

# A comparative study on swarm intelligence for structure learning of Bayesian networks

Junzhong Ji<sup>1</sup> · Cuicui Yang<sup>1</sup> · Jiming Liu<sup>2</sup> · Jinduo Liu<sup>1</sup> · Baocai Yin<sup>1</sup>

Published online: 27 June 2016  
© Springer-Verlag Berlin Heidelberg 2016

**Abstract** A Bayesian network (BN) is an important probabilistic model in the field of artificial intelligence and a powerful formalism used to describe uncertainty in the real world. As science and technology develop, considerable data on complex systems have been acquired by various means, which presents a significant challenge regarding how to accurately and robustly learn a network structure for a complex system. To address this challenge, many BN structure learning methods based on swarm intelligence have been developed. In this study, we perform a systematic comparison of three typical methods based on ant colony optimization, artificial bee colony algorithm, and bacterial foraging optimization. First, we analyze and summarize their main characteristics from the perspective of stochastic searching. Second, we conduct thorough experimental comparisons to examine the roles of different mechanisms in each method by means of multiaspect metrics, i.e., the K2 score, structural differences, and execution time. Next, we perform further experiments to validate the robustness of different algorithms on some benchmark data sets with noise. Finally, we present

the prospects and references for researchers who are engaged in learning BN networks.

**Keywords** Bayesian network structure learning · Swarm intelligence · Ant colony optimization · Artificial bee colony algorithm · Bacterial foraging optimization

## 1 Introduction

Uncertainty is a natural phenomenon that exists widely in the real world, especially in the current era of large data sets. As powerful probabilistic graphical models, Bayesian networks (BNs) (Pearl 1988) have played an increasingly important role in modeling and reasoning with uncertainty. There are two main differentiated areas in the research of BNs: (1) learning a BN that represents a given data set and (2) providing an inference mechanism to issue queries and answers. Along with the development and popularity of machine learning and data mining, BNs have gradually been extensively developed and studied (Koller and Friedman 2009) and have been applied in a variety of different areas, including pattern recognition (Shan et al. 2009), data mining (Romero and Ventura 2010), neuroscience (Bielza and Larrañaga 2014), computational biology (Needham et al. 2007), brain information processing (Mumford and Ramsey 2014), and risk analysis (Weber et al. 2012).

Learning a BN from data includes two main subtasks, first to learn the structure of a BN (i.e., identifying the topology of a BN) and second, to learn the numerical parameters associated with a BN structure (Daly et al. 2011). For the two subtasks, learning a BN structure is a fundamental task, and parameter learning is a subroutine in structure learning. Over the past two decades, learning a BN structure from data has received considerable attention.

---

Communicated by V. Loia.

✉ Junzhong Ji  
jjz01@bjut.edu.cn  
Cuicui Yang  
yangcc\_2008@163.com  
Jiming Liu  
jiming@comp.hkbu.edu.hk

<sup>1</sup> Beijing Municipal Key Laboratory of Multimedia and Intelligent Software Technology, College of Computer Science and Technology, Beijing University of Technology, Beijing, China

<sup>2</sup> Department of Computer Science and Technology, Hong Kong Baptist University, Kowloon Tong, Hong Kong

Researchers have proposed various algorithms (De Campos et al. 2002; Ji et al. 2013; Yang et al. 2016; De Campos and Huete 2000; Cheng et al. 1997; Heckerman 1998; Suzuki 1999; Wong et al. 1999; Alcobó 2004; Cooper and Herskovits 1992) in which the conventional methods mainly include the dependency analysis approach (De Campos and Huete 2000; Cheng et al. 1997) and the score-and-search approach based on a general search strategy (Alcobó 2004; Cooper and Herskovits 1992). The first approach constructs a BN structure by judging dependency and independence relationships among variables, which needs to perform dependency tests. The second approach constantly uses a search method to search for candidate network structures in the space of BN structures until the network structure with the best metric is found. Unfortunately, both conventional approaches have fatal drawbacks. The first approach has to perform an exponential number of dependency tests, which are computationally expensive and may be unreliable (Cheng et al. 1997; Cooper and Herskovits 1992). For the second approach, the space of candidate network structures increases extremely rapidly and becomes very large when the number of variables increases (Robinson 1977;

Chickering et al. 1994), which makes the second approach using deterministic search methods often trapped in a local optimum and not be able to obtain the best solution. To efficiently search the optimum or near-optimum in the whole candidate structure space of BNs, swarm intelligence (SI) methods which incorporate nature-inspired stochastic search were applied to learn BN structures from data in recent years and quickly became a new and significant focus of research.

Over the past decade, several SI methods, such as ant colony optimization (ACO) (Dorigo et al. 1996), artificial immune system (AIS) (Mori et al. 1993), particle swarm optimization (PSO) (Kennedy and Eberhart 1995), artificial bee colony algorithm (ABC) (Karaboga 2005), and bacterial foraging optimization (BFO) (Passino 2002), have been successfully applied to BN structure learning (BNSL). Among them (see Table 1), ACO-B (De Campos et al. 2002), ABC-B (Ji et al. 2013), and BFO-B (Yang et al. 2016) are three representative algorithms. ACO-B opened up the way to learn BN structures using SI methods. ABC-B and BFO-B are recent extensions of ABC and BFO for BNSL, respectively. Moreover, these three SI methods have the greatest

**Table 1** Different SI algorithms on BNSL

Type	Algorithm	References	Space	Representation	Metric	Solving mechanism
ACO	ACO-B	De Campos et al. (2002)	DAGs	Graph	K2	ACO + Local search
	ACO-K2SN	De Campos et al. (2008)	Orderings	Permutation	K2	ACO + K2SN <sup>a</sup> + Local search
	MMACO	Pinto et al. (2009)	DAGs	Graph	BDEu	MMPC <sup>b</sup> + ACO
	ACO-E	Daly and Shen (2009)	PDAGs	Graph	BDEu	ACO + Local search
	I-ACO-B	Ji et al. (2009)	DAGs	Graph	K2	CI <sup>c</sup> + ACO + Local search
	ChainACO&K2ACO HACO-B	Wu et al. (2010) Ji et al. (2011)	Orderings DAGs	Chain permutation Graph	K2 K2	ACO + K2 search CI + ACO + Simulation annealing
AIS	Copt-aiNet	Castro and Zuben (2005)	DAGs	Connectivity matrix	BIC	Copt-aiNet <sup>d</sup>
PSO	BNDP	Heng et al. (2007)	DAGs	Adjacency list	BDEu	PSO
	BN-BQPSO	Zhao et al. (2009)	DAGs	Connectivity matrix	BIC	PSO
	PSO-BNL	Wang and Yang (2010)	DAGs	Connectivity matrix	BDEu	PSO
	IBPSO	Li (2010)	DAGs	Connectivity matrix	MDL	PSO
	PSO-K2	Aouay et al. (2013)	Orderings	Permutation	K2	PSO + K2 search
ABC	ABC-B	Ji et al. (2013)	DAGs	Graph	K2	EABC <sup>e</sup>
BFO	BFO-B	Yang et al. (2016)	DAGs	Graph	K2	BFO

<sup>a</sup> K2SN: An extension of K2 algorithm, which does not require a given ordering (De Campos and Puerta 2001)

<sup>b</sup> MMPC: A local discovery algorithm called max–min parents and children (Tsamardinos et al. 2003)

<sup>c</sup> CI: The conditional independence test used by the dependency analysis approach to judge dependent and independent relationships among variables (Koller and Friedman 2009; De Campos and Huete 2000)

<sup>d</sup> Copt-aiNet: Artificial immune network for combinatorial optimization algorithm, which was proposed in Gomes et al. (2003)

<sup>e</sup> An extended ABC algorithm (Ji et al. 2013)

similarity in BNSL. They use the same representation and metric to find better BN structures in the same search space. Today, BNSL methods based on SI have drawn increasing attention, which is also verified by the state-of-the-art survey on SI (Martens et al. 2011). Authors regarded BNSL as an important application of SI in this survey. However, to the best of our knowledge, there is no previous survey that exclusively focuses on SI for BNSL. Therefore, we will conduct a thorough investigation of SI for BNSL in this paper, whose main contributions include the following aspects:

- (1) We classify the popular BNSL algorithms based on SI into five paradigms and present the search spaces, solution representation forms, metrics, and search mechanisms for different algorithms, which provide a deeper insight into BNSL methods based on SI, and can help the interested researchers to have a better understanding of this domain.
- (2) To explain in detail the working principle of SI methods for BNSL, we take ACO-B, ABC-B, and BFO-B as representatives, discuss their common characteristics, and compare their basic principles: stochastic search mechanisms, local optimization mechanisms, and information transmission mechanisms. And then, we conduct systematic experiments to validate the roles of important mechanisms using derived algorithms.
- (3) We further evaluate the properties of ACO-B, ABC-B, and BFO-B by assessing their robustness in comparison with two non-SI methods on data sets with noisy data.

The rest of this paper is organized as follows. Section 2 briefly presents an overview about the research on BNSL, especially the BN structure learning based on SI. In Sect. 3, we summarize the common characteristics and differences of ACO-B, ABC-B, and BFO-B. Next, experimental and comparison results are given and discussed in Sect. 4. Finally, Sect. 5 summarizes our conclusions and future directions.

## 2 BN and its structure learning

### 2.1 BN

A BN is a directed acyclic graph (DAG) (Bang-Jense and Gutin 2008):  $G = \langle X, A \rangle$ , in which each node  $X_i \in X$  represents a random variable in a domain and each arc  $a_{ij} \in A$  describes a direct dependence relationship between two variables  $X_i$  and  $X_j$ . Associated with each node,  $X_i$ , is a conditional probability distribution represented by  $\theta_i = P(X_i | \Pi(X_i))$ , which quantifies the degree to which the node  $X_i$  depends on its parents  $\Pi(X_i)$ . Because the graph structure  $G$  qualitatively characterizes the independence relationships

among random variables, and these conditional probability distributions quantify the strength of the dependency between each node and its parent nodes, it can be proved that a BN  $\langle X, A \rangle$  uniquely encodes the joint probability distribution of the domain variables  $X = \{X_1, X_2, \dots, X_n\}$  (Pearl 1988):

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

### 2.2 BNSL

The structure of a BN uncovers the underlying probabilistic dependence relationships among nodes and a set of assertions about conditional independencies, which is an important theoretical basis to learn a BN structure from data. The problem of learning a BN structure from data can be stated as follows: Given a sample data  $D = \{X[1], X[2], \dots, X[N]\}$ , where  $X[i]$  is an instance that contains a set of value assignments for  $n$  discrete domain variables, the learning goal is to find a BN structure that matches  $D$  as well as possible. Many algorithms have been proposed to learn a BN structure from data. According to implementation mechanisms, these algorithms are usually classified into two basic categories (Daly et al. 2011; Heckerman 1998; Buntine 1996): (1) the dependency analysis approach and (2) the score-and-search approach. The former views BNSL as a constraint satisfaction problem, whereas the latter takes it as an optimization problem. More specifically, the dependency analysis approach usually uses a statistical method to estimate dependent and independent relationships among variables in the domain and constructs an ideal BN by analyzing these properties. The score-and-search approach always uses a search method to find candidate network structures and uses a scoring metric to evaluate the fitness of candidate structures, until the network structure with the best score is found. The implementation of the dependency analysis approach is relatively simple. However, computations for high-order statistical testing are often complex and unreliable, and the learning quality is difficult to ensure (Cheng et al. 1997; Cooper and Herskovits 1992).

Learning BN structures is an NP-hard problem as the number of variables increases (Chickering et al. 1994), and exact search methods can do nothing when the candidate network space becomes large. Although some heuristic search methods, such as two famous algorithms, incremental hill-climbing search (IHCS) (Alcobó 2004) and max–min hill-climbing (MMHC) (Tsamardinos et al. 2006), can address the problem of large search space, they are local optimization techniques in essence; thus, they can get trapped in local optima. To solve this problem, some stochastic search methods have drawn great attention from all over the world. The stochastic search methods are global search and able to

search the whole space in a reasonable amount of time. They employ meta-heuristic and stochastic mechanisms to escape from a local optimal solution and obtain the global optimal solution or nearly global optimal solution. Up to now, several stochastic methods have been applied to BNSL. These methods can be divided into two categories: the evolutionary method and the SI method. The evolutionary method draws inspiration from evolution and natural genetics, such as genetic algorithm (Larrañaga et al. 2013) and evolutionary programming (Larrañaga et al. 2013). The SI method models the social behavior of certain living creature, such as PSO (Daly et al. 2011), ACO (Daly et al. 2011), AIS (Castro and Zuben 2005), ABC (Ji et al. 2013), and BFO (Yang et al. 2016).

### 2.3 BNSL based on SI

SI algorithms are meta-heuristic search methods that are motivated by the collective behavior of a group of living organisms (such as ants, bees, fish, or bacteria). The essential characteristic of SI is the global optimization mechanism of a population of individuals. Each individual in the population is a simple agent with limited capabilities, such as local interaction with other individuals and the environment, but they can cooperatively perform many complex tasks for their survival. From the perspective of self-organization, a swarm is a group of agents that cooperate to achieve a goal. These agents usually use local perception and moving rules to govern their actions and use interaction and feedback mechanisms to achieve a collective intelligence. That is, although there is no centralized control to guide the behavior of the agents, local interactions among agents always lead to the emergence of global behavior (Eberhart et al. 2001).

Since ACO was used for BNSL in 2002 (De Campos et al. 2002), SI has gradually become a very promising approach for BNSL. Up to now, there are five SI paradigms, i.e., ACO, AIS, PSO, ABC, and BFO, which have been applied in this domain. Table 1 summarizes some representative algorithms based on the five paradigms and lists the algorithm names, the corresponding references, the search spaces, the solution encodings, the scoring metrics, and the mechanisms. According to the different learning goals, there are three search spaces, namely DAGs, orderings (De Campos et al. 2008), and PDAGs (Daly and Shen 2009) (partially DAGs), which represent feasible BN structures, variable ordering combinations, and equivalence classes of BNs, respectively. Graphs, adjacency lists (Heng et al. 2007), and connectivity matrix (Zhao et al. 2009) are three available solution encodings in the search space of DAGs, and a graph is also used as a valid solution encoding in the search space of PDAGs, whereas permutation (De Campos et al. 2008) and chain permutation (Wu et al. 2010) are two

solution forms in the search space of orderings. Based on the penalized maximum likelihood or the marginal likelihood, four different scoring metrics Bayesian information criterion (BIC) (Schwarz 1978), Bayesian Dirichlet equivalence (BDeu) (Heckerman et al. 1995), K2 (a specific case of Bayesian Dirichlet metric) (Cooper and Herskovits 1992), and minimum description length (MDL) (Rissanen 1978) are used to evaluate the quality of solutions in the search process. The last column in the table gives the problem-solving mechanism that corresponds to each algorithm.

