

BNC-PSO: Structure Learning of Bayesian Networks by Particle Swarm Optimization

Abstract Structure learning is a very important problem in the field of Bayesian networks (BNs). It is also an active research area for more than two decades; therefore, many approaches have been proposed in order to find an optimal structure based on training samples. In this paper, a Particle Swarm Optimization (PSO)-based algorithm is proposed to solve the BN structure learning problem; named BNC-PSO (**B**ayesian **N**etwork **C**onstruction algorithm using **PSO**). Edge inserting/deleting is employed in the algorithm to make the particles have the ability to achieve the optimal solution, while a cycle removing procedure is used to prevent the generation of invalid solutions. Then, the theorem of Markov chain is used to prove the global convergence of our proposed algorithm. Finally, some experiments are designed to evaluate the performance of the proposed PSO-based algorithm. Experimental results indicate that BNC-PSO is worthy of being studied in the field of BNs construction. Meanwhile, it can significantly increase nearly 15% in the scoring metric values, comparing with other optimization-based algorithms.

BNC-PSO: Structure Learning of Bayesian Networks by Particle Swarm Optimization

S. Gheisari

Department of Computer, Science and Research
Branch, Islamic Azad University, Tehran, Iran.

S.gheisari@srbiau.ac.ir

M.R. Meybodi

Computer Engineering and Information Technology
Department, Amirkabir University of Technology,
Tehran, Iran.

mmeybodi@aut.ac.ir

Abstract Structure learning is a very important problem in the field of Bayesian networks (BNs). It is also an active research area for more than two decades; therefore, many approaches have been proposed in order to find an optimal structure based on training samples. In this paper, a Particle Swarm Optimization (PSO)-based algorithm is proposed to solve the BN structure learning problem; named BNC-PSO (**B**ayesian **N**etwork **C**onstruction algorithm using **PSO**). Edge inserting/deleting is employed in the algorithm to make the particles have the ability to achieve the optimal solution, while a cycle removing procedure is used to prevent the generation of invalid solutions. Then, the theorem of Markov chain is used to prove the global convergence of our proposed algorithm. Finally, some experiments are designed to evaluate the performance of the proposed PSO-based algorithm. Experimental results indicate that BNC-PSO is worthy of being studied in the field of BNs construction. Meanwhile, it can significantly increase nearly 15% in the scoring metric values, comparing with other optimization-based algorithms.

Keywords Bayesian Information Criteria (BIC); Bayesian Network (BN); Cross over operation; Mutation operation; Particle Swarm Optimization (PSO); structure learning

1 Introduction

Bayesian networks (BNs) are popular within the AI probability and uncertainty community as a method of reasoning under uncertainty [48]. From an informal perspective, A BN is a directed acyclic graph, in which nodes represent random variables, and existence or lack of the arcs represents the dependence relationships between the variables. The relations are further quantified by a set of conditional probability distributions, one for each variable conditioning on its parents. Overall, a BN represents a joint probability distribution over a set of random variables; and provides an efficient device for performing probabilistic inference.

Learning the structure of a BN from data is an important challenge and has been studied extensively during two last decades. It is an *NP-hard* problem [9, 30, 54]; so, inferring complete causal models (i.e., causal BNs) is essentially impossible in large-scale data mining applications with thousands of variables [61].

Generally, there are three main approaches for learning BNs from data: scored-based learning, constraint-based learning, and hybrid methods. Algorithms following the first approach evaluate the quality of BNs structures using a scoring function and select the one with the best score [2, 13, 32]. These methods consider the structure learning problem as a combinatorial optimization problem. However they work well for small datasets, they may fail to find optimal solutions for large datasets. Second group of algorithms typically use statistical tests to identify conditional independence relations from data and build a BN structure that best fits those independence relations [7, 20, 23, 64, 81, 89]. They rely on the results of local statistical tests, so they can often scale to large datasets; however, they are sensitive to the accuracy of the statistical tests, and if there are insufficient or noisy data they may badly work. In comparison, score-based algorithms work well even for datasets with relatively few data points. Finally, hybrid algorithms integrate previous two approaches and use combinations of the score-based and the

constraint-based algorithms to solve the structure learning problem [1, 17, 69, 84]. One popular strategy is to use constraint-based learning to create a skeleton graph and then use score-based learning to find a high-scoring network structure. For further information about BN structure learning, please refer to [45, 47].

Recently, Particle Swarm Optimization (PSO) has been successfully applied in many researches and application areas; structure learning of the BNs is one of these applications. However, the classical PSO only operates in continuous and real-valued space, and some methods, which make it into discrete space to apply it for learning of the BNs, are necessary. In [36, 90] an alphabetic sequence has been used to represent a candidate BN (a particle in PSO), and then, some rules have been defined to make the computations of the velocity and the position updating. These methods have also been applied to dynamic BNs learning [91]. In [51] a memory binary PSO has been introduced to prevent and overcome premature convergence for BNs learning. In [87] a binary encoding scheme has been used to represent a BN; and two modifications of particle moving operations have been proposed. One is the velocity updating rule based on the binary representation, and the other is the position updating rule based on stochastic mutation operation. In [22] a PSO-based approach has been used to feature selection for filtering the irrelevant attributes of the dataset, resulting in a fine BN built with the K2 algorithm. In [50] combining the immune theory in biology with PSO has been studied; but the actual method of BNs structure learning in the article is not clear.

In this paper, we propose a score-based learning algorithm, which uses the PSO principles and called BNC-PSO. Two novel formulas for velocity and position updating are also proposed; these formulas, which are completely new, use stochastic mutation and crossover operations. By presenting these two formulas, we combine PSO with Genetic Algorithm. The structure of the paper is as follows. In section 2, we explain the BN preliminaries and review the problem of learning optimal networks focusing on the score-based approach; related works are also reviewed in this section. Particle Swarm Optimization (PSO) is briefly introduced in section 3. In section 4, we describe that how PSO can be used for structure learning of BNs; complexity analysis and convergence analysis of the proposed algorithm are also discussed in this section. Empirical results obtained with simulations are presented in section 5. Finally, discussion and conclusion are presented in section 6 and section 7, respectively.

2 Bayesian networks and score-based structure learning

In this section, firstly, we provide a brief summary of BNs, and then score-based structure learning preliminaries and its related works are discussed.

2.1 Bayesian networks

A BN is a directed acyclic graph (DAG), which describes the joint probability distribution over a set of random variables (X_1, \dots, X_n) , with defining a series of probability independences and a series of conditional independences [57]. A directed arc from X_i to X_j represents the dependence between two variables; and X_i is identified as a parent of X_j . We use $Pa(X_j)$ to stand for the parent set of X_j . The dependence relations between X_j and $Pa(X_j)$ are quantified using a conditional probability distribution, $P(X_j|Pa(X_j))$. The joint probability distribution represented by BN is factorized as the product of all conditional probability distributions in the network and it can be written according to Eq. (1).

$$P(X_1, \dots, X_n) = \prod_{i=1}^n P(x_i|Pa(X_i)) \quad (1)$$

After constructing a BN, it may be an efficient device to perform probabilistic inference; nevertheless, the problem of constructing the BNs still exists. Given a training dataset D , constructing a BN is the task of finding a network that best fits D , and it is usually considered as two learning subtasks: structure learning and parameter learning. Structure learning determines the topology of the network, while parameter learning defines the numerical parameters (conditional probabilities) for a given network topology. In this work, we focus on structure learning and use the score-based approach, which is

introduced in previous section. Algorithms, which are following this approach, have two main components: a scoring function and a search procedure. While a scoring function evaluates the quality of a BN structure, search procedure determines an algorithm to search throughout all possible networks and finds the structure that optimizes the score. In the rest of this section, we briefly describe each of these two components.

2.2 Scoring function

Usually, several scoring functions have been considered to measure the quality of the constructed BNs. They use the variety of metrics such as *Bayesian* metric, *Minimum Description Length* metric, and *Bayesian Information Criterion*, to mention a few. *Bayesian* metric measures the quality of a BN by computing a *marginal likelihood* of the BN with respect to the given data and inherent uncertainties [27, 66]. *Minimum Description Length* metric is based on the assumption that the number of regularities in the data, encoded by a model, is somehow proportional to the amount of data compression allowed by the model. The *Bayesian Information Criterion* (BIC) is a criterion for model selection among a finite set of models, and it is based on *likelihood function*. Since *BIC* metric is used in this study, below it is explained with more detail.

Given a training dataset D , of N samples each consisting of n variables, the *likelihood function* for a graph structure is given in Eq. (2).

$$LL(D) = \prod_{i=1}^n \prod_{j=1}^{q_i} \frac{\Gamma(\sum_{k=1}^{r_i} \alpha_{ijk})}{\Gamma(N_{ij} + \sum_{k=1}^{r_i} \alpha_{ijk})} \prod_{k=1}^{r_i} \frac{\Gamma(N_{ijk} + \alpha_{ijk})}{\Gamma(\alpha_{ijk})} \quad (2)$$

Where, N_{ij} is the number of samples in the dataset that have the j th value combination for the parent of the i th variable, and likewise, N_{ijk} is the number of samples, in which the i th variable is in its k th state and its parents are in their j th configuration. The number of different states of the i th variable is also given by r_i and the number of possible configurations for its parents is given by q_i . $\Gamma(v)$ is the Gamma function, which for $v \in \mathbb{N}$ is given by $\Gamma(v) = (v - 1)!$, and α_{ijk} is Dirichlet distribution parameter. Using Eq. (2), which is known as *Cooper Herzkovitz likelihood* [13] has some practical difficulties; therefore, the *log likelihood* (Eq. (3)), which is more practical, is used.

$$LL(D) = \sum_{i=1}^n \sum_{k=1}^{r_i} \sum_{j=1}^{q_i} N_{ijk} \ln \frac{N_{ijk}}{N_{ij}} \quad (3)$$

However Eq. (3) is more feasible than Eq. (2), it still cannot be directly used as a scoring function; because it will favour complex graphs with many edges. To favour more simple graphs, a penalization is usually used and finally *BIC* [76] is defined as follow:

$$BIC(D) = LL(D) - \frac{1}{2} (\ln N) |\theta| \quad (4)$$

where, $|\theta| = \sum_{i=1}^n q_i(r_i - 1)$ is the number of required parameters to specify the model of a BN. *Akaike's Information Criterion* (AIC) [3] is another scoring metric, which replaces $\frac{1}{2} (\ln N)$ by N . Any other decomposable scoring functions can also be used instead without affecting the search procedure.

2.3 Search procedure

Most of the score-based algorithms search the space of possible DAGs, which represent feasible BN structures. The number of possible DAGs for n variables is given by the following recursive function [74]:

$$f(n) = \sum_{i=1}^n (-1)^{i+1} \binom{n}{i} 2^{i(n-i)} f(n-i), \quad f(0) = f(1) = 1 \quad (5)$$

It is obvious that searching this huge space for the optimal structure is NP-hard; therefore, a simple greedy algorithm is usually used to build the network. The greedy algorithm adds an edge with the greatest improvement on the current network quality in search step until no more improvement is possible. The initial network structure can be a graph with no edges, or it can take advantage of using the prior information; for example, the best tree, which is computed by the polynomial-time maximum branching algorithm [33, 65, 66]. Extensions to this approach include tabu search with random restarts [30], limiting the number of parents or parameters of each variable [26], searching in the space of equivalence classes [10], searching in the space of variable orderings [83], and searching under the constraints extracted from data [84]. Finally, the optimal reinsertion algorithm (OR), which has been proposed in [56], adds a different operator: a variable is removed from the network; its optimal parents are selected, and the variable is then reinserted into the network with those parents. The parents are selected to ensure that the new network is still a valid BN.

