A Bayesian Network Learning Algorithm Based on Independence Test and Ant Colony Optimization

JI Jun-Zhong[1] ZHANG Hong-Xun[1] HU Ren-Bing[1] LIU Chun-Nian[1]

Abstract To solve the drawbacks of the ant colony optimization for learning Bayesian networks (ACO-B), this paper proposes an improved algorithm based on the conditional independence test and ant colony optimization (I-ACO-B). First, the I-ACO-B uses order-0 independence tests to effectively restrict the space of candidate solutions, so that many unnecessary searches of ants can be avoided. And then, by combining the global score increase of a solution and local mutual information between nodes, a new heuristic function with better heuristic ability is given to induct the process of stochastic searches. The experimental results on the benchmark data sets show that the new algorithm is effective and efficient in large scale databases, and greatly enhances convergence speed compared to the original algorithm.

Key words Uncertainty modeling, Bayesian network structure learning, ant colony optimization (ACO), conditional independence test

Bayesian network (BN) is an important theory model within the community of artificial intelligence, and also a powerful formalism to model the uncertainty knowledge in practise. Recently, learning a BN structure from data has received considerable attentions and researchers have proposed various learning algorithms[1−16]. Especially, there are three efficient approaches using the stochastic search mechanism to tackle the problem of learning Bayesian network. The first one uses genetic algorithm (GA)[5, 7], the second one applies evolutionary programming (EP)[8, 11, 13], and the third one employs ant colony optimization (ACO)[6, 9].

To solve the drawbacks of the ant colony optimization for learning Bayesian networks[6] (ACO-B), this paper proposes a Bayesian network structure learning algorithm based on the conditional independence test and ant colony optimization (I-ACO-B), which not only employs constraint knowledge to reduce the search space, but also takes it as heuristic knowledge to induct the process of stochastic searches. First, the new algorithm uses order-0 independence tests to effectively restrict the available scope of candidate arcs, reduce the space of candidate solutions, and induce ants to avoid many unnecessary searches. And then, by combining the global score increase of a solution with the local mutual information between nodes, a new heuristic function with better heuristic ability is given to induct the process of stochastic searches. The experimental results on the benchmark data sets show that the new algorithm is effective and efficient in large scale databases, and greatly enhances convergence speed compared to the original algorithm.

The paper is organized as follows. In Section 1, we present the background of Bayesian networks and the basic idea of the ant colony optimization for learning Bayesian networks. In Section 2, we describe our new algorithm in detail. Section 3 reports our experimental results. Finally, we conclude the paper in Section 4.

1 Bayesian network structure learning based on ant colony optimization

1.1 Bayesian networks

A Bayesian network (BN) can be denoted as a triple group $\langle X,\ A,\ \Theta\rangle$, where $\langle X, A\rangle$ defines a directed acyclic graph (DAG) structure G, X is the set of nodes, $X_i \in X$ represents a random variable in a special domain; A is a set of directed arcs, $a_{ij} \in A$ describes a direct probabilistic dependency between X_i and X_j, $X_i \leftarrow X_j$; and $\Theta = \{\theta_i\}$ is a set of parameters, $\theta_i = p(X_i|\Pi(X_i))$ is the conditional probability distribution of X_i given the parent set of the variable X_i. As the graph structure G qualitatively characterizes the independence relationship among random variables, and the conditional probability distribution quantifies the strength of dependencies between a node and its parent nodes. Thus, Bayesian network $\langle X,\ A,\ \Theta\rangle$ uses a graph structure and a set of parameters to encode uniquely the joint probability distribution of the domain variables $X = \{X_1, X_2, \cdots, X_n\}$:

$$P(X_1, X_2, \cdots, X_n) = \prod_{i=1}^{n} P(X_i|\Pi(X_i)) \tag{1}$$

1.2 Bayesian network structure learning

The structure of a BN reflects the underlying probabilistic dependence relations among the nodes (corresponding attributes of data) and a set of assertions about conditional independencies. The problem of learning a BN structure can be stated as follows: given a sample data set $D = \{X[1], X[2], \cdots, X[N]\}$, the learning goal is to find the BN structure that best matches D. During the past decade, people have proposed many algorithms on learning Bayesian network structure. Actually, there are two basic realization mechanisms. The first one is an approach based on constraints[2−3], which poses the learning process as a constraint satisfaction problem, and then constructs a network structure by testing the conditional independence relations. The second one is score-and-search approach[1, 4−9], which poses the learning problem as a structure optimization problem. Namely, it uses a score metric to evaluate every candidate network structure, and then, finds a network structure with the best score. Though the implement of the former approach is relatively simple, the computations for high-order testings are complex and irresponsible. Moreover, the precision of learning a model is hard to ensure, thus the score-and-search approach gradually becomes a popular approach for learning Bayesian networks.

Since the parent nodes of each node in a BN, $\Pi(X_i) = \{X_k : k \in \Phi(i)\}$, are only selected from the set of nodes

Received December 24, 2007; in revised form April 15, 2008
Supported by National Natural Science Foundation of China (60496322), Natural Science Foundation of Beijing (4083034), and Scientific Research Common Program of Beijing Municipal Commission of Education (KM200610005020)
1. Beijing Municipal Key Laboratory of Multimedia and Intelligent Software Technology, College of Computer Science and Technology, Beijing University of Technology, Beijing 100124, P. R. China
DOI: 10.3724/SP.J.1004.2009.00281

preceding the current node in a node ordering, namely, $\Phi(i) \subset \{1, 2, \cdots, i-1\}$ (i denotes the sequence number of a node), the number of possible parent sets is 2^{i-1} for each node X_i. Further, the number of possible structures for a BN with n nodes is $2^{n(n-1)/2}$ when a node ordering is known, and then the complexity of a BN structure space is $n! \cdot 2^{n(n-1)/2}$ for the case of a node ordering unknown. Obviously, it is intractable for the complete search based on score to find the global best solution when n is large. Recently, some researches proposed some effective algorithms[4,10,12] with the restriction of having a complete node ordering. Unfortunately, these algorithms still perform complete searching in the worst case, so they are unfitted to learn a BN structure without a complete node ordering.