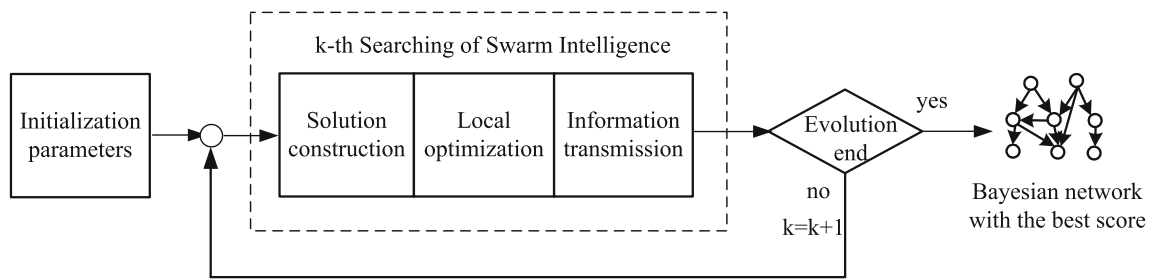
## 3 Qualitative comparisons of ACO-B, ABC-B, and BFO-B

To explain in detail the working principle of SI methods for BNSL, this section takes ACO-B, ABC-B, and BFO-B as examples to discuss their common characteristics and compare their basic principles.

### 3.1 Common characteristic and framework

The ACO-B algorithm (De Campos et al. 2002) is a score-and-search approach for learning BN structures based on ACO, whose main idea is to use ant colony stochastic searching to construct and search for a global optimal solution in the feasible solution space. The ABC-B algorithm (Ji et al. 2013) extends ABC optimization to learn the BN structure. The important characteristic of ABC-B is that employed bees and onlooker bees carry out neighbor searches, while scout bees construct new solutions in the feasible solution space. The BFO-B algorithm is a new approach just proposed in Yang et al. (2016) that makes use of BFO to learn a BN structure. The key feature is that BFO-B merges three principal schemes (i.e., chemotaxis, reproduction, and elimination and dispersal) into the score-and-search process and attempts to achieve balance between exploitation and exploration.

Although the three paradigms simulate different behavioral models and make use of different search methods, their common characteristic is that solution searches are combined into the meta-heuristic optimization mechanism. From the perspective of basic processes, they all perform three phases: solution construction, local optimization, and information transmission. Figure 1 roughly shows the basic flow diagram that uses these SI techniques to learn a BN structure. In general, a swarm in these algorithms can be viewed as a group of agents to cooperatively obtain the BN structure with the best score, in which each agent represents a feasible solution. In this process, all solutions in a swarm are iteratively updated and optimized by swarm foraging behaviors and gradually converge near the global optima.



**Fig. 1** The flow diagram of employing SI for learning BN structures

**Table 2** Feature comparisons of the three learning algorithms based on SI

Algorithms	Agent	Biological phenomena	Main features in three common phases		
			Solution construction	Local optimization	Information transmission
ACO-B	Ant	Ant colony foraging	Locally percept and randomly search ( $\alpha$ , $\beta$ and $q_0$ )	Simple operators ( $l_{step}$ )	Pheromone ( $\rho$ , $\varphi$ and $\tau_0$ )
ABC-B	Bee	Bee colony foraging	Locally percept and randomly search ( $\alpha$ , $\beta$ and $q_0$ )	Neighbor search ( $q_d$ and $limit$ )	Dancing and inductive pheromone ( $\rho$ and $\tau_0$ )
BFO-B	Bacterium	Bacterial flora foraging	Randomly add a small amount of arcs (-)	Swimming and tumbling ( $N_s$ , $N_c$ , $N_{re}$ , $N_{ed}$ and $P_{ed}$ )	Reproduction ( $S_r$ )

Specifically, agents at each iteration use local perception rules to construct their solutions, use simple operators to locally optimize solutions in their neighboring regions, have the aid of the interactions of the entire swarm to transmit the solution information, and eventually achieve their search objectives.

**3.2 Feature comparisons of ACO-B, ABC-B, and BFO-B**

Table 2 lists the biological principles simulated and the main features embodied in the three common phases of BNSL for the three swarm paradigms, in which the control parameters involved in the corresponding phase are provided in parentheses and “-” represents no parameter.

(1) *Agent and biological phenomena* The three algorithms use different agents to simulate different biological phenomena. In ACO-B, an ant is viewed as an agent with sensing and cognitive capabilities. In nature, ants can find the shortest path from their nest to a food source by exploring and exploiting pheromone information that has been deposited on the path. By simulating the behavior of ant colony foraging, ACO-B attempts to search for the BN structure with the highest K2 score in the candidate solution space.

As for ABC-B, three kinds of bees (i.e., employed bees, onlookers, and scout bees) are viewed as different agents that play their respective roles and collaboratively achieve a common task. The intelligent foraging behavior of a honey bee swarm can be briefly described as follows. The employed and onlooker bees carry out exploitive searches in the local area, while the scouts perform exploratory searches in a candidate solution space. Through mutual cooperation and collaboration, the three types of bees efficiently perform the collecting nectar task. By modeling the process of the bee colony collecting nectar, ABC-B attempts to find the BN structure with the best score.

For BFO-B, some species of bacteria like *Escherichia coli* are viewed as a group of agents that can move, reproduce, and perform elimination–dispersal under certain conditions in their lifetimes. As a natural ecosystem, bacterial foraging behavior exhibits chemotactic activity, where the alternation between swimming and tumbling enables each bacterium to search for more nutrients in random directions. To mimic bacterial foraging behavior, BFO-B uses some simple but powerful optimization operators to learn BN structures from a data set.

(2) *Solution construction* The solution construction is one of the most important phases for the three swarm algorithms. The three algorithms have some differences



in solution construction. In ACO-B, each ant uses its perception ability for the pheromone and heuristic information of arcs, starts from an empty graph (arcs-less DAG), and proceeds by adding new arcs to the current graph one by one until there is no way to make the score of a new solution higher. This process is used for all ants in each main loop and has the most core activity to obtain the optimal BN structure. In ABC-B, each employed bee perceives its local environment in traversing nodes and randomly adds some new arcs by means of the inductive pheromone and heuristic information associated with arcs in the initialization phase. In addition, a scout bee also performs a similar process during the exploratory phase. In BFO-B, each bacterium randomly adds a small number of arcs onto an empty graph and carries out the solution construction process to obtain a new initial solution in the initialization and elimination–dispersal phases. For ABC-B and BFO-B, the aim of the construction solutions is to have some new starting points in the search area.

- (3) *Local optimization* The three algorithms have different local optimization mechanisms. To improve the quality of a solution, ACO-B periodically makes use of an extra optimization process that uses the simple operators of addition, deletion, and reversal of arcs to locally optimize the obtained solution. For ABC-B, both employed bees and onlookers perform local neighbor search operators to optimize their associated current solutions each time, which is the most important step for ABC-B to obtain better results. In BFO-B, bacterial chemotaxis is a complex and close combination of swimming and tumbling that keeps bacteria in places with higher scores of BN structures. On the one hand, swimming is more frequent when a bacterium approaches a better solution using the same local optimization operator. On the other hand, tumbling controls the change among different local optimization operators, which is more frequent when a bacterium moves away from a certain solution to search for a better one. This local optimization process plays a crucial role in a chemotaxis activity.
- (4) *Information transmission* Based on different biological principles, the three algorithms use different mechanisms for information transmission. Because pheromones are an important medium in ant colony communication, local and global updating steps are included in ACO-B. The local updating step is performed when each ant adds an arc during the solution construction, and the global updating step is carried out for each arc of the best solution after the ants have executed their searches at every iteration. Obviously, the information transmission mecha-

nism not only keeps the swarm information feasible but also reinforces the effect of the optimal solution on the search process, so it plays an important role in determining the BN structure with the highest score.

In ABC-B, there are two ways to transmit information about solutions among bees. One is the behavior communication (dancing) used in the neighbor search process, that is, employed bees share information about food sources with onlooker bees according to the duration of a dance. The other is a chemical communication (inductive pheromone) in the constructing solution process. The inductive pheromone is a kind of releaser pheromone that is left by the scouts when they walk and is a useful information carrier in constructing solutions. In ABC-B, both methods of communication strengthen the collaboration among bees; this can not only better maintain balance between exploitation and exploration but can also more effectively look into promising regions of the search space.

For BFO-B, an information exchange mechanism implicitly occurs in the reproduction process, in which half of the population of bacteria with higher scores survives, while the other half of the population of bacteria with lower scores dies. In essence, the process represents a fairly abstract model of Darwinian evolution and biological genetics in genetic algorithms, thus achieving the elite information to delivery among swarm agents.

As discussed above, there are several common phases in the SI search, but different algorithms have specific features and mechanisms, where the most significant differences in these algorithms are, and how these different mechanisms play their roles in learning the BN structures, which are worth of in-depth research.

#### 4 Simulation comparisons of ACO-B, ABC-B, and BFO-B

In this section, we test the performance of ACO-B, ABC-B, and BFO-B algorithms from two perspectives. First, we compare each algorithm with its variants to show the effects of their characteristic mechanisms (see Sect. 3.2). Second, the three algorithms are compared with two non-SI algorithms on a set of benchmark data sets with noisy data to check their robustness. To ensure fair comparison, the population sizes of the three SI algorithms are set to 80. The other specific parameters of these algorithms and two non-SI algorithms conform to the best settings as reported in their original papers, which were tuned by a series of hand-tuning experiments. The experimental platform is a PC with Core 2, 2.13 GHz CPU, 2.99 GB RAM, Windows XP, and all algorithms are implemented using Java language.

**Table 3** Data sets used in our experiments

Data set (D)	Original network (G)	Size of D	Number of nodes	Number of arcs	K2 scoring of G
Alarm-1000	Alarm	1000	37	46	-5034.53
Alarm-2000	Alarm	2000	37	46	-9729.13
Alarm-3000	Alarm	3000	37	46	-14412.69
Alarm-4000	Alarm	4000	37	46	-19110.77
Alarm-5000	Alarm	5000	37	46	-23793.81
Alarm-6000	Alarm	6000	37	46	-28358.21
Child-3000	Child	3000	20	25	-15978.37
Credit-3000	Credit	3000	12	12	-13844.74

### 4.1 Benchmark data sets

We adopt the general method to evaluate a BNSL algorithm based on SI. We test a algorithm on data sets generated from famous benchmark networks using probabilistic logic sampling (Henrion 1986). The first network is the Alarm network (Beinlich et al. 1989), which is used for potential anesthesia diagnosis in the operating room and contains 37 nodes and 46 arcs. The second is the Child network (Spiegelhalter et al. 1993), which is applied to a preliminary diagnosis for newborn babies with congenital heart disease and consists of 20 nodes and 25 arcs. The third is the Credit network for assessing an individuals’ credit worthiness that was provided by Gerardina Hernandez as class homework at the University of Pittsburgh. This network is available in the GeNie software or <https://dslpitt.org/genie/> and often appears as an example in textbooks about data mining and contains 12 nodes and 12 arcs. All of the data sets used in our experiments are generated from these three networks. Table 3 shows a summary of the data sets, which include the name of the data set (D), the name of the original network graph (G), the size of the data set, the number of nodes in G, the number of arcs in G, and the K2 score for the original network structure under a given data set used in the experiments.

### 4.2 K2 scoring metric

As described in Sect. 2, there are four popular scoring metrics: K2, BIC, BDEu, and MDL. In this paper, we use the K2 metric due to the following two reasons: (1) None of the four metrics has been proven to be superior to others (Tsamardinos et al. 2006). (2) The five algorithms used in the experiments all adopted the K2 score in their original papers. The K2 metric is one of the most well-known Bayesian scoring methods that can measure how well a BN structure matches a data set. Because the scoring metric is first used in the K2 algorithm (Cooper and Herskovits 1992), it is referred as the K2 metric. Let  $D$  be a given data set and let  $G$  be a possible network structure that contains all of the variables in  $X$ . Each variable

$X_i \in X$  has  $r_i$  possible value assignments:  $(v_{i1}, \dots, v_{ir_i})$  and a set of its parent nodes, which can be represented with a list of variables  $\Pi(X_i)$ .  $q_i$  is the number of possible configurations for the variables in  $\Pi(X_i)$  relative to  $D$ .  $N_{ijk}$  is the number of cases in  $D$  where  $X_i = v_{ik}$  and  $\Pi(X_i)$  are instantiated to the  $j$ th configuration. The initial expression of the K2 metric is:

$$P(G, D) = P(G) \cdot \prod_{i=1}^n \prod_{j=1}^{q_i} \frac{(r_i - 1)!}{(N_{ij} + r_i - 1)!} \prod_{k=1}^{r_i} N_{ijk}!, \quad (2)$$

where  $N_{ij} = \sum_{k=1}^{r_i} N_{ijk}$ .

To obtain a decomposable metric, the logarithm of the above formula is used as the K2 scoring metric  $f(G : D)$  instead of  $P(G, D)$ , and the constant of  $\log P(G)$  is ignored when assuming a uniform prior for each  $P(G)$  (De Campos et al. 2002). Thus, the simplified metric  $f(G : D)$ , which evaluates  $G$  with respect to  $D$ , can be decomposed in the following way:

$$f(G : D) = \log(P(G, D)) \approx \sum_{i=1}^n f(X_i, \Pi(X_i)), \quad (3)$$

where  $f(X_i, \Pi(X_i))$  represents a K2 score for each node  $X_i$  in  $G$  and is formally defined as:

$$f(X_i, \Pi(X_i)) = \sum_{j=1}^{q_i} \left( \log \left( \frac{(r_i - 1)!}{(N_{ij} + r_i - 1)!} \right) + \sum_{k=1}^{r_i} \log(N_{ijk}!) \right). \quad (4)$$

The formula denotes that the K2 score of each node is only related to the local structure involving the node and its parent nodes. Because the joint probability for each candidate network is less than 1, the K2 metric using  $\log(P(G, D))$  always takes a negative value. Whatever form  $(P(G, D)$  or  $\log(P(G, D)))$  takes, the K2 metric is essentially a Bayesian

scoring metric. Hence, the best K2 value is the largest one that corresponds to the optimal BN structure.