Also, stochastic search and learning methods such as, Markov Chain Mont Carlo, Quantum Annealing and Simulated Annealing, and Learning Automata have been applied [19, 29, 32, 37, 59, 61]. These methods explore the solution space using non-deterministic transition between neighbouring network structures, while favouring better solutions. Furthermore, in order to escape local optima and find the best solution, the stochastic moves are used.

Other optimization methods such as Particle Swarm Optimization [36, 51, 87, 90, 91], Genetic Algorithms [38, 48], and Ant Colony Optimization [16, 21, 75] have been applied to learning BN structure as well. These population-based methods maintain a set of candidate solutions throughout their search, and at each step, they create the next generation of the solutions randomly. Since BNs are the acyclic graphs, after each operation, the graph structure must be validated, and all cycles must be removed from the constructed graph.

Moreover, a shortest path perspective has been introduced in [92, 93], which uses an A* algorithm to solve the problem of BNs structure learning.

There are also multiple exact algorithms, which have been developed for structure learning of the BNs. Several dynamic programming algorithms have been proposed, which can find an optimal BN in $O(n2^n)$ in time and space [43, 44, 79, 80]. Furthermore, a branch and bound algorithm has been proposed in [18], and the Integer Linear Programming (ILP) has been used in [14, 35, 39].

In this paper, PSO, which is an optimization method, is used to search the space of possible DAGs. It will be explained in next section.

3. Particle Swarm Optimization (PSO)

Particle Swarm Optimization is a swarm intelligence method, which considers a swarm containing p particles in a D -dimensional continuous solution space. Each i th particle has its own position and velocity. Assuming that the search space is D -dimensional, the position of the i th particle is denoted as a D -dimensional vector: $X_i = (X_{i1}, X_{i2}, \dots, X_{iD})$ and the best particle in the swarm is denoted as P_g , the best previous position of the i th particle is recorded and represented as $P_i = (P_{i1}, P_{i2}, \dots, P_{iD})$, while the velocity for the i th particle can be defined by another D -dimensional vector: $V_i = (V_{i1}, V_{i2}, \dots, V_{iD})$. According to the definitions, the particle position and velocity can be manipulated according to the following equations:

$$V_i^t = w \times V_i^{t-1} + c_1 r_1 (P_g^{t-1} - X_i^{t-1}) + c_2 r_2 (P_g^{t-1} - X_i^{t-1}); \quad (6)$$

$$X_i^t = X_i^{t-1} + V_i^t. \quad (7)$$

Where w is the inertia weight; c_1 and c_2 are acceleration coefficients; r_1 and r_2 are random numbers on the interval $[0,1]$.

The PSO approach utilizes a cooperative swarm of particles, where each particle represents a candidate solution to the problem, to explore the space of possible solutions to the optimization problem of interest. A population of particles with random positions and velocities is initialized and then *fly*, evaluating the fitness at each step of optimization. The fitness function can be defined in different formula according to different real applications. In this study, the *BIC* in Eq. (4) is used. Each particle compares its current fitness value with the fitness value of its previous best position P_i ; if current value is better, then updates P_i with the current value and position. The particle also compares its fitness value with the fitness value of the global best position P_g , and if it is better, then P_g is updated with current value and position of the current particle. The velocity and position of each particle are updated according to Esq. (6,7). If a predefined stopping criterion is met, then P_g and its fitness value is returned, else evaluation step is done.

The classical version of the PSO algorithm operates in continuous search spaces. In order to solve optimization problems in discrete search spaces, several binary discrete PSO algorithms have been proposed. In a binary discrete space, the position of a particle is represented by an N -length bit string and the movement of the particle consists of flipping some of these bits. The first binary version of PSO was introduced in [41]. Two other typical discrete PSO algorithms also exist; discrete PSO algorithm for the travelling salesman problem (TSP) [12], and discrete PSO algorithm for the permutation flow-shop sequencing problem with make-span criteria in [62].

As a swarm-based evolutionary method, the PSO, which was introduced in [24], has been proved to be a powerful optimization tool. The advantages of the PSO over many other optimization algorithms are its simple implementation, and the ability to converge to a reasonably good solution quickly. In the past several years, PSO has been successfully applied in many researches and application areas such as, design of multi-machine power system stabilizers [88], solving non-convex economic dispatch problem in power systems [70], solving optimal power flow problem [86], optimizing the advanced manufacturing process parameters [53], predicting uncertain behaviour and performance analysis of the pulping system in a paper industry [28], optimal tuning of controllers in the power electric industry [4], solving multi-objective reactive power optimization problem [8], and optimal economic operation method for islanded micro-grid [52]. It has been demonstrated that PSO can get better results in a faster and cheaper way compared with other methods.

In this paper, an efficient PSO-based algorithm is designed to deal with the problem of learning BN structure.

4. Proposed PSO-based algorithm for structure learning of BN

In this section, firstly, we introduced our proposed structure learning algorithm, BNC-PSO, and then its complexity and convergence are analyzed.

4.1 Notation and representation

The proposed algorithm represents a candidate BN as a list of DAG edges. For a fixed domain with n random variables and the alphabet $S=\{0,1\}$, a BN structure can be represented by an $n \times n$ connectivity matrix A , where its elements, a_{ij} , verify:

$$a_{ij} = \begin{cases} 1 & \text{if } i \text{ is a parent of } j; \\ 0 & \text{otherwise.} \end{cases}$$

So we represent a particle by string: $a_{11}a_{12} \dots a_{1n}a_{21}a_{22} \dots a_{n1}a_{nn}$, similar to [48].

4.2 Update formulas of the particles

Since the BN structure learning problem is a discrete problem, we employ the novel discrete velocity and position updating method based on genetic operations. The update formula of the particle is represented as:

$$X_i^t = N_3(N_2(N_1(X_i^{t-1}, w), c_1), c_2) \quad (8)$$

where w is an inertia weight; c_1 and c_2 are acceleration constants; N_1 denotes the mutation operation, and finally, N_2 and N_3 denote the crossover operations. Following Esq., numbered (9-11), describe the mutation and the crossover operations. In these Esq. r_1 , r_2 , and r_3 are random numbers on the interval $[0,1)$. The velocity of the particles is updated using N_1 , which is presented in Eq. (9).

$$W_i^t = N_1(X_i^{t-1}, w) = \begin{cases} M(X_i^{t-1}), & r_1 < w \\ X_i^{t-1}, & \text{others} \end{cases} \quad (9)$$

Where w denotes the mutation probability. In the process of the particle mutation, we randomly select one edge of the particle to be muted. Algorithm1 illustrates the pseudo code of mutation operator.

Algorithm 1. Mutation operator

```

1: Input: Particle  $p$ 
2: Output: New particle  $p_{new}$ 
3: while true do
4:    $r=random(1, n \times n); //n is the number of random variables and n \times n is the size of particles$ 
5:    $p_{new}=inverse(r); //if r is equal to one, transform it to zero and vice versa.$ 
6:   if !colored-DFS( $p_{new}$ ); // to check the existence of cycles
7:     break;
10: end while
```

The N_2 , which is the cognitive personal experience of the particles, can be written as:

$$S_i^t = N_2(W_i^t, c_1) = \begin{cases} C_p(W_i^t), & r_2 < c_1 \\ W_i^t, & \text{others} \end{cases} \quad (10)$$

where c_1 denotes the crossover probability of the particle with the personal optimal solution. In the process of the particle crossover with the personal optimal particle, we will generate the new particle, which composes of the common portion between the two particles and the random portions from the two particles. The pseudo code of crossover operator is given in Algorithm 2.

Algorithm 2. Crossover operator

```

1: Input: Particle  $p$  and particle  $q$ 
2: Output: New particle  $p_{new}$ 
3: while true do
4:    $i=1;$ 
5:   while  $i \leq n \times n$  do
6:     if  $p[i]=q[i]=1$ 
7:        $p_{new}[i]=1;$ 
8:     else
9:        $r=random(p,q); // randomly select one particle$ 
10:       $p_{new}[i]=r[i];$ 
11:    end if
12:   end while
12:   if !colored-DFS( $p_{new}$ ); // to check the existence of cycles
13:     break;
16: end while
```

The N_3 , which is the cooperative global experience of particles can be written as:

$$X_i^t = N_3(S_i^t, c_2) = \begin{cases} C_g(S_i^t), & r_3 < c_2 \\ S_i^t, & \text{others} \end{cases} \quad (11)$$

where c_2 denotes the crossover probability of the particle with the global optimal solution. The process of the particle crossover with the global optimal particle is same to the process of particle crossover with the personal optimal particle, and we no longer repeat description.

In the process of building a BN, the update operations take into account as the inserting/deleting the edges. The introduction of edge inserting/deleting might lead to appear a loop and generate the invalid solution in the iterative process, which destroys the soundness of the particle encoding. In order to detect and remove cycles, the search procedure uses a modified version of depth first search algorithm (DFS), named coloured-DFS. In the edge removing process, after detecting all back edges, the search procedure can remove all if it is necessary. This algorithm runs after three update operations, which are described in this section (please see line 6 of Algorithm1 and line 12 of Algorithm2).

4.3. Parameter setting

Property1 the setting of inertia weight affects the balance between local search ability and global search ability of the particles.

As we can see from the velocity update formula, the first part provides the flight impetus of the particle in search space and represents the effect of the previous velocity on the flight trajectory. Thus, inertia weight is a numerical value, which indicates the extent of such influence.

Property2 a larger inertia weight will make the algorithm has strong global search ability.

Property1 and Eq. 6 show that inertia weight decides how much previous velocity will be preserved. Thus, a larger inertia weight can strengthen the capability of searching the unreach area. It is conductive to enhance the global search ability of the algorithm and jump out of the local optima. A smaller inertia weight suggests that the algorithm mainly search near the current solution. It is conductive to enhance the local search ability and accelerate convergence.

In [77], the researchers have proposed a PSO algorithm based on linear decreasing inertia weight. In order to ensure a stronger global search, they employed a larger inertia weight early in the program, and a smaller one in the later stages to guarantee the local search. Simulation on four kinds of different benchmark functions showed that such strategy of parameters setting actually improved the performance of PSO.

Property3 larger acceleration coefficients c_1 may cause wandering in local scope. Larger acceleration coefficients c_2 may make the algorithm prematurely converge on a local optimal solution.

Acceleration coefficients c_1 and c_2 are used in communicating between particles. In [73] a kind of strategy was proposed, which employs a larger c_1 and a smaller c_2 in the early phases and the opposite in the later. In this way, the algorithm will guarantee detailed search in local scope, not have to directly move to the position of global optimal in early phases, and speed up convergence in the later stages. Similarly, the experiments achieved great results.

Based on the above analysis and the obtained experimental results in sensitivity analysis, in our proposed algorithm, parameters are set as follows: population size is 50, w decreases linearly from 0.9 to 0.35 according to Eq. (12), c_1 decreases linearly from 0.84 to 0.52 according to Eq. (13), and finally, c_2 increases linearly from 0.38 to 0.81 according to Eq. (14).

$$w = w_{start} - \frac{w_{start}-w_{end}}{evaluations} \times eval \quad (12)$$

$$c_1 = c_{1_start} - \frac{c_{1_start}-c_{1_end}}{evaluations} \times eval \quad (13)$$

$$c_2 = c_{2_start} - \frac{c_{2_start}-c_{2_end}}{evaluations} \times eval \quad (14)$$

Where, *eval* represents the current iteration number and *evaluations* represents the maximum number of iterations.

