The Sixteenth International Symposium on Artificial Life and Robotics 2011 (AROB 16th '11),
B-Con Plaza, Beppu,Oita, Japan, January 27-29, 2011

A Learning Method for Dynamic Bayesian Network Structures Using a Multi-Objective Particle Swarm Optimizer

Kousuke Shibata, Hidehiro nakano, Arata,Miyauti

Tokyo City University
1-28-1, Tamadutsumi, Setagay–ku, Tokyo, 158-8557 Japan
(Shibata@ic.cs.tcu.ac.jp)

***Abstract*:** In this paper, we present a multi-objective Discrete Particle Optimizer (DPSO) for the learning of Dynamic Bayesian Network (DBN) structures. The proposed method introduces a hierarchical structure consisting of DPSOs and a Multi-Objective Genetic Algorithm (MOGA). Groups of DPSOs find effective DBN sub-network structures and a group of MOGA finds whole of the DBN network structure. Through numerical simulations, the proposed method can find more effective DBN structures and can obtain them faster than the conventional method.

Keywords: Dynamic Bayesian Networks, Structure Learning, Multi-Objective Optimization, Discrete Particle Swarm Optimization

I. INTRODUCTION

The Dynamic Bayesian Network (DBN) describes causal relations in various systems by using stochastic network structures, and represents them as directed acyclic graphs and conditional probabilities for transitions of each observed state variable [1]-[3]. The DBN is an extended model of the Bayesian Network (BN). As compared with the BN, the DBN can describe temporal causal relations of state variables. The DBN has been applied to speech recognition, genetic networks and so on. When the structural topology of a DBN is unknown, the learning of the DBN structure is needed [2][3]. In the learning, two trade-off characteristics must be considered. One is a characteristic which indicates how well the network fits the observed data. Another is structural complexity of the network. A fully-connected network can represent all relationships between each state variable. However, such a redundant network cannot adapt target model well, and the network size becomes extremely large. Therefore, it is preferred that networks should be constructed as simple as possible. That is, the degree of connectivity in the networks must be controlled in learning methods. Many learning methods introduce criteria which evaluate networks in terms of both probabilistic likelihood and structural complexity. However, they generally use a single evaluation function to combine these factors by a weight parameter. Since appropriate weight parameters depend on target models, it is difficult to determine a unique parameter value. The learning methods based on Multi-Objective GA (MOGA) [4] or Immune Algorithm (IA) can overcome this problem. However, in these algorithms, the learning for large scale DBN structure requires significantly long computation time.

In this paper, we present a multi-objective Discrete Particle Optimizer (DPSO) for the learning of DBN structures. The proposed method introduces a hierarchical structure. First, the objective DBN is divided into plural sub-networks depending on observed state variables. The lower layer consists of groups of DPSO. The DPSO is well-known as one of the fast solvers for various optimization problems [5]. In the proposed method, each particle has binary states corresponding to causal relations between state variables in each sub-network. Each group of particles in the lower layer finds each effective sub-network structure. The higher layer consists of MOGA [4]. A group of individuals in the higher layer finds the whole of the DBN structure sharing information from the lower layer by migration. Evaluation values of each particle or individual are given as Pareto solutions for likelihood and complexity to obtained DBN structures. The hierarchical structure in the proposed method can reduce computation time for the learning of larger scale DBN structures. We evaluate both likelihood and complexity to obtained DBN structures, and compare with the conventional learning method based on MOGA [3]. Through numerical simulations, the proposed method can find more effective DBN structures and can obtain them faster than the conventional method. Generally, there exist trade-off relations between likelihood and complexity to DBN structures in various actual applications. The proposed method can also provide many candidate structures of DBNs.

II. BACK GROUND

A Dynamic Bayesian Network (DBN) is a kind of probabilistic networks which represent temporal relationships between observed state variables. A DBN has three elements represented by (i) Node: observed state variables, (ii) Edge: dependencies between each node, (iii) Conditional Probability Table (CPT): degree of dependencies. An end point node and an origin node of an edge are a child node and a parent node, respectively. Let $X_1[t],\cdots,X_N[t]$ be N discrete state variables at time t. A DBN consists of (i) a prior network DBN_0 that specifies prior probabilities $\Pr(X[0])$ and (ii) a transition network DBN_T that specifies transition probabilities $\Pr(X[t+1]|[X[t])$. The Joint probabilities over all the variables for time $t=0,1,2,\cdots T$ is

$$DBN_0(X[0])\cdot\prod_{t=0}^{T-1}DBN_T(X[t+1]|X[t]) \quad (1)$$

For simplicity, this paper considers on the transition network only.