Though some improved hill-climbing algorithms[15−16] can solve the problem of learning a BN structure with an unknown node ordering, they usually get a local optimal solution of the model. Recently, the development of stochastic search technology has provided effective and feasible methods to tackle the problem, and genetic algorithms, simulated annealing[14], evolutionary programming and ant colony optimization have been applied to learning Bayesian networks one after the other. These methods perform stochastically iterative search and find the global optimal solution by means of simulating various natural phenomena. In the following, we introduce the ACO-B algorithm, which is an effective ant colony optimization for learning Bayesian networks.

1.3 Learning Bayesian networks using ACO (ACO-B)

Ant colony optimization (ACO), proposed by Dorigo in 1990 s[17−18], is a new meta-heuristic search algorithm, which is often used to solve combinatorial optimization problems. The mechanism is the simulation of the intelligent behaviors of real ant colonies looking for food. The frame work of ACO has gradually grown up[19−21] for many years, and there are many successful applications in a wide range of different fields[22], such as data mining, machining learning, and bioinformatics. ACO-B algorithm[6] is a score-and-search approach based on the ant colony optimization for learning Bayesian networks, whose main idea is to use the $K2$ metric as a score measure $f(G:D) = \sum_{i=1}^{n} f(X_i, \Pi(X_i))$ to evaluate a BN structure, and induce ants to search the global maximum in a feasible solution space.

Let a be the number of ants in an ant colony, $\tau_{ij}(t)$ be the pheromone intensity associated with the directed arc a_{ij} at time t, and the initial pheromone intensity of every directed arc be a constant value C, i.e., $\tau_{ij}(0) = C$. During constructing a solution, each ant $k\,(k = 1, 2, \cdots, a)$ starts from the empty graph G_0 (arcs-less DAG) and proceeds by adding an arc at one time. The construction process of a BN for an ant is shown in Fig. 1, where the current state G_h of an ant is a graph with all nodes $X_i \in X$, exactly h arcs and no directed cycle. Suppose there are m candidate directed arcs. In terms of the pheromone and heuristic information of candidate arcs, the ant selects the s-th arc a_{ij} as a new component of a solution, thus the new state by adding an arc a_{ij} can be denoted as $G_{h+1} = G_h \cup \{a_{ij}\}$. Once there is no way to make the score of a BN structure more higher by adding an arc, the construction process is ended and the ant gets its solution G_g.

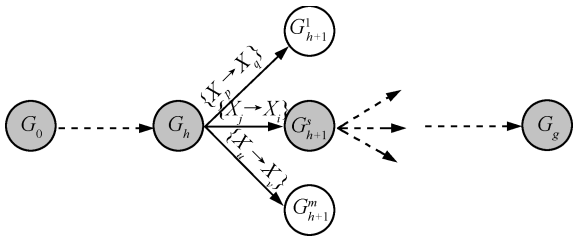


Fig. 1 The construction process of a BN for an ant

The detailed process of constructing a solution can be described as follows. At time t, the probabilistic transition rule that an ant k selects a directed arc a_{ij} from the current candidate arcs is defined as

$$i, j = \begin{cases} \arg \max\limits_{r,l \in DA_k(t)} \{[\tau_{rl}(t)] \cdot [\eta_{rl}(t)]^{\beta}\}, & \text{if } q \le q_0 \\ I, J, & \text{otherwise} \end{cases} \tag{2}$$

where $\eta_{rl}(t)$ represents the heuristic information of the directed arc a_{rl}, β is the weighted coefficient which controls $\eta_{rl}(t)$ to influence the selection of arc. $DA_k(t)$ $(r, l \in DA_k(t))$ is the set of all candidate arcs that satisfy constraint conditions and heuristic information is larger than zero, q_0 $(0 \le q_0 \le 1)$ is an initial parameter that determines the relative importance of exploitation versus exploration, q $(q \in [0, 1])$ is a random number; I and J are a pair of nodes randomly selected according to the probabilities in (3), with $\alpha = 1$.

$$p^k_{ij}(t) = \begin{cases} \dfrac{[\tau_{ij}(t)]^{\alpha} \cdot [\eta_{ij}(t)]^{\beta}}{\sum\limits_{r,l \in DA_k(t)} [\tau_{rl}(t)]^{\alpha} \cdot [\eta_{rl}(t)]^{\beta}}, & \text{if } i, j \in DA_k(t) \\ 0, & \text{otherwise} \end{cases} \tag{3}$$

where parameter α depicts the relative importance of the pheromone $\tau_{rl}(t)$ left by the real ants. As the learning goal is to achieve the best BN structure whose $K2$ score is the maximum, the heuristic information function of a directed arc can be interpreted as the greatest increase produced in $K2$ score when the arc is added to the graph. Since the metric $K2$ is decomposable, the heuristic information function can be defined as

$$\eta_{ij}(t) = f(X_i, \Pi(X_i) \cup X_j) - f(X_i, \Pi(X_i)) \tag{4}$$

After each iteration of the ant colony is performed, ACO-B algorithm will carry out the pheromone updating process, which includes local and global updating steps. First, while building a solution, if an ant selects an arc a_{ij}, then the pheromone level of the corresponding arc is changed in the following way

$$\tau_{ij}(t+1) = (1 - \psi)\tau_{ij}(t) + \psi\tau_0 \tag{5}$$

where τ_0 is a constant related with the initial solution, $0 < \psi \le 1$ is a parameter that controls the pheromone evaporation. And then, from all feasible solutions, the algorithm finds the best solution obtained so far by means of the $K2$ metric, and performs the global updating for each arc of the current best solution, the global updating rule is

$$\tau_{ij} = (1 - \rho)\tau_{ij} + \rho \triangle \tau_{ij}$$
$$\triangle \tau_{ij} = \begin{cases} \dfrac{1}{|f(G^+ : D)|}, & \text{if } a_{ij} \in G^+ \\ \tau_{ij}, & \text{otherwise} \end{cases} \tag{6}$$

where $0 < \rho \leq 1$ is also a parameter of the pheromone evaporation and $f(G^+ : D)$ is the metric value of the best solution G^+.

