A Probability-based Evolutionary Algorithm with Mutations to Learn Bayesian Networks

Sho Fukuda, Yuuma Yamanaka, and Takuya Yoshihiro

Wakayama University, Sakaedani, Wakayama, Japan

***Abstract* — Bayesian networks are regarded as one of the essential tools to analyze causal relationship between events from data. To learn the structure of highly-reliable Bayesian networks from data as quickly as possible is one of the important problems that several studies have been tried to achieve. In recent years, probability-based evolutionary algorithms have been proposed as a new efficient approach to learn Bayesian networks. In this paper, we target on one of the probability-based evolutionary algorithms called PBIL (Probability-Based Incremental Learning), and propose a new mutation operator. Through performance evaluation, we found that the proposed mutation operator has a good performance in learning Bayesian networks.**

***Keywords* — Bayesian Networks, PBIL, Evolutionary Algorithms**

I. Introduction

BAYESIAN network is a well-known probabilistic model that represents causal relationships among events, which has been applied to so many areas such as Bioinformatics, medical analyses, document classifications, information searches, decision support, etc. Recently, due to several useful tools to construct Bayesian networks, and also due to rapid growth of computer powers, Bayesian networks became regarded as one of the promising analytic tools that help detailed analyses of large data in variety of important study areas.

To learn a near-optimal Bayesian network structure from a set of target data, efficient optimization algorithm is required that searches an exponentially large solution space for near-optimal Bayesian network structure, as this problem was proved to be NP-hard [1]. To find better Bayesian network structures with less time, several efficient search algorithms have been proposed so far. Cooper et al., proposed a well-known deterministic algorithm called K2 [2] that searches for near-optimal solutions by applying a constraint of the order of events. As for the general cases without the order constraint, although several approaches have been proposed so far, many of which uses genetic algorithms (GAs), which find good Bayesian network structures within a reasonable time [3][4][5]. However, because recently we are facing on large data, more efficient algorithms to find better Bayesian network models are expected.

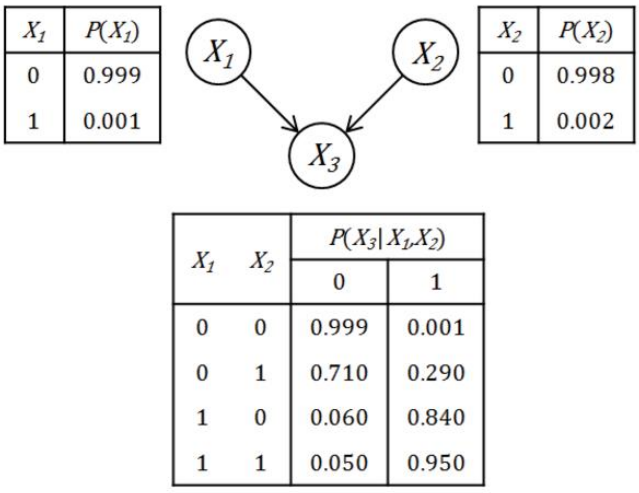
To meet this requirement, recently, a new category of algorithms so called EDA (Estimation of Distribution Algorithm) has been reported to provide better performance in learning Bayesian Networks. EDA is a kind of genetic algorithms that evolves statistic distributions to produce individuals over generations. There are several types of EDA such as UMDA (Uni-variate Marginal Distribution Algorithm) [12], PBIL (Population-Based Incremental Learning) [7], MIMIC (Mutual Information Maximization for Input Clustering) [13], etc. According to the result of Kim et al. [11], PBIL-based algorithm would be the most suitable for learning Bayesian networks.

The first PBIL-based algorithm for Bayesian networks was presented by Blanco et al. [9], which learns good Bayesian net- works within short time. However, because this algorithm does not include mutation, it easily falls into local minimum solution. To avoid converging at local minimum solutions, Handa et al. introduced a *bitwise mutation* into PBIL and showed that the mutation operator improved the quality of solutions in four-peaks problem, Fc4 function, and max-sat problem[10]. Although this operator was not applied to Bayesian networks, Kim et al. later proposed a new mutation operator transpose mutation specifically for Bayesian networks, and compares the performance of EDA-based Bayesian network learning with several mutation variations including bitwise mutation [11].