If a DBN structure is unknown, some learning methods by using search algorithms and evaluation criteria are necessary. Evolutionary computation and Bayesian Information Criterion (BIC) [3] have been widely used to the learning of DBN structures. BIC has two terms that indicate likelihood and complexity of networks. BIC is represented by the following formula.

$$BIC = Likelihood - \omega\cdot Complexity \quad (2)$$

where ω is the weight parameter which balances between Likelihood and Complexity. The Likelihood term signifies the plausibility of the network. It is calculated by counting occurrences within sequences. The likelihood is then defined as:

$$Likelihood=\sum_i\sum_{j_i}\sum_{k_i}N_{i,j_i,k_i}\cdot\log\left(\frac{N_{i,j_i,k_i}}{\sum_{k_i}N_{i,j_i,k_i}}\right) \quad (3)$$

where N_{i,j_i,k_i} denotes the number of occurrences in observation data sequences such that a child node i has a value k_i and its parent node has a value j_i. The Complexity term signifies the structural complexity of the network which is defined as:

$$Complexity=\sum_i|parent(X_i)| \quad (4)$$

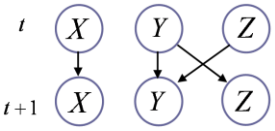
where $parent(\cdot)$ is the number of parent nodes.

In case of using BIC, we must determine an optimum weight parameter between two terms. If setting of this parameter is improper, the resulting graph is too complex or sparse. However, it is difficult to determine a unique optimum weight beforehand, because it depends on target models and training data sets. To solve this problem, the learning method by using a Multi-Objective Genetic Algorithm (MOGA) has been proposed [3]. The method applies a Pareto ranking scheme to the MOGA [4]. In the Pareto ranking scheme, each vector is evaluated by the number of the other vectors which have better values about all objectives. Using this method, a variety of solutions about likelihood and complexity can be obtained.

However, multi-objective methods tend to need much iteration to convergence, compared with single-objective methods. Since these methods acquire solutions for a variety of likelihood and complexity, the solutions are dispersed in search space. Typically, evaluation of the network requires enormous computation time which is proportional to complexity and amount of data. Therefore, it is desirable to reduce computation time to the learning convergence.

III. PROPOSED METHOD

In this paper, we use Discrete Particle Swarm Optimizers (DPSOs) [5] as a search algorithm to reduce computation cost to structure learning of DBNs. The DPSO is an optimization method that is a kind of swarm intelligence. In DPSO, particles efficiently search solutions in target problems, by updating their positions and velocities based on personal best solutions which each particle has and a global best solution which all the particles have. The DPSO is known as simple and fast algorithm. In the proposed method, a structure of a DBN is represented by binary variables in the DPSO. Particles have binary variables which denote existence of connections in the DBN. If a connection exists between a child node and a parent node, the value of the binary variable is 1. Otherwise, it is 0 (see Fig 1). If N variables can be observed, $N\times N$ strings are required.



	X[t]	Y[t]	Z[t]
X[t+1]	1	0	0
Y[t+1]	0	1	1
Z[t+1]	0	1	0

Fig.1 A DBN structure and binary variables in DPSO.

In order to solve multi-objective problems by the DPSO, some schemes are needed. In this paper, we use the existing method. We apply a multi-objective optimization scheme by using the Archive scheme [6]. The Archive has two features that are the Archive controller and the Grid. The Archive controller determines whether to save solutions into Pareto solutions. If the Archive has no solution with better evaluation values about all objectives compared with a selected solution, the Archive controller saves the selected solution into the Archive. The Grid produces well-distributed Pareto front. The function-space is divided into grids. If particles are dense in a grid, one of solutions in the grid is eliminated. With these features, a variety of solutions can be obtained.

