2006 IEEE Congress on Evolutionary Computation
Sheraton Vancouver Wall Centre Hotel, Vancouver, BC, Canada
July 16-21, 2006

Evolutionary Aggregation and Refinement of Bayesian Networks

Kyung-Joong Kim and Sung-Bae Cho
Dept. of Computer Science, Yonsei University
134 Shinchon-dong, Sudaemoon-ku, Seoul, Korea
kjkim@cs.yonsei.ac.kr, sbcho@cs.yonsei.ac.kr

***Abstract*—**Bayesian network (BN) is a useful tool to represent joint probability distribution in the form of graphical model providing flexible inference and uncertainty handling. If there is enough knowledge about domain, it is possible to design the structure and parameters of BN by expert. Also, it can be learned from massive dataset with statistical learning algorithm. Usually, because the search space of Bayesian networks is relatively huge compared to the other models, evolutionary algorithms have been used to find optimal structure and parameters by many researchers. In this paper, we have focused on the topic of adaptation of constructed models for better performance. If there are a number of models constructed or learned by different experts or sources, it is better to fuse them into one model by considering all the information of each model. However, the complexity of the integrated model is relatively higher than previous isolated models. Minimizing the complexity of the integrated model using evolutionary algorithm is proposed. After integrating models into single one, it needs to adapt to the new data from the environment. It is likely to provide wrong results to the newly generated data from the environment and slightly modifying the joint probability distribution is necessary. The refinement process is also guided by the evolutionary algorithm because the space of search is large. Experimental results on a benchmark network show that the proposed adaptation methods with evolutionary algorithm can perform better than heuristics or greedy approaches.

I. Introduction

There are two different approaches for adaptation of models: long-term and short-term adaptation [19]. Because greedy or heuristics are very fast to find solution, it is appropriate for short-term adaptation. It has a risk of getting stuck into the local optima. On the other hands, evolutionary algorithm needs more time to optimize the solutions but they are well in global search because of population-based search. It is better to use evolutionary approach for long-term adaptation.

Adaptation is the process of reconfiguration of the basic installed system to the change of the environment. If human has no adaptation capability, he cannot survive or travel in many different areas because preparing all the information is not possible. In the perspective of Bayesian networks (BN), adaptation process means the modification of joint probability distribution by deleting, adding, or removing edges and changing parameters. The original model is defined as *O* and new information sources represented by Bayesian networks or raw data sets are referred as *N*. The purpose of adaptation process of BN is defined as a search process of appropriate BN that is similar to *O* and adjusted to the *N*.

If the new information is represented as a BN, the process of adaptation is called as aggregation. It is simply fusion of two models into an integrated one. On the other hands, if the new information is a data set, the adaptation is called as refinement. Although, the structure and parameter of Bayesian networks are the targets of the adaptations, we focus only on the structural one because the structure of the model is more important than the parameters.

The procedure of adaptation of Bayesian networks can be formulated as learning given original and new sources. In this paper, we have applied evolutionary algorithm to find the appropriate Bayesian networks given two sources (original and new). Because the aggregation and refinement of Bayesian network structure can be formulated in a different manner, the procedures of applying evolutionary algorithm into adaptation are a bit different. In aggregation part, basic aggregation algorithm of two Bayesian networks is from [3]. As noted by [4], the task of minimizing the number of arcs in their combined directed acyclic graph is NP-hard. Given a number of Bayesian networks, the evolutionary algorithm is used to minimize the number of arcs of combined model. In refinement part, scoring measure based on the minimum description length is adopted from [5]. In this paper, we have applied genetic algorithm to find the appropriate solution given the measure. Experimental results on ASIA network, one of the representative benchmark networks, show that the proposed methods can perform better than the heuristics or simple greedy algorithms.

Contribution of this paper is like this. (1) The first experimentation on the optimization of the number of edges of aggregation based on the fusion method [3]. (2) Proposing chromosome representation based on permutation index for aggregation optimization. (3) The first evolutionary approach for the refinement measure proposed by [5].

II. Backgrounds

A. Bayesian Networks

Bayesian network is an acyclic directed graph that represents joint probability distributions of random variables.

0-7803-9487-9/06/$20.00/©2006 IEEE

Its node represents the random variable and the edges between nodes mean that the two nodes are probabilistically dependent on each other. Each node has its own conditional probability table. It describes conditional probabilities of itself given the states of parents. If there are n variables in the domain, the set of variables is $X = \{x_1, x_2, \cdots, x_n\}$. The basic meaning of edges between two nodes is cause and effect relationships. If there is link from parent to child node, it means that parent node is a cause and the child node is a result of the cause. The conditional probability of the node is defined as $\Pr(x_i \mid Pa(x_i))$. $Pa(x_i)$ is a set of parent nodes of x_i. The conditional probabilities define the strength between the parent and child. The joint probability distribution of n variables given such dependency (parent-child relationship) and conditional probabilities is defined using this formula.