### 4.3 Metrics of performance

Generally, there are two basic ways to use the score-and-search approach to evaluate the learned results. One way is based on a scoring metric (e.g., the K2 scoring metric in this paper), whereas the other is based on the metric of structural differences between the learned network and the original one. According to the first way, a higher K2 score of the learned network indicates a better corresponding algorithm. In contrast, a smaller structural difference indicates a better learning algorithm in the view of the latter metric. Essentially, the network with the highest K2 score is not necessarily the network with the smallest structural difference; thus, there may be conflict between the two ways. In this paper, from the perspective of optimization, we put the K2 scoring metric in the first place to evaluate the learned results. The evaluation metrics used in this section are listed as follows.

- HKS: the Highest K2 Score obtained over all trials.
- LKS: the Lowest K2 Score obtained over all trials.
- AKS: the Average K2 Score obtained over all trials.
- BSD: the Biggest Structural Difference (including arcs accidentally added, deleted, and inverted) obtained over all trials.
- SSD: the Smallest Structural Difference over all trials.
- ASD: the Average Structural Difference obtained that converges over all trials.
- SET: the Smallest Execution Time obtained over all trials.
- LET: the Longest Execution Time obtained over all trials.
- AET: the Average Execution Time obtained over all trials.

Obviously, HKS, LKS, and AKS are three scoring metrics; higher corresponding values indicate better learned networks are. BSD, SSD, and ASD are three structural difference metrics; smaller corresponding values indicate better learned networks are. The last three items are time metrics; smaller values indicate better time performance of the corresponding algorithm.

To study the comparison results with a certain confidence level, we also use the schema of statistical analysis in [Rubio-Largo et al. \(2012\)](#) to distinguish whether the algorithm performances differ significantly. According to the schema, we first perform Kolmogorov–Smirnov tests on the experimental results and know that our results do not follow a Gaussian distribution. Thus, we make use of Kruskal–Wallis tests to carry out nonparametric analysis on the results. In this paper, the confidence level is always set 95 %, which means the probability of producing a difference by chance is not more than 5 %. The statistical results are shown as  $p$  values. That is, if the  $p$  value obtained in a statistical test is less than

5 %, we can assume that significant differences exist in the corresponding experimental results.

In addition to the nine metrics mentioned above, we also use the log-loss metric to measure how well the learned networks model the target distribution. Generally, the higher the log-loss value, the better the algorithm. The process is as follows. Randomly generate a test data set with 10,000 cases and then compute the average log-loss of a learned network to the test data set in the case of a variety of data loss, where the average log-loss of a learned network on a test data set can be denoted as:

$$\ell = \frac{1}{Num.} \sum_{i=1}^{Num.} \log P_B(x^i) \quad (5)$$

where  $Num.$  is the size of the data set (i.e., 10,000),  $B$  is the learned network structure,  $x^i \in D$  is an instance in the data set, and  $P_B(x^i)$  is the probability that the instance happens in light of the current  $B$ .

### 4.4 Effects of different mechanisms in ACO-B, ABC-B, and BFO-B

To study which mechanism in the three paradigms strongly influence performance, we compared each algorithm with its variants on four data sets: Alarm-2000, Alarm-6000, Child-3000, and Credit-3000. Tables 4, 5, 6, and 7 give the corresponding results on the four data sets. The bold numbers in these tables indicate the best values for the corresponding metrics. More specifically, ACO-B1 and ACO-B2 are two variants of ACO-B. ACO-B1 removes the process of pheromone updating locally and globally while keeping the ant colony construction and local optimization processes. ACO-B2 removes both the local optimization process in the ant colony iterative search and the local optimization for ants in the last time. Similarly, ABC-B1 and ABC-B2 are two variants of ABC-B. ABC-B1 removes the effects of the inductive pheromone by removing inductive pheromone updating from ABC-B, whereas ABC-B2 removes scout exploring from ABC-B. As two variants of BFO-B, BFO-B1 removes the reproduction process from BFO-B, whereas BFO-B2 removes the elimination and dispersal process from BFO-B. These results in the four tables are obtained over 20 independent runs under the same parameter set for each algorithm. Table 4 shows the parameter sets for each basic paradigms, which are the optimal configurations provided in the original studies.

From the perspective of scoring metrics, the different mechanisms in ACO-B, ABC-B, and BFO-B play their respective roles in obtaining the best score. (1) ACO-B, whether by pheromone updating or by local optimization, has obvious roles in finding better solutions, and the former



**Table 4** Performance comparisons among each algorithm and its variations on Alarm-2000

Algorithm	Scoring			Structure			Time (s)		
	HKS	LKS	AKS <sup>d</sup>	SSD	BSD	ASD <sup>d</sup>	SET	LET	AET <sup>d</sup>
ACO-B <sup>a</sup>	<b>-9717.46</b>	<b>-9723.84</b>	<b>-9718.96 ± 2.18</b>	5	<b>13</b>	<b>7.40 ± 2.03</b>	221	416	286.2 ± 55.1
ACO-B1	-9720.09	-9734.17	-9727.53 ± 4.02	<b>3</b>	20	13.20 ± 4.38	<b>57</b>	<b>249</b>	215.6 ± 59.3
ACO-B2	<b>-9717.46</b>	-9726.96	-9722.34 ± 3.14	5	15	10.30 ± 2.51	146	272	<b>195.4 ± 39.5</b>
ABC-B <sup>b</sup>	<b>-9717.46</b>	<b>-9723.48</b>	<b>-9718.15 ± 1.80</b>	5	<b>9</b>	<b>5.90 ± 1.61</b>	82	116	101.6 ± 10.9
ABC-B1	<b>-9717.46</b>	-9733.10	-9721.45 ± 3.89	5	18	8.65 ± 3.02	68	90	82.5 ± 6.0
ABC-B2	-9723.55	-9733.25	-9727.36 ± 3.73	<b>4</b>	18	12.15 ± 3.94	55	<b>78</b>	<b>67.7 ± 5.6</b>
BFO-B <sup>c</sup>	<b>-9717.46</b>	<b>-9721.28</b>	<b>-9717.83 ± 0.98</b>	<b>3</b>	<b>9</b>	<b>5.25 ± 1.09</b>	<b>209</b>	<b>361</b>	<b>300.5 ± 40.1</b>
BFO-B1	<b>-9717.46</b>	-9723.34	-9718.86 ± 1.78	<b>3</b>	10	5.25 ± 2.12	253	366	308.5 ± 34.8
BFO-B2	<b>-9717.46</b>	-9725.97	-9718.72 ± 2.67	<b>3</b>	11	5.60 ± 2.03	263	401	324.6 ± 45.6

<sup>a</sup> The parameters of ACO-B:  $A = 80, \rho = \psi = 0.4, \beta = 2.0, q_0 = 0.8, N = 150$   
<sup>b</sup> The parameters of ABC-B:  $K = 80, \alpha = 1, \beta = 2, \rho = 0.1, q_0 = 0.8, N = 150, limit = 3$   
<sup>c</sup> The parameters of BFO-B:  $S = 80, N_s = 4, N_c = 30, N_{re} = 4, N_{ed} = 3, P_{ed} = 0.10$   
<sup>d</sup> Results in the form  $\mu \pm \delta$  indicate the mean  $\mu$  and the standard deviation  $\delta$

**Table 5** Performance comparisons among each algorithm and its variations on Alarm-6000

Algorithm	Scoring			Structure			Time (s)		
	HKS	LKS	AKS	SSD	BSD	ASD	SET	LET	AET
ACO-B	<b>-28347.11</b>	<b>-28353.88</b>	<b>-28348.19 ± 1.55</b>	<b>3</b>	<b>12</b>	<b>5.95 ± 2.33</b>	690	1248	914.8 ± 120.8
ACO-B1	-28350.24	-28387.34	-28365.45 ± 8.40	5	20	10.75 ± 4.22	<b>188</b>	<b>806</b>	<b>617.5 ± 239.3</b>
ACO-B2	-28347.14	-28375.23	-28354.83 ± 10.50	4	17	9.00 ± 4.36	471	1155	762.1 ± 145.2
ABC-B	<b>-28347.11</b>	-28360.11	<b>-28347.99 ± 2.83</b>	3	<b>8</b>	<b>4.95 ± 1.28</b>	284	346	319.9 ± 17.9
ABC-B1	-28347.21	-28370.60	-28354.63 ± 8.89	<b>1</b>	12	5.65 ± 2.73	<b>220</b>	281	250.5 ± 17.1
ABC-B2	-28358.31	-28387.85	-28373.45 ± 8.44	4	17	10.80 ± 3.47	184	<b>220</b>	<b>206.8 ± 8.98</b>
BFO-B	<b>-28347.11</b>	<b>-28349.19</b>	<b>-28347.90 ± 0.93</b>	<b>1</b>	<b>5</b>	2.60 ± 1.20	564	<b>810</b>	<b>684.7 ± 59.5</b>
BFO-B1	<b>-28347.11</b>	-28356.34	-28348.46 ± 2.57	<b>1</b>	6	3.00 ± 1.38	618	878	769.3 ± 60.5
BFO-B2	<b>-28347.11</b>	-28352.37	-28348.46 ± 1.32	<b>1</b>	7	<b>2.40 ± 1.56</b>	<b>540</b>	860	728.1 ± 77.5

**Table 6** Performance comparisons among each algorithm and its variations on Child-3000

Algorithm	Scoring			Structure			Time (s)		
	HKS	LKS	AKS	SSD	BSD	ASD	SET	LET	AET
ACO-B	<b>-15975.66</b>	<b>-15975.66</b>	<b>-15975.66 ± 0.00</b>	<b>1</b>	<b>1</b>	<b>1.00 ± 0.00</b>	15	40	26.1 ± 6.8
ACO-B1	<b>-15975.66</b>	-15976.38	-15975.76 ± 0.26	<b>1</b>	4	1.45 ± 1.07	<b>9</b>	37	<b>23.4 ± 10.4</b>
ACO-B2	<b>-15975.66</b>	-15976.82	-15975.75 ± 0.29	<b>1</b>	4	1.25 ± 0.77	13	<b>35</b>	25.5 ± 5.8
ABC-B	<b>-15975.66</b>	<b>-15976.38</b>	<b>-15975.69 ± 0.16</b>	<b>1</b>	<b>4</b>	<b>1.15 ± 0.65</b>	12	30	18.2 ± 4.9
ABC-B1	<b>-15975.66</b>	<b>-15976.38</b>	-15975.73 ± 0.22	<b>1</b>	<b>4</b>	1.30 ± 0.90	<b>9</b>	17	13.4 ± 1.9
ABC-B2	<b>-15975.66</b>	-15976.82	-15976.19 ± 0.36	<b>1</b>	<b>4</b>	3.05 ± 1.36	10	<b>15</b>	<b>12.6 ± 1.4</b>
BFO-B	<b>-15975.66</b>	<b>-15976.38</b>	<b>-15975.76 ± 0.26</b>	<b>1</b>	<b>4</b>	<b>1.45 ± 1.07</b>	52	<b>72</b>	62.6 ± 6.3
BFO-B1	<b>-15975.66</b>	<b>-15976.38</b>	<b>-15975.76 ± 0.26</b>	<b>1</b>	<b>4</b>	<b>1.45 ± 1.07</b>	56	91	71.3 ± 7.7
BFO-B2	<b>-15975.66</b>	<b>-15976.38</b>	-15975.87 ± 0.33	<b>1</b>	<b>4</b>	1.90 ± 1.37	<b>50</b>	73	<b>60.0 ± 5.9</b>

plays a greater role in relative terms. Compared with the results of ACO-B in Tables 4, 5, 6 and 7, the HKS values of ACO-B1 have some variances on Alarm-2000 (-9720.09) and Alarm-6000 (-28,350.24) and keep the same value as those of ACO-B on Child-3000 (-15,975.66) and Credit-

3000 (-13,842.16). For the HKS values of ACO-B2, there is little change on Alarm-6000. However, both the LKS and AKS values of ACO-B1 and ACO-B2 in Tables 4, 5 and 6 have obvious variances compared with the results of ACO-B. In general, the two types of values go bad when the ACO