In this paper, we propose a new mutation operator called *probability mutation* for PBIL-based Bayesian Network learning. Through evaluation, we show that our new mutation operator is also efficient to find good Bayesian network structures.

The rest of this paper is organized as follows: In Section 2, we give the basic definitions on Bayesian networks and also describe related work in this area of study. In Section 3, we propose a new mutation operator called probability mutation to achieve better learning performance of Bayesian networks. In Section 4, we describe the evaluation results, and finally we conclude this paper in Section 5.

DOI: 10.9781/ijimai.2014.311



X_1	$P(X_1)$
0	0.999
1	0.001

X_2	$P(X_2)$
0	0.998
1	0.002

X_1	X_2	$P(X_3 \mid X_1, X_2)$	
		0	1
0	0	0.999	0.001
0	1	0.710	0.290
1	0	0.060	0.840
1	1	0.050	0.950

Fig. 1. A Bayesian Network Model

II. Learning Bayesian Networks

A. Bayesian Network Models

A Bayesian network model visualizes the causal relationship among events through graph representation. In a Bayesian network model, events are represented by nodes while causal relationships are represented by edges. See Figure 1 for example. Nodes X_1, X_2, and X_3 represent distinct events where they take 1 if the corresponding events occur, and take 0 if the events do not occur. Edges $X_1 \to X_3$ and $X_2 \to X_3$ represent causal relationships, which mean that the probability of $X_3 = 1$ depends on events X_1 and X_2. If edge $X_1 \to X_3$ exists, we call that X_1 is a parent of X_3 and X_3 is a child of X_1. Because nodes X_1 and X_2 do not have their parents, they have own prior probabilities $P(X_1)$ and $P(X_2)$. On the other hand, because node X_3 has two parents X_1 and X_2, it has a conditional probability $P(X_3 \mid X_1, X_2)$. In this example, the probability that X_3 occurs is 0.950 under the assumption that both X_1 and X_2 occur. Note that, from this model, Bayesian inference is possible: if X_3 is known, then the posterior probability of X_1 and X_2 can be determined, which enables us to infer events that causes the child event.

The Bayesian networks can be learned from the data obtained through the observation of events. Let $O = \{o_j\}, 1 \le j \le S$ be a set of observations, where S is the number of observations. Let $o_j = (x_{j1}, x_{j2}, ..., x_{jN})$ be a j-th observation, which is a set of observed values x_{ji} on event X_i for all $i (1 \le i \le N)$, where N is the number of events. We try to learn a good Bayesian network model θ from the given set of observations. Note that the model θ should be able to explain the observation O, i.e., O should be likely to be observed under θ. As an evaluation criterion to measure the level of fitting between θ and O, we use AIC (Akaike's Information Criterion) [6], which is one of the best known criterion used in Bayesian networks. Formally, the problem of learning Bayesian networks that we consider in this paper is defined as follows:

Problem 1: From the given set of observations O, compute a Bayesian network model θ that has the lowest AIC criterion value.

B. K2 Algorithm

K2 [2] is one of the best-used traditional algorithms to learn Bayesian network models. Note that searching good Bayesian network models is generally time consuming because the problem to learn Bayesian networks is NP-hard [1]. K2 avoids the problem of running time by limiting the search space through the constraint of totally order of events. Namely, for a given order of events $X_1 < X_2 < ... < X_N$, causal relationship $X_k \to X_l$, where $k > l$ is not allowed. Note that this constraint is suitable for some cases: if events have their time of occurrence, an event X_k that occurred later than X_l cannot be a cause of X_l. Several practical scenes would be the case.

The process of K2 algorithm applied to a set of events $X_1, X_2, ..., X_N$ with the constraint $X_1, X_2, ..., X_N$ is described as follows:

(1) Select the best structure using two events X_N and X_{N-1}. Here, the two structures, i.e., $X_{N-1} \to X_N$ and the independent case, can be the candidates, and the one with better criterion value is selected.