$$JPD(x_1, x_2, \cdots x_n) = \prod_{i=1}^{n} \Pr(x_i \mid Pa(x_i))$$

If there is enough knowledge about cause-effect relationships and their strength among variables, it is not difficult to design Bayesian networks that represent the knowledge. However, such kinds of knowledge are not allowable in many domains excluding some special areas such as medical science, psychology, and troubleshooting of systems. If there is no available knowledge, it is possible to learn the structure and parameters of Bayesian networks using statistical learning algorithm. Structure of Bayesian networks means the parent-child relationships (edge of variables) and parameters are conditional probability tables. Learning of Bayesian networks are based on a scoring function that returns the degree of fitting of the network to the given datasets. The problem of searching for appropriate Bayesian networks given massive dataset is defined as searching problem in the Bayesian network space.

B. Evolutionary Approaches for BN

The search space of Bayesian networks is exponentially extended, if the number of variables increases. Enumerating all possible Bayesian networks is not feasible, heuristic search algorithm is necessary. The representative method is greedy heuristics. It incrementally adds an edge that maximizes the score increase given current structures. Though, it can get stuck into local optima, it is widely used because of its simplicity of implementation and speed of search. However, the risk of getting stuck into local optima causes the adoption of evolutionary algorithm for inducting Bayesian networks from data. There are two different approaches for inducing Bayesian networks using evolutionary algorithms.

The first approach uses connection matrix as a chromosome that represents Bayesian network [2]. When the number of variables is n, the size of matrix is $n \times n$. Each entry indicates whether there is edge between two nodes. e_{ij} represents the entry of the i th row and the j th column. If the value of the entry is 1, it means that the j th variable is the parent of the i th node. The search space of n variables is $2^{n \times n}$. Because BN does not allow cycles, there could be many invalid Bayesian networks. Also, some of Bayesian networks that have different structures might refer the identical probabilistic distribution and it can distort the search space. Some researchers attempt to solve this problem by applying the concept of equivalent classes [6][7]. Genetic operations on the model can generate invalid Bayesian networks and some repairing operators are needed to transfer the invalid one to a valid BN.

The second approach is searching for topological order for BN [1]. The order restricts the connection among variables by prohibiting the connection from low-ordered variables to high-ordered variables. The purpose of evolutionary algorithm is to find appropriate topological order. Given the order found, Bayesian networks are chosen using simple local search algorithm. This approach can save the search space significantly but still the size of space is relatively large. Some genetic operators used for traveling salesman problem can be adopted. Because it always guarantees the validity of Bayesian networks (topological order prohibits the cyclic links), repairing operations are not needed.

Matrix representation is easy to implement and intuitive to the programmer but needs additional effort to deal with large search space and invalid Bayesian networks. Compared to the former, order-based representation requires some knowledge about genetic operator design and it needs additional local search methods given the order found. However, it can reduce the search space and does not need repairing operators.

There are some different approaches for learning Bayesian networks. Lam *et al.* used evolutionary programming to evolve Bayesian networks and proposed knowledge-guided mutation for improving performance [16]. Novobilski *et al.* [17] proposed a genetic programming approach [17]. There is research on the development of new representation and genetic operators [18]. Campos *et al.* used evolutionary algorithm to enhance the inference procedure of Bayesian networks [20].

C. Aggregation and Refinement of BN

Kim *et al.* proposed evolutionary algorithm with fitness sharing to generate diverse Bayesian networks given massive dataset. At the final generation, the inference results of the representative Bayesian networks are combined for better inference [8]. Chen *et al.* uses a collective approach to learning a Bayesian network from distributed heterogeneous data. Chen *et al.* used union of nodes and edges of local Bayesian networks and non-local Bayesian networks [9][10]. Given a number of sites, they have learned each local Bayesian networks using their local data and non-local Bayesian network is learned using meta-information collected from each site. Moral *et al.* indicated shortcomings of previous Bayesian networks combination methods and proposed the union and intersection of independencies of each Bayesian network [11]. Pennock *et al.* investigated previously assumed context of consensus belief such as the

separation of conditional probability when authors find the combined structure [12].

Lam *et al*. proposed a refinement measure based on the minimum description length [13]. In this approach, current network was used as a summarization of previously seen data and Friedman *et al*. proposed a similar method based on maximum a-posteriori probability approach [14]. Tian *et al*. improved the work of Friedman *et al*. and applied evolutionary algorithms to avoid getting into a local maximum [15]. In our approach, we proposed an evolutionary approach for Lam's refinement measures.

III. Adaptation of BN Using Evolutionary Algorithms

A. Aggregation of Multiple Bayesian Networks

If there are a number of authors of Bayesian networks about the same domain, there could be a variety of models that describe similar things. Because they have different expertise about the domain, it is better to integrate them into a single model. The easiest way of combining them is to use intersection and union operations. In the case of intersection operator, the common structure of all Bayesian networks is used as a global Bayesian network. On the other hands, union operator put all of the edges and variables of the networks into a global network.

