GMR Genetics and Molecular Research
Online Journal
ISSN:1676-5680

Bayesian network structure learning based on the chaotic particle swarm optimization algorithm

Q. Zhang, Z. Li, C.J. Zhou and X.P. Wei

Key Laboratory of Advanced Design and Intelligent Computing,
Dalian University, Ministry of Education, Dalian, China

Corresponding author: Q. Zhang
E-mail: zhangq@dlu.edu.cn

Genet. Mol. Res. 12 (4): 4468-4479 (2013)
Received February 2, 2013
Accepted August 15, 2013
Published October 10, 2013
DOI http://dx.doi.org/10.4238/2013.October.10.12

ABSTRACT. The Bayesian network (BN) is a knowledge representation form, which has been proven to be valuable in the gene regulatory network reconstruction because of its capability of capturing causal relationships between genes. Learning BN structures from a database is a nondeterministic polynomial time (NP)-hard problem that remains one of the most exciting challenges in machine learning. Several heuristic searching techniques have been used to find better network structures. Among these algorithms, the classical K2 algorithm is the most successful. Nonetheless, the performance of the K2 algorithm is greatly affected by a prior ordering of input nodes. The proposed method in this paper is based on the chaotic particle swarm optimization (CPSO) and the K2 algorithm. Because the PSO algorithm completely entraps the local minimum in later evolutions, we combined the PSO algorithm with the chaos theory, which has the properties of ergodicity, randomness, and regularity. Experimental results show that the proposed method can improve the convergence rate of particles and identify networks more efficiently and accurately.

Key words: Bayesian network structure learning; Convergence; Particle swarm optimization; Chaos theory; Ergodicity

INTRODUCTION

In the 1950s, the discovery of the double helix structure of DNA unveiled a new era of molecular biology. Since then, the study of genes and gene expression at the molecular level promoted the development of biology. As basic building blocks of life, genes as well as their products do not work independently. They interact with each other and form a complicated network. That is, some genes can activate or inhibit the expression of another gene. It is possible to infer gene regulatory relationships between genes from the gene expression levels measured from experimental data of the cell cycle. Based on these relationships, a gene regulatory network can be constructed that can facilitate biomedical research, such as cancer research, drug discovery, and disease prevention.

At present, through the analysis of gene expression microarray data, many methods and models have been developed and applied to infer gene regulatory networks. These studies contributed to insight into gene functions, gene regulation, cell biological functions, and working mechanisms through the development of tools such as weight matrices (Weaver et al., 1999), Boolean networks (Akutsu et al., 1999), differential equations (Chen et al., 1999), and Bayesian networks (BNs) (Friedman et al., 2000).

The BN is one of the most promising models for learning genetic regulatory networks because of its ability to deal with the noise in experimental measurements and because it can handle missing data and incomplete knowledge about the biological system. However, learning BNs is a nondeterministic polynomial time (NP)-hard problem.

Bayesian networks and particle swarm optimization

Bayesian networks and structure learning

BNs, known as directed acyclic graphs (DAGs), can be used to represent causal relationships among a large number of random variables in a graph. This is essentially a model of the pattern of an uncertain inference network that is based on probability. The BN was first proposed by Pearl Judea in 1987 (Judea, 1987). BN learning includes structure learning and parameter learning. The network structure, which is the qualitative part of the model, is used to describe the probability dependencies between variables. The DAG consists of two parts: nodes and directed arcs, where the nodes represent random variables and arcs represent statistical dependence relations among the variables in the problem areas. The conditional probability table indicates the dependence degree between the variables. If X is a non-root node, its conditional probability distribution can be expressed as $P(X \mid Pa(X))$, where $Pa(X)$ is the parents set of node X; if X is a root node, its marginal distribution is $P(X)$. The conditional probability distribution and the marginal distribution of all nodes form the conditional probability table.

Assume that the network has N nodes, marked as $X_1 \ldots X_N$, and P is the conditional probability of the node X_i that can be calculated as follows:

$$P(X_1, \cdots X_N) = \prod_{i=1}^{N} P(X_i \mid Pa(X_i)) \quad \text{(Equation 1)}$$

Definition 1: Let $G = (V, E)$ be any of a DAG and P is the joint probability distribution defined on the set V of random variables that satisfy the Markov condition. Then, $B = (G, P) = (V, E, P)$ is defined as the BN (Neapolitan, 2002).