4.4 Procedure of the proposed algorithm

Using the concepts and the basics, which are mentioned above, the detail procedure of the proposed algorithm, BNC-PSO can be summarized as follows:

Step 1: Randomly generate the initial valid population.

Step 2: Calculate the fitness value of each particle according to *BIC* using Eq.(4).

Step 3: Update the personal optimal solution of each particle.

Step 4: Adjust the position and velocity of each particle according to Esq. (8-11). Update and validate the new particle.

Step 5: Recalculate the *BIC* value for each particle.

Step 6: Update the global optimal solution of the population.

Step 7: check the termination condition (a good enough position or the maximum number of iterations is reached); if it has fulfilled, the run is terminated and output the global optimal solution, i.e., the final solution. Otherwise, go to Step 3.

4.5 Complexity analysis

Lemma1 Assume that the population size is *p*, the number of iterations is *iters* and the number of random variables is *n*. The time complexity of the proposed algorithm is $O(\text{iter} \times p \times n^2)$.

Proof The inner loop of the proposed algorithm, from Step 3 to Step 6, includes mutation, crossover and *BIC* computing. In the mutation and crossover operations, the storing steps determine those operations' time complexity, $O(n \log n)$, because the time of cycle removing is just more than linear. Besides, the time complexity of computing the *BIC* is $O(Mn^2)$, where *M* is the number of training samples. Therefore, the complexity of the inner loop is $O(n \log n + n^2) = O(n^2)$. The outer loop is related to the number of particles *p* and the number of iterations *iters*, consequently, the time complexity of the proposed algorithm is $O(\text{iter} \times p \times n^2)$.

4.6 Convergence analysis

Definition 1 (*Finite Markov Chain*) Let X , ($X = X_k, k = 1, 2, \dots$) be the stochastic process of discrete parameters defined in probability space (Ω, F, P) over a finite state space S . If X has Markov properties, i.e., for any non-negative integer *k* and state $i_0, i_1, \dots, i_{k-1} \in S$, then X is a finite Markov chain when,

$$P(X_{k+1} = i_{k+1} | X_0 = i_0, X_1 = i_1, \dots, X_k = i_k) = P(X_{k+1} = i_{k+1} | X_k = i_k) \quad (15)$$

$$P(X_0 = i_0, X_1 = i_1, \dots, X_k = i_k) > 0.$$

$P(X_{k+m} = j | X_k = i)$ is called m -step transition probability of X , which is the conditional probability of process from state i at time k , after the m th step, to state j at time $k + m$, denoted as $P_{ij}(k, k + m)$. For $i, j \in S$, if $P_{ij}(k, k + 1)$, P_{ij} for short, does not depend on the time k , the Markov chain is said to be homogenous. $P = [P_{ij}]$ is called transition matrix with P_{ij} as the element of i th row and j th column. The long-term behaviour of the homogeneous finite Markov chain is completely determined by its initial distribution and first step transition probability.

Theorem1 *The Markov chain of BNC-PSO is finite and homogeneous.*

Proof In its global search, BNC-PSO obtains the global and individual optimum through updating the positions of the particles by stochastic mutation and crossover operators. Judging from the process of global searching, the generation of a new population depends on the current population. Thus, the conditional probability of the search process, from a state to a certain specific state, satisfies Eq. (15). That means, it satisfies the property of Markov. Therefore, the Markov chain of BNC-PSO is finite and homogeneous. In this algorithm the set constituted of all the populations $\{p_1, p_2, \dots, p_m\}$ is finite. That is, events occurring at time $k = 0, 1, \dots$ all belong to a finite countable event aggregate; thus, its Markov chain is finite as well. Hence, the analysis theories and methods of Markov chain can be directly applied to the analysis of this algorithm.

Theorem2 *Transition probability matrix of the Markov chain, made up of BNC-PSO, is positive definite.*

Proof While searching, the population transits from state $i_i \in S$ to state $i_j \in S$, through mutation operator and crossover operators with the global optimum and the individual optimum. The transition probabilities of these three operators are m_{ij} , g_{ij} , p_{ij} , respectively. And the stochastic matrixes, what they consist of, are $M = \{m_{ij}\}$, $G = \{g_{ij}\}$, $D = \{d_{ij}\}$, respectively, let $P = MGD$, then $m_{ij} > 0$, $\sum_{i_j \in E} m_{ij} = 1$; $g_{ij} \geq 0$, $\sum_{i_j \in E} g_{ij} = 1$; $d_{ij} \geq 0$, $\sum_{i_j \in E} d_{ij} = 1$.

Therefore, M , G , D are all stochastic and M is a positive definite. We can prove that P is a positive definite too. Let $B = GD$; for $\forall i_i \in S, i_j \in S$, we have $b_{ij} = \sum_{\lambda_k \in E} g_{ik} d_{kj} \geq 0$, then $\sum_{\lambda_j \in E} b_{ij} = \sum_{\lambda_j \in E} \sum_{\lambda_k \in E} g_{ik} d_{kj} = \sum_{\lambda_k \in E} g_{ik} \sum_{\lambda_j \in E} d_{kj} = \sum_{\lambda_k \in E} g_{ik} = 1$.

Hence, B is a stochastic matrix, similarly, we get $d_{ij} = \sum_{\lambda_k \in E} b_{ik} m_{kj} > 0$.

Theorem3 (Limit theorem for Markov chain) *Assume that P is a positive stochastic transition matrix of definite homogeneous Markov chain, then:*

- (1) *There exists a unique probability vector $\bar{P}^T > 0$, which satisfies $\bar{P}^T P = \bar{P}^T$.*
- (2) *For any initial state i (e_i^T as its corresponding initial probability), we get $\lim_{k \rightarrow \infty} e_i^T P^k = \bar{P}^T$.*
- (3) *Limit probability matrix $\lim_{k \rightarrow \infty} P^k = \bar{P}$, where \bar{P} is a $n \times n$ stochastic matrix, and all its rows equal to \bar{P}^T .*

The limit theorem explains that the long-term probability of Markov chain does not depend on its initial states. This theorem is the basis for the convergence of an algorithm.

Lemma2 *If mutation probability $m > 0$, the algorithm is an ergodic irreducible Markov chain, which has only one limited distribution and nothing to do with the initial distribution; moreover, the probability at a random time and random state is greater than zero.*

Proof At the t th time, the j th state probability distribution of population $X(t)$ is:

$$P_j(t) = \sum_{i \in S} P_i(1)P_{ij}^{(t)}, \quad t = 1, 2, \dots \quad (16)$$

According to Theorem3, we can get the formulation as following:

$$P_j(\infty) = \lim_{t \rightarrow \infty} \left(\sum_{i \in S} P_i(1)P_{ij}^{(t)} \right) = \sum_{i \in S} P_i(1)P_{ij}^{(\infty)} > 0, \forall j \in S. \quad (17)$$

Definition2 Suppose a stochastic variant $Z_t = \max\{f(x_k^{(t)}(i)) | k = 1, 2, \dots, N\}$, which represents individual best fitness at the t th step and i th state of the population. Then, the algorithm converges to the global optimum, if and only if,

$$\lim_{t \rightarrow \infty} P\{Z_t = Z^*\} = 1; \quad (18)$$

where $Z^* = \max\{f(x) | x \in S\}$ represents the global optimum.

Theorem4 for any i and j , the time transiting of an ergodic Markov chain from the i th state to the j th state is limited.

Theorem5 BNC-PSO algorithm can converge to the global optimum.

Proof Suppose that $i \in S, Z_t < Z^*$ and $P_i(t)$ is the probability of BNC-PSO algorithm at i th state and t th step. Obviously, $P\{Z_t \neq Z^*\} \geq P_i(t)$; hence, we can know that $P\{Z_t = Z^*\} \leq 1 - P_i(t)$.

According to Lemma2, the i th state probability of the operator in BNC-PSO algorithm is $P_i(\infty) > 0$, then,

$$\lim_{t \rightarrow \infty} P\{Z_t = Z^*\} \leq 1 - P_i(\infty) < 1. \quad (19)$$

Observe a new population, such $X_t^+ = \{Z_t, X_t\}, t \geq 1, x_{ti} \in S$ denoting the search space (which is a finite set or a countable set), where Z_t , the same to that in Definition2, represents individual best fitness in current population, and X_t denotes the population during the search. As it is easy to prove that the group shift process $\{X_t^+, t \geq 1\}$ is still a homogeneous and ergodic Markov chain, we can know that,

$$\begin{aligned} P_j^+(t) &= \sum_{i \in S} P_i^+(1)P_{ij}^+(t); \\ P_{ij}^+ &> 0 (\forall i \in S, \forall j \in S_0); \\ P_{ij}^+ &= 0 (\forall i \in S, \forall j \notin S_0). \end{aligned} \quad (20)$$

So

$$\begin{aligned} (P_{ij}^+)^t &\rightarrow 0 (t \rightarrow \infty); \\ P_j^+(\infty) &\rightarrow 0 (j \notin S_0); \\ \lim_{t \rightarrow \infty} P\{Z_t = Z^*\} &= 1. \end{aligned} \quad (21)$$

5. Experimental results

In order to test the behaviour of BNC-PSO, several algorithms and networks are selected. All the algorithms have been implemented and executed in .Net framework in a PC, which has a single CPU of Intel(R) Core™ 2 Duo 3.33GHz and a 1GB memory.

5.1 Databases

To show the feasibility and flexibility of the proposed algorithm four well-known networks are selected. Selected networks are mainly obtained from real-decision support systems that cover different range of real-life applications, such as medicine. Selected networks are: ALARM [5], INSURANCE [6], ASIA [49], and BOBLO [72]. The ALARM is a medical diagnostic alarm message system for patient monitoring, and it contains 37 nodes and 46 arcs. The INSURANCE network, which contains 27 variables and 52 arcs, is a network for evaluating car insurance risks. ASIA network is a simple network with eight binary nodes and eight arcs. It was introduced by Lauritzen and Spiegelhalter to illustrate their method of propagation of evidence, considers a small piece of factious qualitative medical knowledge. Finally, the BOBLO network is a system, which helps in the verification of the percentage of Jersey cattle through blood type identification, and contains 23 variables and 24 arcs.

From each of the networks, we randomly sampled 4 training datasets with 500, 1500, 3000, and 5000 cases. All datasets are available on the associated web-pages online, or are sampled using the Netica tool [60].

5.2 Algorithms and parameter settings

We have carried out an empirical comparison of BNC-PSO and some other algorithms; they are, Genetic Algorithm [48] (GA), Greedy Equivalent Search [10] (GS), Ant Colony Optimization [21] (ACO), max-min hill climbing [84] (MMHC), previous PSO-based algorithm [87], and finally, an HC algorithm with the standard operators of arc addition, deletion and reversal [34] (HCST). Notice that HCST is a deterministic algorithm, and we use it as a comparison reference.

As mentioned in 4.3, parameters in the proposed BNC-PSO algorithm are given as follows: population size is 50, w decrease linearly from 0.95 to 0.4 according to Eq. (12), c_1 decreasing linearly from 0.82 to 0.5 according to Eq. (13), and finally c_2 increasing from 0.4 to 0.83 according to Eq. (14).

The GA uses the following parameters: population size (λ) is equal to 50, crossover probability (p_c) is equal to 0.9, and mutation rate (p_m) is equal to 0.01. Experiments are done without assuming ordering between the nodes and with local optimizer as in [49].

The ACO algorithm has been used with following parameters: number of ants (m) is equal to 10, the parameter that determines the relative importance of exploitation versus exploration (q_0) is equal to 0.8, ρ which controls the pheromone evaporation and ψ are equal to 0.4, and the importance weight of pheromone (β) is equal to 2.0; according to [21].