(2) Add X_{N-2} to the structure. Namely, select the best structure from every possible cases where X_{N-2} has edges connected to X_{N-1} and X_N. Namely, from the cases (i) $X_{N-2} \to X_{N-1}$ and $X_{N-2} \to X_N$, (ii) $X_{N-2} \to X_{N-1}$ only, (iii) $X_{N-2} \to X_N$ only, and (iv) where X_{N-2} has no edge.

(3) Repeat step (2) to add events to the structure in the order $X_{N-3}, ..., X_2, X_1$.

P		Parent Node					
		X_1	X_2	...	X_i	...	X_N
Child node	X_1	0.0	0.5	...	p_{i1}	...	0.5
	X_2	0.5	0.0	...	p_{i2}	...	0.5
	⋮	⋮	⋮	⋱	⋮	...	⋮
	X_j	p_{1j}	p_{2j}	...	p_{ij}	...	p_{Nj}
	⋮	⋮	⋮	⋮	⋮	⋱	⋮
	X_N	0.5	0.5	...	p_{iN}	...	0.0

Fig. 2. A Probability Vector

(4) Output the final structure composed of all events.Although K2 requires low computational time due to the constraint

of event order, many problems do not allow the constraint. In such cases, we require to tackle the NP-hard problem using a heuristic algorithm for approximate solutions.

C. Related Work for Un-ordered Bayesian Network Models

Even for the cases where the constraint of order is not allowed, several approaches to learn Bayesian network models has been proposed. One of the most basic method is to use K2 with random order, where randomly generated orders are applied repeatedly to K2 to search for good Bayesian network models.

As more sophisticated approaches, several ideas have been proposed so far. Hsu, et al. proposed a method to use K2 algorithm to which the orders evolved by genetic algorithms are applied [3]. Barrière, et al. proposed an algorithm to evolve Bayesian network models based on a variation of genetic algorithms called co-evolving processes [4]. Tonda, et al. proposed another variation of genetic algorithms that applies a graph-based evolution process [5]. However, with these approaches, the performance seems to be limited, and a new paradigm of the algorithm that learn Bayesian networks more efficiently is strongly required.

D. Population-Based Incremental Learning

Recently, a category of the evolutionary algorithms called EDA (Estimation Distribution Algorithm) appears and reported to be efficient to learn Bayesian network models. As one of EDAs, PBIL [7] is proposed by Baluja et al. in 1994, which is based on genetic algorithm, but is designed to evolve a probability vector. Later, Blanco et al. applied PBIL to the Bayesian network learning, and showed that PBIL efficiently works in this problem [9].

In PBIL, an individual creature s is defined as a vector $s=(v_1,v_2,...,v_L)$, where $v_i\,(1\le s\le L)$ is the i-th element that takes a value 0 or 1, and L is the number of elements that consist of an individual. Let $P=(p_1,p_2,...,p_L)$ be a probability vector where $p_i\,(1\le i\le L)$ represents the probability to be $v_i=1$. Then, the algorithm of PBIL is described as follows:

(1) As initialization, we let $p_i=0.5$ for all $i=1,2,...,L$.
(2) Generate a set S that consists of C individuals according to P. Namely, element v_i of each individual is determined according to the corresponding probability p_i.
(3) Compute the evaluation value for each individual $s\in S$.
(4) Select a set of individuals S' whose members have evaluation values within top C′ in S , and update the probability vector according to the following formula:

$$p_i^{new}=ratio(i)\times\alpha+p_i\times(1.0-\alpha) \qquad (1)$$

where p_i^{new} is the updated value of the new probability vector P^{new} (P is soon replaced with P^{new}), $ratio(i)$ is the function that represents the ratio of individuals in S' that include link i (i.e., $v_i=1$), and α is the parameter called learning ratio.

(5) Repeat steps (2)-(4).

By merging top-C′ individuals, PBIL evolves the probability vector such that the good individuals are more likely to be generated. Different from other genetic algorithms, PBIL does not include "crossover" between individuals. Instead, it evolves the probability vector as a "parent" of the generated individuals.