Definition 2: Let $G = (V, E)$ be a DAG with a node $X \in V$ that is defined as the following:

$$a_{ij} = \begin{cases} 1, (X_i, X_j) \in E \\ 0, (X_i, X_j) \notin E \end{cases} \quad \text{(Equation 2)}$$

Then, $A = (a_{ij})$ is the adjacency matrix of graph G (Liu, 2001).

BN structure learning is used to find the best DAG match to the data set. Usually, the structure learning method is used to evaluate whether the DAG is a good match of this data set by defining a score function. Then, the structure graph is identified using the optimal score. There are common scoring criteria, such as the maximum likelihood score, Bayesian information criterion (BIC) score, minimum description length (MDL) score, and Bayesian Dirichlet equivalence (BDe). This paper intends to use the BDe score criteria, which can be used to calculate the posterior probability P of the structure G under the premise of a given data set to evaluate the BN.

So far, many people have presented different approaches to learning the structures of BNs. In 1992, the classical K2 algorithm proposed by Cooper and Herskovits (Cooper and Herskovis, 1992) was found to be the most successful algorithm, but it is heavily dependent on a prior order. In 2003, Chickering presented the greedy search (GS) algorithm (Chickering, 2002), which easily falls into the local optimal value. In 2007, Heng et al. proposed the discrete particle swarm optimization (PSO) (Heng et al., 2007). Subsequently, Sahina et al. introduced the distributed discrete PSO (Sahina et al., 2007). In 2010, T Wang et al. first proposed BN learning based on binary PSO (Wang and Yang, 2010), and J Zhao et al. presented the discrete binary quantum-behaved PSO algorithm (Zhao et al., 2009). This paper explored the combination of the chaotic theory with the PSO algorithm (CPSO) for learning BN structure.

Particle swarm optimization

PSO, which was first proposed by Eberhart and Kennedy in 1995 (Kennedy and Eberhart, 1995), is an evolutionary computing technology based on swarm intelligence. This algorithm is inspired by the social behavior of flocking birds. In PSO, each single solution is a "bird" in the search space. The PSO algorithm has the advantages of simple program design, few adjustable parameters, and a fast convergence rate at the early stage of the search. However, it easily falls into the local optimum later. The population of PSO is called a swarm, and each individual in the population of PSO is called a particle. Each particle's position is denoted as a D-dimensional vector $X_i = (x_{i1}\, x_{i2...}\, x_{iD})$. The i_{th} particle's velocity is represented by $V_i = (v_{i1}\, v_{i2...}\, v_{iD})$. In every search iteration, each particle is updated by following the two best values. The first value is the best solution (fitness) that the model has achieved so far. The fitness value is also stored. This value is denoted as $p_i = (p_{i1}\, p_{i2...}\, p_{iD})$. Another best value that is tracked by the particle swarm optimizer is the best value that is obtained so far by any particle in the population. This best value is the global best and is denoted by $P_g = (p_{g1}\, p_{g2...}\, p_{gD})$. After finding the two best values, the particle updates its velocity and positions with the following formulas:

$$v_{id}^{k+1} = \omega v_{id}^{k} + c_1 r_1 (p_{id}^{k} - x_{id}^{k}) + c_2 r_2 (p_{gd}^{k} - x_{id}^{k}) \quad \text{(Equation 3)}$$

$$x_{id}^{k+1} = x_{id}^{k} + v_{id}^{k+1} \quad \text{(Equation 4)}$$

Here, p_i is the best previous position of the i_{th} particle (also known as pbest), and p_g is the best position among all of the particles in the swarm (also known as gbest). k represents the iterative number, $i = 1,2,\ldots, N$ and $d = 1,2,\ldots, D$. The variables c_1 and c_2 are learning factors, where usually $c_1 = c_2 = 2$, and r_1 and r_2 are random values in the range of [0, 1]. ω is the inertia weight in the equation of velocity updating. A larger inertia weight facilitates global exploration, and a smaller inertia weight tends to facilitate local exploration to fine-tune the current search area. The termination criterion for the iterations is determined according to whether the max generation or a designated value of the fitness of p_g is reached.