In the union operation, there are critical problem causing cycles. Given intermediate global network, inserting a new edge from one of networks could make a cycle and it must be discarded. However, it causes a loss of information. Using a reverse operation is a solution to the problem (figure 1). By reversing the direction of edge that makes a cycle, the edge can be incorporated to the intermediate global Bayesian network. Using this operation, the networks are able to encode the same relationships among variables with different settings of individual conditional probabilities.

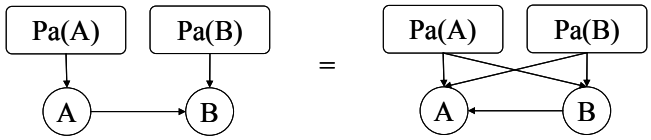

Figure 1: A reverse operation

A set of Bayesian networks that need to be integrated into one model is defined as $S = \{B_1, B_2, ..., B_N\}$. N is the number of networks for combination. The integration process is conducted incrementally. At first, B_1 and B_2 are combined and the result is referred as B_{12}. Then, B_3 is combined with B_{12} and it produces B_{123}. Like this, the combination procedure is continued until the last Bayesian network is fused into the integrated model. The final model is referred as global Bayesian network or $B_{123\cdots N}$.

If there are N Bayesian networks, the problem is how to determine the order of Bayesian networks for combination. According to the order, the result of integration process is different and the number of edges for the model varies. For example, the result of $B_{123\cdots N}$ and $B_{21345\cdots N}$ is not the same. For N Bayesian networks, there are $N!$ cases of combination. The details of combination of two Bayesian networks are described in [3].

There are source network and target network. B_{12} is the result of the combination of source network B_1 and target network B_2. B_{123} is the combination of source network B_{12} and target network B_3. The combination algorithm fuses the structural information of target network into source network. The edges of target network are classified into three categories: DIR, REV, and EQ. The decision of the category is based on the topological values of the variables of source network. The topological value of variable x_i means the length of the longest path from the root node of topological graph of source network to the variable. The connection from the node that has low topological value to the node with high topological value does not make cycles. However, cycles are generated in the case of the opposite way.

DIR means that the edge of target network has the right direction that is directly inserted into the source network without modification. REV means that the edge that needs to be reversed using the reverse operation for the insertion. The reverse operation generates a number of new edges and they need to be classified into the three categories. EQ means the edges of two variables that have the same topological value. Insertion of the EQ edge makes the change of topological values of source network.

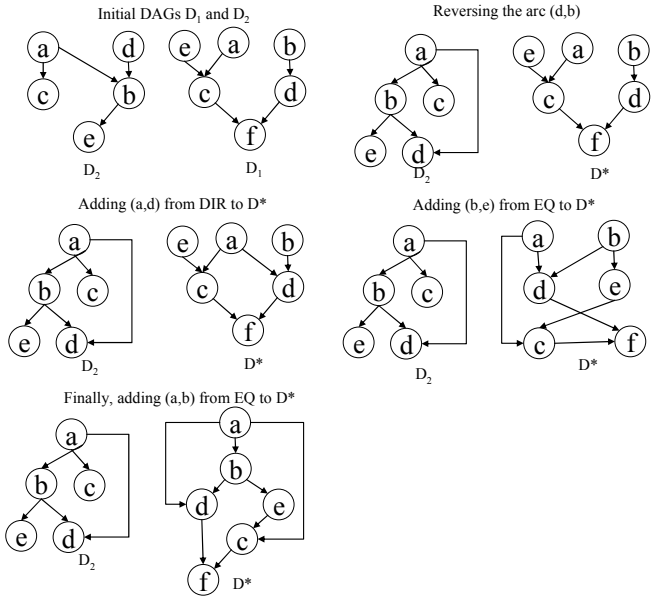

Figure 2: An example of topological fusion [3].

The algorithm has six steps. 1) Calculating the topological values of the variables in the source network. 2) Classifying the categories of the edges into DIR, REV and EQ. 3) For each edge in the REV, applying reversing operation to the target network and classifying the new edges from the operation into the three categories are done. 4) Inserting edges in the DIR into the source network from target network.

5) For each edge in the EQ, add the edge into the network and update the topological value of source network (some edges in EQ is transferred to the DIR). 6) After clearing all the edges in the three categories, the process is finished. The detailed insertion order of edges in the REV and EQ is described in the [3]. Figure 2 shows an example of topological fusion.