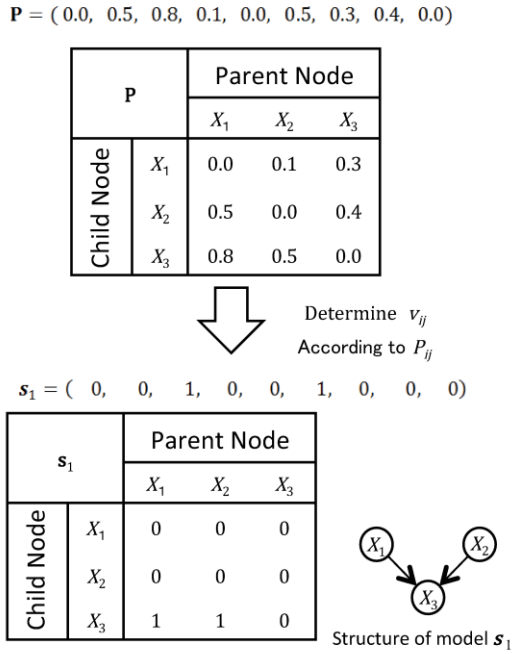


Fig. 3. Step (2): Generating Individuals

III. PBIL-BASED BAYESIAN NETWORK LEARNING

In this section, we present a PBIL-based algorithm to learn Bayesian network models to which we apply a new mutation operator. Since our problem (i.e., Problem 1) to learn Bayesian networks is a little different from the general description of PBIL shown in the previous section, a little adjustment is required.

In our algorithm, individual creatures correspond to each Bayesian network model. Namely, with the number of events N, an individual model is represented as $s=(v_{11},v_{12},...,v_{1N},v_{21},v_{22},...,v_{N1},v_{N2},...,v_{NN})$, where v_{ij} corresponds to the edge from events X_i to X_j, i.e., if $v_{ij}=1$ the edge from X_i to X_j exists in s, and if $v_{ij}=0$ it does not exist. Similarly, we have the probability vector P to generate individual models as $P=(p_{11},p_{12},...,p_{1N},p_{21},p_{22},...,p_{N1},\ p_{N2},...,p_{NN})$ where

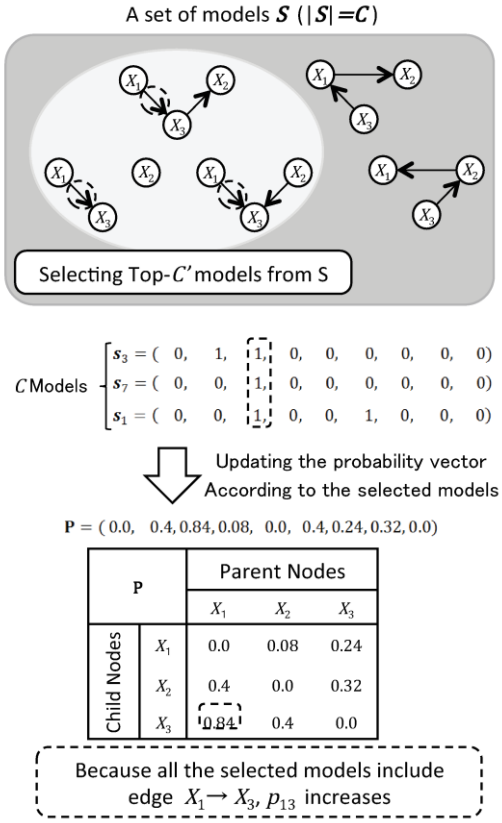


Fig. 4. Step (3)(4): Updating the Probability Vector

p_{ij} is the probability that the edge from X_i to X_j exists. A probability vector can be regarded as a table as illustrated in Fig. 2. Note that, because Bayesian networks do not allow self-edges, p_{ij} is always 0 if $i = j$.

The process of the proposed algorithm is basically obtained from the steps of PBIL. Namely, the basic steps are described as follows:

(1) Initialize the probability vector P as $p_{ij} = 0$ if $i = j$ and $p_{ij} = 0.5$ otherwise.