Currently, the PSO algorithm has been used to develop calculation models in the form of position-speed, such as the inertial weight-PSO model, convergence factors-PSO model, and discrete and binary-PSO models. The performance of PSO models is different when solving different optimization problems because inertia weight can effectively regulate global and local searches. The design of the inertia coefficient has become the focus. This paper will design an inertia weight coefficient to improve the PSO models and gain the optimum network structure.

Structure learning algorithm based on chaotic PSO

Chaos theory

It is well recognized that chaos theory can be applied as a very useful technique in practical applications. The chaotic system can be described by a phenomenon in which a small change in the initial condition will lead to a nonlinear change in future behavior. Chaos optimization is a new search algorithm; its basic idea is to transform the variables from the chaos space to solution space. It searches optima by means of regularity, ergodicity, and intrinsic stochastic properties of chaotic motion, and it can identify the global optimum. The chaos optimization algorithm has the advantages of global asymptotic convergence and a fast convergence rate. Besides, it is easy to skip the local minima. These advantages can overcome the drawback of the standard PSO algorithm. When particles get into a local extremum, the chaos search strategy is introduced into PSO (Meng et al., 2004) to guide the particle swarm toward the best solution.

The chaotic map has a determinate form and does not contain any random factors. However, it can produce a dynamic change phenomenon that seems completely random, and it is extremely sensitive and dependent on the parameters.

One of the simplest maps, the logistic map proposed by May in 1976 (May, 1976), can be described by the following equation:

$$\omega_{i+1}^{*} = \mu * \omega_i^{*} (1 - \omega_i^{*}) \quad i = 1,2,\cdots, N \quad \text{(Equation 5)}$$

Here, $\mu > 3.57$ and $\omega_i \notin \{0.0, 0.25, 0.5, 0.75, 1.0\}$. When $\mu = 4$, system (5) iteratively produces a status of the pseudo-random distribution, which is a completely chaotic state.

Tian and Zhao (2010) improved the algorithm so that the tent-PSO has not only the powerful ability to search the global optimum but also effectively avoid the premature convergence in time. The tent map that was proposed (Shan et al., 2005) is given as the following:

$$\omega_{i+1} = \begin{cases} 2*\omega_i, 0 \le \omega_i \le 0.5 \\ 2*(1-\omega_i), 0.5 \le \omega_i \le 1 \end{cases} \quad \text{(Equation 6)}$$

An arbitrary initial value $\omega_0 \in [0.0, 1.0]$ can iterate a certain sequence ω_1, ω_2, ω_3, etc. Because of the sensitive dependence on the chaotic initial value, we take the experienced value $\omega = 0.7298$, which iteratively produces a certain sequence ω_1, ω_2, ω_3, etc. Finally, the chaotic sequence replaces the linear decreasing weight to identify the optimal particle.

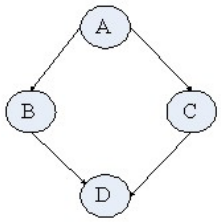
The chaotic sequence is generated in [0.0, 1.0] by (6). This paper intends to limit the inertia weight value in [0.4, 0.9], so we selected the improved tent map (Zhang et al., 2008) to limit the range in [0.4, 0.9]. The chaotic sequence is generated by the following formula:

$$\omega_{i+1} = \begin{cases} \omega_i / a, 0.4 \le \omega_i \le a \\ (1-\omega_i)/(1-a), a \le \omega_i \le 0.9 \end{cases} \quad \text{(Equation 7)}$$

In summary, we can use the chaotic characteristics to effectively improve the diversity of the population and the ergodicity of the particle search, create a better balance between the global and local optimization search, and avoid premature convergence.

Coding scheme

Because of the special form of the initial particle, we encoded the BN structure utilizing the coding scheme introduced by Poza (Larrañaga et al., 1996). A BN structure B with N nodes is represented by an $N * N$ adjacent matrix where any dimension is dispersed as 0 or 1. In the model of PSO, every particle is a BN structure, and each adjacent matrix is one solution of the problem. An example is shown in Figure 1:



$$X = \begin{bmatrix} 0 & 1 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 \end{bmatrix}$$

Figure 1. The structure of BN and its adjacent matrix.