Assume that ACO-B algorithm will end after running NC times. Then, the global optimal solution is $G^+_{best-all} = \arg\max_l f(G^+_{(l)} : D)$ where $l \in (1, 2, \cdots, NC)$ is the number of iterations, and $G^+_{(l)} = \arg\max_k f(G_k : D)$ $(k \in (1, 2, \cdots, a))$ is the best solution at the l-th iteration.

Moreover, ACO-B algorithm employs a local optimizing process to prevent from getting into a local maximum, more specifically, it uses the standard operators of arc addition, arc deletion, and arc reversal to locally optimize the obtained solution and improve the quality of the solution. ACO-B algorithm adopts the stochastic search mechanism based on ACO, so it could get the global best solution and its quality is higher than that of solutions obtained by many deterministic search methods[6]. However, there are two drawbacks of ACO-B algorithm, namely, the iteration number is too large and the convergence time is too long. The main reason lies on that even if ACO-B does not traverse the candidate solution space, ants may select some candidate arcs that are impossible to be components of the best solution during each iteration. In other words, ACO-B might gain many useless combinations, which makes the search space large and wastes too much running time. Therefore, a hybrid algorithm coupled with conditional independence tests and ACO is presented in the following.

2 I-ACO-B algorithm

2.1 Main ideas

As mentioned above, a BN is a graph for the probability dependence relationships among random variables, so a BN structure can be determinated by means of performing effective conditional independence tests on sample data and distinguishing all connected relationship among nodes. That is just the basic idea of the learning approach based on constraints. On the other hand, the learning model of a BN structure can be defined as $M = (S, \Omega, f)$ from the view of score-and-search, where S is a search space which defines a set of graphs including all nodes in X and possible arcs connected one another; Ω is the set of constraint conditions among nodes, the basic constraint is that all nodes construct a directed acyclic graph; the score function f is a mapping from S to the set of real numbers ($f : S \to \mathbf{R}$), the function extremum (maximum or minimum) corresponds to the best network structure. A feasible solution $s \in S$ can be denoted a connect graph which satisfies all constraints in Ω, and a solution $s* \in S$ is called a global optimum if and only if $f(s*) \geq f(s)$ (or $f(s*) \leq f(s)$), $\forall s \in S$. It is obvious that the search space, the constraint set, and the score function are the three elements influencing the performance of the algorithm based on score and search. Because the research about the common score functions (Bayes scoring and MDL scoring) has grown up, and in theory, the more constraint conditions in Ω, the smaller the search space S of BNs, thus the search efficiency will be much higher. Hence, it is very necessary for people to research how to discover the knowledge and employ it to reduce the search space. Therefore, considering the BN′s own characteristic, we combine the ideas of two basic approaches to learn a BN structure with ACO. First, some order-0 independence tests with low cost are performed to discover a few potential constraints (i.e., independence knowledge) from the sample data D. Second, the search space is effectively reduced by using the obtained knowledge. Third, the obtained knowledge is reused to revise the heuristic function and ants carry out fast searching in the reduced space.

2.2 Reducing search space using conditional independence test

In ACO-B, ants start from G_0 (arcs-less DAG) in light of the complete connect graph, construct their respective feasible solution by adding a directed arc to the current graph each time. As ACO-B is an iterative optimization algorithm based on a stochastic search for the space of all feasible solutions, each ant could select a satisfied arc from a candidate complete connect graph at every iteration, thus the complexity of the initial candidate connect graph determines the complexity of ACO-B algorithm to a large extent. In other words, if some strategies are adopted and the initial connect graph is simplified, then the search space of the algorithm will be greatly reduced. In light of the idea of the constraint satisfaction, I-ACO-B algorithm first uses conditional independence (CI) test to reduce the search space before ants search.

For a BN structure learning, CI test is a typical method that validates the conditional independence relationship between two variables given the conditional set. The basic of CI is the measure of an information flow in information theory. A simple and natural measure for an information flow between X_i and X_j is the mutual information:

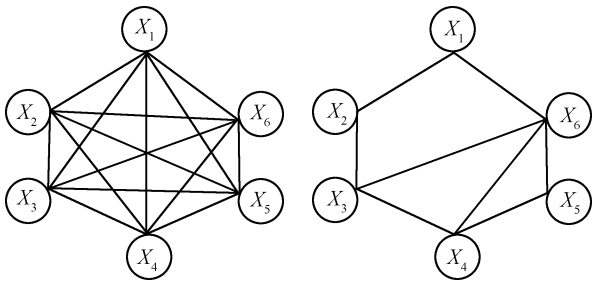
$$\inf(X_i, X_j|Z) = \sum_{x_i, x_j, z} \hat{P}(x_i, x_j, z) \log \frac{\hat{P}(x_i, x_j|z)}{\hat{P}(x_i|z)\hat{P}(x_j|z)} \tag{7}$$

where Z is the given condition set, $\hat{P}$ denotes an empirical probability estimate for various cases in the sampling data set D, and x_i, x_j, z correspond to the observed values of the variables and the condition variable set, respectively.

Considering the reliability and less computational cost of the low order CI, I-ACO-B algorithm only adopts the order-0 independence tests (Z is a null set). More specially, we first build an undirected complete graph including all nodes, and then, compute the mutual information $\inf(X_i, X_j)$ for each arc of the complete graph. Given the confidence level, we confirm the undirected relations among modes by means of χ^2 test, and get the forbidden connect set FA using just obtained relations. Finally, we remove these redundant connects in FA from the complete connect graph, and change the complete connect graph to a possible connect graph.