GS and MMHC algorithms use their operators of addition, deletion and reversal of arcs as mentioned in [10] and [84], respectively.

Parameter settings of the previous PSO-based algorithm are: population size is equal to 50, $C_1=C_2=2.0$, $Weight=0.9-0.4$; according to [87].

Finally, stopping criteria is defined as, if the score of the constructed network does not improve for a pre-defined number of iterations, It_{MAX} , the run will be stopped.

5.3 Measures of the performance

To evaluate and compare the performance of different algorithms, we have employed several performance metrics, which measure the quality of the results or the complexity of the algorithms. They can be listed as below:

- The value of BIC , Eq. (4). Its interpretation is: the higher this parameter, the network is better. In the origin implementation of some selected algorithms, MDL or $K2$ are used as scoring metrics, which have equivalent computation as BIC .

- The NI [62], defined as:

$$NI(G|D) = \sum_{i=1, Pa(x_i) \neq \emptyset}^n I(x_i, Pa(x_i)) \quad (22)$$

Where $I(., .)$ is the measure of mutual information. It can be said that $NI(G|D)$ is a decreasing monotonic transformation of Kullback distance [46] between the probability distribution associated with the database, and the probability distribution associated with the network. NI 's interpretation is: the higher this parameter, the network is better.

- Normalized Hamming Distance between the learned and the original network (HD). We define the Hamming Distance between two DAGs as the number of the following operators required to make the DAGs match: add or delete an undirected edge, and reverse the orientation of a directed edge. The lower HD indicates the better network.

Mentioned measures evaluate the quality of the constructed network; however, there are other measures, which evaluate the complexity of the algorithms:

- Execution time (Ext). As mentioned before, all algorithms are implemented and executed in .Net framework in a PC which has a single CPU of Intel(R) Core™ 2 Duo 3.33GHz and a 1GB memory. To measure the computation time, algorithms run with no prior limitation in the number of iterations, until more repetition does not increase the score.
- Total number of calls to test of the independence (TI). It can be also called the number of statistical performed by an algorithm.
- The number of iterations, where the best individual was found (It).

5.4 Experimental results and analysis

5.4.1 Performance analysis of the BNC-PSO

To study the performance of BNC-PSO, we use it to construct the BNs, which are introduced in section 5.1. The training datasets with 5000 training samples are used, and results are summarized as below. The constructed network for ALARM is identical to its origin; except that two arcs, {21-31 and 12-32}, are missed (to see the ALARM network, please refer to [5]). A subsequent analysis has revealed that missing arcs are not supported by training samples, and their nodes are actually independent in the employed database. However, the other BNs are completely the same with their origin networks.

Table1. Experimental results of BNC-PSO performance for constructing the ALARM network using different number of samples.

ALARM	500	1500	3000	5000
BIC	9842.72±12.61 (9850.03)	12273.56±6.75 (12278.30)	15730.12±1.45 (15731.50)	15902.48±1.02 (15903.00)
NI	6.821	7.221	9.421	9.473
HD	8.1	4.5	2.0	2.0
Ext	5.43	5.00	3.31	3.01
TI	12.50E06	70.11E05	61.12E05	49.26E05
It	80	61.5	49.0	47

Table2. Experimental results of BNC-PSO performance for constructing the INSURANCE network using different number of samples.

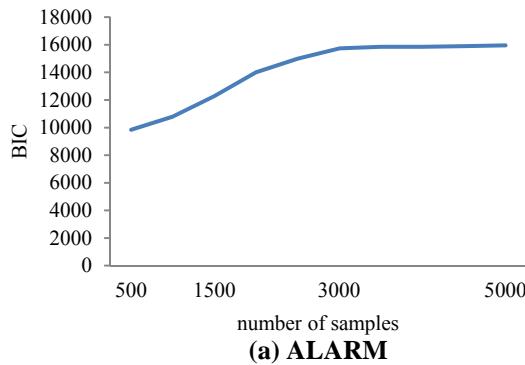
INSURANCE	500	1500	3000	5000
BIC	30298.05±10.61 (30307.55)	45433.23±5.55 (45438.20)	54225.77±1.22 (54226.12)	56970.36±0.50 (56970.80)
NI	5.70	6.00	8.45	8.52
HD	14.0	9.0	0.11	0.0
Ext	6.12	5.20	4.02	3.83
TI	10.03E06	58.12E05	31.56E05	29.75E05
It	191.6	143.4	73.2	71

Table3. Experimental results of BNC-PSO performance for constructing the BOBLO network using different number of samples.

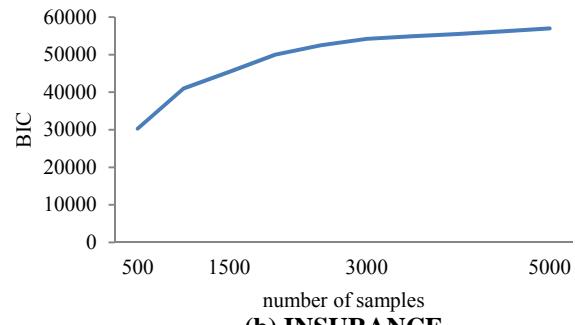
BOBLO	500	1500	3000	5000
BIC	6756.67±14.41 (6767.88)	9191.17±10.35 (9199.45)	11500.61±2.25 (11502.86)	11891.82±1.50 (11893.05)
NI	5.015	5.981	7.492	7.518
HD	8.5	4.0	0.03	0.0
Ext	6.07	5.10	4.66	3.45
TI	69.67E05	48.19E05	30.01E05	29.12E05
It	213	190	147	40

Table4. Experimental results of BNC-PSO performance for constructing the ASIA network using different number of samples.

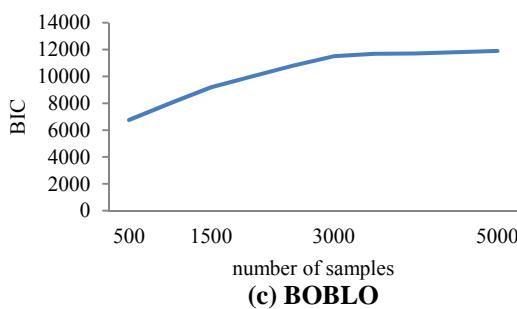
ASIA	500	1500	3000	5000
BIC	545.07±7.12 (551.15)	2145.33±4.45 (2149.77)	3443.28±0.72 (3443.98)	3613.54±0.20 (3613.56)
NI	8.920	9.210	9.510	9.510
HD	4.0	1.0	0.0	0.0
Ext	2.76	2.01	1.35	1.27
TI	30.08E05	19.87E05	12.21E05	10.01E05
It	70	49	32	30



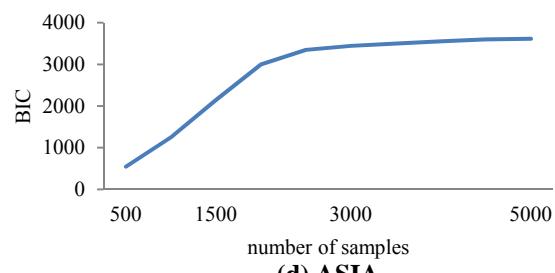
(a) ALARM



(b) INSURANCE



(c) BOBLO



(d) ASIA

Fig. 1. The sensitivity of the size of training datasets on the constructed networks' scores.

Next, the effect of the size of training datasets on the performance of BNC-PSO has been studied for four selected networks. To do this, different subsets of datasets, consisting of the first 500, 1500, 3000 and 5000 cases, are considered and the proposed algorithm is employed to construct the networks using them. Tables 1-4 represent the results of the experiments. Each reported statistic is the average over 50 independent runs of the algorithm on each datasets. In these tables $\mu \pm \sigma$ (for BIC) indicates the mean, and the standard deviation over the executions carried out, and the value inside (.) is the best result found along the experimentation. This information helps us to show the robustness and reliability of BNC-PSO. The results indicate that by using 3000-5000 cases, BNC-PSO shows good performance; while, the constraint-based learning methods need larger training datasets to show such results. For example, the proposed algorithm in [20] needs at least 10000 training samples to achieve such results for the ALARM network. The sensitivity of the size of the training datasets on the scores of the constructed networks (BIC) is also shown in Fig. 1.

5.4.2 Comparison with some other algorithms

Next, we compare the BNC-PSO algorithm with some other algorithms. The results of experiments are displayed in Tables 5-8. Each reported statistic is the average over 50 runs of each algorithm on datasets with 5000 samples.

The most important result which can be considered to prove the goodness of the proposed algorithm is the *BIC* values for the true graphical structures. For ALARM, it is 15902.48, for INSURANCE, it is 56970.36, for BOBLO, it is 11891.82, and finally, for ASIA, it is 3613.54. We also compare the deterministic HCST algorithm and stochastic algorithms in order to obtain significant comparison. Again, BNC-PSO shows almost better results compare with HCST except in time, which is clear that HCST is the fastest. Notice that HCST is a deterministic algorithm and only one execution was carried out. Fig.2 illustrates the score convergence of the algorithms on different networks.

The experimental results reported here show that the proposed algorithm is superior to other related algorithms based on the quality and performance measures. All in all, BNC-PSO improves the score of the constructed networks 15.30 percent compared with GA, 14.76 percent compared with ACO, 11.97 percent compared with Max-Min HC and 17.29 percent compared with GS. In general, the proposed algorithm has improved the score of the constructed networks 14.83% compared with other score-based algorithms.

Table5. Experimental results of the performances of different algorithms on the ALARM network.

ALARM	BNC-PSO	Prev. PSO	GA	ACO	MMHC	GS	HCST
BIC	15902.48±1.02 (15903.00)	14327.02±22.01 (14330.11)	14401.43±16.21 (14417.09)	14399.55±8.98 (14406.50)	14304.03±35.44 (14338.50)	14058.28±62.12 (14100.30)	14420
NI	9.473	9.230	9.231	9.230	9.229	9.230	9.220
HD	2.0	2.50	2.1	2.12	2.73	2.62	2.50
Ext	3.01	4.78	3.29	4.04	5.34	9.37	2.99
TI	49.26E05	72.812E06	71.08E05	75.48E05	80.75E05	18.89E06	154637
It	47	72	56	70	87	385	-

Table6. Experimental results of the performances of different algorithms on the INSURANCE network.

INSURANCE	BNC-PSO	Prev. PSO	GA	ACO	MMHC	GS	HCST
BIC	56970.36±0.50 (56970.80)	47833.02±12.22 (47839.23)	47717.22±10.25 (47726.12)	48084.49±6.06 (48090.00)	49925.15±22.34 (49945.44)	47008.05±40.74 (47048.55)	56191
NI	8.52	8.430	8.446	8.440	8.450	8.401	8.231
HD	0.0	0.5	0.3	0.2	0.0	1.0	0.0
Ext	3.83	4.55	3.98	4.34	5.12	7.77	3.01
TI	29.75E05	44.016E05	43.07E05	43.45E05	38.25E05	44.80E06	7.66E04
It	71	95	90	98	107	238	-

Table7. Experimental results of the performances of different algorithms on the BOBLO network.