In addition, we apply a hierarchical structure to DPSOs in order to reduce computation cost. In formulas (3) and (4), BIC can be divided into computations for each child node i. This means that the learning of DBN structures can be split for each child node. In this paper, we propose a learning method by two layers. In the lower layer, each group of Multi-objective DPSOs finds each sub-network structure divided for each child node. That is, in the lower layer, the number of groups corresponds to the number of child nodes. In the higher layer, a group of MOGA finds the whole of the DBN structure. For every iteration, the lower and higher layers exchange respective solutions by migration. The overview of this hierarchical structure is shown in Fig 2.

The proposed algorithm is described by the following steps:

(step1) In the each layer, initialize particles and individuals.

(step2) Evaluate particles and individuals.

(step3) Update the positions of each particle in the lower layer by equation (5).

$$if(rand() < sig(v_{id}))\ then\ x_{id} = 1 \qquad else\ x_{id} = 0$$
$$sig(v_{id}) = \frac{1}{1+e^{-v_{di}}} \tag{5}$$

(step4) Update the each Archive.

(step5) Update the velocities of each particle in the lower layer by equation (6).

$$v_{id}^{k+1} = w \cdot v_{id}^{k} + c_1 \cdot r_1 \cdot (P_{id} - x_{id}^{k}) + c_2 \cdot r_2 \cdot (G_d - x_{id}^{k}) \tag{6}$$

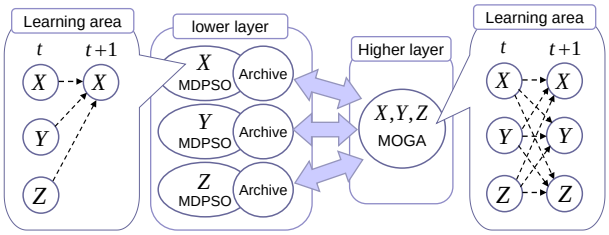


Fig.2 Overview of a hierarchical learning method.

(step6) Manipulate individuals in the higher layer.

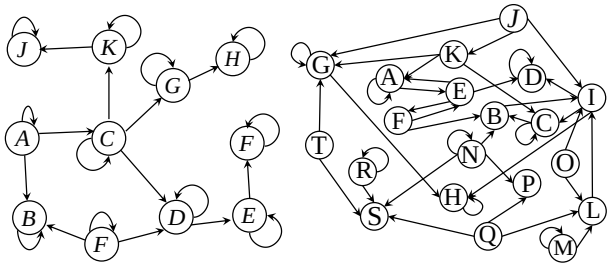
(step7) Migrate between each layer.

(step8) Repeat from (step2) to (step7).

In migration steps, a particle which is chosen randomly overwrites to a part of individual whose rank is not 1.

VI. SIMULATION RESULT

The proposed method is applied to several benchmarks which have a variety of complexities and performances. In order to compare the proposed method with the conventional method, we use some benchmarks which were tested by the conventional method. Target networks (a) and (b) are shown in Fig 3. Table 1 shows training data sets. These nodes in the networks generate discrete probabilistic values according to dependencies. If nodes have no parents, they generate random values. Parameters used for the simulation are shown in Table 2.



(*a*) (*b*)

Fig.3 Target networks.

Table 1 Training data sets.

Network	Variables	Links	Examples	Parent Links
(a)	10	20	100	0-3
(b)	20	40	200	0-4

Table 2 Simulation parameters.

Parameter	Conventional Method	Proposed Method
iteration	50	50
population	200,500	100,250
particles	-	100,250
GA operation	tournament	tournament

Figs. 4 and 5 show the simulation results for the networks (a) and (b). It can be found that the proposed method obtains a variety of solutions which have better values with less iteration than the conventional method. A structure which has high complexity and good likelihood becomes complex and has extra edges. On the other hand, a structure which has low complexity and bad likelihood has many missing edges. Generally, there exist trade-off relations between likelihood and complexity to DBN structures in various actual applications. The proposed method can also provide many candidate structures of DBNs.

V. CONCLUSION

In this paper, we have proposed a hierarchical multi-objective DPSO for the structure learning of DBNs. Dividing computations, the proposed method requires less iteration to learning convergence than the conventional. In addition, diversity and accuracy of solutions are equal or higher than the conventional method. Since the hierarchical structure can reduce the computation cost for the learning, the proposed method is effective in high-dimensional problems.

Future problems include (1) application to other benchmarks, (2) evaluation for learning speed, and (3) consideration of appropriate group size of particles or individuals.