It can be seen from the adjacent matrix in Figure 1 that x_{ij} represented the relationship between node i and node j: if node i is a parent of node j, then the matrix element $x_{ij} = 1$; otherwise $x_{ij} = 0$. In the vector set $x_i = (x_{1i}\ x_{2i}...x_{ni})$, x_i is the parent vector of node i and represents the edges from other nodes to node i in the network. The velocity, v_{ij}, represented the variation between node i and node j. This is a special probability parameter

to change the particle position. A transfer function $\sigma(v_{ij}^{k+1})$ is expressed by the following equation (Wang and Yang, 2010):

$$\sigma(v_{ij}^{k+1}) = \frac{1.0}{1.0 + e^{-v_{ij}^{k+1}}} \quad \text{(Equation 8)}$$

Equation (3) is used to update the velocity vector of the particle, and equation (8) can obtain a σ value of the velocity. The new position of the particle is obtained using the following equation:

$$v_{ij}^{k+1} = \begin{cases} 1, if\ v_{ij}^{k+1} > \rho \\ 0, otherwise \end{cases} \quad \text{(Equation 9)}$$

In this equation, ρ is a uniform random number in the range [0.0, 1.0].

The fitness is calculated using the BDe Bayesian score function mentioned above, and the equation is given below:

$$F = \lg\left(\prod_{i=1}^{n}\prod_{j=1}^{q_i}\frac{(r_i-1)!}{(N_{ij}+r_i-1)!}\prod_{k=1}^{r_i}N_{ijk}!\right) \quad \text{(Equation 10)}$$

Here, r_i is the number of states of node i, the first product N is over the nodes in the network, the second product q_i is over the set of permutations of the parents of node i, and the third product r_i is over the states of the node. Also N_{ij} is defined as:

$$N_{ij} = \sum_{k=1}^{r_i} N_{ijk} \quad \text{(Equation 11)}$$

Here, N_{ijk} is the elements of node i in the conditional probability table (CPT), which contains occurrences of joint instantiations of the parents where each permutation is indexed with j of node i, which is in state k. Hence, the sum N_{ij} is the total of a column of the conditional probability table where each column enumerates occurrences of node i in each state for a specific instantiation set of parents.

The function value obtained by equation (10) represents the score of the network structure learning and the degree of fit with the known network structure. The higher the value is, the greater the degree of similarity fitting.

Novel learning algorithm: BN-CPSO

In this paper, we used the proposed CPSO algorithm to identify the global best position p_g, which is a network structure, and to obtain the ordering of the nodes. Then, the K2 algorithm uses the node ordering as an input parameter to obtain the optimal network structure. The following steps show the CPSO and ordered K2 algorithm.

1) Initialize the particle swarms, the parameters containing the number of swarms,

iterations, c_1, c_2, and ω.

2) Encode according to the BN structure and its adjacent matrix. Adopt the maximum weight spanning tree algorithm (MWST) to get the initial DAG.

3) Obtain all adjacency graphs of the DAG using the mk_dag_topo algorithm and select a certain n umber as the initial particle swarm.

4) Calculate the fitness value through the function F of each particle at its current position and find the best value as the global optimal particle.

5) Compare the performance of each particle to its best performance:

if $F(\chi_i(t)) < F(p_i)$

then, $F(p_i) = F(\chi_i(t))$ and $p_i = \chi_i(t)$.

6) Compare the performance of each particle to the global performance:

if $F(\chi_i(t)) < F(p_g)$

then, $F(p_g) = F(\chi_i(t))$ and $p_g = \chi_i(t)$.

7) Use the tent map equation (7) to obtain the chaotic sequence ω_i, and limit the sequence in the range of [0.4, 0.9].

8) Calculate a new velocity using equation (3) to change each particle, and discretize it into 0 or 1 according to equations (8) and (9).

9) Change each particle to a new position using equation (4).

10) Go to step (3), repeat the process until convergence or the maximum number of iterations have been reached, and record the global optimal particle p_g.

11) Transfer the encoding to a matrix and obtain the node ordering from the DAG constructed in the previous step.

12) Run the K2 algorithm to obtain the network structure using the ordered nodes as the input of the K2 algorithm.