For example, consider the two different initial connect graphs with 6 nodes in Fig. 2 and assume that we get 7 conditional independency assertions $I(X_1, X_3)$, $I(X_1, X_4)$, $I(X_1, X_5)$, $I(X_2, X_4)$, $I(X_2, X_5)$, $I(X_2, X_6)$, and $I(X_3, X_5)$ by the order-0 CI tests. The complete connect graph shown in Fig. 2 (a) includes $2 \cdot C_6^2 = 30$ directed arcs, while the possible connect graph shown in Fig. 2 (b) only includes 16 directed arcs by employing the obtained constraint knowledge to delete corresponding redundant arcs. Because all possible network structures including these redundant arcs will be prevented from construction, the search space is greatly reduced. The reduction of the search space in I-ACO-B will directly influence the efficiency of ACO. The deletion of redundant arcs greatly restricts the selecting scope of ants, reduces some sightless searchings, and avoids many constructing and scoring processes for those network structures including these redundant arcs. Hence, the strategy can shorten score and search time and improve the search efficiency. Moreover, the step lies on the inner circle of an ant iterative search, so the improvement will be repeatedly

magnified during an ant colony searching, and then induce the performance of I-ACO-B algorithm to improve remarkably.



(a) A complete connect graph (b) A connect graph satisfying constraints

Fig. 2 The initial candidate connect graph

2.3 Heuristic function with a weighted factor

In ACO-B, the heuristic function is defined as the score increase introduced by an arc addition. According to the decomposability of $K2$ metric, the operator that adds an arc $X_j \rightarrow X_i$ to the current G_h will bring $f(G_{h+1} : D) - f(G_h : D) = f(X_i, \Pi(X_i) \cup X_j) - f(X_i, \Pi(X_i))$. Apparently, the definition also reflects the score increase of the structure changing, so represents the global information of the solution. However, the definition has a drawback, namely, it only gives heuristic information of arcs from evaluating of an arc combination (solution structure), but does not consider arc′s own connecting specialities implied in sample data D. That is apt to make the heuristic information unilateral and might influence the heuristic ability. Therefore, we redefine the heuristic function of a directed arc:

$$\eta_{ij}(t) = \omega \cdot (f(X_i, \Pi(X_i) \cup X_j) - f(X_i, \Pi(X_i))) \qquad (8)$$

where ω is a weighted factor concerned with arc connecting speciality, its value is larger than 1. Because the mutual information $\inf(X_i, X_j)$ can objectively reflects whether the two nodes in BN are dependent and how much the dependency is, namely, they are independent each other when $\inf(X_i, X_j) = 0$, otherwise the more the value of the mutual information is, the stronger the dependence between the two nodes is. It shows that the value of the mutual information can be used as heuristic knowledge to induce ant selecting arcs. Thus, the weighted factor is defined as $\omega = 1 + \inf(X_i, X_j)$, which employs the local dependency information (mutual information) of arcs to control ant to select an arc. Obviously, when the dependency intensity is strong and the score increase is large, the heuristic information is great, and vice versa. In other words, new definition integrates the global solution information with the local component information and guides together an ant to select arcs. The experimental results in next section also show that the strategy can enhance the ability of the heuristic function.

2.4 Algorithm description

Based on the main idea of ACO-B, I-ACO-B employs two strategies to improve the original algorithm. First, the CI tests are introduced so that the knowledge from dependency tests is exploited to restrict the search space, namely, the dependence information is effectively used to avoid many unnecessary searches. Second, the knowledge from CI tests is reused in the new heuristic function, which enhances the heuristic ability during searching. In other words, the new algorithm not only makes use of the constraint knowledge to reduce the search space, but also takes it as the heuristic information to induce searching. In contrast to ACO-B, there are two differences as follows: 1) using order-0 independence tests to obtain an initial structure graph so that the search space is greatly reduced; 2) combining the mutual information with the score increase and giving a heuristic function with more powerful heuristic ability. The pseudo code of I-ACO-B is shown in the following, where each ant uses the function called *AntconstructGraph*() to construct its respective solution, and the *Optimization*() function is employed to perform a local optimization for the solution G_k.

Algorithm. I-ACO-B
Begin Procedure
1) Initialization;
Initialize a, NC, $G(0)$, l_{step}, $G^+ = G(0)$, $\tau_{ij} = 1/n \times |f(G(0) : D)|$;
2) Condition independent test phase;
for every pair of nodes $(X_i, X_j) \in X$ do:
Perform order-0 CI tests;
for every pair of nodes $(X_i, X_j) \in FA$ do:
$\eta_{ij} = -\infty, \eta_{ji} = -\infty$;
3) Search phase;
a) NC times iterations;
for $l = 1$ to NC do:
i) for $k = 1$ to a do:
$G_k = AntConstructGraph()$;
if (l mod $l_{step} = 0$) then $G_k = Optimization(G_k)$;
ii) $G^+_{(l)} = \arg\max_k f(G_k : D)$;
iii) if $(f(G^+_{(l)} : D) \geq f(G_k : D))$ then $G^+ = G^+_{(l)}$;
iv)Perform global pheromone updating by (6);
b) Local optimization;
i) for $k = 1$ to a do: $G_k = Optimization(G_k)$;
ii) $G^+_{(l)} = \arg\max_k f(G_k : D)$;
iii) if $(f(G^+_{(l)} : D) \geq f(G_k : D))$ then $G^+ = G^+_{(l)}$;
4) Return G^+;
End Procedure

2.5 Algorithm analysis

The main cost of ACO algorithm is the computation of statistic factors, just as the other stochastic search algorithms for learning BNs. Each new search object needs to carry out new statistic count, thus each iteration of ants needs much more computing cost, and in the case of same sample capacity, the more the number of iterations, the more the computing cost. In contrast to ACO-B, I-ACO-B not only employs constraint knowledge to reduce the search space, but also takes it as a heuristic knowledge to induct the process of stochastic searches. The essence of the two improvement strategies is to decrease the computing cost of stochastic searches and improve the time performance of ants optimization by reducing the computing of statistic factors, scoring of structures, and the numbers of comparisons.