**Table 7** Performance comparisons among each algorithm and its variations on Credit-3000

Algorithm	Scoring			Structure			Time (s)		
	HKS	LKS	AKS	SSD	BSD	ASD	SET	LET	AET
ACO-B	<b>-13842.16</b>	<b>-13842.16</b>	<b>-13842.16 ± 0.00</b>	<b>4</b>	<b>4</b>	<b>4.00 ± 0.00</b>	<b>2</b>	<b>3</b>	<b>2.5 ± 0.5</b>
ACO-B1	<b>-13842.16</b>	-13844.84	-13842.70 ± 1.07	<b>4</b>	6	4.40 ± 0.80	<b>2</b>	6	3.3 ± 1.1
ACO-B2	<b>-13842.16</b>	<b>-13842.16</b>	<b>-13842.16 ± 0.00</b>	<b>4</b>	<b>4</b>	<b>4.00 ± 0.00</b>	<b>2</b>	<b>3</b>	2.7 ± 0.5
ABC-B	<b>-13842.16</b>	<b>-13842.16</b>	<b>-13842.16 ± 0.00</b>	<b>4</b>	<b>4</b>	<b>4.00 ± 0.00</b>	<b>2</b>	<b>4</b>	2.8 ± 0.4
ABC-B1	<b>-13842.16</b>	-13847.52	-13843.65 ± 2.10	<b>4</b>	7	5.05 ± 1.43	<b>2</b>	9	3.5 ± 2.0
ABC-B2	<b>-13842.16</b>	-13847.74	-13845.74 ± 1.39	<b>4</b>	9	6.60 ± 0.92	<b>2</b>	5	<b>2.3 ± 0.6</b>
BFO-B	<b>-13842.16</b>	<b>-13844.00</b>	<b>-13842.35 ± 0.55</b>	<b>1</b>	<b>4</b>	<b>3.70 ± 0.90</b>	7	66	12.8 ± 13.9
BFO-B1	<b>-13842.16</b>	-13844.84	-13842.65 ± 1.02	<b>1</b>	7	3.95 ± 1.24	<b>6</b>	59	13.9 ± 13.0
BFO-B2	<b>-13842.16</b>	-13845.57	-13843.08 ± 1.33	<b>1</b>	7	4.10 ± 1.70	7	<b>9</b>	<b>7.9 ± 0.8</b>

paradigm does not use pheromone updating or local optimization. For the Alarm network, which has a larger number of nodes and number of arcs, the effect of pheromone updating has a more significant influence on the LKS and AKS values than does local optimization, such that the LKS values drop about 11 and 34 and the AKS values drop about 9 and 17 on Alarm-2000 and Alarm-6000, respectively. Of course, the role of local optimization should not be overlooked due to the fact that the LKS and AKS values drop about 21 and 6, respectively, on Alarm-6000. For both Child and Credit networks, which have fewer nodes and arcs, there are no significant differences in the role of the two mechanisms. For instance, the LKS and AKS values of ACO-B1 and ACO-B2 drop about 1 and 0.1, respectively, on Child-3000; the LKS and AKS values of ACO-B1 drop about 2 and 0.5, respectively; and LKS and AKS values of ACO-B2 have the same values as those of ACO-B on Credit-3000. These results also mean that the role of pheromone updating is greater than that of local optimization in learning BN network structures. (2) As for ABC-B, both inductive pheromone updating and scout exploring have obvious roles in finding better solutions, and the latter plays a greater role in relative terms. Based on the HKS values in Table 4, 5, 6, and 7, ABC-B1 without the inductive pheromone obtains the same results as ABC-B on Alarm-2000, Child-3000, and Credit-3000, whereas it achieves a slightly lower ( $-28,347.21$ ) value on Alarm-6000. ABC-B2 without scout exploring can obtain the same results as ABC-B on Child-3000 and Credit-3000, whereas it only achieves two obviously lower values ( $-9723.55$  and  $-28,358.31$ ) on Alarm-2000 and Alarm-6000. From the LKS and AKS metrics, both ABC-B1 and ABC-B2 achieve obviously smaller values than ABC-B except for Child-3000, which shows that the two mechanisms are necessary to obtain the optimal solution by means of ABC. According to the fluctuations in these values, ABC-B2 has worse results than ABC-B1, which means that the scout exploring process is a more important mechanism in ABC-B than the induc-

tive pheromone. (3) As for BFO-B, both the reproduction process and the elimination and dispersal process have a certain impact on solution scores, and there is little difference between the effects of the two processes. It is not difficult to find that both BFO-B1 and BFO-B2 can obtain the same HKS values as those of BFO-B on four different data sets and only obtain slightly worse AKS values than that of BFO-B except for BFO-B1 on Child-3000. For the LKS values, BFO-B1 and BFO-B2 gain the same result as BFO-B on Child-3000 and also obtain slightly lower values on Credit-3000 and worse results than BFO-B only on Alarm-2000 and Alarm-6000. These results reflect that the chemotaxis process is more essential to BFO-B than the other two mechanisms.

From the perspective of structural metrics, the use of mechanisms from ACO-B, ABC-B, and BFO-B affects the network structure obtained to a certain extent. (1) Although these variant algorithms remove some mechanisms, the basic random search frameworks are still in keeping with their respective paradigms. Therefore, there are few changes in the SSD values over 20 runs in Tables 4, 5, 6, and 7, whereas the SSD values of ACO-B1 on Alarm-2000 and Alarm-6000 differ slightly from that of ACO-B, the SSD values of ACO-B1 on Child-3000 and Credit-3000 are the same as that of ACO-B, the SSD value of ACO-B2 on Alarm-6000 is slightly larger than that of ACO-B, and the SSD values of ACO-B2 on the other three data sets are the same as those of ACO-B. ABC-B1 obtains better SSD values on Alarm-6000. ABC-B2 also obtains better SSD values on Alarm-2000 but worse results on Alarm-6000, which shows the randomness of ABC-B2. In other cases, ABC-B1 and ABC-B2 obtain the same SSD values as ABC-B. BFO-B1 and BFO-B2 obtain the same SSD values as BFO-B for all four data sets, which shows that the chemotaxis mechanism plays the greatest role in the BFO paradigm. (2) For the BSD metric, there are obvious changes in most cases because each paradigm is missing an important mechanism. When compared with ACO-B, the values for ACO-B1 decline by 7, 8, 3, and 2 for Alarm-2000,

Alarm-6000, Child-3000, and Credit-3000, respectively, and the values for ACO-B2 decline by 2, 5, and 3 for Alarm-2000, Alarm-6000, and Child-3000, respectively. Compared with ABC-B, the BSD values for ABC-B1 and ABC-B2 decline from 9 to 18 for Alarm-2000. The BSD value for ABC-B1 declines from 8 to 12 on Alarm-6000, whereas that for ABC-B2 declines from 8 to 17. The BSD value for ABC-B1 declines from 4 to 7 on Credit-3000, whereas that for ABC-B2 declines from 4 to 9. Compared with BFO-B, the BSD value of BFO-B1 declined 1, 1, and 3 for Alarm-2000, Alarm-6000, and Credit-3000, respectively, whereas those of BFO-B2 declined 2, 2, and 3, respectively. (3) In almost all cases, the ASD values of these variants become worse than those of their initial algorithms, mainly because of the fluctuations in the BSD values. From the ASD metric, BFO-B2 seems to obtain a better value than for BFO-B on Alarm-6000. However, the smaller average structural difference does not mean that the result is better; the ASD metric is a mathematical mean, so it implies only that BFO-B2 is more stable than BFO-B with these data.

From the perspective of time metrics, the execution time mostly depends on the mechanisms used in the three par-

adigms. As a whole, ACO-B1, ACO-B2, ABC-B1, and ABC-B2 use less time than their initial algorithms for complicated networks such as Alarm and Child because the removal of a certain mechanism from ACO-B or ABC-B may save time in obtaining an optimal result. However, BFO-B1 and BFO-B2 use more time than BFO-B for the same data sets because the reproduction process and the elimination and dispersal process are two indispensable mechanisms in the BFO paradigm; the deletion of any mechanism will affect the convergence of BFO-B, which causes an increase in the execution time. For the Credit network with less nodes and arcs, the three paradigms remain relatively stable on the SET metric; however, these variants give different time performances for the LET and AET metrics, which shows that application of SI to the solution of a simpler optimization problem has a chance of convergence. From Table 7, we learn that BFO-B2 can save much time in finding a simpler BN structure because it removes the elimination and dispersal process.

To further illustrate the roles of different mechanisms in the three algorithms, we performed Kruskal–Wallis tests on the results over 20 independent runs in pairs. Tables 8, 9, and 10 give the  $p$  values of the statistical tests for the ACO,

**Table 8** The  $p$  value<sup>a</sup> of Kruskal–Wallis test on different data sets for ACO-B<sup>b</sup>

Data sets	Algorithm	Scoring		Structure		Time	
		ACO-B1	ACO-B2	ACO-B1	ACO-B2	ACO-B1	ACO-B2
Alarm-2000	ACO-B	<b>2.765e-07</b>	<b>0.0004578</b>	<b>4.17e-05</b>	<b>0.0006286</b>	<b>0.0001147</b>	<b>1.042e-06</b>
	ACO-B1		<b>0.0002816</b>		<b>0.01042</b>		<b>0.002031</b>
Alarm-6000	ACO-B	<b>6.086e-07</b>	<b>0.004829</b>	<b>0.000101</b>	<b>0.00335</b>	<b>1.917e-05</b>	<b>0.003636</b>
	ACO-B1		<b>0.0009117</b>		0.1875		0.2914
Child-3000	ACO-B	0.07536	0.1521	0.07536	0.1521	0.5423	0.8921
	ACO-B1		0.5877		0.5877		0.6844
Credit-3000	ACO-B	0.07536	NA	0.07536	NA	<b>0.0042</b>	0.2094
	ACO-B1		0.07536		0.07536		<b>0.03562</b>

<sup>a</sup> The confidence level is 95 %

<sup>b</sup> The parameters of ACO-B:  $A = 80, \rho = \psi = 0.4, \beta = 2.0, q_0 = 0.8, N = 150$

**Table 9** The  $p$  value<sup>a</sup> of Kruskal–Wallis test on different data sets for ABC-B<sup>b</sup>

Data sets	Algorithm	Scoring		Structure		Time	
		ABC-B1	ABC-B2	ABC-B1	ABC-B2	ABC-B1	ABC-B2
Alarm-2000	ABC-B	<b>1.719e-05</b>	<b>4.262e-08</b>	<b>3.022e-06</b>	<b>6.525e-08</b>	<b>7.733e-07</b>	<b>6.125e-08</b>
	ABC-B1		<b>2.288e-05</b>		<b>0.002389</b>		<b>6.071e-07</b>
Alarm-6000	ABC-B	<b>3.022e-06</b>	<b>6.525e-08</b>	0.4218	<b>9.888e-07</b>	<b>6.257e-08</b>	<b>6.222e-08</b>
	ABC-B1		<b>4.803e-06</b>		<b>4.011e-05</b>		<b>6.691e-08</b>
Child-3000	ABC-B	0.5533	<b>2.924e-05</b>	0.5533	<b>3.364e-05</b>	<b>0.0002844</b>	<b>2.262e-08</b>
	ABC-B1		<b>0.0001301</b>		<b>0.0001678</b>		0.1811
Credit-3000	ABC-B	<b>0.004193</b>	<b>2.262e-08</b>	<b>0.004025</b>	<b>1.675e-08</b>	0.7862	<b>6.327e-05</b>
	ABC-B1		<b>0.005002</b>		<b>0.003905</b>		<b>0.008311</b>

<sup>a</sup> The confidence level is 95 %

<sup>b</sup> The parameters of ABC-B:  $K = 80, \alpha = 1, \beta = 2, \rho = 0.1, q_0 = 0.8, N = 150, limit = 3$

**Table 10** The  $p$  value<sup>a</sup> of Kruskal–Wallis test on different data sets for BFO-B<sup>b</sup>

Data sets	Algorithm	Scoring		Structure		Time	
		BFO-B1	ABC-B2	BFO-B1	BFO-B2	BFO-B1	BFO-B2
Alarm-2000	BFO-B	<b>0.02795</b>	0.4133	0.3643	0.9383	0.5699	0.1515
	BFO-B1		0.2300		0.2722		0.33
Alarm-6000	BFO-B	0.829	0.1794	0.3959	0.4324	<b>0.0002342</b>	<b>0.04248</b>
	BFO-B1		0.2267		0.1304		0.06989
Child-3000	BFO-B	1	0.262	1	0.262	<b>0.0006096</b>	0.1843
	BFO-B1		0.262		0.262		<b>6.466e-05</b>
Credit-3000	BFO-B	0.3376	<b>0.04531</b>	0.4327	0.3351	0.782	0.09457
	BFO-B1		0.2702		0.7304		0.1214

<sup>a</sup> The confidence level is 95 %.

<sup>b</sup> The parameters of BFO-B:  $S = 80$ ,  $N_s = 4$ ,  $N_c = 30$ ,  $N_{re} = 4$ ,  $N_{ed} = 3$ ,  $P_{ed} = 0.10$

ABC, and BFO paradigms, respectively. In the three tables, a  $p$  value  $< 0.05$  (in bold) indicates that the two corresponding algorithms have significant differences in the performance metric and vice versa.

Table 8 shows the  $p$  values of the Kruskal–Wallis tests on the four data sets for ACO-B, where NA represents a case in which both algorithms obtain the same results over 20 independent runs. (1) For a larger network like Alarm, both ACO-B1 and ACO-B2 have significant differences from the initial ACO-B on all three types of metrics, and in most cases, there are significant differences between them except for the structure and time metrics on Alarm-6000. (2) For Child-3000, no two of the three algorithms have a significant difference by means of  $p$  values. (3) For Credit-3000, ACO-B1 differs significantly from ACO-B and from ACO-B2 on the time metric. There are no significant differences for any two of the three algorithms on the scoring and structural metrics. These testing results show that pheromone updating and local optimization are two important processes in accurately learning a complex BN model, even though they cost more time, but they have no significant role on the solution quality when the BN model is simple.

As shown in Table 9, it is easy to see that: (1) From the perspective of the scoring metric, the scoring values obtained by ABC-B and ABC-B1 have significant differences for Alarm-2000, Alarm-6000, and Credit-3000; the scoring values obtained by ABC-B and ABC-B2 have significant differences for Alarm-2000, Alarm-6000, Child-3000, and Credit-3000; and the scoring values obtained by ABC-B1 and ABC-B2 also have significant differences for all four data sets. (2) From the perspective of the structural metric, the results obtained by ABC-B and ABC-B1 have significant differences for Alarm-2000 and Credit-3000 and the results obtained by ABC-B and ABC-B2 or by ABC-B1 and ABC-B2 have significant differences for all four data sets. (3) From the perspective of the time metric, the time cost of ABC-B and ABC-B1 differs significantly for Alarm-2000, Alarm-6000, and Child-3000; the time cost of ABC-B and

ABC-B2 differs significantly for four data sets; and the time cost of ABC-B1 and ABC-B2 differs significantly for Alarm-2000, Alarm-6000, and Credit-3000. These test results show that inductive pheromone updating and scout exploring are two basic mechanisms that have significant effects on the use of the ABC paradigm to learn BN structures. Moreover, the two mechanisms have very different roles and thus cannot be alternatives to one another.

Table 10 gives some interesting information on BFO-B: (1) From the perspective of the scoring metric, the scoring values obtained by BFO-B and BFO-B1 have significant differences only on Alarm-2000, whereas the scoring values obtained by BFO-B and BFO-B2 also have significant differences only for Credit-3000. (2) From the perspective of the structural metric, the results obtained by any two algorithms have no significant difference for all four data sets. (3) From the perspective of the time metric, the time costs of BFO-B and BFO-B1 have significant differences for Alarm-6000 and Child-3000, the time costs of BFO-B and BFO-B2 have a significant differences for Alarm-6000, and the time costs of BFO-B1 and BFO-B2 also have significant differences for Child-3000. These test results show that the reproduction process and the elimination and dispersal process have certain effects on the solution scoring and a great improvement in the execution time, but there is no significant role in the structural optimization, which just demonstrates that the chemotaxis process is the most important and indispensable mechanism for the BFO paradigm to learn BN structures from another point of view.