MATERIAL AND METHODS

Experiment

In this section, the implementation of the algorithm was written in MATLAB and compiled using MATLAB 2009b. Our algorithm is based on the BN Toolbox (BNT) (Murphy et al., 1997-2002) and its structure learning package (BNT_SLP). The code is completely free to the public. In addition, the system has good scalability, which makes our learning and improvement algorithm more convenient than previous algorithms.

Parameters

In order to quantitatively evaluate the authenticity of the reconstructed network, this paper used a number of evaluation criteria. Above all, the Bayesian score was adopted. The higher the Bayesian score is, the better the obtained network structure will be. A previous report (Kim et al., 2004) mentioned two criteria: sensitivity and specificity. The following formulas are for the sensitivity and specificity calculation, respectively.

$$P_{se} = \frac{E_c}{E_{ta}} \quad \text{(Equation 12)}$$

Here, E_c represents the correctly estimated edges and E_{ta} indicates all edges in the target network. P_{se} is the sensitivity, which represents the proportion of the edges that were correctly estimated by the algorithm compared to the real regulation relationships in the network.

$$P_{sp} = \frac{E_c}{E_a} \quad \text{(Equation 13)}$$

Here, E_c represents the correctly estimated edges and E_a indicates all estimated edges. P_{sp} is the specificity, which denotes the proportion of the correct estimations in the network.

We defined D as the sum of the removed edges, the added edges, and the reversed edges. We averaged the results from ten experiments.

For the purpose of comparison, all of the simulations used the same parameter settings for the algorithm except the inertia weight ω. Table 1 shows the parameter values for the three algorithms.

Table 1. Parameter values for three algorithms .

Algorithms	Max-parents	k	V_{max}	ω	c_1	c_2
K2	2	-	-	-	-	-
BPSO	2	10	4	0.4-0.9	2	2
CPSO	2	10	4	0.4-0.9	2	2

k = number of particle; V_{max} = limit exploration of velocity; ω = additional inertia weight; c_1 = local learning factors; c_2 = global learning factors.

Here, max-parents is the upper boundary of the number of parents a node may have to compute efficiency, k is the number of particles, V_{max} is the limit exploration of velocity. ω is an additional inertia weight in the updated equation of velocity and c_1 and c_2 are local and global learning factors, respectively.

RESULTS AND DISCUSSION

In order to assess the performance of CPSO_BN, we compared with classic K2 algorithm (Leray and François, 2004), the BPSO algorithm (Wang and Yang, 2010), and the chaos hybrid genetic algorithm (CHGA) (Shen et al., 2012) by using data sets that were generated from a known BN that used a probabilistic logic sampling known as ASIA. The Asia network was initially presented by Lauritzen and Spiegelhalter (Lauritzen et al., 1988) and is a small network (eight nodes) that is used in the domain of medical knowledge. The training set with 2000 samples is generated from the original network.

We also applied these algorithms to gene expression data to infer genetic regulatory networks. The microarray dataset we employed is from a previous study on yeast cell-cycle gene regulation (Spellman et al., 1998) and is publicly available. We chose eight histone genes - HHT1, HHT2, HHF1, HHF2, HTA1, HTA2, HTB1, and HTB2 - as our experimental data set. In this paper, we used 77 data samples for these eight genes and compared our results with those from Chen et al. (Chen et al., 2006). The two original networks were shown in Figure 2, and the experimental comparison results were shown in Figure 3 and Figure 4.

From Figure 3, we know that the proposed CPSO and the CHGA (Shen et al., 2012) algorithms both can reconstruct the network efficiently. The proposed method can better predict the initial particle swarm by computing all of the adjacent matrices for the graph of the MWST algorithm.