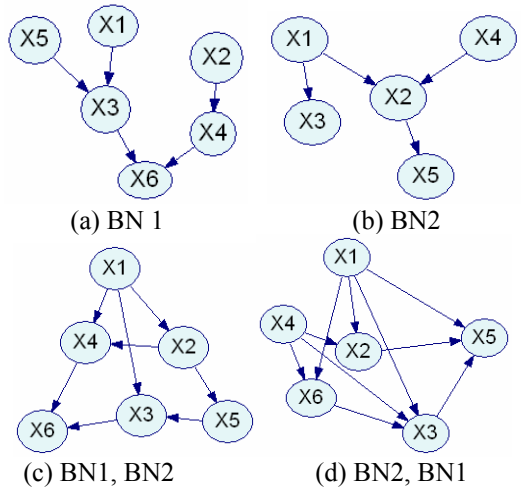

Figure 3: The effect of the order

The goal of the evolution of the order of Bayesian networks for combination is to minimize the edge of the final network. Figure 2 shows the effect of order for combination. The change of order results in different final network. The complexity of model is related to the generalization ability and the degree of difficulty to insert the parameters. In Bayesian network, the insertion of one edge can increase a huge number of parameters when the number of parents is relatively high or the number of discrete states of variables is large.

To design the evolutionary algorithm, we have investigated the search space of the problem. The number of candidates for the combination is the factorial of the number of Bayesian networks. If the number of Bayesian networks is 15, the number of candidates (the size of solution space) is 1.3×10^{12}. Enumerative approach for the problem is not feasible because the combination process itself has computational cost and the number of candidates is growing exponentially.

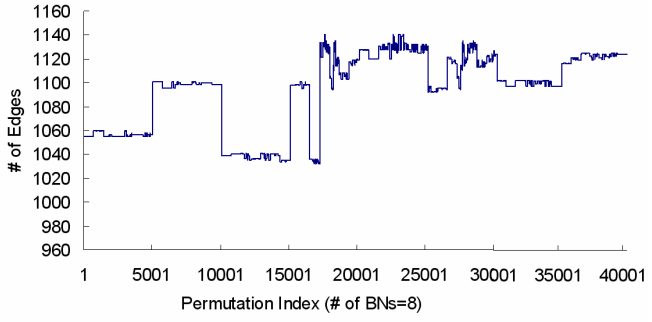

Figure 4: The fitness landscape in the case of 8 Bayesian networks.

To get some information about the search space, we have generated a number of random Bayesian networks. The number of variables of each Bayesian network ranges from 3 to 40. A set of variables is $X=\{x_1,x_2,\cdots,x_{50}\}$. If the number of variables of BN is 37, 37 variables are chosen from X without duplication. When the number of Bayesian networks is larger than 8, it is impossible to enumerate all the candidates and draw fitness landscape. Figure 4 shows the fitness landscape of Bayesian network combinations for 8 Bayesian networks. The global optimum is located around at 16561~17281. The x-scale of the figure represents permutation index of 8 Bayesian networks. For example, $B_{12345678}$ is 1 and $B_{87654321}$ is the last index (40320) . Figure 5 shows an example of permutation index.

Permutation Index	Order of combination (Each number represents BN ID)							
0	1	2	3	4	5	6	7	8
1	1	2	3	4	5	6	8	7
2	1	2	3	4	5	7	6	8
3	1	2	3	4	5	7	8	6
4	1	2	3	4	5	8	6	7
5	1	2	3	4	5	8	7	6
…	…	…	…	…	…	…	…	…
40318	8	7	6	5	4	3	1	2
40319	8	7	6	5	4	3	2	1

Figure 5: Permutation index for 8 BNs.

```
1:/* N : the number of Bayesian networks for combination */
2: /* POP: Population */
3: /* order[] : array with length N */
4: /* Permutation(permutation index): return the order of N
   items given the inputted permutation index */
5: /* fitness(j) : fitness of jth individual */
6: /* Initialization(population): Each individual is
   initialized as a value from 0 to N!-1 */
7: /* θ: random variable */
8:
9: Initialization(POP);
10:
11: For i=1 to MAX_GEN {
12:   For j=1 to |POP| {
13:     order = Permutation(POP(j));
14:     fitness(j) = Fusion (order); }
15:   POP=Selection(POP, fitness); // selection
16:   For j=1 to |POP|/2  // crossover
17:   {Parent(); // select two indexes
        POP(children)=
        θ×POP(parent1)+(1-θ)×POP(parent2);}
18:   For j=1 to |POP| // mutation
19:     POP(j)=Random(POP(j), 0, N!-1)
20:}
```

Figure 6: The pseudo code for evolutionary aggregation

In this paper, we have used permutation index as a representation of evolutionary algorithms because the fitness landscape is partially continuous (not extremely fluctuated). If the small change of order makes huge difference of performance, it is difficult to use such representation. But, in this case, it is relatively continuous at neighbor of point though there are some sudden decreases. The adoption of the representation makes the implementation of the genetic operations very easy compared to the order-based representation. In permutation index, 1 is used to represent the combination order 12345678. The crossover operator is simply defined as the middle point of two parents. Figure 6 shows the pseudo code of the evolutionary aggregation.