#### 4.5 Robustness comparisons

SI is a kind of intelligent algorithms based on population. The intelligence emerges from the swarm and does not directly from the single agent. An individual agent that fails in the process of solving a problem will not affect functionality of the whole swarm, which naturally gives SI good robustness to noise. In this section, we perform a large number

of experiments on Alarm data sets with noise to evaluate the robustness properties of ACO-B, ABC-B, and BFO-B. There are two reasons for only choosing Alarm data sets to do the experimental comparisons: (1) Alarm network is the most popular benchmark in the domain of learning BN structures. Nearly all the algorithms mentioned in this paper used the Alarm network to test the performance of algorithm in their original study. (2) Alarm network is more complex than another two networks (Child and Credit) used in this paper. The more complex the network, the more difficult for a algorithm to learn its structure. For a algorithm for BNSL, the performance played on a complex network more fully embodies its ability to learn a BN structure. Based on the two reasons described above, we only select data sets generated from Alarm network in this section. To show the better robustness of SI, we compare the three algorithms with two non-SI methods: HCST (hill-climbing searcher technique) and TS-B (tabu search for BNs). According to the basic idea of a greedy construction heuristic in [Alcobó \(2004\)](#), [Tsamardinos et al. \(2006\)](#), we developed an HCST algorithm to learn BNs from data that uses the three operators of arc addition, arc deletion, and arc reversion to maximize the increase in the K2 scores. TS-B ([Ji et al. 2008](#)) is an effective global searching approach that makes use of the tabu list and aspiration criteria to guide the search of BN structures, and performs local neighborhood searches by adding arc, reducing arc, and reversing arc operators. Obviously, HCST is a simple and common heuristic-based approach, whereas TS-B is a stochastic global optimization algorithm. The main reason for the selection of these two algorithms is similarity of their three operators with those of the three SI algorithms.

Based on benchmark data sets, we first generate test databases with noisy data. The method is as follows. (1) Randomly generate two integers  $i$  and  $j$  and perform a modulo operation on the two numbers:  $i \% SD$  and  $j \% NN$  ( $SD$  represents the size of the data set and  $NN$  represents the number of nodes in the network). (2) Take the  $i \% SD$  case and the  $j \% NN$  node as the location of the insertion of noise. (3) Randomly generate an integer located at  $[0, NV)$  ( $NV$  represents the total number of candidate values for the variable at the current location) and replace the initial value with the new integer generated. (4) Repeat the three steps  $p * SD$  times to obtain the data set needed ( $p$  is a noise ratio). By means of this method, various test data sets with 5, 10, 15, 20, and 25 % noisy data are obtained.

The detailed comparison results on six Alarm data sets with different proportions of noisy data for five algorithms are presented in Tables 11, 12, 13, 14, 15, and 16, where the first column represents the noise ratio deliberately added; the second column lists the five algorithms used; the third, fourth, and fifth columns give three scoring values; the sixth, seventh, and eighth columns give three structure values; the

ninth, tenth, and eleventh columns give three time values; and the last column shows the average log-loss values.

- (1) From the perspective of scores in these six tables, we can see that each algorithm obtains different scoring values under different data capacities and different noise ratios; the more the noise is added, the lower the score becomes. The three SI algorithms acquire almost unanimously the best results in each case compared with non-SI methods. Among the SI methods, ABC-B gets the highest scores in most cases. TS-B sometimes obtain the best results, but is significantly inferior to three SI algorithms in most cases, and the difference becomes greater as the noise increases except in a few cases on Alarm-6000. HCST has the worst performance among the five algorithms in all cases, and the scoring values get worse as the data capacity increases. More strangely, HCST seems insensitive to the increase in the noise, because its performance is nearly identically poor in all cases. The reasons for these experimental results are as follows. ACO-B, ABC-B, BFO-B, and TS-B are global search methods and can obtain a near-optimal solution. Further, ACO-B, ABC-B, and BFO-B are three SI methods that make use of collective intelligence to search for optima, and are better than an independent search of a single agent like TS-B. So the three SI methods show the best performance and robustness. HCST is a local search method that only can obtain a local optimal solution in a larger solution space, so its performance in learning the BN structures is the poorest.
- (2) From the perspective of the structural differences in the six tables, we can see that when the data capacity increases, the structural differences generally decrease and the learning quality improve, which means that high-quality solutions are ensured using plenty of instances. The results obtained by ACO-B, ABC-B, and BFO-B are nearly the same, and overall BFO-B obtains the best results among the three algorithms. TS-B and HCST obtain larger structural differences than the three SI algorithms. More specifically, HCST performs better than TS-B when the data capacity is smaller, but TS-B greatly improves as the data capacity increases. These results show that the SI-based algorithms can obtain high-quality solutions in all cases and a local search like HCST only obtains better results than a global search like TS-B when the data capacity is smaller. Intuitively, a greater amount of noise leads to larger structural differences. However, because all of the test algorithms take the K2 score as an optimization goal, the network with the smallest structural difference does not actually correspond to the one with the best score. Moreover, the SI methods are a self-organizing search that has the ability to reduce the effects



**Table 11** Experimental results on Alarm-1000 with different proportions of noisy data for five algorithms

Noise ratio	Algorithm	Scoring			Structure			Time (s)			LogLoss
		HKS	LKS	AKS <sup>f</sup>	SSD	BSD	ASD <sup>f</sup>	SET	LET	AET <sup>f</sup>	
0 %	ACO-B <sup>a</sup>	-5023.28	-5030.06	-5025.63 ± 2.31	6	20	15.15 ± 4.32	93	182	123.5 ± 29.8	-15.4251
	ABC-B <sup>b</sup>	-5023.28	-5029.22	-5024.70 ± 1.75	9	23	16.00 ± 3.36	37	54	48.1 ± 4.1	-15.4231
	BFO-B <sup>c</sup>	-5023.34	-5035.45	-5026.53 ± 2.74	6	24	14.75 ± 5.87	178	282	222.8 ± 30.1	-15.4484
	TS-B <sup>d</sup>	-5031.78	-5031.78	-5031.78 ± 0.00	18	18	18.00 ± 0.00	115	119	116.0 ± 0.7	-15.4444
	HCST <sup>e</sup>	-5113.67	-5113.67	-5113.67 ± 0.00	16	16	16.00 ± 0.00	2	2	2.0 ± 0.0	-15.7282
5 %	ACO-B	-5099.10	-5104.62	-5101.50 ± 1.61	6	24	12.55 ± 4.88	56	157	115.4 ± 29.6	-15.4236
	ABC-B	-5099.07	-5104.43	-5101.02 ± 1.14	9	21	12.25 ± 3.77	28	66	48.1 ± 8.1	-15.4202
	BFO-B	-5099.10	-5107.11	-5101.58 ± 2.06	7	25	14.45 ± 5.66	160	316	239.1 ± 38.9	-15.4287
	TS-B	-5120.68	-5120.68	-5120.68 ± 0.00	19	19	19.00 ± 0.00	134	149	138.7 ± 4.4	-15.4883
	HCST	-5183.13	-5183.13	-5183.13 ± 0.00	16	16	16.00 ± 0.00	2	2	2.0 ± 0.0	-15.7281
10 %	ACO-B	-5160.90	-5167.21	-5163.23 ± 1.91	12	28	18.70 ± 4.73	72	194	128.0 ± 32.2	-15.4685
	ABC-B	-5160.90	-5165.45	-5162.22 ± 1.24	12	23	19.70 ± 3.41	34	53	46.8 ± 5.0	-15.4643
	BFO-B	-5161.46	-5167.13	-5163.80 ± 1.12	9	25	16.10 ± 4.93	151	285	227.9 ± 39.3	-15.4680
	TS-B	-5184.49	-5184.49	-5184.49 ± 0.00	20	20	20.00 ± 0.00	130	136	132.7 ± 2.0	-15.4830
	HCST	-5247.02	-5247.02	-5247.02 ± 0.00	16	16	16.00 ± 0.00	2	2	2.0 ± 0.0	-15.7273
15 %	ACO-B	-5192.38	-5197.78	-5194.51 ± 1.55	9	19	13.05 ± 2.62	57	188	112.8 ± 31.5	-15.4674
	ABC-B	-5192.69	-5196.54	-5193.28 ± 0.83	11	14	12.00 ± 1.10	36	50	43.7 ± 4.5	-15.4654
	BFO-B	-5193.36	-5196.83	-5194.59 ± 0.98	9	23	12.85 ± 3.45	161	293	222.8 ± 40.1	-15.4685
	TS-B	-5217.01	-5217.01	-5217.01 ± 0.00	21	21	21.00 ± 0.00	130	137	134.8 ± 3.3	-15.5019
	HCST	-5272.03	-5272.03	-5272.03 ± 0.00	16	16	16.00 ± 0.00	2	2	2.0 ± 0.0	-15.7283
20 %	ACO-B	-5310.35	-5315.36	-5312.67 ± 1.57	11	23	15.80 ± 3.82	66	157	107.7 ± 23.7	-15.4672
	ABC-B	-5310.35	-5313.01	-5312.28 ± 1.06	9	23	13.55 ± 3.49	29	52	40.5 ± 7.9	-15.4632
	BFO-B	-5311.86	-5317.08	-5313.60 ± 1.27	8	26	12.75 ± 4.09	175	292	241.8 ± 38.7	-15.4655
	TS-B	-5316.59	-5316.59	-5316.59 ± 0.00	21	21	21.00 ± 0.00	107	115	110.7 ± 2.7	-15.4972
	HCST	-5389.17	-5389.17	-5389.17 ± 0.00	20	20	20.00 ± 0.00	2	2	2.0 ± 0.0	-15.7421
25 %	ACO-B	-5322.78	-5322.78	-5325.71 ± 2.39	11	34	23.50 ± 6.09	61	167	123.5 ± 29.3	-15.3947
	ABC-B	-5322.78	-5331.39	-5325.14 ± 2.63	11	29	21.20 ± 4.80	39	56	47.6 ± 5.6	-15.3952
	BFO-B	-5324.57	-5331.26	-5323.17 ± 1.78	11	28	18.95 ± 4.90	180	280	226.0 ± 26.2	-15.4195
	TS-B	-5331.73	-5331.73	-5331.73 ± 0.00	20	20	20.00 ± 0.00	132	143	136.5 ± 0.00	-15.5245
	HCST	-5410.50	-5410.50	-5410.50 ± 0.00	19	19	19.00 ± 0.00	2	2	2.0 ± 0.0	-15.7377

<sup>a</sup> The parameters of ACO-B:  $A = 80, \rho = \psi = 0.4, \beta = 2.0, q_0 = 0.8, N = 150$   
<sup>b</sup> The parameters of ABC-B:  $K = 80, \alpha = 1, \beta = 2, \rho = 0.1, q_0 = 0.8, N = 150, \text{limit} = 3$   
<sup>c</sup> The parameters of BFO-B:  $S = 80, N_s = 4, N_c = 30, N_{re} = 4, N_{ed} = 3, P_{ed} = 0.10$   
<sup>d</sup> The parameters of TS-B:  $\text{maxiter} = 1200, \text{tabulength} = 112$   
<sup>e</sup> The parameters of HCST:  $\text{maxiter} = 1200$   
<sup>f</sup> Results in the form  $\mu \pm \delta$  indicate the mean  $\mu$  and the standard deviation  $\delta$

**Table 12** Experimental results on Alarm-2000 with different proportions of noisy data for five algorithms

Noise ratio (%)	Algorithm	Scoring					Structure					Time (s)			LogLoss
		HKS		LKS		AKS	SSD	BSD	ASD	SET	LET	AET			
0	ACO-B	-9717.46	-9723.84	-9718.96 ± 2.18	5	13	7.40 ± 2.03	221	416	286.2 ± 55.1	-15.4175				
	ABC-B	-9717.46	-9723.48	-9718.15 ± 1.80	5	9	5.90 ± 1.61	82	116	101.6 ± 10.9	-15.4156				
	BFO-B	-9717.46	-9721.28	-9717.83 ± 0.98	3	9	5.25 ± 1.09	209	361	300.5 ± 40.1	-15.4169				
	TS-B	-9717.54	-9717.54	-9717.54 ± 0.00	6	6	6.00 ± 0.00	158	162	159.9 ± 1.1	-15.4156				
	HCST	-9822.23	-9822.23	-9822.23 ± 0.00	11	11	11.00 ± 0.00	4	5	4.4 ± 0.5	-15.6288				
5	ACO-B	-9882.59	-9889.18	-9883.54 ± 1.97	5	11	6.20 ± 1.83	179	366	253.9 ± 49.5	-15.4169				
	ABC-B	-9882.59	-9883.07	-9882.66 ± 0.11	5	7	5.10 ± 0.44	81	111	89.3 ± 9.5	-15.4155				
	BFO-B	-9882.59	-9887.04	-9883.35 ± 1.55	5	8	5.15 ± 1.11	227	319	280.0 ± 29.6	-15.4173				
	TS-B	-9893.78	-9893.78	-9893.78 ± 0.00	11	11	11.00 ± 0.00	157	162	159.4 ± 1.4	-15.4260				
	HCST	-9990.21	-9990.21	-9990.21 ± 0.00	13	13	13.00 ± 0.00	4	5	4.1 ± 0.3	-15.6407				
10	ACO-B	-10016.87	-10023.21	-10018.13 ± 2.05	5	13	7.20 ± 2.52	124	324	217.4 ± 44.9	-15.4175				
	ABC-B	-10016.87	-10022.96	-10017.21 ± 1.32	5	9	5.60 ± 1.11	68	103	88.4 ± 9.1	-15.4155				
	BFO-B	-10016.87	-10034.11	-10018.88 ± 4.15	3	16	5.80 ± 2.95	199	371	294.4 ± 43.6	-15.4174				
	TS-B	-10024.54	-10024.54	-10024.54 ± 0.00	14	14	14.00 ± 0.00	154	157	155.6 ± 1.2	-15.4256				
	HCST	-10142.07	-10142.07	-10142.07 ± 0.00	16	16	16.00 ± 0.00	4	5	4.3 ± 0.4	-15.6101				
15	ACO-B	-10150.74	-10156.10	-10151.61 ± 1.20	5	11	8.05 ± 2.09	148	374	244.7 ± 49.6	-15.4154				
	ABC-B	-10150.74	-10151.92	-10151.01 ± 0.39	5	9	7.30 ± 1.14	70	98	86.9 ± 9.3	-15.4148				
	BFO-B	-10150.74	-10162.92	-10153.06 ± 3.30	5	13	7.30 ± 2.47	266	383	326.9 ± 39.8	-15.4170				
	TS-B	-10167.71	-10167.71	-10167.71 ± 0.00	22	22	22.00 ± 0.00	180	182	181.2 ± 0.9	-15.4264				
	HCST	-10275.87	-10275.87	-10275.87 ± 0.00	16	16	16.00 ± 0.00	4	4	4.0 ± 0.0	-15.6215				
20	ACO-B	-10301.42	-10308.54	-10302.81 ± 1.95	5	15	9.55 ± 3.19	167	301	224.0 ± 31.4	-15.4139				
	ABC-B	-10300.63	-10301.42	-10301.07 ± 0.40	7	13	9.00 ± 1.67	80	112	96.4 ± 9.0	-15.4125				
	BFO-B	-10300.63	-10308.76	-10303.28 ± 2.60	5	15	8.20 ± 2.40	216	405	314.0 ± 43.5	-15.4201				
	TS-B	-10310.50	-10310.50	-10310.50 ± 0.00	17	17	17.00 ± 0.00	202	235	216.3 ± 10.4	-15.6288				
	HCST	-10400.00	-10400.00	-10400.00 ± 0.00	14	14	14.00 ± 0.00	4	7	4.6 ± 0.9	-15.6394				
25	ACO-B	-10430.08	-10436.57	-10431.10 ± 1.37	5	11	8.75 ± 2.23	210	329	236.2 ± 38.4	-15.4180				
	ABC-B	-10430.08	-10431.44	-10430.46 ± 0.46	5	11	8.05 ± 1.88	67	108	90.1 ± 10.1	-15.4129				
	BFO-B	-10430.24	-10433.95	-10430.86 ± 0.99	5	14	7.10 ± 1.97	232	340	280.1 ± 34.8	-15.4170				
	TS-B	-10431.44	-10431.44	-10431.44 ± 0.00	11	11	11.00 ± 0.00	182	188	185.9 ± 1.5	-15.4178				
	HCST	-10524.18	-10524.18	-10524.18 ± 0.00	13	13	13.00 ± 0.00	4	5	4.3 ± 0.4	-15.6428				

