0	0	0	0	0	0	0.0(0.0)	0

Table 6: The structural differences of the best BAYESIAN network found by the MDLEP algorithm with different values of q for the PRINTD problem with 5000 cases. The numbers in brackets are standard deviations

4.3 Learning the ALARM Network Using MDLEP

In sub-section 4.1, we observe that MDLEP is superior to the classical GA proposed by Larrañaga *et. al.* [14]. Thus, more experiments are done to determine the performance of MDLEP on learning the ALARM network with 1000, 2000, 3000, and

5000 cases. The parameter values of MDLEP described in sub-section 4.1 are also used here. Ten trials of each experiment are conducted. A number of statistics are presented in Table 7. The structural differences between the best Bayesian network structure found in each trial and the original ALARM network for all experiments are summarized in Table 8.

	1000	2000	3000	5000
AOM	18021.33	33825.42	49431.21	81148.56
	(64.65)	(125.29)	(284.52)	(262.25)
SMM	17905.80	33745.70	49230.08	81004.00
ANG	3914.6	3554.1	3962.8	3931.7
	(674.39)	(667.74)	(884.20)	(505.19)
AME	3.59	3.63	3.64	3.64
	(0.02)	(0.01)	(0.02)	(0.02)
APS	1.16	1.29	1.36	1.43
	(0.03)	(0.02)	(0.02)	(0.02)
AIB	20.81	21.61	22.65	22.62
	(0.45)	(0.38)	(0.59)	(0.65)
AED	1.29	1.32	1.35	1.36
	(0.01)	(0.02)	(0.02)	(0.03)

Table 7: Performance of MDLEP for the ALARM problem with 1000, 2000, 3000, and 5000 cases. The numbers in brackets are standard deviations

From Table 7, we can observe that the values of AOM and SMM for all experiments are smaller than the corresponding MDL metric of the original ALARM network. In other words, there are Bayesian network structures with smaller MDL metric in the search space and MDLEP can successfully find them in all experiments. The values of ANG show that the best network structures are obtained at generations between 3554 and 3963. The values of AME, APS, AIB, AED for all experiments are within a small interval. However, for a problem with larger number of cases, their values are normally larger than those with smaller number of cases. In other words, the ALARM problem with larger number of cases is harder and takes longer to solve than the one

	number of cases			
	1000	2000	3000	5000
Trial 1	22	9	9	7
Trial 2	20	12	8	9
Trial 3	22	9	16	13
Trial 4	15	10	17	10
Trial 5	27	10	15	10
Trial 6	15	8	18	11
Trial 7	23	10	8	8
Trial 8	16	12	9	10
Trial 9	26	12	11	17
Trial 10	24	13	9	11
ASD	21.0(4.40)	10.5(1.65)	12(4.03)	10.6(2.80)
SSD	15	8	8	7

Table 8: The structural differences between the best Bayesian network structure found and the original ALARM network. The numbers in brackets are standard deviations

with smaller number of cases. Table 8 indicates that MDLEP can produce network structures that are similar to the original ALARM network.

5 Conclusions

We have presented a novel approach (MDLEP) to learning Bayesian network structures based on the Minimum Description Length (MDL) principle and Evolutionary Programming (EP). Specifically, our approach employs a MDL-based Bayesian network learning technique which is founded on information theory and integrates an optimization process which is based on evolutionary programming. An important characteristic of our approach is that, we have designed a knowledge-guided operator which incorporates a MDL learning scheme. Essentially we have made use of the network structure discovery knowledge for developing the genetic operators. We have conducted a series of experiments to demonstrate the performance of our MDLEP approach and to compare the performance of the MDLEP approach with the classical

GA approach described in [14]. The empirical results illustrate that our approach is superior both in terms of quality of the solutions and computational time in most data sets we have tested. For future work, we will compare the MDLEP approach with other algorithms including stochastic hillclimbing and simulated annealing methods [18] .

References

[1] I. A. Beinlich, H. J. Suermondt, R. M. Chavez, and G. F. Cooper. The ALARM monitoring system: A case study with two probabilistic inference techniques for belief networks. In *Proceedings of the 2nd European Conference on Artificial Intelligence in Medicine*, pages 247–256, 1989.

[2] D. Chickering, D. Geiger, and D. Heckerman. Learning Bayesian networks: Search methods and experimental results. In *Proceedings of the Fifth Conference on Artificial Intelligence and Statistics*, pages 112–128, 1995.

[3] G. F. Cooper and E. Herskovits. A Bayesian method for the induction of probabilistic networks from data. *Machine Learning*, 9:309–347, 1992.

[4] D. B. Fogel. An introduction to simulated evolutionary optimization. *IEEE Trans. on Neural Network*, 5:3–14, 1994.

[5] L. Fogel, A. Owens, and M. Walsh. *Artificial Intelligence through Simulated Evolution.* New York: John Wiley and Sons, 1966.

[6] D. Goldberg. *Genetic Algorithms in Search, Optimization, and Machine Learning.* Reading MA: Addison-Wesley, 1989.

[7] D. Heckerman, D. Geiger, and D. M. Chickering. Learning Bayesian networks: The combination of knowledge and statistical data. *Machine Learning*, 20(3):197–243, 1995.

[8] D. Heckerman and M. Wellman. Bayesian networks. *Communications of the ACM*, 38(8):27–30, 1995.

[9] E. Herskovits and G. Cooper. KUTATO: An entropy-driven system for construction of probabilistic expert systems from databases. Technical report, Knowledge Systems Laboratory, Medical Computer Science, Stanford University, KSL-90-22, 1990.

[10] J. Holland. *Adaptation in Natural and Artificial Systems.* Cambridge MA: MIT Press, 1992.

[11] W. Lam. Bayesian network refinement via machine learning approach. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 20(3):240–251, 1998.

[12] W. Lam and F. Bacchus. Learning Bayesian belief networks - an approach based on the MDL principle. *Computational Intelligence*, 10(3):269–293, 1994.

[13] P. Larrañaga, C. Kuijpers, R. Murga, and Y. Yurramendi. Learning Bayesian network structures by searching for the best ordering with genetic algorithms. *IEEE Transactions on Systems, Man, and Cybernetics - Part A: Systems and Humans*, 26(4):487–493, 1996.

[14] P. Larrañaga, M. Poza, Y. Yurramendi, R. Murga, and C. Kuijpers. Structure learning of Bayesian network by genetic algorithms: A performance analysis of control parameters. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 18(9):912–926, 1996.

[15] J. Rissanen. Modeling by shortest data description. *Automatica*, 14:465–471, 1978.

[16] P. Spirtes, C. Glymour, and R. Scheines. *Causation, Prediction, and Search.* Springer-Verlag, 1993.

[17] P. Spirtes and C. Meek. Learning Bayesian networks with discrete variables from data. In *Proceedings of the First International Conference on Knowledge Discovery and Data Mining*, pages 294–299, 1995.

[18] P. J. M. van Laahoven and E. H. L. Aarts. *Simulated annealing: Theory and applications.* Dordrecht: D. Reidel Publishing Company, 1988.