Theoretically, the more the constraint knowledge obtained by CI tests, the smaller the search space, and the higher the searching efficiency. However, the results of higher-order CI tests may be unreliable[2], and there is some extra computing cost even if for lower-order CI tests, e.g, the number of order-0 CI tests is C_n^2, the computing complexity is $O(n^2)$; the number of order-1 CI tests is $C_n^2 \cdot C_{n-2}^1$, the computing complexity is $O(n^3)$; the number of order-2 CI tests is $C_n^2 \cdot C_{n-2}^2$, the computing complexity is $O(n^4)$. Therefore, I-ACO-B only uses order-0 CI tests to reduce the search space. Even so, the experimental results show that the pruning based on order-0 CI tests may accidentally

delete a few candidate arcs in the optimal solution. Though there is no thicking step as that of constraint-based learning method[3], the local optimization (adding arcs operator) of I-ACO-B can also get the losing arcs back. In our experiments, when the confidence value of the χ^2 test was 99.5 %, the order-0 CI tests accidentally deleted an arc, which was subsequently put back to optimal solution by the local optimization process.

The convergence of ant colony optimization is problem-independent. Reference [21] used a theorem to prove the convergence of the ACS algorithm. For $P^*(t)$, the probability that the algorithm finds an optimal solution at least once within the first t iterations. The theorem pointed that the $P^*(t)$ is asymptotically close to 1 for a sufficiently large t, and that the theorem is not affected by the form of the heuristic information if we have $0 < \eta_{ij} < +\infty$ for each pair (i, j) and $\beta < +\infty$. Both ACO-B and I-ACO-B are ACO algorithms in the ACS formalism, and the heuristic function of I-ACO-B is weighted for the one of ACO-B. As the $K2$ scoring increase of ACO-B is larger than 0, and the weighted factor $\omega \geq 1$, both I-ACO-B and ACO-B satisfy the conditions by which the theorem holds in [21], consequently ensuring the convergence of I-ACO-B.

3 Experimental evaluation

To study the performance of I-ACO-B, we adopted the well-known benchmark data set–ALARM (http://www.cs.huji.ac.il/labs/compbio/Repository/) to conduct the following experiments and compared it with the original ACO-B. The experimental platform was a personal computer with Pentium 4, 2.8 GHz CPU, 512 M memory, and Windows XP. The algorithm was implemented by Java, the final experimental parameters were confirmed by large numbers of experiments. These parameters were set as follows: the confidence value of χ^2 tests is 99.5 %, $\alpha = 1$, $\beta = 2$, $\rho = \psi = 0.4$, $q_0 = 0.8$, $a = 10$, $NC = 100$, and $l_{step} = 20$.

3.1 Performance analysis of two strategies

We employed three algorithms to learn a BN structure from ALARM data sets with different sizes. The three algorithms were respectively the original ACO-B, an improved ACO-B1 (only adding order-0 CI tests), and another improved ACO-B2 (only using the new heuristic function). The experimental results are shown in Table 1, where $K2$ denotes the $K2$ metric values for the solutions obtained for different sample capacities, $It.$ is the smallest number of iterations when the algorithm finds an optimal structure, time is the execution time when the algorithm finds an optimal structure, and $\mu \pm \sigma$ indicates the mean μ and the standard deviation σ over 10 executions independently carried out by the corresponding algorithm. Moreover, numbers in parentheses of the $K2$ row are the best results found over 10 executions, and numbers in parentheses of $It.$ and time rows are the smallest numbers of the iterations and the shortest running time when the best solutions were obtained.

By analyzing these data in Table 1, we can draw the following conclusions: 1) Except for the 1000 cases, the introduction of order-0 CI tests does not evidently affect the solution quality ($K2$ value) of the original algorithm, and the convergence performance (the number of the iterations and the running time) of ACO-B1 is significantly improved compared to ACO-B. Especially, the improvement of the running time is remarkable, which shows that the strategy can effectively enhance the time performance. The reasons are as follows: when the sample size is small (e.g., 1 000), the reliability for order-0 CI tests is not ensured, so some useful arcs may be accidentally deleted, and the solution quality is degraded. Despite that order-0 CI tests spend some extra time, the search space is greatly reduced. As the time saved during searching is much longer than the time increased in CI testing (especially when the sample size is large), the strategy of order-0 CI tests can improve