(2) Generate S as a set of C individual models according to P. (This step is illustrated in Fig. 3.)

(3) Compute values of the evaluation criterion for all individual models $s \in S$.

(4) Select a set of individuals S' whose members have top-C' evaluation values in S, and update the probability vector according to the formula (1). (These steps (3) and (4) are illustrated in Fig. 4.)

(5) Repeat steps (2)-(4).

Same as PBIL, the proposed algorithm evolves the probability vector to be likely to generate better individual models. However, there is a point specific to Bayesian networks, that is, a Bayesian network model is not allowed to have cycles in it. To consider this point in our algorithm, step 2 is detailed as follows:

(2a) Create a random order of pairs (i, j), where $1 \le i, j \le N$ and $i \ne j$.

(2b) Determine the values of v_{ij} according to P, with the ordercreated in step (2a); every time v_{ij} is determined, if v_{ij} is determined as 1, we check whether this edge from X_i to X_j creates a cycle with all the edges determined to exist so far. If it creates a cycle, let v_{ij} be 0.

(2c) Repeat steps (2a) and (2b) until all pairs (i, j) in the order are processed.These steps enable us to treat the problem of learning good Bayesian network models within the framework of PBIL. Note that checking the cycle creation in step (2b) can be done efficiently using a simple table that manages the taboo edges that create cycles when they are added to the model.

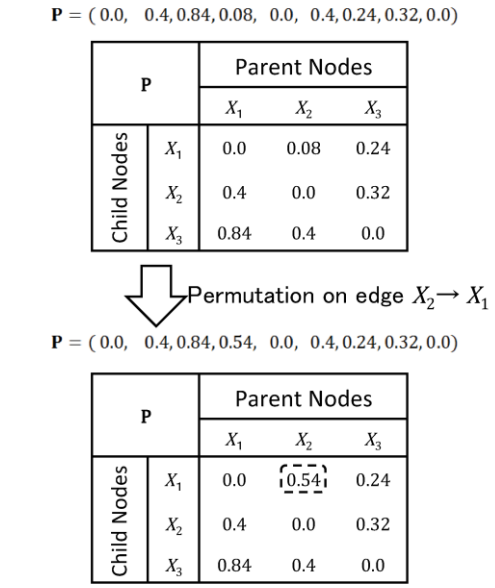


Fig. 5. Probability Mutation (PM)

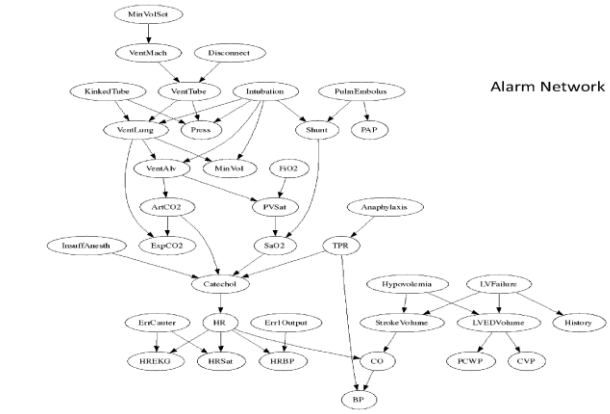


Fig. 6. The Alarm Network

A. Mutation Operators

Note that the algorithm introduced in the previous section does not include mutation operator. Thus, naturally, it is easy to converge to a local minimum solution. Actually, PBIL-based algorithm to learn Bayesian networks proposed by

Blanco et al. [9] stops when the solution converges to a minimal solution, i.e., when score does not improve for recent k generations. However, local minimum solutions prevent us to search for better solutions, thus it should be avoided.