BOBLO	BNC-PSO	Prev. PSO	GA	ACO	MMHC	GS	HCST
BIC	11891.82±1.50 (11893.05)	11344.11±12.07 (11350.67)	11445.34±18.81 (11461.50)	11425.21±9.01 (11431.77)	11587.55±38.54 (11612.12)	11202.13±65.55 (11265.50)	11959
NI	7.518	9.230	9.49	9.49	9.500	9.489	7.472
HD	0.0	1.0	1.0	1.1	0.0	1.62	0.0
Ext	3.45	5.00	4.50	4.57	5.04	8.84	2.90
TI	29.12E05	39.18E06	38.38E05	39.12E05	36.15E05	80.09E06	37385
It	40	70	64	75	89	312	-

Table8. Experimental results of the performances of different algorithms on the ASIA network.

ASIA	BNC-PSO	Prev. PSO	GA	ACO	MMHC	GS	HCST
BIC	3613.54±0.20 (3613.56)	3080.38±7.40 (3085.33)	3082.74±8.70 (3090.11)	3095.22±4.06 (3098.82)	311031±16.13 (3125.48)	3078.62±22.27 (3100.00)	3340
NI	9.510	9.230	9.321	9.331	9.410	9.210	9.450
HD	0.0	0.0	0.0	0.0	0.0	0.0	0.0
Ext	1.27	1.78	1.39	1.54	2.03	3.37	1.20
TI	10.01E05	12.81E05	12.12E05	12.85E05	14.54E05	19.02E06	17312
It	30	42	53	58	62	80	-

5.5 Predictive ability of the proposed algorithm

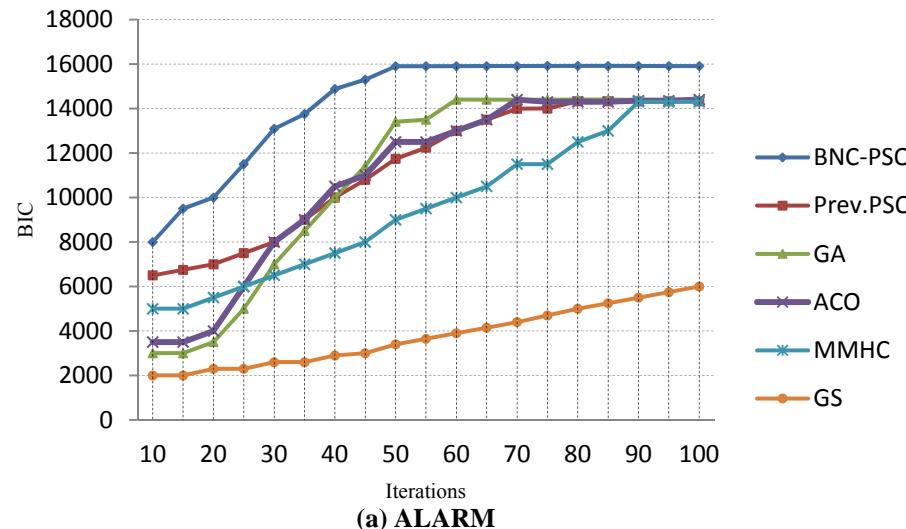
A constructed BN is an efficient device to perform probabilistic inference, whereupon, predictive and estimation [39] ability in different applications is one of the important issues in the field of BNs. Classification is one of the prominent applications of the BNs, which is used in varied fields such as, recommender systems for estimating users' ratings based on their implicit preferences, bank direct marketing for predicting clients' willingness of deposit subscription, disease diagnosis for assessing patients' breast cancer risk [25], and simultaneous fault diagnosis [31, 67]. In order to show the productivity of the proposed algorithm, in this section, we briefly explain about BNs classifiers and then, choosing datasets of different applications, we evaluate the classification accuracy and classification time of different algorithms.

5.5.1 BN classifiers

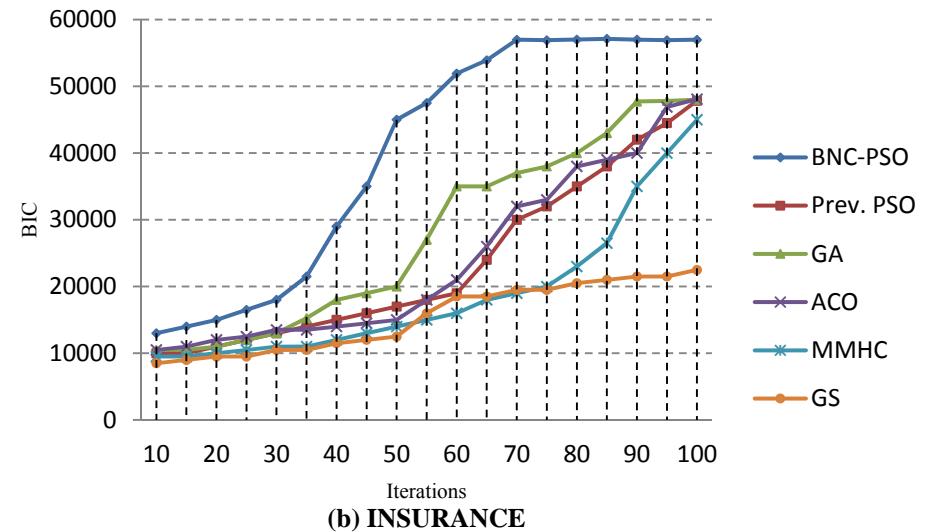
Suppose that each training sample is a vector of attributes $(X_1, X_2 \dots X_{v-1}, C)$. The goal of classification is predicting the right value of class variable $c=x_v$ having $(x_1, x_2 \dots x_{v-1})$. If the performance measure is the percentage of correct predictions on test samples (classification accuracy), the correct prediction for $(x_1, x_2 \dots x_{v-1})$ is a class that maximizes $P(c|x_1, x_2 \dots x_{v-1})$. If there is a BN over $(x_1, x_2 \dots x_{v-1}, C)$, we could compute these probabilities by inference on it. After the structure of a BN is specified, estimating the parameters, so that the network can provide the best prediction for the value of the class variable in the test samples, is important; however, it is out of the scope of this study, and we simply use Maximum Likelihood (ML) to estimate the parameters' values.

5.5.2 Comparison the classification accuracy of BNC-PSO against other classifiers

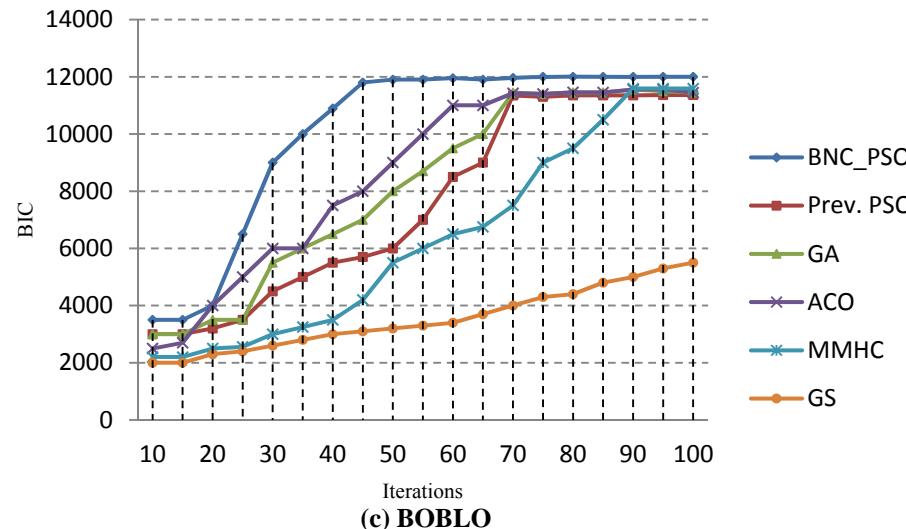
In order to evaluate the classification accuracy of BNC-PSO compared with other algorithms, 15 datasets from UCI repository [58] and [42] are used. Table9 shows a brief description of these datasets.



(a) ALARM



(b) INSURANCE



(c) BOBLO

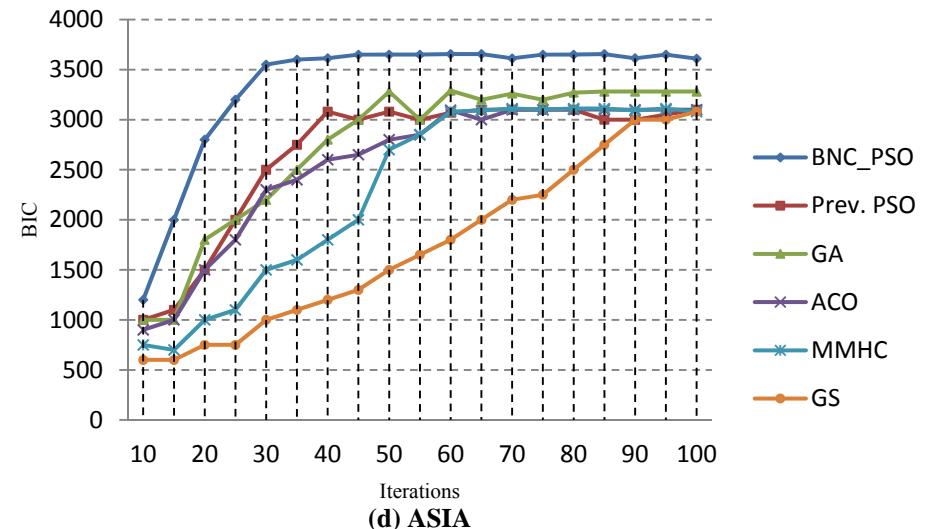


Fig. 2. The score convergence of different algorithms on: (a) the ALARM, (b) the INSURANCE, (c) the BOBLO, and (d) the ASIA networks.

All implemented classifiers are described as follows:

- BNC-PSO-based classifier,
- Naïve Bayes classifier (NB),
- TAN-based classifier (TAN),
- Hill climbing-based classifier (HC),
- Genetic algorithm-based classifier (GA),
- And finally, ACO-classifier (ACO).

In order to construct more efficient classifiers, another scoring function is used, which is proposed in [68], named classification rate:

$$CR = \frac{1}{|D|} \sum_{m \in D}^{|D|} \delta(B_D(x_{1:N}^m), c^m), \quad (22)$$

where, $|D|$ is the number of training samples. The equation simply represents the rate of samples that are classified correctly by the network. And $\delta(B_D(x_{1:N}^m), c^m) = 1$ if the BN classifier $B_D(x_{1:N}^m)$, which is trained using D , predicts the right value of the class variable c^m having attributes $x_{1:N}^m$.

Table10 represents the average error rate and the standard deviation of each different algorithm for classification of different datasets. For each algorithm, we have carried out 50 independent runs over each dataset with a fixed execution time. The best results are highlighted.

As a reference to the goodness of the results, we can consider the average error rate for all 15 datasets, which are **0.128413** for BNC-PSO, 0.144127 for GA, 0.149407 for TAN, 0.158947 for HC, 0.159107 for ACO and 0.173547 for NB. The details of the results show that, for ten datasets, BNC-PSO has shown the best results; and for five remaining datasets (Breast, Heart, mofn-3-7-10, soybean-large, and Vehicle), the best results belong to NB and TAN. However, for these five datasets the differences between the best results and the results of BNC-PSO are negligible, particularly with regard to their standard deviations values.

Table9. Datasets and their samples.

Dataset	Number Of Classes	Number Of Attributes	Number Of Samples	Dataset	Number Of classes	Number Of attributes	Number Of Samples
Australian	2	14	690	Letter	26	16	15000
Breast	2	10	683	mofn-3-7-10	2	10	300
Chess	2	36	2130	Pima	2	8	768
Crx	2	15	653	shuttle-small	7	9	3866
Flare	2	10	1066	soybean-large	19	35	562
Glass	7	9	214	Vehicle	4	18	846
Heart	2	13	270	Vote	2	16	435
Iris	3	4	150				

Fig. 3 represents the scatter charts which directly compare the BNC-PSO classifier with other classifiers; points above the line $y=x$, represent wherever the BNC-PSO has shown better performance in comparison with other algorithms.