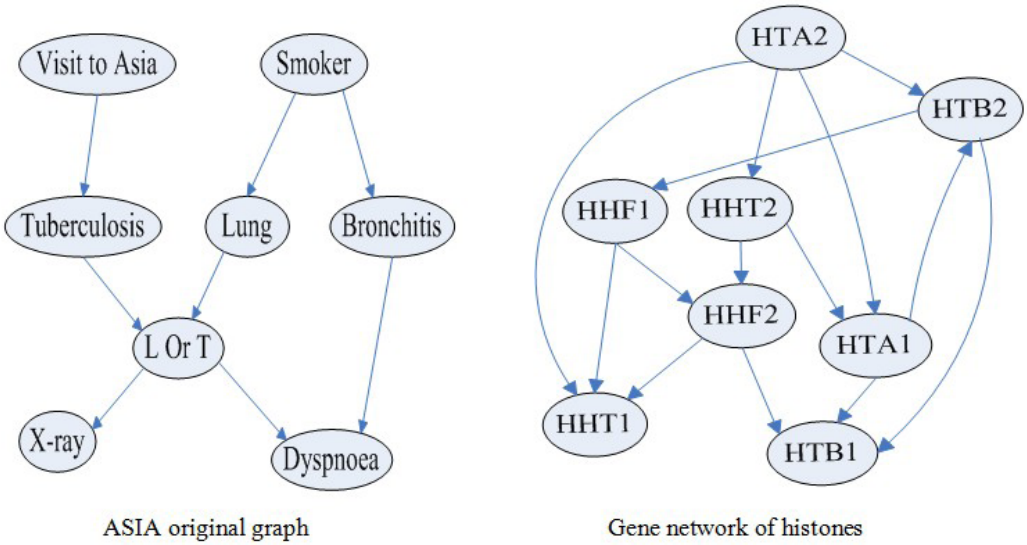


Figure 2. Standard ASIA and gene network of histones.

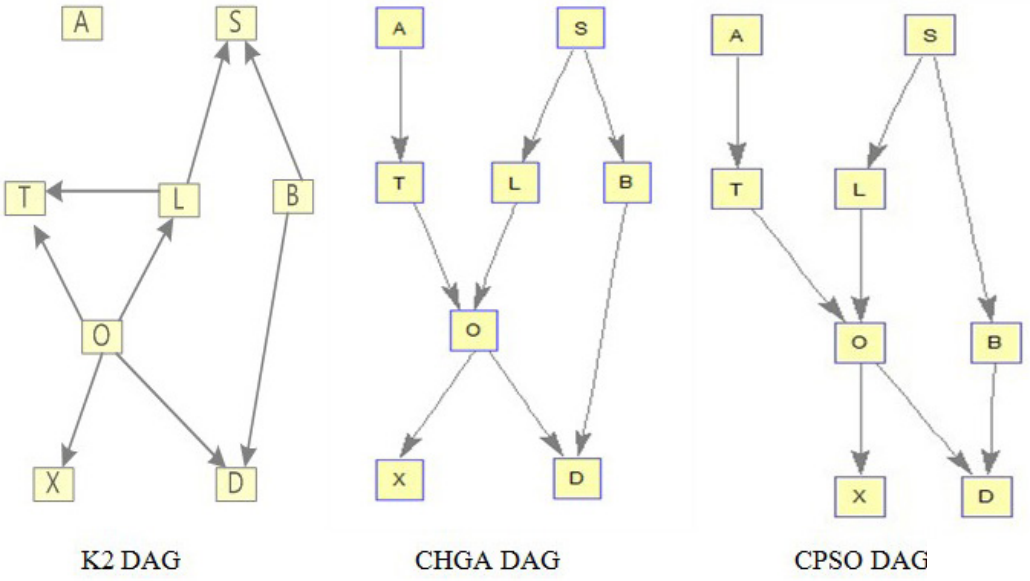


Figure 3. Comparison of ASIA network structure learning.

According to the DAG, we get the 2nd, 3rd, and 4th column data in Table 2 by using the formulas (12) and (13). The data in the 5th column are obtained through the Bayesian scoring function (9).