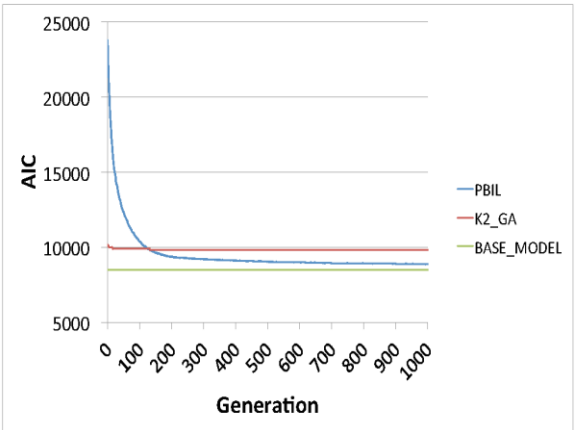


Fig. 7. Performance of the PBIL-based Algorithm

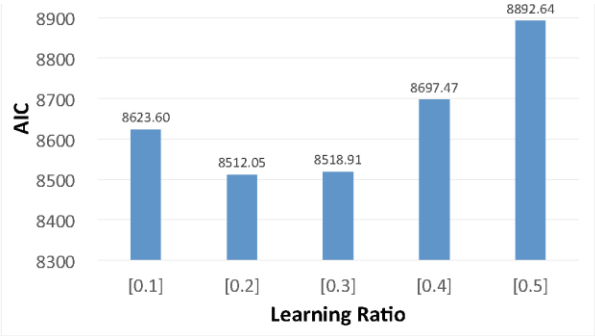


Fig. 8. AIC Scores under Variation of Learning Ratio

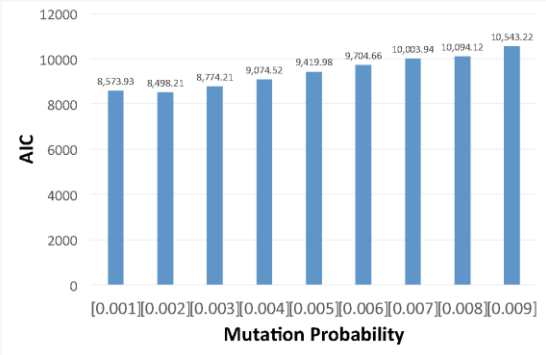


Fig. 9. AIC Scores under Variation of Mutation Probability

To avoid converging to the local minimum solution and to improve the performance of the algorithm, typically several mutation operations are inserted between steps (2) and (3). The most popular mutation operator is called *bitwise mutation* (BM) introduced by Handa [10], which apply mutations to each link in each individual, as described in the following step:

BM: For each individual in S generated in step (2), we flip each edge with probability p_{mut}. Namely, for each pair of nodes i and $j(1 \le i, j \le N)$, $v_{ij} \leftarrow 1$ if $v_{ij} = 0$, and $v_{ij} \leftarrow 0$ otherwise, with probability p_{mut}.

The other mutation operator we try in this paper is called *transpose mutation* (TM) introduced by [11]. This operation is proposed based on the observation that that reverse-edges frequently appear in the solutions. To avoid this, transpose mutation changes the direction of edges in the individuals produced in each generation. The specific operation inserted between steps (2) and (3) is in the following.

TM: For each individual in S generated in step (2), with probability p_{mut}, we do the following operation: we reverse all edges in the individual with probability p_{mut}, namely, $v_{ij} \leftarrow v_{ji}$ for all i and j.

In contrast to these conventional mutations shown above, our new mutation operator called *probability mutation* (PM) does not manipulate individuals produced in each generations. Instead, we manipulate the probability vector P to generate better individuals in the next generation, which is inserted between steps (4) and (5). The specific operation of this mutation is shown and in the following (See also Fig. 5):

PM: Apply mutations on the new probability vector P : For all pairs of events (X_i, X_j), $i \neq j$, we apply the following formula with probability p_{mut}, where the function $rand()$ generates a random value from range [0,1].