**Table 13** Experimental results on Alarm-3000 with different proportions of noisy data for five algorithms

Noise ratio (%)	Algorithm	Scoring					Structure					Time (s)			LogLoss		
		HKS					AKS					SET				LET	AET
		LKS	AKS	SSD	BSD	ASD	SSD	BSD	ASD	SET	LET	AET					
0	ACO-B	-14412.26	-14401.29	-14410.41	-14404.09 ± 3.38	1	12	5.85 ± 2.90	297	562	380.5 ± 63.0	-15.4178					
	ABC-B	-14410.41	-14401.29	-14410.41	-14402.60 ± 2.76	3	10	5.50 ± 1.75	130	173	154.1 ± 13.9	-15.4170					
	BFO-B	-14412.98	-14401.91	-14412.98	-14403.97 ± 3.34	1	13	<b>3.95 ± 3.37</b>	295	456	375.0 ± 45.7	-15.4169					
	TS-B	-14401.29	-14401.29	-14401.29	-14401.29 ± 0.00	5	5	5.00 ± 0.00	209	211	209.5 ± 0.9	-15.4158					
	HCST	-14560.86	-14560.86	-14560.86	-14560.86 ± 0.00	10	10	10.00 ± 0.00	7	8	<b>7.8 ± 0.4</b>	-15.6290					
5	ACO-B	-14646.36	-14635.51	-14646.36	-14638.28 ± 4.00	3	11	5.70 ± 2.10	303	496	399.6 ± 61.8	-15.4183					
	ABC-B	-14636.36	-14635.51	-14636.36	-14635.65 ± 0.30	5	7	5.20 ± 0.60	107	174	147.6 ± 16.0	-15.4158					
	BFO-B	-14644.20	-14635.75	-14644.20	-14637.11 ± 2.58	1	6	<b>2.90 ± 1.04</b>	289	446	359.2 ± 38.7	-15.4170					
	TS-B	-14648.45	-14648.45	-14648.45	-14648.45 ± 0.00	8	8	8.00 ± 0.00	195	201	198.8 ± 1.8	-15.4266					
	HCST	-14792.08	-14792.08	-14792.08	-14792.08 ± 0.00	10	10	10.00 ± 0.00	7	7	<b>7.0 ± 0.0</b>	-15.6290					
10	ACO-B	-14814.40	-14804.11	-14814.40	-14805.79 ± 2.28	2	13	5.55 ± 2.58	358	567	402.5 ± 86.5	-15.4163					
	ABC-B	-14810.40	-14804.11	-14810.40	-14804.67 ± 1.46	2	9	4.25 ± 1.26	117	169	141.9 ± 17.0	-15.4158					
	BFO-B	-14813.26	-14804.11	-14813.26	-14805.75 ± 2.44	2	6	3.65 ± 1.06	343	477	401.0 ± 36.0	-15.4167					
	TS-B	-14805.81	-14805.81	-14805.81	-14805.81 ± 0.00	2	2	<b>2.00 ± 0.00</b>	238	245	240.1 ± 2.5	-15.4160					
	HCST	-14953.06	-14953.06	-14953.06	-14953.06 ± 0.00	11	11	11.00 ± 0.00	7	8	<b>7.1 ± 0.2</b>	-15.6287					
15	ACO-B	-15044.54	-15035.36	-15044.54	-15036.47 ± 2.04	5	10	7.70 ± 1.42	231	494	416.7 ± 79.6	-15.4175					
	ABC-B	-15044.53	-15035.36	-15044.53	-15036.68 ± 3.51	5	11	7.40 ± 1.32	116	168	145.0 ± 11.6	-15.4191					
	BFO-B	-15040.08	-15035.37	-15040.08	-15036.64 ± 1.90	4	11	6.40 ± 1.96	299	442	364.0 ± 36.6	-15.4172					
	TS-B	-15037.25	-15037.25	-15037.25	-15037.25 ± 0.00	6	6	<b>6.00 ± 0.00</b>	235	246	239.5 ± 2.3	-15.4170					
	HCST	-15186.71	-15186.71	-15186.71	-15186.71 ± 0.00	12	12	12.00 ± 0.00	7	7	<b>7.0 ± 0.0</b>	-15.6274					
20	ACO-B	-15247.18	-15247.18	-15247.18	-15248.67 ± 2.18	5	14	8.80 ± 2.44	332	495	389.9 ± 47.8	-15.4133					
	ABC-B	-15247.18	-15247.18	-15247.18	-15247.70 ± 2.06	8	13	<b>8.50 ± 1.36</b>	114	182	154.9 ± 18.0	-15.4130					
	BFO-B	-15256.92	-15247.18	-15256.92	-15252.30 ± 3.50	6	17	10.35 ± 2.74	324	439	376.5 ± 34.9	-15.4157					
	TS-B	-15293.55	-15293.55	-15293.55	-15293.55 ± 0.00	20	20	20.00 ± 0.00	251	262	254.7 ± 3.1	-15.4301					
	HCST	-15324.16	-15324.16	-15324.16	-15324.16 ± 0.00	14	14	14.00 ± 0.00	7	8	<b>7.1 ± 0.3</b>	-15.4785					
25	ACO-B	-15427.75	-15427.75	-15439.66	-15428.96 ± 2.57	7	16	9.35 ± 1.98	228	465	358.5 ± 61.7	-15.4153					
	ABC-B	-15427.75	-15427.75	-15429.44	-15428.00 ± 0.46	7	13	<b>9.00 ± 1.26</b>	116	179	150.2 ± 14.8	-15.4146					
	BFO-B	-15437.00	-15427.75	-15437.00	-15429.68 ± 3.02	7	15	9.65 ± 2.22	331	443	384.6 ± 32.1	-15.4155					
	TS-B	-15499.79	-15499.79	-15499.79	-15499.79 ± 0.00	21	21	21.00 ± 0.00	272	276	274.2 ± 1.6	-15.4413					
	HCST	-15548.43	-15548.43	-15548.43	-15548.43 ± 0.00	21	21	21.00 ± 0.00	7	7	<b>7.0 ± 0.0</b>	-15.5203					

**Table 14** Experimental results on Alarm-4000 with different proportions of noisy data for five algorithms

Noise ratio (%)	Algorithm	Scoring			Structure			Time (s)			LogLoss
		HKS	LKS	AKS	SSD	BSD	ASD	SET	LET	AET	
0	ACO-B	-19098.41	-19105.06	-19100.14 ± 1.87	2	10	5.90 ± 2.14	233	679	517.5 ± 121.0	-15.4156
	ABC-B	-19098.41	-19101.10	-19098.62 ± 0.62	5	7	5.20 ± 0.51	181	221	194.8 ± 15.1	-15.4158
	BFO-B	-19098.59	-19105.18	-19099.89 ± 2.15	2	9	<b>4.20 ± 1.78</b>	371	581	479.5 ± 53.5	-15.4160
	TS-B	-19098.41	-19098.41	-19098.41 ± 0.00	5	5	5.00 ± 0.00	212	218	214.3 ± 2.2	-15.4158
	HCST	-19324.95	-19324.95	-19324.95 ± 0.00	10	10	10.00 ± 0.00	9	<b>10</b>	<b>9.7 ± 0.5</b>	-15.6290
5	ACO-B	-19402.65	-19418.48	-19403.92 ± 3.43	4	11	5.45 ± 2.04	405	667	527.6 ± 90.7	-15.4164
	ABC-B	-19402.65	-19417.21	-19403.72 ± 3.21	2	7	<b>4.25 ± 1.37</b>	174	286	219.0 ± 31.2	-15.4164
	BFO-B	-19402.65	-19415.83	-19404.76 ± 4.13	2	8	4.40 ± 1.11	397	583	484.1 ± 41.6	-15.4166
	TS-B	-19487.43	-19487.43	-19487.43 ± 0.00	13	13	13.00 ± 0.00	186	187	186.0 ± 0.3	-15.4213
	HCST	-19640.71	-19640.71	-19640.71 ± 0.00	13	13	13.00 ± 0.00	9	<b>9</b>	<b>9.0 ± 0.0</b>	-15.5889
10	ACO-B	-19667.91	-19671.67	-19668.86 ± 1.14	1	7	4.00 ± 1.95	405	758	541.0 ± 87.2	-15.4162
	ABC-B	-19667.91	-19670.27	-19668.20 ± 0.64	1	7	3.40 ± 1.36	165	226	203.1 ± 19.2	-15.4162
	BFO-B	-19667.91	-19674.42	-19668.95 ± 1.63	1	8	<b>3.30 ± 1.71</b>	400	562	472.0 ± 40.9	-15.4168
	TS-B	-19746.35	-19746.35	-19746.35 ± 0.00	10	10	10.00 ± 0.00	177	183	179.2 ± 2.2	-15.4222
	HCST	-19894.43	-19894.43	-19894.43 ± 0.00	13	13	13.00 ± 0.00	9	<b>9</b>	<b>9.0 ± 0.0</b>	-15.5889
15	ACO-B	-19981.66	-19984.39	-19982.41 ± 0.93	5	10	6.95 ± 1.66	445	891	546.2 ± 104.3	-15.4119
	ABC-B	-19981.66	-19984.29	-19982.06 ± 0.72	7	10	7.40 ± 1.32	195	236	212.1 ± 12.5	-15.4119
	BFO-B	-19981.77	-19983.73	-19982.63 ± 1.09	3	11	<b>4.75 ± 1.73</b>	437	567	489.0 ± 38.1	-15.4125
	TS-B	-20048.36	-20048.36	-20048.36 ± 0.00	13	13	13.00 ± 0.00	188	252	212.3 ± 19.7	-15.4202
	HCST	-20197.82	-20197.82	-20197.82 ± 0.00	13	13	13.00 ± 0.00	9	<b>9</b>	<b>9.0 ± 0.0</b>	-15.5889
20	ACO-B	-20222.02	-20237.10	-20223.97 ± 4.19	5	13	6.85 ± 2.24	236	663	502.4 ± 87.2	-15.4152
	ABC-B	-20222.02	-20223.56	-20222.17 ± 0.37	3	7	<b>5.20 ± 0.87</b>	186	239	204.0 ± 18.2	-15.4139
	BFO-B	-20222.02	-20229.82	-20223.51 ± 2.40	3	10	5.25 ± 1.64	381	547	480.9 ± 43.4	-15.4156
	TS-B	-20276.00	-20276.00	-20276.00 ± 0.00	10	10	10.00 ± 0.00	181	183	182.6 ± 0.5	-15.4334
	HCST	-20311.05	-20311.05	-20311.05 ± 0.00	13	13	13.00 ± 0.00	9	<b>9</b>	<b>9.0 ± 0.0</b>	-15.4808
25	ACO-B	-20504.07	-20506.14	-20504.73 ± 0.72	10	13	11.20 ± 1.12	446	764	526.4 ± 71.7	-15.4153
	ABC-B	-20504.07	-20505.54	-20504.21 ± 0.36	10	12	11.70 ± 0.84	171	222	194.1 ± 13.9	-15.4156
	BFO-B	-20504.07	-20521.28	-20507.70 ± 3.68	8	14	<b>10.55 ± 1.76</b>	430	521	477.6 ± 30.3	-15.4157
	TS-B	-20575.02	-20575.02	-20575.02 ± 0.00	18	18	18.00 ± 0.00	204	206	205.1 ± 0.7	-15.4362
	HCST	-20705.93	-20705.93	-20705.93 ± 0.00	18	18	18.00 ± 0.00	9	<b>10</b>	<b>9.3 ± 0.4</b>	-15.5823