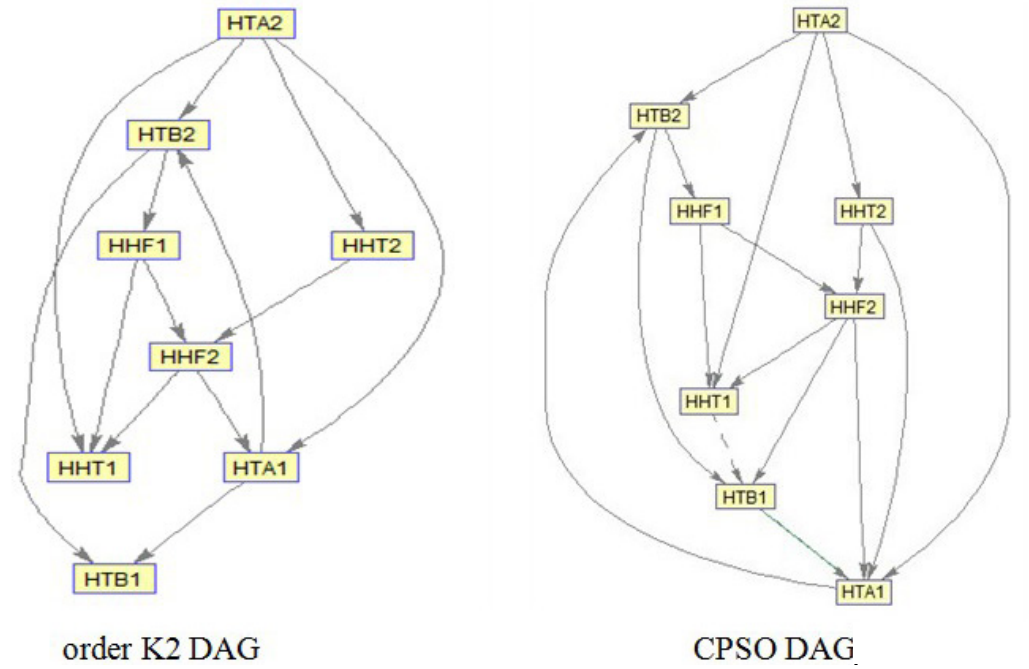


Figure 4. Comparison of gene network of histones structure learning.

Table 2. Comparison of the various algorithms.

Algorithms	P_{se}	P_{sp}	D	Score
K2 (Leray and François, 2004)	0.375	0.333	6.0	-2291.4
CHGA (Shen et al., 2012)	1.0	1.0	0.2	-
CPSO	1.0	1.0	0.1	-2143.6

From Table 3, we can see that, for all data sets, the Bayesian score of our CPSO algorithm is higher than that from the K2 and BPSO algorithms, which indicates that our algorithm can identify networks that are close to the optimal structures.

Table 3. Comparison of BDe with various datasets.

Algorithms	Asia500	Asia1000	Asia2000	Asia3000	Asia5000
K2 (Leray and François, 2004)	-586.5	-1173.5	-2291.4	-3287.3	-5703.2
BPSO (Wang and Yang, 2010)	-547.1	-1078.5	-2149.1	-3239.2	-
CPSO	-539.6	-1067.4	-2143.6	-	-5665.4

As illustrated in Figure 4, compared with the results from Chen et al. (Chen et al., 2006), the CPSO method uncovered 93% (13 out of 14) of the currently known interactions among them. The dashed line predicts the unsupported edge, and the green line indicates the reversed edge. Experimental evidence may support some currently unsupported edges of the

constructed network in the future. Therefore, these unsupported edges are not necessarily false ones. The interaction between HTA1 and HTB1 may be a mutual regulatory relationship; we will study the relationship in equivalence class space.

CONCLUSIONS

This paper presented a new method for BN learning. The proposed method identifies the ordering of nodes for the K2 algorithm using CPSO. We used the chaotic map to update the particles, and the proposed method improved the quality of the obtained network structure and enhanced the convergence of the particle. Although the algorithm uncovered some interactions that were novel and currently unknown, the validation of the new interactions is beyond the scope of this article. The possibility of mutual regulatory relationships provides a future research direction.

ACKNOWLEDGMENTS

Research supported by the National Natural Science Foundation of China (#31370778, #31170797, #30870573, #61103057), the Program for Changjiang Scholars and Innovative Research Team in University (#IRT1109), the Program for Liaoning Excellent Talents in University (#LR201003), and the Program for Liaoning Science and Technology Research in University (#LS2010179).

REFERENCES

Akutsu T, Miyano S and Kuhara S (1999). Identification of genetic networks from a small number of gene expression patterns under the Boolean network model. *Pac. Symp. Biocomput.*17-28.

Chen T, He HL and Church GM (1999). Modeling gene expression with differential equations. *Pac. Symp. Biocomput.*29-40.

Chen XW, Anantha G and Wang X (2006). An effective structure learning method for constructing gene networks. *Bioinformatics* 22: 1367-1374.

Chickering DM (2002). Optimal structure identification with greedy search. *J. Mach. Learn. Res.* 11: 507-554.