$$p_i^{new} = rand() \cdot b + p_i \cdot (1-b) \qquad (2)$$

IV. Evaluation

A. Methods

In order to reveal the effectiveness of PBIL-based algorithms, we first evaluate the PBIL-based algorithm with probability mutation in comparison with K2 with its constraint (i.e., the order of events) evolved with genetic algorithms, which is a representative method among traditional approaches to learn Bayesian networks. In this conventional algorithm, we repeat creating Bayesian network models, in which its constraints (i.e., order of nodes) are continuously evolved with a typical genetic algorithm over generations, and output the best score among those computed ever. The results are described in Sec. IV-B. We next compare the performance of three mutation operators BM, TM, and PM applied to the PBIL-based algorithm. With this evaluation, we show that the new mutation operator PM proposed in this paper has good performance. The results are described in Sec. IV-C. In our experiment, we use Alarm Network [8] shown in Fig. 6, which is a Bayesian network model frequently used as a benchmark problem in this area of study. We create a set of 1000 observations according to the structure and the conditional probability of Alarm Network, and then learn Bayesian network models from the observations using those two algorithms. As the evaluation criterion, we use AIC, one of the representative criterion in this area. Namely, we compare the AIC values in order to evaluate how good is the Bayesian

network models obtained by these two algorithms. As for parameters, we use $C = 1000$, $C' = 1$, $\alpha = 0.2$, $\beta = 0.5$, and $p_{mut} = 0.002$.

B. Result 1: Performance of PBIL-based Algorithms

The first result is shown in Fig. 7, which indicates the AIC score of the best Bayesian network model found with the growth of generations. In this figure, the AIC score of the original Alarm Network, which is the optimal score, is denoted by "BASE MODEL." The proposed algorithm with *probability mutation* (represented as PBIL in the figure) converges to the optimal score as time passes, whereas K2-GA stops improving in the early stage. We can conclude that the performance of the PBIL-based algorithm is better than the conventional algorithm in that the PBIL-based algorithm computes better Bayesian network models according to time taken in execution. Note that the running time per generation in the proposed method is far shorter than K2-GA; the difference is more than 250 times in our implementation.

Fig. 8 and 9 show the performance of the proposed algorithm with variation of learning ratio α and mutation probability p_{mut} in 10,000 generations. These results show that the performance of the proposed method depends on α and p_{mut}, which indicates that we should care for these values

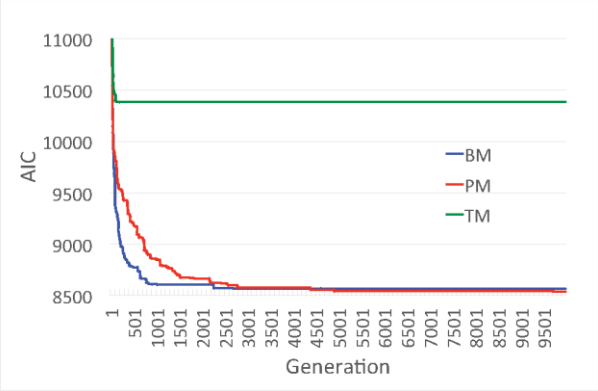


Fig. 10. Performance Comparison in Mutation Variations

to improve the performance of the proposed algorithm. Note that, from these results, we have the best-performance values $\alpha = 0.2$ and $p_{mut} = 0.002$, which are used as the default values in our experiment.

C. Result 2: Comparison of Mutation Variations

We further compared the performance of the PBIL-based algorithm with three mutations, bitwise mutation (BM), transpose mutation (TM), and probability mutation (PM). Facing on this experiment, we carefully choose the mutation probability of each method through preliminary experiments. For BM, we examined the performance of the mutation probability in range [0.001:0.2], and chose the value of the best performance, 0.005. For TM, we similarly tried the performance of the mutation probability in range [0.05:0.5], and chose 0.1 as the best value. For PM, from the result shown in Fig. 9, we chose the mutation probability 0.002, which is the same value as our first result shown in Fig. 7.

The result is shown in Fig. 10. We see that BM and PM continue improving as generation passes, whereas TM stops improving at the early stage of generation. Also, we see that the curve of BM and PM are slightly different where BM reach better scores in the early stage while PM outperforms BM in the late stage. This result shows that the newly proposed mutation operator PM is also useful especially in long-term learning of Bayesian network models under PBIL-based algorithms.