5.5.3 Comparison the classification time of BNC-PSO with other classifiers

Finally, classification execution time of BNC-PSO is compared against other algorithms. All algorithms are implemented and executed in .Net framework in a PC, which has a single CPU of Intel(R) Core™ 2 Duo 3.33GHz and a 1GB memory. Table 11 shows the results after the 50 independent runs for five chosen dataset. The chosen datasets for these experiments are: Australian, Iris, vehicle, glass, and soybean-large. The results indicate that BNC-PSO needs less time for classification, in comparison with other algorithms. For a better comparison of different algorithms, Fig. 4 is added, which presents the bar chart of the classification execution time.

Table10. Experimental results of classification error rate and standard deviation on different datasets.

Algorithms Datasets	BNC-PSO	NB	TAN	HC	GA	ACO
Australian	0.1149±0.008	0.1489±0.010	0.1751±0.012	0.1391±0.010	0.1296±0.008	0.1488±0.009
Breast	0.0411±0.004	0.0245±0.017	0.0351±0.010	0.0668±0.008	0.0425±0.006	0.0339±0.008
Chess	0.0469±0.002	0.1266±0.003	0.076±0.011	0.0966±0.006	0.0522±0.004	0.06±0.004
Crx	0.1401±0.004	0.1505±0.003	0.1631±0.003	0.167±0.007	0.158±0.005	0.1505±0.004
Flare	0.174±0.007	0.2024±0.010	0.178±0.010	0.181±0.016	0.1804±0.012	0.1813±0.012
Glass	0.3992±0.003	0.4412±0.006	0.4578±0.006	0.4412±0.007	0.4173±0.007	0.422±0.005
Heart	0.1751±0.008	0.155±0.016	0.1847±0.012	0.2221±0.012	0.1874±0.010	0.155±0.010
Iris	0.0411±0.001	0.0699±0.003	0.0763±0.003	0.0414±0.004	0.042±0.002	0.0485±0.002
Letter	0.1061±0.003	0.3068±0.003	0.1752±0.003	0.1896±0.005	0.183±0.004	0.3068±0.003
mofn-3-7-10	0.0866±0.004	0.1328±0.003	0.085±0.004	0.0859±0.008	0.0859±0.006	0.1367±0.006
Pima	0.1823±0.008	0.2571±0.012	0.2384±0.010	0.255±0.016	0.2666±0.010	0.2505±0.010
shuttle-small	0.0041±0.000	0.014±0.004	0.0093±0.003	0.0145±0.008	0.0077±0.003	0.0083±0.003
soybean-large	0.0786±0.004	0.0852±0.008	0.0644±0.010	0.0922±0.012	0.0754±0.008	0.092±0.008
Vehicle	0.2907±0.008	0.3892±0.008	0.2758±0.016	0.3451±0.016	0.2922±0.010	0.3453±0.010
Vote	0.0374±0.001	0.0991±0.003	0.0509±0.003	0.0467±0.005	0.0417±0.003	0.047±0.002
Averages	0.128413±0.004	0.173547±0.007	0.149407±0.007	0.158947±0.009	0.144127±0.006	0.159107±0.006

6. Discussion

In this section, advantages and disadvantages of the proposed algorithm, and also our plans for future works are discussed.

Advantages: PSO has many advantages compared with other optimization algorithms. First, the concept of PSO is simple and it can be implemented in a few lines of code, which requires some primitive operators, and it is computationally inexpensive, both in memory and runtime. Second, a particle swarm system has memory; it retains the knowledge of where in the search space it performed the best, a memory of an experience. In a GA, for example, if an individual is not selected for elitism or crossover,

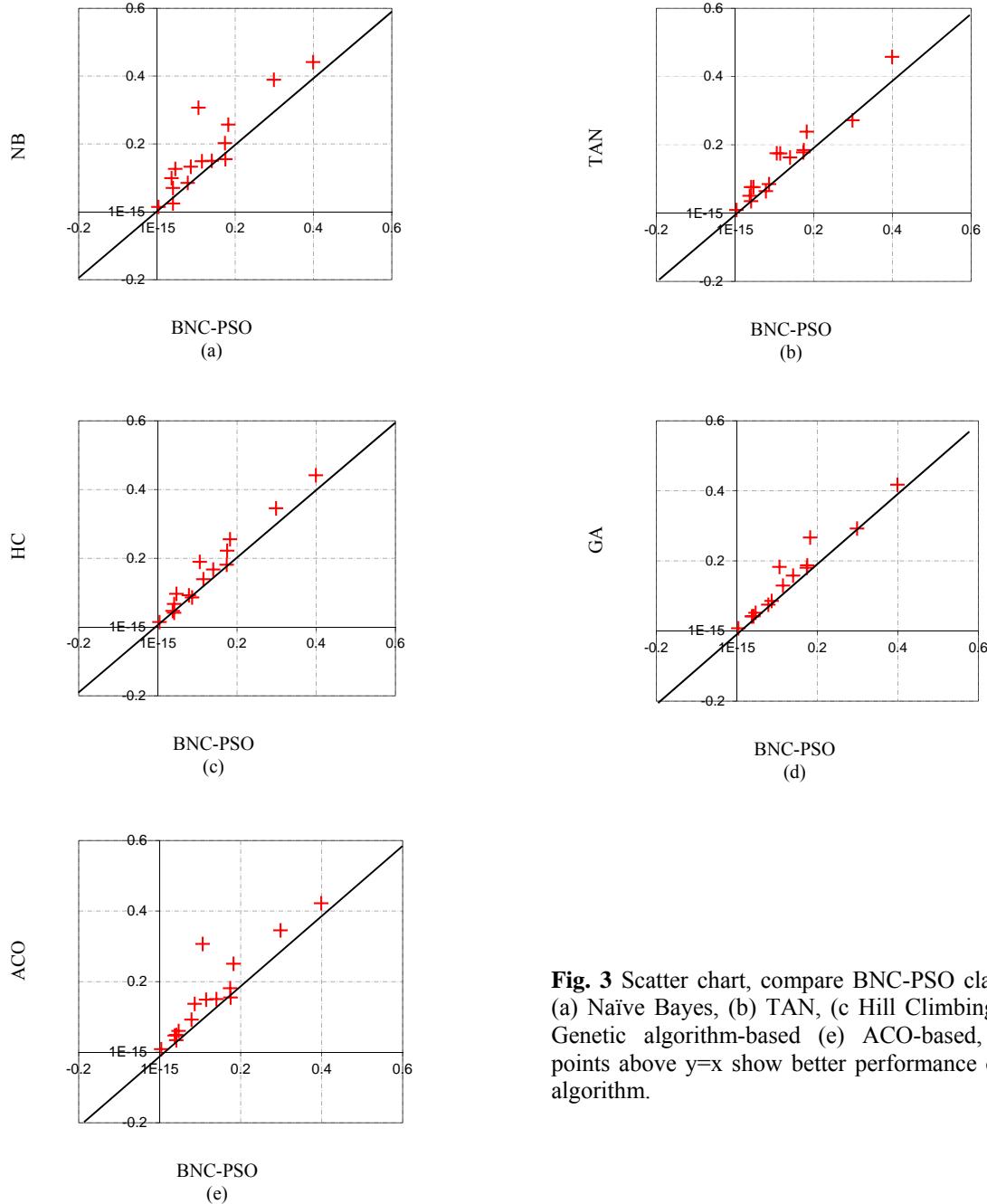


Fig. 3 Scatter chart, compare BNC-PSO classifier with (a) Naïve Bayes, (b) TAN, (c) Hill Climbing-based, (d) Genetic algorithm-based (e) ACO-based, classifiers; points above $y=x$ show better performance of proposed algorithm.

the information contained by that individual is lost; as for PSO, individuals who fly past the optima are tugged to return towards them. Indeed, in PSO, knowledge of good solutions is retained by all particles [24]. Third, every particle flies in the candidate problem space, adjusts their velocity and position according to the local best and the global best; so, all the particles have a powerful search capability, which can help the swarm find the optimal solution. While, for most of the optimization algorithms, after finding a sub-optimal solution, they cannot find a better one.

Disadvantages: The running time of PSO is affected less by the problem dimension (number of random variables), but more by the number of generations. It means that the computational time increases quickly with the number of generations; and it is due to the communications between the particles after each generation. Hence, in terms of computational time with a higher number of generations, other approaches such as GA may work faster. However, PSO converges fast, and it can obtain the best solution in a small number of iterations; this may help to reduce its final execution time. Moreover, as mentioned in section 1, algorithms, which belong to the scored-based structure learning approach, work well for small datasets but they may fail to find optimal solution for large datasets, compared with constrained-based learning; although BNC-PSO is not an exception, mentioned features of PSO help to find desirable solution even for large datasets.

Table11. Classification execution time of different algorithms

Algorithms \ Datasets	BNC-PSO	NB	TAN	HC	GA	ACO
Australian	38.32	52.12	50.00	55.40	41.67	46.33
Iris	14.50	19.90	17.50	22.53	15.30	17.88
vehicle	40.47	62.13	61.22	65.72	47.50	54.67
glass	20.00	32.40	30.75	35.27	27.04	29.14
soybean-large	72.36	101.00	98.41	112.05	85.95	92.84

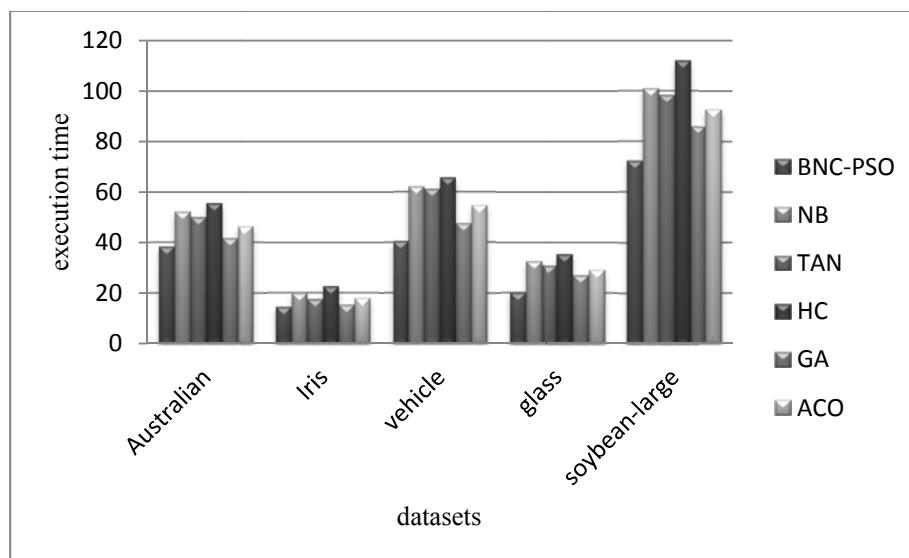


Fig. 4. The bar chart of classification execution time of different algorithms

Future works: BNs have also been used to model time-series data; a choice for modelling time-series data is to use directed graphical models, which can appropriately capture the forward flow of time. If all arcs of the model are directed, both within and between different time slices, while the structure is unchanged, the resulting model is called *Dynamic Bayesian networks* (DBN). An interesting future work that can complete this work is to apply proposed PSO-based algorithm for DBN such as [15, 55, 63, 91]. The writers are also enthusiastic to study more about the behaviour of the BNs, which reflect the dependency between continuous variables [82], or investigate about structure learning in super-structure BNs and its requirements. Finally, we plan to apply the BNs obtained by our model, as learning and reasoning tool in cognitive networks [71], which are an attractive research area in the field of computer communications.