B. Refinement of Bayesian Networks

The refinement process of Bayesian network is defined as a machine learning problem. The input of learning is the original network and new input data. The output is refined network. The goal of refinement is to find the appropriate refined network given the original network and new input data. Original network that is learned from the accumulated data or designed manually can be wrong to the newly inputted data. The original network has good estimation about the domain but the recent change of the distribution makes the model has difficulty to estimate properly to the new data. Because it is a sudden change, the change of whole distribution based on the new data can cause severe problem when the environment produces usual cases.

Lam *et al*. proposed a minimum description length (MDL)-based score metric for the refinement process [5]. They used MDL as a basic principle for the refinement learning. It finds a Bayesian network that has relatively low complexity with the similarity to the original network and new distribution cased by the newly inputted data. In the MDL, the best network is one with the minimized sum of description lengths of the learning data and model itself. The problem is to search the best one among a number of candidates. Let us define H_p as the candidate network and H_O as the original network. Newly inputted data are represented as D. Basic components of the score metric are NDL, DDL, and NDSL.

(1) Network description length of H_p

(2) Data description length of D given H_p

(3) Network description length of source H_O given H_p

The refinement metric is defined as the sum of (1), (2) and (3). The purpose of learning is the minimization of the sum. If the domain has n variables, $X = \{x_1, x_2, \cdots, x_n\}$. k_i represents the number of parents of x_i. d is the necessary number of bits for storing real-value. s_i is the number of states of x_i. ps_j represents the number of states of the jth parents of x_i. The first term of the metric is defined using this formula.

$$\sum_{i=1}^{n} (k_i \log_2(n) + d(s_i - 1) \prod_{j=1}^{k_i} ps_j)$$

q_i represents the number of all instances of parents of x_i. R means the relative frequency in the D. M is the number of samples in D. Data description length is calculated using relative frequency.

$$-M \sum_{i=1}^{n} (W(x_i, Pa(x_i))$$

$$W(x_i, Pa(x_i)) = \sum_{j=1}^{s_i} \sum_{k=1}^{q_i} R(x_i = s_j, Pa(x_i) = q_k) \log_2 \frac{R(x_i = s_j, Pa(x_i) = q_k)}{R(x_i = s_j) R(Pa(x_i) = q_k)}$$

r, m, a represent the number of reversed edges, removed edges, and added edges of H_O compared to H_p, respectively. The description length of source is based on the number of operations (add, deletion, and reverse).

$$(r + m + a) \log_2(n(n-1))$$

```
1: /* H_O : Original network */
2: /* D : new data */
3: /* POP: Population of Bayesian networks */
4: /* Initialization(): Initialization of Bayesian networks */
5: /* Refine( H_O , D , H_p ): return the MDL score */
6: /* array[]: 1-d array */
7:
8: Initialization(POP);
9:
10: For i=1 to MAX_GEN{
11:  For j=1 to |POP|{
12:    fitness(j)=Refine( H_O , D , H_j );}
13:  POP=Selection(POP);
14:  For j=1 to |POP|/2 { // crossover
       Parent(); // select two indexes
       array1=parent1; array2 = parent2;
       POP(children) = crossover(array1,array2);}
15:  For j=1 to |POP| // mutation
16:    POP(children)
         = Random(addition, deletion of edges);
17:  Repairing(); // break cycle
18:}
```

Figure 7: The pseudo code for evolutionary aggregation

The evolutionary algorithm uses connection matrix as a representation of H_p. The topological order of H_p could also be choice for the representation. Because the purpose of our research is to investigate the usefulness of evolutionary algorithm for the adaptation of Bayesian network, simple connection matrix is used without deep discussion about the comparison of two representations. The crossover is as

follows. The $n \times n$ matrix can be represented as a 1-dimensional array of n^2 members. Each parent is transformed to the 1-D array and the information is exchanged by 1-point crossover. The mutation is simple addition and deletion of edges. Repairing operator randomly deletes one of edges that form a cycle. Figure 7 shows the evolutionary approach for refinement.

IV. Experimental Results

A. Aggregation of Multiple Bayesian Networks

Randomly generated networks are used to test the performance of evolutionary approaches for combination (Table 1). The problem of fusion of 8 Bayesian networks (#1, #2, #3, #4, #5, #6, #7, and #8) is to find the best order that minimizes the number of edges of the final network. Table 2 shows the results of enumeration of all candidates. # of BN=2 means that it is the result of the combination of BN#1 and BN#2. Table 3 shows the percentage of the best solutions and the enumerative evaluation time. It is not feasible to do that for N>8.