REFERRENCES

[1] K.Murphy, "Dynamic Bayesian Networks: Representation, Inference and Learning," PhD thesis, Computer Science Division, UC Berkeley, 2002
[2] W.Guo, X.Gao & Q.Xiao, "Bayesian Optimization Algorithm for Learning Structure of Dynamic Bayesian Networks from Incomplete Data," Proc. CCDC 2008, pp. 2088-2093, 2008.
[3] B.J.Ross & E.Zuvria, "Evolving dynamic Bayesian networks with Multi-objective genetic algorithms," Applied Intelligence, vol. 26, no. 1, pp. 13-23, 2007.
[4] R.E.Steuer, *Multiple Criteria Optimization: Theory, Computations, and Application*. New York: John Wiley & Sons, Inc., ISBN 047188846X, 1986.
[5] J.Kennedy & R.Eberhart, "A discrete binary version of the particle swarm optimization algorithm," Proc. SMC '97, pp.4104-4109, 1997.
[6] C.A.C.Coello, G.T.Pulido & M.S.Lechuga, "Handling Multiple Objectives with Particle Swarm Optimization," IEEE Transactions on Evolutionary Computation, vol. 8, no. 3, pp. 256-279, 2004.

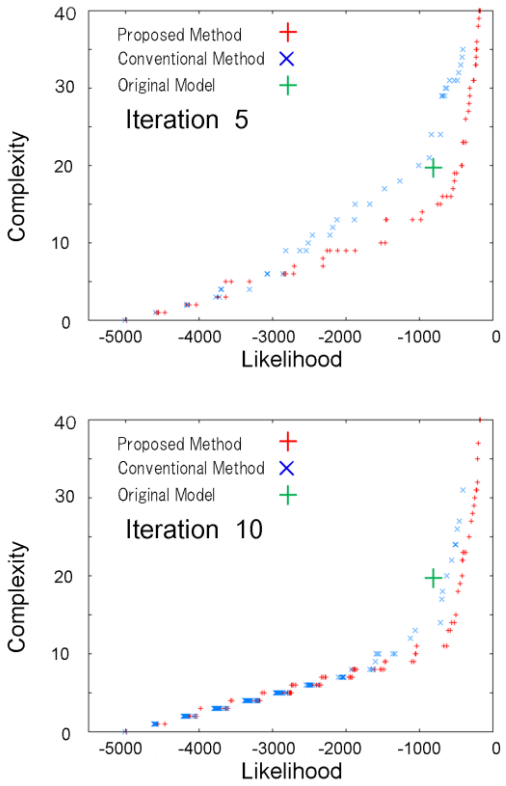


Fig.4 Pareto solutions for the network (a). Upper: iteration=5. Lower: iteration=10.

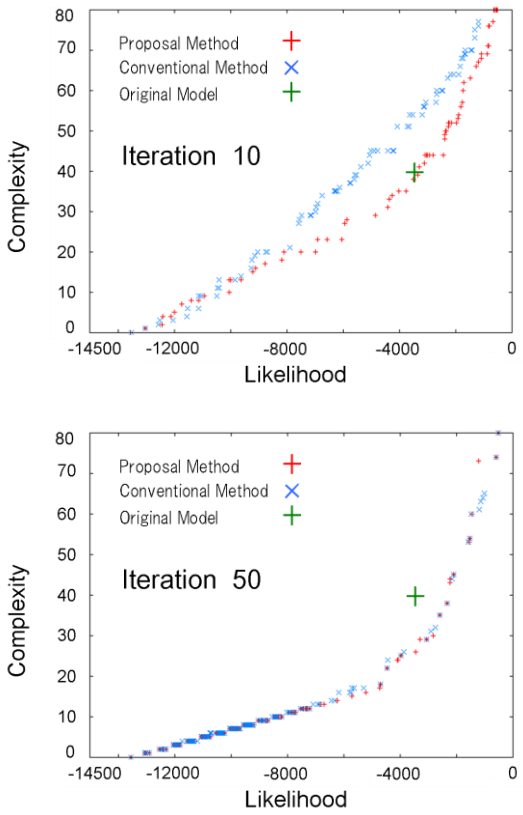


Fig.5 Pareto solutions for the network (b). Upper: iteration=10. Lower: iteration=50.