7. Conclusion

Bayesian networks (BNs) belong to an important class of probabilistic graphical models, which have been proven to be very useful and effective for reasoning in the uncertain domain. One of the important challenges in the field of BNs is constructing the network topology from data; in this paper, we have proposed an efficient algorithm for structure learning of BN using Particle Swarm Optimization (PSO). Stochastic crossover and mutation operations and also a cycle-removing procedure are defined to make the evolutionary process more effective. Time and convergence analysis are carried out and the convergence of the proposed algorithm to an optimal solution is proven. Using simulations of some benchmark networks, we carry out a performance analysis on the proposed BNC-PSO algorithm. Experimental results, which are reported in section 5.4.1, show the acceptable performance of the BNC-PSO algorithm. BNC-PSO is also compared with some other score-based algorithms such as Genetic Algorithm (GA), Ant Colony Optimization (ACO), Hill-Climbing (HC), and Greedy Search (GS) algorithms. The experimental results reported in section 5.4.2 show that the proposed algorithm is superior to the other algorithms based on the quality and performance measures. On average, BNC-PSO has improved the score of the constructed network 15.30% compared with GA, 14.76% compared with ACO, 11.97% compared with HC, and 17.29% compared with GS. Moreover, in order to examine the predictive ability of BNC-PSO in classification, we have also evaluated the classification accuracy and the classification time in section 5.5. Experimental results show that BNC-PSO classifies data with less error rate and in less computational time.

References

- [1] Acid, Silvia, and Luis M. de Campos. "A hybrid methodology for learning belief networks: BENEDICT." *International Journal of Approximate Reasoning* 27, no. 3 (2001): 235-262.
- [2] Adabor, Emmanuel S., George K. Acquaah-Mensah, and Francis T. Oduro. "SAGA: A hybrid search algorithm for Bayesian Network structure learning of transcriptional regulatory networks." *Journal of biomedical informatics* 53 (2015): 27-35.
- [3] Akaike, Hirotugu. "A new look at the statistical model identification." *Automatic Control, IEEE Transactions on* 19, no. 6 (1974): 716-723.
- [4] Babahajani, P., F. Habibi and H. Bevrani. "An On-Line PSO-Based Fuzzy Logic Tuning Approach: Microgrid Frequency Control Case Study." In *Handbook of Research on Novel Soft Computing Intelligent Algorithms: Theory and Practical Applications*, ed. Pandian M. Vasant, 589-616 (2014), accessed December 15, 2015. doi:10.4018/978-1-4666-4450-2.ch020
- [5] Beinlich, Ingo A., Henri J. Suermondt, R. Martin Chavez, and Gregory F. Cooper. *The ALARM monitoring system: A case study with two probabilistic inference techniques for belief networks*. Springer Berlin Heidelberg, 1989.
- [6] Binder, John, Daphne Koller, Stuart Russell, and Keiji Kanazawa. "Adaptive probabilistic networks with hidden variables." *Machine Learning* 29, no. 2-3 (1997): 213-244.
- [7] Borboudakis, Giorgos, and Ioannis Tsamardinos. "Bayesian network learning with discrete case-control data." In *Uncertainty in Artificial Intelligence (UAI)*. 2015.

- [8] Cao, Shengrang, Xiaoqun Ding, Qingyan Wang, and Bingyan Chen. "Opposition-Based Improved PSO for Optimal Reactive Power Dispatch and Voltage Control." *Mathematical Problems in Engineering* 501 (2015): 754582.
- [9] Chickering, David Maxwell. "Learning Bayesian networks is NP-complete." In *Learning from data*, pp. 121-130. Springer New York, 1996.
- [10] Chickering, David Maxwell. "Learning equivalence classes of Bayesian-network structures." *The Journal of Machine Learning Research* 2 (2002): 445-498.
- [11] Chickering, David Maxwell, David Heckerman, and Christopher Meek. "Large-sample learning of Bayesian networks is NP-hard." *The Journal of Machine Learning Research* 5 (2004): 1287-1330.
- [12] Clerc, Maurice. "Discrete particle swarm optimization, illustrated by the traveling salesman problem." In *New optimization techniques in engineering*, pp. 219-239. Springer Berlin Heidelberg, 2004.
- [13] Cooper, Gregory F., and Edward Herskovits. "A Bayesian method for the induction of probabilistic networks from data." *Machine learning* 9, no. 4 (1992): 309-347.
- [14] Cussens, James, and Mark Bartlett. "Advances in Bayesian network learning using integer programming." *arXiv preprint arXiv:1309.6825* (2013).
- [15] Dai, Jingguo, and Jia Ren. "Unsupervised Evolutionary Algorithm for Dynamic Bayesian Network Structure Learning." In *Advanced Methodologies for Bayesian Networks*, pp. 136-151. Springer International Publishing, 2015.
- [16] Daly, Rónán, and Qiang Shen. "Learning Bayesian network equivalence classes with ant colony optimization." *Journal of Artificial Intelligence Research* 35, no. 1 (2009): 391.
- [17] Dash, Denver, and Marek J. Druzdzel. "A hybrid anytime algorithm for the construction of causal models from sparse data." In *Proceedings of the Fifteenth conference on Uncertainty in artificial intelligence*, pp. 142-149. MorGan Kaufmann Publishers Inc., 1999.
- [18] De Campos, Cassio P., and Qiang Ji. "Efficient structure learning of Bayesian networks using constraints." *The Journal of Machine Learning Research* 12 (2011): 663-689.
- [19] De Campos, Luis M., and J. Miguel Puerta. "Stochastic local algorithms for learning belief networks: Searching in the space of the orderings." In *Symbolic and Quantitative Approaches to Reasoning with Uncertainty*, pp. 228-239. Springer Berlin Heidelberg, 2001.
- [20] De Campos, Luis M., and Juan F. Huete. "A new approach for learning belief networks using independence criteria." *International Journal of Approximate Reasoning* 24, no. 1 (2000): 11-37.
- [21] De Campos, Luis M., Juan M. Fernandez-Luna, José A. Gámez, and José M. Puerta. "Ant colony optimization for learning Bayesian networks." *International Journal of Approximate Reasoning* 31, no. 3 (2002): 291-311.
- [22] del Carmen Chávez, María, Gladys Casas, Rafael Falcón, Jorge E. Moreira, and Ricardo Grau. "Building fine bayesian networks aided by pso-based feature selection." In *MICAI 2007: Advances in Artificial Intelligence*, pp. 441-451. Springer Berlin Heidelberg, 2007.
- [23] Dünder, Emre, Mehmet Ali Cengiz, and Haydar Koç. "INVESTIGATION OF THE IMPACTS OF CONSTRAINT-BASED ALGORITHMS TO THE QUALITY OF BAYESIAN NETWORK STRUCTURE IN HYBRID ALGORITHMS FOR MEDICAL STUDIES." *Journal of Advanced Scientific Research* 5, no. 1 (2014).
- [24] Eberhart, Russ C., and James Kennedy. "A new optimizer using particle swarm theory." In *Proceedings of the sixth international symposium on micro machine and human science*, vol. 1, pp. 39-43. 1995.
- [25] Feng, Guang, Jia-Dong Zhang, and Stephen Shaoyi Liao. "A novel method for combining Bayesian networks, theoretical analysis, and its applications." *Pattern Recognition* 47, no. 5 (2014): 2057-2069.
- [26] Friedman, Nir, Iftach Nachman, and Dana Peér. "Learning bayesian network structure from massive datasets: the «sparse candidate »algorithm." In *Proceedings of the Fifteenth conference on Uncertainty in artificial intelligence*, pp. 206-215. MorGan Kaufmann Publishers Inc., 1999.

- [27] Gallagher, Marcus, Ian Wood, Jonathan Keith, and George Sofronov. "Bayesian inference in estimation of distribution algorithms." In *Evolutionary Computation, 2007. CEC 2007. IEEE Congress on*, pp. 127-133. IEEE, 2007.
- [28] Garg, Harish, Monica Rani and S.P. Sharma. "Predicting Uncertain Behavior and Performance Analysis of the Pulping System in a Paper Industry using PSO and Fuzzy Methodology." In *Handbook of Research on Novel Soft Computing Intelligent Algorithms: Theory and Practical Applications*, ed. Pandian M. Vasant, 414-449 (2014), accessed December 15, 2015. doi:10.4018/978-1-4666-4450-2.ch014
- [29] Gheisari, S., M. R. Meybodi, M. Dehghan, and M. M. Ebadzadeh. "Bayesian network structure training based on a game of learning automata." *International Journal of Machine Learning and Cybernetics* 7 (2016) 1-13.
- [30] Glover, Fred. "Tabu search: A tutorial." *Interfaces* 20, no. 4 (1990): 74-94.
- [31] He, Yu-Lin, Ran Wang, Sam Kwong, and Xi-Zhao Wang. "Bayesian classifiers based on probability density estimation and their applications to simultaneous fault diagnosis." *Information Sciences* 259 (2014): 252-268.
- [32] Heckerman, David. *A tutorial on learning with Bayesian networks*. Springer Netherlands, 1998.
- [33] Heckerman, David. "A tutorial on learning with Bayesian networks." In *Innovations in Bayesian Networks*, pp. 33-82. Springer Berlin Heidelberg, 2008.
- [34] Heckerman, David, Dan Geiger, and David M. Chickering. "Learning Bayesian networks: The combination of knowledge and statistical data." *Machine learning* 20, no. 3 (1995): 197-243.
- [35] Hemmecke, Raymond, Silvia Lindner, and Milan Studený. "Characteristic imsets for learning Bayesian network structure." *International Journal of Approximate Reasoning* 53, no. 9 (2012): 1336-1349.
- [36] Heng, Xing-Chen, Zheng Qin, Xian-Hui Wang, and Li-Ping Shao. "Research on learning bayesian networks by particle swarm optimization." *Information Technology Journal* 5, no. 3 (2006): 540-545.
- [37] Hesar, Alireza Sadeghi. "Structure Learning of Bayesian Belief Networks Using Simulated Annealing Algorithm." *Middle-East Journal of Scientific Research* 18, no. 9 (2013): 1343-1348.
- [38] Hsu, William H., Haipeng Guo, Benjamin B. Perry, and Julie A. Stilson. "A Permutation Genetic Algorithm For Variable Ordering In Learning Bayesian Networks From Data." In *GECCO*, vol. 2, pp. 383-390. 2002.
- [39] Jaakkola, Tommi, David Sontag, Amir Globerson, and Marina Meila. "Learning Bayesian network structure using LP relaxations." In *International Conference on Artificial Intelligence and Statistics*, pp. 358-365. 2010.
- [40] Jarraya, Aida, Philippe Leray, and Afif Masmoudi. "Discrete exponential Bayesian networks: definition, learning and application for density estimation." *Neurocomputing* 137 (2014): 142-149.
- [41] Kennedy, James, and Russell C. Eberhart. "A discrete binary version of the particle swarm algorithm." In *Systems, Man, and Cybernetics, 1997. Computational Cybernetics and Simulation., 1997 IEEE International Conference on*, vol. 5, pp. 4104-4108. IEEE, 1997.
- [42] Kohavi, Ron, and George H. John. "Wrappers for feature subset selection." *Artificial intelligence* 97, no. 1 (1997): 273-324.
- [43] Koivisto, Mikko. "Advances in exact Bayesian structure discovery in Bayesian networks." *arXiv preprint arXiv:1206.6828* (2012).
- [44] Koivisto, Mikko, and Kismat Sood. "Exact Bayesian structure discovery in Bayesian networks." *The Journal of Machine Learning Research* 5 (2004): 549-573.
- [45] Koski, Timo JT, and John M. Noble. "A review of bayesian networks and structure learning." *Annales Societatis Mathematicae Polonae. Series 3: Mathematica Applicanda* 40, no. 1 (2012): 53-103.
- [46] Kullback, Solomon. *Information theory and statistics*. Courier Corporation, 1968.
- [47] Larrañaga, Pedro, Hossein Karshenas, Concha Bielza, and Roberto Santana. "A review on evolutionary algorithms in Bayesian network learning and inference tasks." *Information Sciences* 233 (2013): 109-125.