Table 1 The influence of CI tests and new heuristic function on ACO-B

Sample		1 000	2 000	3 000	4 000	5 000	6 000
ACO-B	$K2$	$-5\,024.14 \pm 0.34$ ($-5\,023.28$)	$-9\,717.64 \pm 0.11$ ($-9\,717.46$)	$-14\,402.01 \pm 0.37$ ($-14\,401.29$)	$-19\,099.64 \pm 0.65$ ($-19\,098.41$)	$-23\,782.17 \pm 0.13$ ($-23\,781.98$)	$-28\,347.17 \pm 0.03$ ($-28\,347.11$)
	$It.$	75.20 ± 4.61 (79)	59.30 ± 6.96 (30)	72.10 ± 6.32 (61)	66.70 ± 6.07 (60)	72.30 ± 4.78 (48)	65.60 ± 4.71 (40)
	Time (s)	54.58 ± 2.10 (55.95)	95.71 ± 5.76 (67.06)	196.93 ± 9.35 (177.66)	247.79 ± 10.97 (256.45)	272.58 ± 10.67 (213.70)	315.59 ± 8.91 (265.58)
ACO-B1	$K2$	$-5\,025.54 \pm 0.09$ ($-5\,025.28$)	$-9\,717.47 \pm 0.00$ ($-9\,717.46$)	$14\,401.54 \pm 0.10$ ($-14\,401.29$)	$-19\,099.44 \pm 0.50$ ($-19\,098.41$)	$-23\,782.53 \pm 0.45$ ($-23\,782.06$)	$-28\,347.88 \pm 0.30$ ($-28\,347.11$)
	$It.$	56.00 ± 7.77 (60)	52.00 ± 5.33 (40)	56.00 ± 6.53 (20)	54.00 ± 3.06 (40)	52.00 ± 6.80 (20)	50.00 ± 7.45 (40)
	Time (s)	19.70 ± 2.03 (10.51)	31.80 ± 1.77 (28.42)	44.20 ± 2.81 (28.58)	56.17 ± 1.87 (49.83)	72.04 ± 4.71 (49.88)	83.74 ± 6.79 (77.95)
ACO-B2	$K2$	$-5\,024.16 \pm 0.58$ ($-5\,023.28$)	$-9\,717.57 \pm 0.08$ ($-9\,717.46$)	14401.88 ± 0.35 ($-14\,401.29$)	$-19\,099.26 \pm 0.69$ ($-19\,098.41$)	$-23\,782.72 \pm 0.52$ ($-23\,781.98$)	$-28\,347.67 \pm 0.34$ ($-28\,347.11$)
	$It.$	65.80 ± 8.39 (32)	50.30 ± 7.45 (38)	61.40 ± 5.42 (40)	62.50 ± 7.95 (20)	66.50 ± 5.06 (55)	46.90 ± 5.49 (20)
	Time (s)	48.44 ± 3.83 (33.89)	84.05 ± 5.77 (77.17)	144.92 ± 6.61 (117.77)	193.09 ± 11.33 (122.81)	248.16 ± 8.06 (235.95)	266.83 ± 14.19 (201.53)

the time performance of ACO-B. 2) The new heuristic function can improve the convergence performance while keeping the solution quality. The main reason is that the improvement of the heuristic ability increases the diversity of solutions and reduces the number of the iterations and the running time.

3.2 Comparing I-ACO-B with ACO-B on performances

Table 2 provides a summary of the performance comparison between I-ACO-B and ACO-B algorithms. The two different algorithms were independently executed 10 times for each data set, the figures were, therefore, an average of 10 trails. In Table 2, $A.$, $D.$ and $I.$ are used to denote the structure differences between the learned and the original network, namely, the number of arcs accidentally added ($A.$), deleted ($D.$) and inverted ($I.$), compared with the original network. Total number represents the total number of statistics evaluated for the instances of all local structures during the learning process, and practical number represents the number of statistics truly computed from data. Since we used the hashing techniques to cache the results, we avoided the necessity of recomputing previously calculated values and greatly saved the running time. The meanings of other items and data formats are the same as those of Table 1.

Compared with ACO-B, I-ACO-B can always find better or equally good network structures for all the data sets in terms of both $K2$ score and structure difference, so I-ACO-B can get a better solution quality. On the other hand, $It.$, time, and the practical number. are evidently reduced, thus the computation complexity of I-ACO-B is greatly improved.

3.3 Comparison of the convergence and time performance

We compared with the typical runs of I-ACO-B and ACO-B on the same data set by large number of experiments. For each algorithm, we measured the $K2$ score of the best-so-far solution averaged over 10 runs as the iteration proceed. Fig. 3 shows the curves of the convergence performance on Alarm data set with 3 000 cases. In Fig. 3, the abscissa depicts the number of iterations, and the Y-coordinate depicts the $K2$ score. We observed that I-ACO-B converges much faster than ACO-B, and gets the best $K2$ score ($-14\,402.2$) at about 40 iterations while ACO-B needs over 60 iterations.

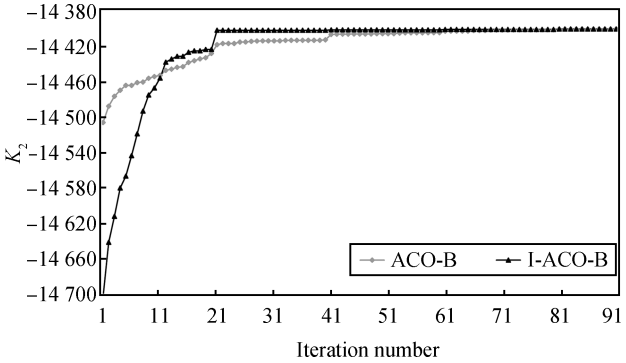


Fig. 3 Comparison of the convergence performances on 3 000 samples data for both algorithms