Table 1. The randomly generated BNs

BN ID	# of variables	# of edges	BN ID	# of variables	# of edges
#1	38	229	#15	12	33
#2	31	96	#16	29	14
#3	33	182	#17	12	23
#4	5	4	#18	28	129
#5	33	22	#19	33	136
#6	14	6	#20	22	65
#7	27	114	#21	32	69
#8	22	103	#22	38	12
#9	33	128	#23	24	63
#10	34	56	#24	30	35
#11	20	87	#25	29	23
#12	13	11	#26	21	25
#13	28	128	#27	10	1
#14	8	13	#28	5	4

The parameters of evolutionary algorithms are as follows. The number of population size is 20, the number of maximum generation is 50, crossover rate is 0.8 and mutation rate is 0.01.

The evolutionary approach is compared to the various strategies. They are as follows.

(1) Heuristic #1: The order is based on the number of edges. If the BN has more edges than others, it will be combined later.
(2) Heuristic #2: It is also based on the number of edges. If the BN has more edges than others, it will be combined early.
(3) Greedy #1: This is a greedy strategy. It adds a BN that minimizes the number of edges of the final network.
(4) Greedy #2: It adds a BN that maximizes the number of edges of the final network.

Table 2. The result of enumeration of all candidates (the values indicate the number of edges)

# of BN	# of candidates	Average	Max	Min
2	2	609.0±135.7645	705	513
3	6	917.5± 61.7243	994	855
4	24	926.1± 53.6475	995	858
5	120	970.5± 62.6244	1056	880
6	720	982.6± 61.3339	1065	883
7	5040	1056.0± 39.3754	1119	984
8	40320	1093.4± 31.0356	1140	1032

Table 3. The best solutions and the enumerative evaluation time

# of BN	# of minimum solutions	% of minimum solutions	Time
2	1	50.0	-
3	2	33.3	-
4	6	25.0	-
5	18	15.0	-
6	90	12.5	-
7	5	0.1	1h 40m
8	96	0.23	22h

Table 4. The results of each strategy (the values indicate the number of edges)

# of BN	Heuristic #1	Heuristic #2	Greedy #1	Greedy #2	GA
2	513	705	513	705	513
3	855	989	855	904	855
4	858	990	990	905	858
5	880	1005	1041	932	880
6	883	1005	1046	933	883
7	1014	1093	1105	993	986
8	1055	1118	1126	1039	1034

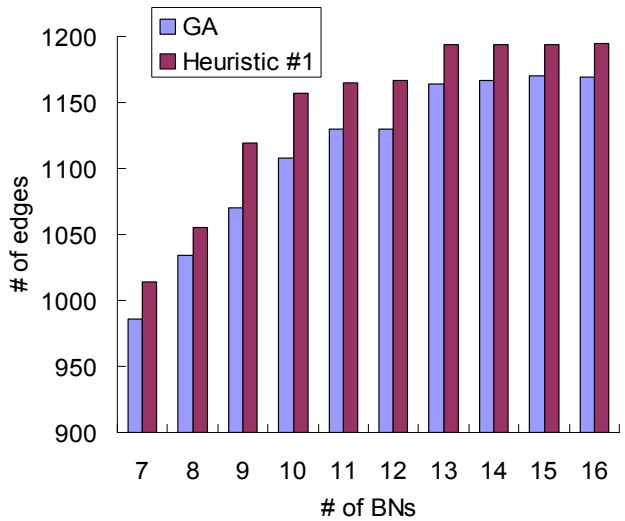

Figure 8: The results of comparison between Heuristic #1 and genetic algorithm

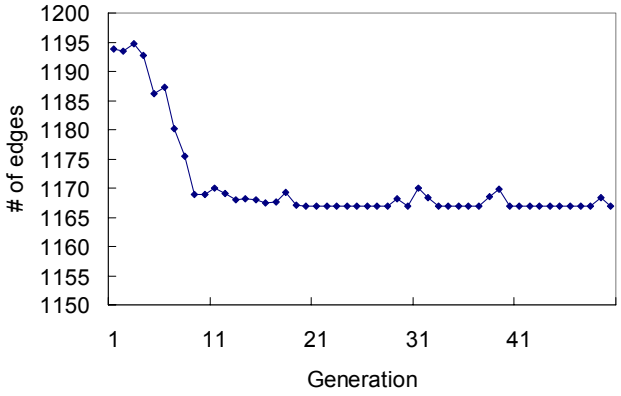

Figure 9: Average fitness change over generation (# of BN = 14)

Table 4 shows the number of edges of the final network. The results show that Heuristic #1 shows good performance compared to other methods. Heuristic #2 shows the worst performance. Greedy #1 shows relatively low performance compared to the Heuristic #1. Greedy #2 shows low performance when the number of Bayesian networks is small, but it shows improved performance when the number of Bayesian networks is increased. Figure 8 shows the comparison of performance between GA and Heuristic #1. It shows that the GA shows better performance than the Heuristic #1. Greedy #2 shows comparable results to the genetic algorithm but it shows low performance when the number of Bayesian networks is small. Figure 9 shows the change of average fitness. It converges in the early generation.