- [48] Larrañaga, Pedro, Mikel Poza, Yosu Yurramendi, Roberto H. MurGa, and Cindy MH Kuijpers. "Structure learning of Bayesian networks by genetic algorithms: A performance analysis of control parameters." *Pattern Analysis and Machine Intelligence, IEEE Transactions on* 18, no. 9 (1996): 912-926.
- [49] Lauritzen, Steffen L., and David J. Spiegelhalter. "Local computations with probabilities on graphical structures and their application to expert systems." *Journal of the Royal Statistical Society. Series B (Methodological)* (1988): 157-224.
- [50] Li, Xiao-Lin. "A Particle Swarm Optimization and Immune Theory-Based Algorithm for Structure Learning of Bayesian Networks." *Int. J. Database Theory Appl* 3, no. 2 (2010): 61-69.
- [51] Li, Xiao-Lin, Shuang-Cheng Wang, and Xiang-Dong He. "Learning Bayesian networks structures based on memory binary particle swarm optimization." In *Simulated Evolution and Learning*, pp. 568-574. Springer Berlin Heidelberg, 2006.
- [52] Ma, Yiwei, Ping Yang, Zhuoli Zhao, and Yuewu Wang. "Optimal Economic Operation of Islanded Microgrid by Using a Modified PSO Algorithm." *Mathematical Problems in Engineering* 501 (2015): 379250.
- [53] Majumder, Arindam and Abhishek Majumder. "Application of Standard Deviation Method Integrated PSO Approach in Optimization of Manufacturing Process Parameters." In *Handbook of Research on Artificial Intelligence Techniques and Algorithms*, ed. Pandian Vasant, 536-563 (2015), accessed December 15, 2015. doi:10.4018/978-1-4666-7258-1.ch017
- [54] Malone, Brandon. "Empirical Behavior of Bayesian Network Structure Learning Algorithms." In *Advanced Methodologies for Bayesian Networks*, pp. 105-121. Springer International Publishing, 2015.
- [55] Marini, Simone, Emanuele Trifoglio, Nicola Barbarini, Francesco Sambo, Barbara Di Camillo, Alberto Malovini, Marco Manfrini, Claudio Cobelli, and Riccardo Bellazzi. "A Dynamic Bayesian Network model for long-term simulation of clinical complications in type 1 diabetes." *Journal of biomedical informatics* 57 (2015): 369-376.
- [56] Moore, Andrew, and Weng-Keen Wong. "Optimal reinsertion: A new search operator for accelerated and more accurate Bayesian network structure learning." In *ICML*, vol. 3, pp. 552-559. 2003.
- [57] Murphy, Kevin. "An introduction to graphical models." *Rap. tech* (2001).
- [58] Murphy, P., and David W. Aha. "UCI repository of machine learning databases--a machine-readable repository." (1995).
- [59] Myers, James W., Kathryn Blackmond Laskey, and Tod Levitt. "Learning Bayesian networks from incomplete data with stochastic search algorithms." In *Proceedings of the Fifteenth conference on Uncertainty in artificial intelligence*, pp. 476-485. MorGan Kaufmann Publishers Inc., 1999.
- [60] Netica. Netica Bayesian network software from Norsys. <http://www.norsys.com>
- [61] O'Gorman, Bryan, R. Babbush, A. Perdomo-Ortiz, Alan Aspuru-Guzik, and Vadim Smelyanskiy. "Bayesian network structure learning using quantum annealing." *The European Physical Journal Special Topics* 224, no. 1 (2015): 163-188.
- [62] Pan, Quan-Ke, M. Fatih Tasgetiren, and Yun-Chia Liang. "A discrete particle swarm optimization algorithm for the no-wait flowshop scheduling problem." *Computers & Operations Research* 35, no. 9 (2008): 2807-2839.
- [63] Pasquini Santos, Fernando, and Carlos Dias Maciel. "A PSO approach for learning transition structures of Higher-Order Dynamic Bayesian Networks." In *Biosignals and Biorobotics Conference (2014): Biosignals and Robotics for Better and Safer Living (BRC), 5th ISSNIP-IEEE*, pp. 1-6. IEEE, 2014.
- [64] Pearl, Judea, and Stuart Russell. *Bayesian networks*. Computer Science Department, University of California, 1998.
- [65] Pelikan, Martin. "Bayesian optimization algorithm." In *Hierarchical Bayesian optimization algorithm*, pp. 31-48. Springer Berlin Heidelberg, 2005.
- [66] Pelikan, Martin, and David E. Goldberg. "Bayesian optimization algorithm: From single level to hierarchy." *University of Illinois at Urbana-Champaign, Champaign, IL* (2002).
- [67] Perkusich, Mirko, Gustavo Soares, Hyggo Almeida, and Angelo Perkusich. "A procedure to detect problems of processes in software development projects using Bayesian networks." *Expert Systems with Applications* 42, no. 1 (2015): 437-450.
- [68] Pernkopf, Franz. "Bayesian network classifiers versus selective k-NN classifier." *Pattern Recognition* 38, no. 1 (2005): 1-10.

- [69] Perrier, Eric, Seiya Imoto, and Satoru Miyano. "Finding optimal Bayesian network given a super-structure." *Journal of Machine Learning Research* 9, no. 2 (2008): 2251-2286.
- [70] Polprasert, Jirawadee, Weerakorn Ongsakul, and Vo Ngoc Dieu. "A New Improved Particle Swarm Optimization for Solving Nonconvex Economic Dispatch Problems." *International Journal of Energy Optimization and Engineering (IJEDE)* 2, no. 1 (2013): 60-77.
- [71] Quer, Giorgio, Hemanth Meenakshisundaram, Bheemarjuna Tamma, B. S. Manoj, Ramesh Rao, and Michele Zorzi. "Cognitive Network Inference through Bayesian Network Analysis." In *Global Telecommunications Conference (GLOBECOM 2010), 2010 IEEE*, pp. 1-6. IEEE, 2010.
- [72] Rasmussen, Lene, K. "Bayesian Network for blood typing and parentage verification of cattle." PhD. thesis, *Research Center Foulum*, Denmark, 1995.
- [73] Ratnaweera, AsanGa, Saman K. HalGamuge, and Harry C. Watson. "Self-organizing hierarchical particle swarm optimizer with time-varying acceleration coefficients." *Evolutionary Computation, IEEE Transactions on* 8, no. 3 (2004): 240-255.
- [74] Robinson, Robert W. "Counting unlabeled acyclic digraphs." In *Combinatorial mathematics V*, pp. 28-43. Springer Berlin Heidelberg, 1977.
- [75] Salama, Khalid M., and Alex A. Freitas. "Ant colony algorithms for constructing Bayesian multi-net classifiers." *Intelligent Data Analysis* 19, no. 2 (2015): 233-257.
- [76] Schwarz, Gideon. "Estimating the dimension of a model." *The annals of statistics* 6, no. 2 (1978): 461-464.
- [77] Shi, Yuhui, and Russell Eberhart. "A modified particle swarm optimizer." In *Evolutionary Computation Proceedings, 1998. IEEE World Congress on Computational Intelligence., The 1998 IEEE International Conference on*, pp. 69-73. IEEE, 1998.
- [78] Silverstein, Craig, Sergey Brin, Rajeev Motwani, and Jeff Ullman. "Scalable techniques for mining causal structures." *Data Mining and Knowledge Discovery* 4, no. 2-3 (2000): 163-192.
- [79] Silander, Tomi, and Petri Myllymaki. "A simple approach for finding the globally optimal Bayesian network structure." *arXiv preprint arXiv:1206.6875* (2012).
- [80] Singh, Ajit P., and Andrew W. Moore. "Finding optimal Bayesian networks by dynamic programming." (2005).
- [81] Spirtes, Peter, Clark N. Glymour, and Richard Scheines. *Causation, prediction, and search*. Vol. 81. MIT press, 2000.
- [82] Suzuki, Joe. "Consistency of learning Bayesian network structures with continuous variables: an information theoretic approach." *Entropy* 17, no. 8 (2015): 5752-5770.
- [83] Teyssier, Marc, and Daphne Koller. "Ordering-based search: A simple and effective algorithm for learning Bayesian networks." *arXiv preprint arXiv:1207.1429* (2012).
- [84] Tsamardinos, Ioannis, Laura E. Brown, and Constantin F. Aliferis. "The max-min hill-climbing Bayesian network structure learning algorithm." *Machine learning* 65, no. 1 (2006): 31-78.
- [85] Villanueva, Edwin, and Carlos Dias Maciel. "Efficient methods for learning Bayesian network super-structures." *Neurocomputing* 123 (2014): 3-12.
- [86] Vo, Dieu Ngoc and Peter Schegner. "An Improved Particle Swarm Optimization for Optimal Power Flow." In *Meta-Heuristics Optimization Algorithms in Engineering, Business, Economics, and Finance*, ed. Pandian M. Vasant, 1-40 (2013), accessed December 15, 2015. doi:10.4018/978-1-4666-2086-5.ch001.
- [87] Wang, Tong, and Jie Yang. "A heuristic method for learning Bayesian networks using discrete particle swarm optimization." *Knowledge and information systems* 24, no. 2 (2010): 269-281.
- [88] Welhazi, Yosra, Tawfik Guesmi, and Hsan Hadj Abdallah. "Eigenvalue Assignments in Multimachine Power Systems using Multi-Objective PSO Algorithm." *International Journal of Energy Optimization and Engineering (IJEDE)* 4, no. 3 (2015): 33-48.

- [89] Xie, Xianchao, and Zhi Geng. "A recursive method for structural learning of directed acyclic graphs." *The Journal of Machine Learning Research* 9 (2008): 459-483.
- [90] Xing-Chen, Heng, Qin Zheng, Tian Lei, and Shao Li-Ping. "Learning bayesian network structures with discrete particle swarm optimization algorithm." In *Foundations of Computational Intelligence, 2007. FOCI 2007. IEEE Symposium on*, pp. 47-52. IEEE, 2007.
- [91] Xing-Chen, Heng, Qin Zheng, Tian Lei, and Shao Li-Ping. "Research on structure learning of dynamic Bayesian networks by particle swarm optimization." In *Artificial Life, 2007. ALIFE'07. IEEE Symposium on*, pp. 85-91. IEEE, 2007.
- [92] Yuan, Changhe, and Brandon Malone. "Learning Optimal Bayesian Networks: A Shortest Path Perspective." *J. Artif. Intell. Res.(JAIR)* 48 (2013): 23-65.
- [93] Yuan, Changhe, Brandon Malone, and Xiaojian Wu. "Learning optimal Bayesian networks using A* search." In *IJCAI Proceedings-International Joint Conference on Artificial Intelligence*, vol. 22, no. 3, p. 2186. 2011.