**Table 15** Experimental results on Alarm-5000 with different proportions of noisy data for five algorithms

Noise ratio (%)	Algorithm	Scoring					Structure					Time (s)			LogLoss																																
		HKS					LKS					AKS					SSD					BSD					ASD					SET					LET					AET					
0	ACO-B	-23781.98	-23788.58	-23783.32 ± 1.66	3	12	5.95 ± 1.99	413	798	647.6 ± 101.6	-15.4159																																				
	ABC-B	-23781.98	-23784.17	-23782.67 ± 2.17	3	13	6.40 ± 1.91	205	290	256.6 ± 20.5	-15.4158																																				
	BFO-B	-23782.15	-23792.47	-23784.18 ± 3.97	1	13	3.60 ± 2.52	515	655	585.2 ± 40.9	-15.4164																																				
	TS-B	-23879.78	-23879.78	-23879.78 ± 0.00	12	12	12.00 ± 0.00	271	281	274.6 ± 1.9	-15.4221																																				
	HCST	-24092.57	-24092.57	-24092.57 ± 0.00	13	13	13.00 ± 0.00	12	13	12.4 ± 0.5	-15.5889																																				
5	ACO-B	-24138.65	-24144.67	-24139.98 ± 1.41	2	12	6.55 ± 2.44	529	921	685.0 ± 111.4	-15.4153																																				
	ABC-B	-24138.65	-24140.04	-24138.77 ± 0.41	4	8	7.10 ± 1.18	237	307	268.8 ± 22.7	-15.4154																																				
	BFO-B	-24138.70	-24145.90	-24139.62 ± 1.65	2	13	4.45 ± 2.27	484	655	575.2 ± 45.4	-15.4156																																				
	TS-B	-24222.13	-24222.13	-24222.13 ± 0.00	15	15	15.00 ± 0.00	268	269	268.4 ± 0.1	-15.4212																																				
	HCST	-24428.15	-24428.15	-24428.15 ± 0.00	13	13	13.00 ± 0.00	11	13	11.5 ± 0.6	-15.5889																																				
10	ACO-B	-24564.57	-24582.05	-24566.37 ± 3.72	3	9	5.95 ± 1.77	326	746	627.4 ± 85.1	-15.4159																																				
	ABC-B	-24564.57	-24566.03	-24564.72 ± 0.45	3	9	5.10 ± 0.99	220	317	269.1 ± 22.0	-15.4154																																				
	BFO-B	-24564.57	-24573.95	-24565.24 ± 2.05	3	9	5.10 ± 1.18	531	689	587.4 ± 43.1	-15.4153																																				
	TS-B	-24643.66	-24643.66	-24643.66 ± 0.00	16	16	16.00 ± 0.00	255	256	255.2 ± 0.4	-15.4209																																				
	HCST	-24844.70	-24844.70	-24844.70 ± 0.00	14	14	14.00 ± 0.00	11	12	11.2 ± 0.4	-15.5885																																				
15	ACO-B	-24865.31	-24867.94	-24866.36 ± 0.61	7	13	9.30 ± 1.76	543	939	678.5 ± 106.8	-15.4111																																				
	ABC-B	-24865.31	-24866.08	-24865.61 ± 0.28	7	11	8.85 ± 1.15	215	318	267.6 ± 26.0	-15.4110																																				
	BFO-B	-24865.31	-24870.44	-24866.66 ± 1.55	7	10	7.85 ± 1.11	543	670	614.7 ± 38.9	-15.4113																																				
	TS-B	-24947.36	-24947.36	-24947.36 ± 0.00	21	21	21.00 ± 0.00	257	258	257.7 ± 0.1	-15.4160																																				
	HCST	-25135.66	-25135.66	-25135.66 ± 0.00	18	18	18.00 ± 0.00	11	12	11.0 ± 0.2	-15.5844																																				
20	ACO-B	-25275.81	-25286.39	-25277.48 ± 2.69	5	20	8.50 ± 3.14	285	888	652.1 ± 131.6	-15.4119																																				
	ABC-B	-25275.81	-25286.58	-25275.89 ± 0.23	8	9	7.20 ± 0.60	221	304	265.8 ± 21.5	-15.4116																																				
	BFO-B	-25275.81	-25288.19	-25278.00 ± 2.71	5	14	7.70 ± 2.32	511	678	624.0 ± 55.6	-15.4116																																				
	TS-B	-25339.71	-25339.71	-25339.71 ± 0.00	17	17	17.00 ± 0.00	305	306	305.7 ± 0.2	-15.4197																																				
	HCST	-25527.12	-25527.12	-25527.12 ± 0.00	17	17	17.00 ± 0.00	11	12	11.3 ± 0.5	-15.5816																																				
25	ACO-B	-25577.24	-25579.01	-25577.81 ± 0.59	9	14	10.20 ± 1.85	543	939	678.5 ± 106.8	-15.4135																																				
	ABC-B	-25577.24	-25583.96	-25577.85 ± 1.45	9	11	10.70 ± 1.73	230	320	259.7 ± 22.5	-15.4136																																				
	BFO-B	-25577.24	-25586.37	-25579.84 ± 2.17	7	14	9.85 ± 2.26	499	645	592.3 ± 41.3	-15.4127																																				
	TS-B	-25649.56	-25649.56	-25649.56 ± 0.00	15	15	15.00 ± 0.00	245	246	245.3 ± 0.2	-15.4206																																				
	HCST	-25822.71	-25822.71	-25822.71 ± 0.00	19	19	19.00 ± 0.00	11	12	11.2 ± 0.4	-15.5821																																				



**Table 16** Experimental results on Alarm-6000 with different proportions of noisy data for five algorithms

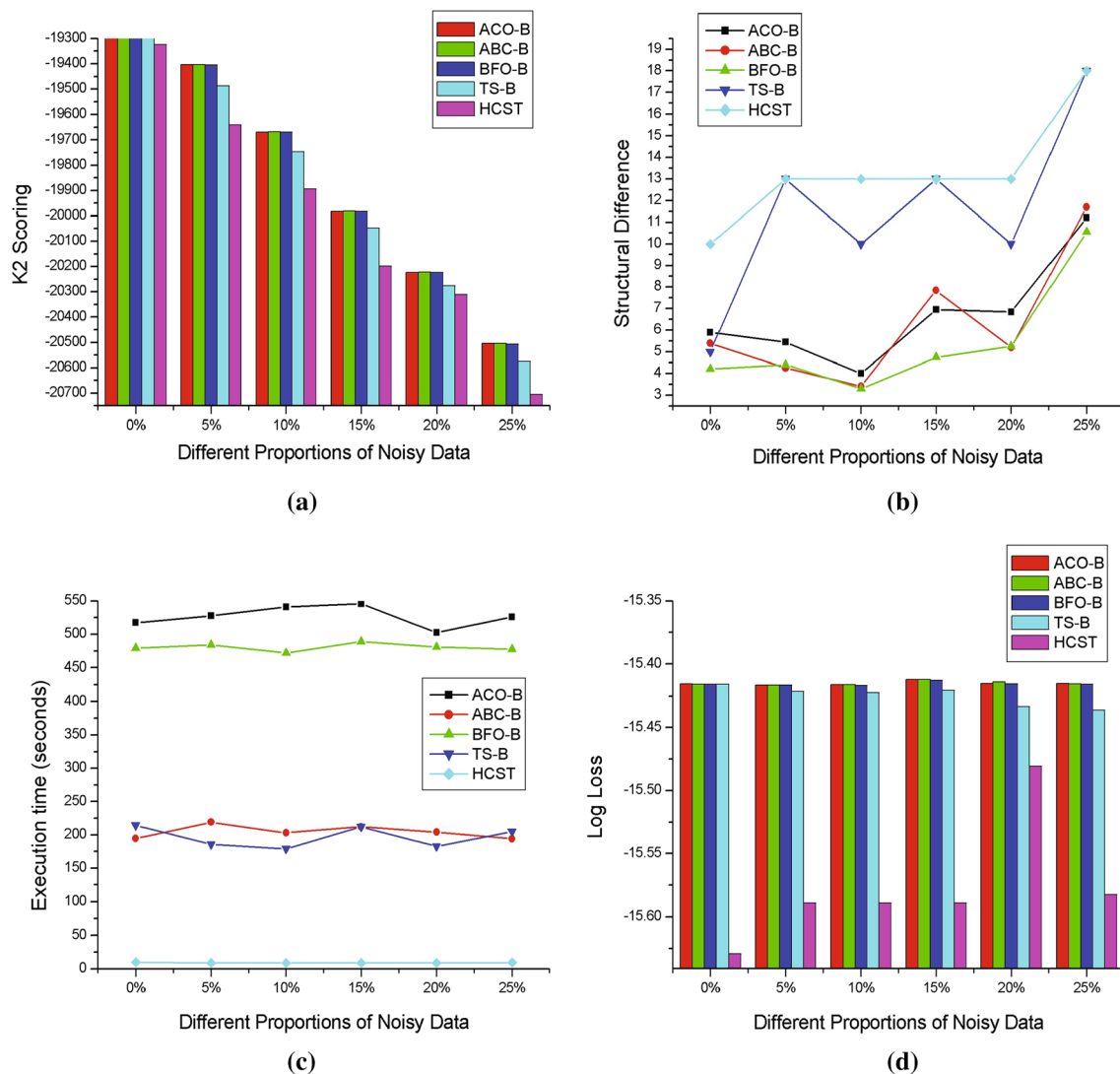
Noise ratio (%)	Algorithm	Scoring					Structure					Time (s)				LogLoss
		HKS	LKS	AKS	SSD	BSD	ASD	SET	LET	AET						
0	ACO-B	-28347.11	-28353.88	-28348.19 ± 1.55	3	12	5.95 ± 2.33	690	1248	914.8 ± 120.8	-15.4162					
	ABC-B	-28347.11	-28360.11	-28347.99 ± 2.83	3	8	4.95 ± 1.28	284	346	319.9 ± 17.9	-15.4161					
	BFO-B	-28347.11	-28349.19	-28347.90 ± 0.93	1	5	2.60 ± 1.20	564	810	684.7 ± 59.5	-15.4161					
	TS-B	-28450.76	-28450.76	-28450.76 ± 0.00	10	10	10.00 ± 0.00	258	263	260.6 ± 1.4	-15.4222					
	HCST	-28710.68	-28710.68	-28710.68 ± 0.00	14	14	14.00 ± 0.00	14	15	14.2 ± 0.4	-15.5841					
5	ACO-B	-28819.44	-28825.49	-28820.56 ± 1.45	1	10	4.50 ± 2.20	438	1052	797.9 ± 156.7	-15.4160					
	ABC-B	-28819.44	-28821.18	-28819.65 ± 0.47	1	7	3.30 ± 1.14	275	356	312.2 ± 22.5	-15.4162					
	BFO-B	-28819.44	-28822.52	-28819.88 ± 0.84	1	7	3.15 ± 1.19	613	775	708.0 ± 41.2	-15.4162					
	TS-B	-28909.32	-28909.32	-28909.32 ± 0.00	10	10	10.00 ± 0.00	250	251	250.7 ± 0.3	-15.4222					
	HCST	-28984.90	-28984.90	-28984.90 ± 0.00	13	13	13.00 ± 0.00	13	13	13.0 ± 0.0	-15.4760					
10	ACO-B	-29272.67	-29279.80	-29274.05 ± 1.45	4	12	7.70 ± 2.19	687	1099	879.9 ± 143.0	-15.4145					
	ABC-B	-29272.67	-29293.02	-29275.49 ± 4.76	5	12	8.20 ± 2.29	262	362	298.2 ± 22.1	-15.4148					
	BFO-B	-29272.70	-29279.80	-29274.29 ± 2.28	3	12	5.60 ± 2.58	611	768	682.7 ± 47.5	-15.4146					
	TS-B	-29279.59	-29279.59	-29279.59 ± 0.00	13	13	13.00 ± 0.00	247	248	247.2 ± 0.2	-15.4133					
	HCST	-29727.21	-29727.21	-29727.21 ± 0.00	12	12	12.00 ± 0.00	13	17	13.2 ± 0.9	-15.6984					
15	ACO-B	-29605.35	-29620.11	-29606.64 ± 3.17	2	12	5.10 ± 2.12	610	1020	818.9 ± 120.2	-15.4162					
	ABC-B	-29605.35	-29605.88	-29605.42 ± 0.15	2	6	4.40 ± 1.01	254	347	294.5 ± 31.1	-15.4159					
	BFO-B	-29605.35	-29614.00	-29606.15 ± 1.92	2	12	4.95 ± 2.06	610	838	738.8 ± 64.2	-15.4163					
	TS-B	-29684.42	-29684.42	-29684.42 ± 0.00	11	11	11.00 ± 0.00	239	240	239.8 ± 0.1	-15.4221					
	HCST	-29750.85	-29750.85	-29750.85 ± 0.00	14	14	14.00 ± 0.00	13	13	13.0 ± 0.0	-15.4757					
20	ACO-B	-30056.08	-30073.30	-30057.13 ± 3.73	6	12	9.60 ± 1.20	736	1079	874.0 ± 106.1	-15.4126					
	ABC-B	-30056.08	-30075.94	-30057.14 ± 4.32	8	14	10.00 ± 1.10	249	368	306.2 ± 30.7	-15.4138					
	BFO-B	-30056.62	-30066.21	-30059.17 ± 2.94	6	14	9.45 ± 2.80	630	831	736.2 ± 48.8	-15.4116					
	TS-B	-30122.55	-30122.55	-30122.55 ± 0.00	14	14	14.00 ± 0.00	239	240	239.8 ± 0.1	-15.4194					
	HCST	-30187.77	-30187.77	-30187.77 ± 0.00	18	18	18.00 ± 0.00	13	13	13.0 ± 0.0	-15.4732					
25	ACO-B	-30524.00	-30527.46	-30524.42 ± 0.77	7	11	8.55 ± 1.28	534	1204	874.5 ± 158.5	-15.4098					
	ABC-B	-30524.00	-30524.48	-30524.12 ± 0.21	7	9	8.60 ± 0.80	239	334	298.3 ± 28.4	-15.4094					
	BFO-B	-30524.38	-30538.31	-30528.23 ± 4.09	7	14	9.80 ± 2.06	662	797	746.3 ± 38.7	-15.4109					
	TS-B	-30606.51	-30606.51	-30606.51 ± 0.00	20	20	20.00 ± 0.00	273	275	274.1 ± 0.5	-15.4159					
	HCST	-30824.60	-30824.60	-30824.60 ± 0.00	19	19	19.00 ± 0.00	13	14	13.4 ± 0.5	-15.5704					

of noise. Therefore, the structural differences sometimes fluctuate as the noise changes.