B. Refinement of Bayesian Networks

ASIA, one of representative benchmark networks, is a small Bayesian network that calculates the probability of a patient having tuberculosis, lung cancer or bronchitis respectively based on different factors, for example whether or not the patient has been to Asia recently (Figure 10). Shortness-of-breath (dyspnoea) may be due to tuberculosis, lung cancer, bronchitis, more than one of these diseases or none of them. A recent visit to Asia increases the risk of tuberculosis, while smoking is known to be a risk factor for both lung cancer and bronchitis. The results of a single chest X-ray do not discriminate between lung cancer and tuberculosis, as neither does the presence or absence of dyspnoea.

Figure 11 shows the ways of performance evaluation of the proposed genetic algorithm refinement. The ASIA network is the original network and 2000 data are sampled from the network using probabilistic logic sampling. After modifying the original network with some ratio, it will be recovered using new data and the likeness between the recovered one and original one is measured. Modification ratio δ means δ×100% edges are modified using addition, deletion and reverse operations (each δ/3×100%).

The number of new data and the rate of modification are changed. Figure 12 shows the performance comparison of genetic algorithm and greedy approaches. It shows that the genetic algorithm performs better than greedy algorithm. It is the average of 10 runs.

The increase of the new data makes the performance improvement but more than 1800 shows decrease of performance. It might be the result of biased optimization to the distribution of new data. The modification ratio degrades the performance as expected but evolutionary approach shows robust performance even for 0.4~0.8 ratio.

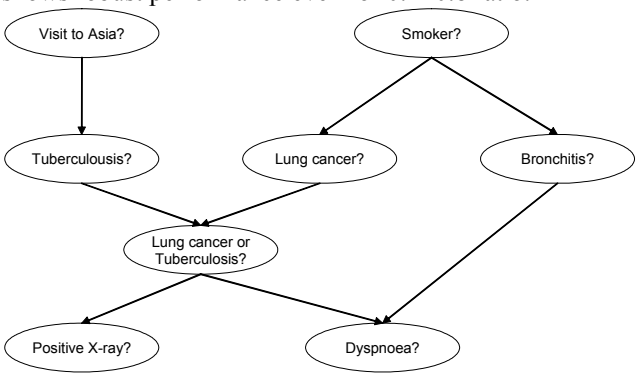

Figure 10: ASIA network

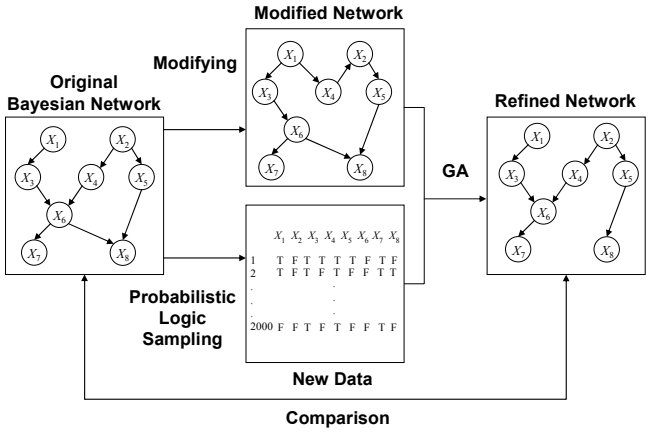

Figure 11: The procedure of comparison

V. CONCLUSIONS

In this paper, we have proposed two evolutionary approaches for adaptation of Bayesian networks. A preliminary test with a benchmark problem indicates that the genetic algorithm approach can provide more improved results than the heuristics or greedy search. Further investigation with a variety of realistic problems might strengthen the point of this paper.

ACKNOWLEDGEMENT

This research was supported by the MIC (Ministry of Information and Communication), Korea, under the ITRC (Information Technology Research Center) support program supervised by the IITA (Institute of Information Technology Assessment) (IITA-2005-(C1090-0501-0019))

REFERENCES

[1] P. Larranaga, C.M.H. Kuijpers, R. H. Murga, and Y. Yurramendi, "Learning Bayesian network structures by searching for the best ordering with genetic algorithm," *IEEE Transactions on Systems, Man and Cybernetics -Part A*, vol. 26, no. 4, pp. 487-493, 1996.

[2] P. Larranaga, M. Poza, Y. Yurramendi, R. H. Murga, and C.M.H. Kuijpers, "Structure learning of Bayesian networks by genetic algorithms: A performance analysis of control parameters," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 18, no. 9, pp. 912-926, 1996.

[3] I. Matzkevich, and B. Abramson, "The topological fusion of Bayes nets," *Proceedings of the 8th Annual Conference on Uncertainty in Artificial Intelligence*, pp. 191-198, 1992.

[4] I. Matzkevich, and B. Abramson, "Some complexity considerations in the combination of belief networks," *Proceedings of the 9th Conference on Uncertainty in Artificial Intelligence*, pp. 159-165, 1993.