Table 2 The results for two algorithms on Alarm data with different capacities

Sample	Statistic	Algorithm	
		ACO-B	I-ACO-B
2 000	$K2$	$-9\,717.64 \pm 0.11$ ($-9\,717.46$)	$-9\,717.47 \pm 0.00$ ($-9\,717.46$)
	$A.$	3.20 ± 0.13 (3)	3.0 ± 0.0 (3)
	$D.$	1.0 ± 0.0 (1)	1.0 ± 0.0 (1)
	$I.$	1.6 ± 0.4 (1)	1.1 ± 0.1 (1)
	$It.$	59.3 ± 7.55 (30)	50.0 ± 7.45 (20)
	Time (s)	95.71 ± 5.76 (67.06)	27.91 ± 2.51 (18.80)
	Total number	79.14E05 $\pm$ 0.57 (79.66E05)	90.51E05 $\pm$ 0.70 (95.47E05)
	Practical number	$36\,108.2 \pm 174.66$ (36 615)	$9\,954.2 \pm 85.70$ (10 105)
3 000	$K2$	$-14\,402.01 \pm 0.36$ ($-14\,401.29$)	$-14\,401.66 \pm 0.10$ ($-14\,401.29$)
	$A.$	2.30 ± 0.33 (2)	1.4 ± 0.16 (2)
	$D.$	1.0 ± 0.00 (1)	1.0 ± 0.00 (1)
	$I.$	2.3 ± 0.3 (2)	1.4 ± 0.16 (2)
	It.	72.10 ± 6.32 (61)	50.00 ± 8.03 (20)
	Time (s)	196.93 ± 9.35 (177.66)	47.73 ± 4.53 (29.00)
	Total number	82.20005 $\pm$ 0.70 (83.60E05)	77.81E05 $\pm$ 0.81 (75.19E05)
	Practical number	$40\,200.9 \pm 311.24$ (40 490)	$11\,405.6 \pm 113.48$ (11 054)
4 000	$K2$	$-19\,099.64 \pm 0.65$ ($-19\,098.41$)	$-19\,099.53 \pm 0.46$ ($-19\,098.41$)
	$A.$	2.40 ± 0.22 (2)	1.9 ± 0.18 (2)
	$D.$	1.0 ± 0.00 (1)	1.0 ± 0.00 (1)
	$I.$	2.5 ± 0.43 (2)	1.8 ± 0.33 (2)
	It.	66.70 ± 6.07 (60)	54.00 ± 6.70 (40)
	Time (s)	247.79 ± 10.97 (256.45)	66.96 ± 4.37 (60.78)
	Total number	82.51E05 $\pm$ 1.58 (76.27E05)	84.06E05 $\pm$ 1.18 (83.44E05)
	Practical number	$40\,475.8 \pm 512.67$ (42 122)	$11\,961.6 \pm 64.50$ (12 091)

In Fig. 4, we compared the iteration numbers of two algorithms on 10 data sets with different sample capacities. For each algorithm, we recorded the iteration number averaged over 10 runs when obtaining the best $K2$ score on the respective data sets. Although we observed the iteration number was not specifically related with the sample capacity from Fig. 4, which illuminates the stochastic property of ACO, I-ACO-B was observed to perform better than ACO-B in terms of the iteration number of the final solution obtained over the whole scope of sample capacity.

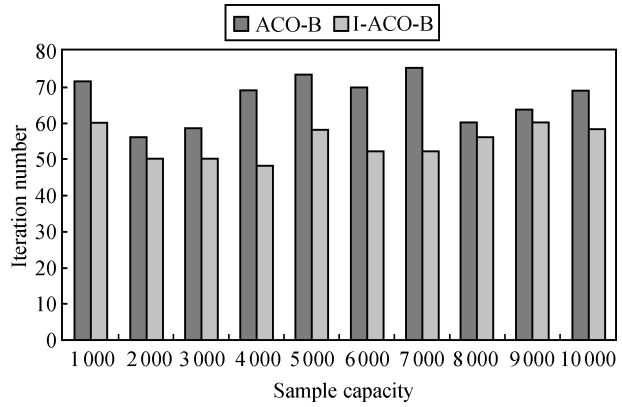


Fig. 4 Comparison of the iteration numbers on Alarm network for both algorithms

Fig. 5 gives the running time corresponding to Fig. 4. We can see that I-ACO-B performs better than ACO-B in terms of the running time on all data sets. Moreover, the advantage is very obvious when the data set is large, namely, the bigger the sample size, the more obvious the improvement. There are two reasons. One is that I-ACO-B takes the learned mutual information as a heuristic knowledge to revise the heuristic function, enhances the heuristic ability, and improves the time performance. The another reason is that I-ACO-B employs order-0 CI tests with less cost to effectively reduce the search space and cuts down many computing of statistic factors, scoring of structures, and comparisons of the solutions, thus greatly enhances the time performance. The experimental results also showed that the running time of ACO-B fast increases as the sample capacity increases, however I-ACO-B is not sensitive to the increase of the sample capacity. The fact that the running time of I-ACO-B increases slowly suggests that I-ACO-B is able to handle very large data sets and is a more promising algorithm for learning BNs.

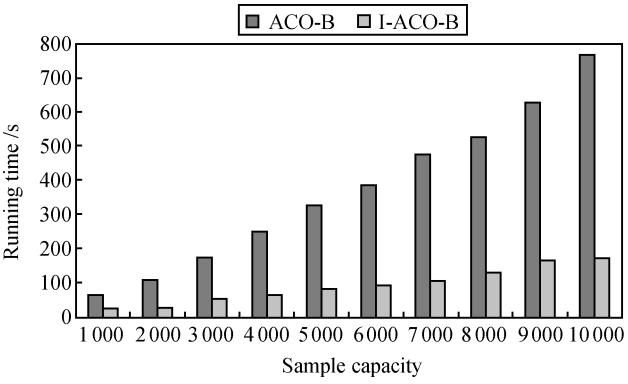


Fig. 5 Comparison of the time performances on Alarm network for both algorithms

Moreover, comparing the iteration numbers in Fig. 4 with the running time in Fig. 5 on different data sets, we can draw that the learning algorithms based on ACO cost less time on smaller data sets even if their convergence needs much more iterations. Just as mentioned in the algorithm analysis, the main cost of the ACO for learning BNs lies on computations on statistic factors. Each computing for a statistic factor needs to scan the sample data, thus the more the sample capacity is, the more time it costs. Therefore, the running time of the ACO for learning BNs is essentially related with the sample capacity.

4 Conclusion

Bayesian networks (BNs) are popular within the community of uncertainty in artificial intelligence. Nowadays, learning BNs from data is a research hotspot in data mining and machine learning. Based on ACO-B algorithm, this paper proposes a new algorithm, I-ACO-B, which combines the condition independence tests with ant colony optimization. I-ACO-B first employs order-0 independence tests to effectively restrict the space of candidate solutions, so that many unnecessary searches for ants can be avoided. Then, it merges the global score increase of a solution with the local mutual information between nodes, and introduces a new heuristic function with better heuristic ability to enhance the optimization efficiency. The empirical results illustrate that the new algorithm is superior both in terms of quality of the solutions and computational time in all data sets we have tested, especially, our algorithm is effective and efficient in large scale data sets, and greatly enhances the convergence speed compared to the original algorithm. Our future work is applying our algorithm to some real-life data mining problems and extending our study to some more complicated problems for learning BNs, e.g., problems with incomplete data, hidden variables, and multi-relational data.