Cooper GF and Herskovis E (1992). A Bayesian method for the induction of probabilistic networks from data. *Mach. Learn.* 9: 309-347.

Friedman N, Linial M, Nachman I and Pe'er D (2000). Using Bayesian networks to analyze expression data. *J. Comput. Biol.* 7: 601-620.

Heng XC, Qin Z, Tian L and Shao LP (2007). Learning Bayesian Network Structures with Discrete Particle Swarm Optimization Algorithm. Foundations of Computational Intelligence, 2007. FOCI 2007. IEEE Symposium on, 47-52.

Judea P (1987). Evidential reasoning using stochastic simulation of causal models. *Artif. Intell.* 32: 245-257.

Kennedy J and Eberhart RC (1995). Particle Swarm Optimization. Proceedings of IEEE International Conference on Neural Network, Piscataway, 1942-1948.

Kim S, Imoto S and Miyano S (2004). Dynamic Bayesian network and nonparametric regression for nonlinear modeling of gene networks from time series gene expression data. *Biosystems* 75: 57-65.

Larrañaga P, Poza M, Yurramendi Y, Murga RH, et al. (1996). Structure learning of Bayesian networks by genetic algorithms: a performance analysis of control parameters. IEEE Transactions on Pattern Analysis and Machine Intelligence, 18: 912-926.

Lauritzen SL and Spiegelhalter DJ (1988). Local Computations with Probabilities on Graphical Structures and Their Application to Expert Systems. *J. R. Stat. Soc. Series B* 50: 157-224.

Leray P and François O (2004). BNT Structure Learning Package: Documentation and Experiments. PSI Technical Reports, Rouen.

Liu ZQ (2001). Causation, Bayesian Networks, and Cognitive Maps. *Acta Automatica Sin.* 27: 552-566.

Neapolitan RE (2002). Learning Bayesian Networks. Northeastern Illinois University.

May RM (1976). Simple mathematical models with very complicated dynamics. *Nature* 261: 459-467.
Meng HJ, Zheng P, Wu RY, Hao XJ, et al (2004). A Hybrid Particle Swarm Algorithm with Embedded Chaotic Search. Cybernetics and Intelligent Systems, 2004 IEEE Conference on, 371 -367.
Murphy K (1997-2002). Bayes Net Toolbox for Matlab. Available at [http://people.cs.ubc.ca/ murphyk/Software/BNT/bnt.html]. Accessed October 19, 2007.
Sahina F, Yavuz MÇ, Ziya A and Önder U (2007). Fault diagnosis for airplane engines using Bayesian networks and distributed particle swarm optimization. *J. Parallel Comput.* 33: 124-143.
Shan L, Qiang H, Li J and Wang ZQ (2005). Chaotic optimization algorithm based on tent map. *Control Decision.* 20: 179-182.
Shen JJ, Lin F, Sun W and Chang KC (2012). Bayesian Network Structure Learning Using Chaos Hybrid Genetic Algorithm. Proceedings SPIE 8392, Signal Processing, Sensor Fusion, and Target Recognition XXI.
Spellman PT, Sherlock G, Zhang MQ, Iyer VR, et al. (1998). Comprehensive identification of cell cycle-regulated genes of the yeast Saccharomyces cerevisiae by microarray hybridization. *Mol. Biol. Cell* 9: 3273-3297.
Tian DP and Zhao TX (2010). Particle awarm optimization based on tent map and logistic map. *J. Shanxi Univ. Sci. Technal.* 28: 17-23.
Wang T and Yang J (2010). A heuristic method for learning Bayesian networks using discrete particle swarm optimization. *Knowl. Inf. Syst.* 24: 269-281.
Weaver DC, Workman CT and Stormo GD (1999). Modeling regulatory networks with weight matrices. *Pac. Symp. Biocomput.*112-123.
Zhang H, Zhang TN, Shen JH and Li Y (2008). Research on decision-makings of structure optimization based on improved tent PSO. *Control Decision.* 23: 857-862.
Zhao J, Sun J, Xu WB and Zhou D (2009). Structure Learning of Bayesian Networks Based on Discrete Binary Quantum-Behaved Particle Swarm Optimization Algorithm. ICNC '09 Proceedings of the 2009 Fifth International Conference on Natural Computation, 86-90.