[5] W. Lam, "Bayesian network refinement via machine learning approach," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 20, no. 3, pp. 240-251, 1998.

[6] J. Muruzabal, and C. Cotta, "A primer on the evolution of equivalence classes of Bayesian network structure," *The 8th International Conference on Parallel Problem Solving from Nature*, pp. 612-621, 2004.

[7] D. M. Chickering, "Learning equivalence classes of Bayesian network structures," *Journal of Machine Learning Research*, vol. 2, pp. 445-498, 2002.

[8] K.-J. Kim, J.-O. Yoo and S.-B. Cho, "Robust inference of Bayesian networks using speciated evolution," *Lecture Notes in Artificial Intelligence*, vol. 3488, pp. 92-101, 2005.

[9] R. Chen, K. Sivakumar, and H. Kargupta, "Learning Bayesian network structure from distributed data," *Proceedings of the 3rd SIAM International Data Mining Conference*, pp. 284-288, 2003.

[10] R. Chen, K. Sivakumar, and H. Kargupta, "Collective mining of Bayesian networks from distributed heterogeneous data," *Knowledge and Information Systems*, vol. 6, no. 2, pp. 164-187, 2004.

[11] J. D. Sagrado, and S. Moral, "Qualitative combination of Bayesian networks," *International Journal of Intelligent Systems*, vol. 18, no. 2, pp. 237-249, 2003.

[12] D. M. Pennock, and D. Wellman, "Graphical representations of consensus belief," *Proceedings of 15th Conference on University in Artificial Intelligence*, pp. 531-540, 1999.

[13] W. Lam and F. Bacchus, "Using new data to refine a Bayesian network," *Proceedings of the Uncertainty in Artificial Intelligence*, pp. 383-390, 1994.

[14] N. Friedman, and M. Goldszmidt, "Sequential update of Bayesian network structure," *Proceedings of 13th Conference on Uncertainty in Artificial Intelligence*, 1997.

[15] F. Tian, H. Zhong, Y. Lu and C. Shi, "Incremental learning of Bayesian networks with hidden variables," *1st IEEE International Conference on Data Mining*, pp. 651, 2001.

[16] M. L. Wong, W. Lam, and K. S. Leung, "Using evolutionary programming and minimum description length principle for data mining of Bayesian networks," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 21, no. 2, pp. 174-178, 1999.

[17] A. J. Novobilski, and F. Kamangar, "Bayesian learning with selective subsets of populations in genetic programming," *The conference on Smart Engineering System Design: Neural Networks, Fuzzy Logic, Evolutionary Programming, Complex Systems and Data Mining (ANNIE)*, 2002.

[18] A. J. Novobilski, "The random selection and manipulation of legally encoded Bayesian networks in genetic algorithms," *Proceedings of the International Conference on Artificial Intelligence*, pp. 438-443, 2003.

[19] R. Forster, and C. O. Wilke, "Tradeoff between short-term and long-term adaptation in a changing environment," *Physical Review E*, vol. 72, no. 4, art. no. 041922, 2005.

[20] L. M. De Campos, J. A. Gamez, and S. Moral, "Partial abductive inference in Bayesian belief networks – an evolutionary computation approach by using problem-specific genetic operators," *IEEE Transactions on Evolutionary Computation*, vol. 6, no. 2, pp. 105-131, 2002.

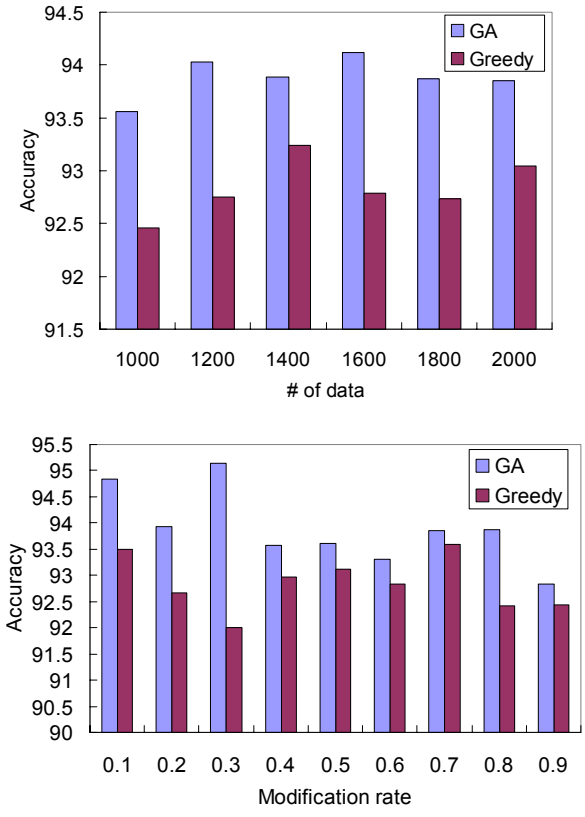

Figure 12: The performance comparison