References

1 Heckerman D. *A Tutorial on Learning Bayesian Networks*. Kluwer: Learning in Graphical Models, 1996. 301−354

2 de Campos L M, Huete J F. A new approach for learning belief networks using independence criteria. *International Journal of Approximate Reasoning*, 2000, **24**(1): 11−37

3 Cheng J, Greiner R, Kelly J, Bell D, Liu W. Learning Bayesian networks from data: an information theory based approach. *Artificial Intelligence*, 2002, **137**(1-2): 43−90

4 Suzuki J. Learning Bayesian belief networks based on the minimum description length principle: basic properties. *IEICE Transactions on Fundamentals*, 1999, **82**(10): 2237−2245

5 Liu Da-You, Wang Fei, Lu Yi-Nan, Xue Wan-Xin, Wang Song-Xi. Research on learning Bayesian network structure based on genetic algorithms. *Journal of Computer Research and Development*, 2001, **38**(8): 916−922 (in Chinese)

6 de Campos L M, Fernandez-Luna J M, Gamez J A, Puerta J M. Ant colony optimization for learning Bayesian networks. *International Journal of Approximate Reasoning*, 2002, **31**(3): 291−311

7 Larranaga P, Poza M, Yurramendi Y, Murga R H, Kuijpers C M H. Structure learning of Bayesian networks by genetic algorithms: a performance analysis of control parameters. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 1996, **18**(9): 912−926

8 Wong M L, Lam W, Leung K S. Using evolutionary programming and minimum description length principle for data mining of Bayesian networks. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 1999, **21**(2): 174−178

9 José A G, José M P. Searching for the best elimination sequence in Bayesian networks by using ant colony optimization. *Pattern Recognition Letters*, 2002, **23**(1-3): 261−277

10 Qiang Lei, Xiao Tian-Yuan, Qiao Gui-Xiu. An improved Bayesian networks learning algorithm. *Journal of Computer Research and Development*, 2002, **39**(10): 1221−1226 (in Chinese)

11 Wong M L, Lee S Y, Leung K S. A hybrid approach to discover Bayesian networks from databases using evolutionary programming. In: Proceedings of IEEE International Conference on Data Mining. Japan: IEEE, 2002. 498−505

12 Ji Jun-Zhong, Liu Chun-Nian, Yan Jing. A fast Bayesian network structure learning algorithm. *Journal of Computer Research and Development*, 2007, **44**(3): 412−419 (in Chinese)

13 Wong M L, Leung K S. An efficient data mining method for learning Bayesian networks using an evolutionary algorithm-based hybrid approach. *IEEE Transactions on Evolutionary Computation*, 2004, **8**(4): 378−404

14 Mantin J, Jan N. A simulated annealing-based method for learning Bayesian networks from statistical data: research articles. *International Journal of Intelligent Systems*, 2006, **21**(3): 335−348

15 Tsamardinos I, Brown L E, Alieris C F. The max-min hill-climbing Bayesian network structure learning algorithm. *Machine Learning*, 2006, **65**(1): 31−78

16 Alcobe J R. Incremental hill-climbing search applied to Bayesian network structure learning. In: Proceedings of the 15th European Conference on Machine Learning. Pisa, Italy: IEEE, 2004. 1−10

17 Dorigo M, Maniezzo V, Colorni A. The ant system: optimization by a colony of cooperating agents. *IEEE Transactions on Systems, Man, and Cybernetics, Part B: Cybernetics*, 1996, **26**(1): 29−41

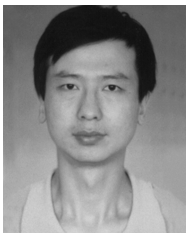
18 Dorigo M, Gambardella L M. Ant colony system: a cooperative learning approach to the traveling salesman problem. *IEEE Transactions on Evolutionary Computation*, 1997, **1**(1): 53−66

19 Blum C, Dorigo M. Search bias in ant colony optimization: on the role of competition-balanced systems. *IEEE Transactions on Evolutionary Computation*, 2005, **9**(2): 159−174

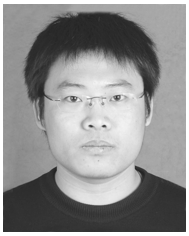
20 Blum C, Dorigo M. The hyper-cube framework for ant colony optimization. *IEEE Transactions on Systems, Man, and Cybernetics, Part B: Cybernetics*, 2004, **34**(2): 1161−1172

21 Stutzle T, Dorigo M. A short convergence proof for a class of ant colony optimization algorithms. *IEEE Transactions on Evolutionary Computation*, 2002, **6**(4): 358−365

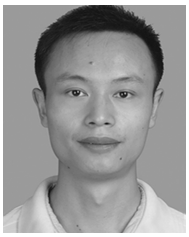
22 Dorigo M, Birattari M, Stutzle T. Ant colony optimization: artificial ants as a computational intelligence technique. *IEEE Computational Intelligence Magazine*, 2006, **1**(4): 28−39



JI Jun-Zhong Professor. He received his Ph. D. degree from Beijing University of Technology. His research interests covers machine learning, web intelligence, and computation intelligence. Corresponding author of this paper.
E-mail: jjz01@bjut.edu.cn



ZHANG Hong-Xun Master student. His research interest covers data mining and web intelligence.
E-mail: zhanghx@emails.bjut.edu.cn



HU Ren-Bing Master student. His research interest covers data mining and web intelligence.
E-mail: hurenbing@emails.bjut.edu.cn

LIU Chun-Nian Professor. His research interest covers data mining, inductive logic programming, constraint programming, machine learning, and soft computing. E-mail: ai@bjut.edu.cn