V. CONCLUSION

In this paper, we introduced the literature of PBIL-based learning of Bayesian network models, and proposed a new mutation operator called probability mutation that manipulates probability vector of PBIL. Through evaluation of these algorithms, we found that (i) the PBIL-based algorithm outperforms K2-based traditional algorithms with the long-term continuous improvement, and (ii) probability mutation works well under PBIL-based algorithms especially in long-term computation to obtain high-quality Bayesian network models. Designing more efficient search algorithms based on EDA is one of the most attractive future tasks.

REFERENCES

[1] D.M. Chickering, D. Heckerman, C. Meek, "Large-Sample Learning of Bayesian Networks is NP-Hard," Journal of Machine Learning Research, Vol.5, pp.1287—1330, 2004.

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

[3] W.H. Hsu, H. Guo, B.B. Perry, and J.A. Stilson, A permutation genetic algorithm for variable ordering in learning Bayesian networks from data In Proceedings of the Genetic and Evolutionary Computation Conference (GECCO), 2002.

[4] O.Barrière, E. Lutton, P.H. Wuillemin, "Bayesian Network Structure Learning using Cooperative Coevolution," In Proceedings of the Genetic and Evolutionary Computation Conference (GECCO), pp.755-762, 2009.

[5] A.P Tonda, E. Lutton, R. Reuillon, G. Squillero, and P.H. Wuillemin, Bayesian network structure learning from limited datasets through graph evolution, In Proceedings of the 15th European conference on Genetic Programming (EuroGP'12), pp.254-265, 2012.

[6] Akaike, H., "Information theory and an extension of the maximum likelihood principle", Proceedings of the 2nd International Symposium on Information Theory, pp.267-281 (1973).

[7] Shumeet Baluja. Population-based incremental learning: A method for integrating genetic search based function optimization and competitive learning. Technical Report No. CMU-CS-94-163, Carf Michigan, Ann Arbor, 1994.

[8] Beinlich, I,A., Suermondt, H.J., Chavez R.M., Cooper G.F. The ALARM monitoring System: A Case Study with Two Probabilistic Inference Techniques for Belief Networks" In: Second European Conference on Artificial Intelligence in Medicine. Volume 38., London, Great Britain, Springer-Verlag, Berlin 247—256, 1989.

[9] R. Blanco, I. Inza, P. Larrañaga, "Learning Bayesian Networks in the Space of Structures by Estimation of Distribution Algorithms," International Journal of Intelligent Systems, Vol.18, pp.205–220, 2003.

[10] H. Handa, "Estimation of Distribution Algorithms with Mutation," Lecture Notes in Computer Science, Vol.3448, pp.112-121, 2005.

[11] D.W. Kim, S. Ko, and B.Y. Kang, "Structure Learning of Bayesian Networks by Estimation of Distribution Algorithms with Transpose Mutation," Journal of Applied Research and Technology, Vol.11, pp.586– 596, 2013.
[12] H. Muhlenbein, "The Equation for Response to Selection and Its Use for Prediction," Evolutionary Computation, Vol.5, No.3, pp. 303–346, 1997.
[13] J.S. De Bonet et al., "MIMIC: Finding Optima by Estimating Probability Densities," Advances in Neural Information Processing Systems, Vol.9, pp.424–430, 1997.

Sho Fukuda received his B.E. and M.E. degrees from Wakayama University in 2012 and 2014, respectively. He is currently working with Intec Hankyu Hanshin Co.Ltd. He is interested in Data Analytics with large data sets.

Yuuma Yamanaka is currently pursuing his Bachelor's degree in Faculty of Systems Engineering, Wakayama University. He is interested in Data Analytics and Machine Learning.

Takuya Yoshihiro received his B.E., M.I. and Ph.D. degrees from Kyoto University in 1998, 2000 and 2003, respectively. He was an assistant professor in Wakayama University from 2003 to 2009. He has been an associate professor in Wakayama University from 2009. He is currently interested in the graph theory, distributed algorithms, computer networks, wireless networks, medical applications, bioinformatics, etc. He is a member of IEEE, IEICE, and IPSJ.