- (3) From the perspective of the execution time in the six tables, it is easy to see that the global search takes more time than the greedy local search. ACO-B and BFO-B take the longest run times, and ABC-B has the shortest time cost among the three SI algorithms that can compete with TS-B. HCST is the fastest and far superior to the other four algorithms. The reasons for these results are as follows. The SI algorithms are a kind of population-based optimization method in which each individual has to perform an independent optimization task and complete information exchange among the individuals at each iteration; therefore, the time cost is the highest.
- (4) From the perspective of the log-loss values in the six tables, it is difficult to determine which of the three

SI algorithms are good or bad in all cases. TS-B performs well when there is no noise, even better than the three SI algorithms in some cases. However, when noise is present, its log-loss value significantly decreases. Although HCST is not sensitive to noise, its performance is always the worst. These results show that the three SI methods have the strongest robustness, followed by TS-B, and HCST is the worst in the robustness.

To further exhibit the differences in robustness of the five algorithms on the same data, we take Alarm-4000 as an example to illustrate the performance from four aspects. As shown in Fig. 2a, ACO-B, ABC-B, and BFO-B consistently have the best performance in light of the K2 score in all cases, and there is little difference among the three algorithms. That is,

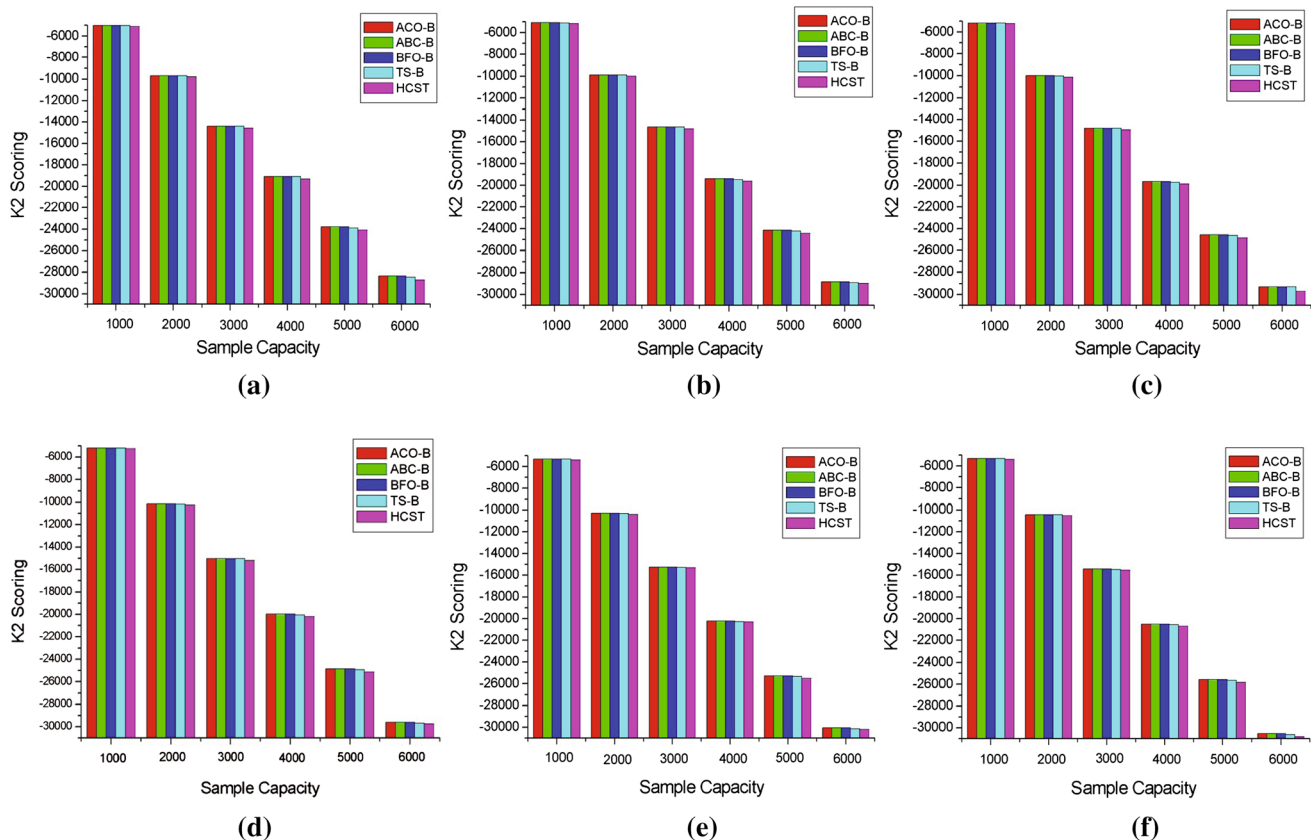


**Fig. 2** Different performance comparisons for five algorithms on Alarm-4000. **a** K2 scoring comparison. **b** Structural difference comparison. **c** Execution time comparison. **d** Log-loss comparison

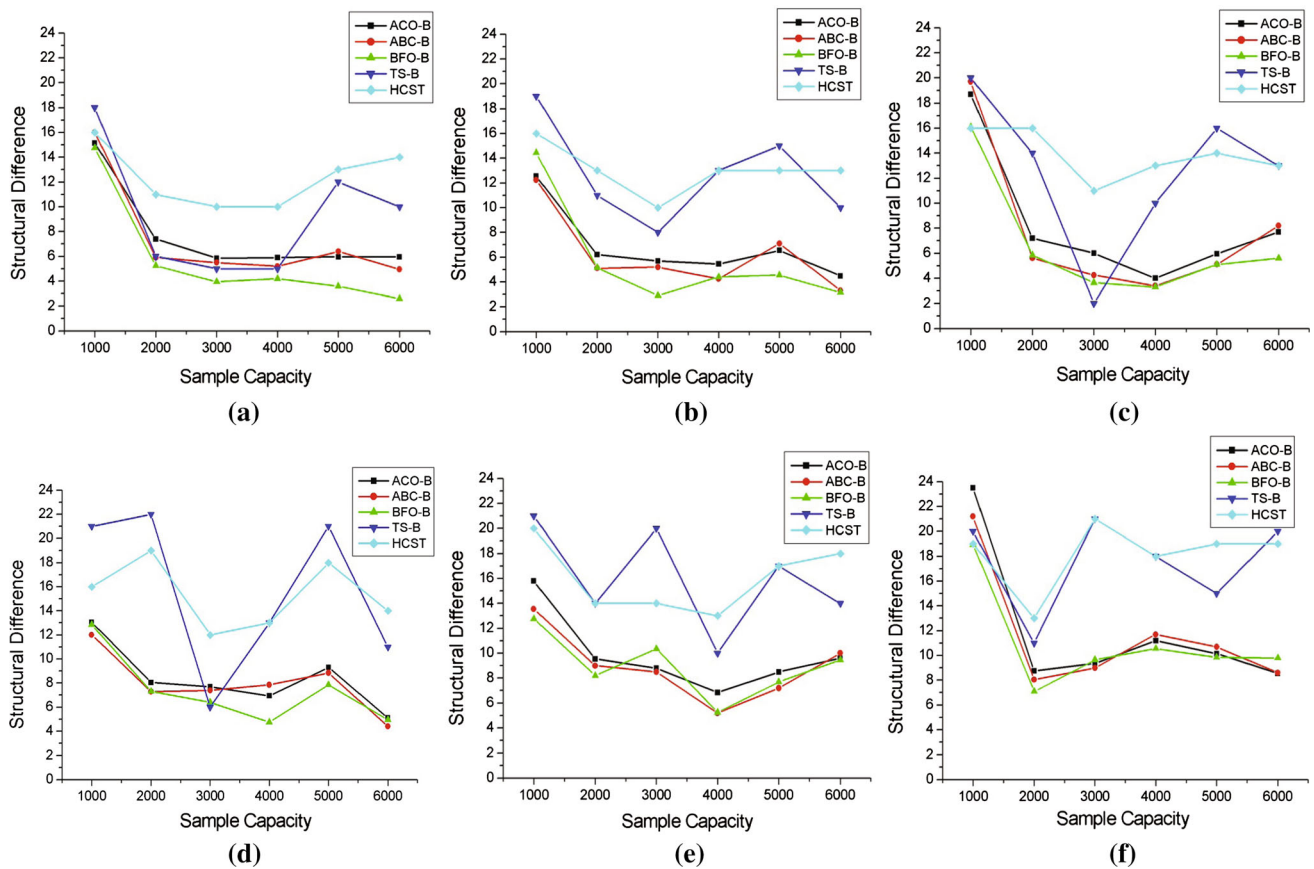
the three SI algorithms perform well both with and without noise. HCST plays the worst in all cases. When there is no noise, TS-B obtains slightly better K2 score than the three SI algorithms. However, when noise is present in data, TS-B is not as good as the three SI algorithms, even though it is much better than HCST. Figure 2b shows structure difference comparison of the five algorithms. BFO-B has the smallest structural difference in almost all cases; it has only about five error arcs even when the noise ratio reaches 20%. ACO-B and ABC-B have similar trends and variation ranges, although as a whole they are inferior to BFO-B. When there is no noise, TS-B obtains a relatively good result with five error arcs. However, once noise is introduced into the data set, there is a great gap between the standard structure and the structure learned by TS-B. Compared with the other four algorithms, HCST has the largest structural difference in all cases. These results further demonstrate the advantages of the SI-based algorithms. Figure 2c shows the curve of the execution time of the five algorithms, which demonstrates that the execution time of each algorithm has nothing to do with the noise and depends only on its time complexity. In all cases, HCST has the best time performance (less than 10s) and has an obvious advantage over the other four algorithms.

Although ABC-B and TS-B are inferior to HCST, they are two most competitive algorithms among the four stochastic global optimization methods. ACO-B has the worst time performance, and BFO-B is slightly better than ACO-B. It is obvious that ABC-B is the most simple and effective algorithm based on SI. Figure 2d shows the histogram of the average log-loss values of five algorithms. When there is no noise, the four stochastic global optimization algorithms have very similar log-loss values. When noise is introduced, the three SI-based algorithms maintain a higher level of log-loss even when the noise ratio reaches 25%. The log-loss value of TS-B gradually decreases as the noise ratio increases. The log-loss values of HCST are significantly lower than those of the other four algorithms in all cases, which suggests that the BN structure learned by HCST does not reflect the target data distribution well. Moreover, because HCST is a greedy search method and only finds local optimal solutions, the log-loss values obtained by HCST show some fluctuations in different noise ratios. In a word, global optimization algorithms have better solution performance and robustness than the local optimization algorithms.

Figures 3, 4, 5, and 6 show the evolution trend of different algorithm in terms of various evaluation metrics on



**Fig. 3** K2 scoring comparisons for five algorithms on Alarm network with different proportions of noisy data. **a** 0% of noisy data, **b** 5% of noisy data, **c** 10% of noisy data, **d** 15% of noisy data, **e** 20% of noisy data **f** 25% of noisy data

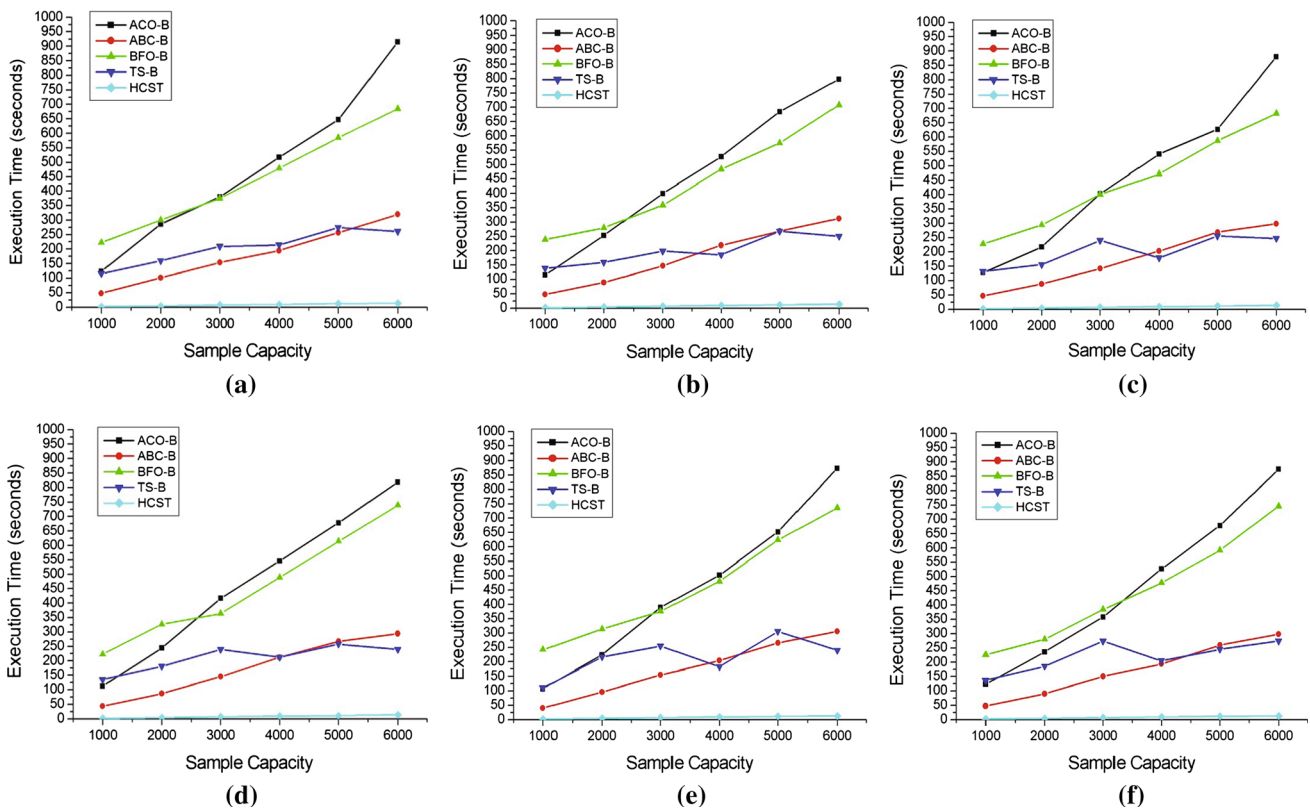


**Fig. 4** Structural difference comparisons for five algorithms on Alarm network with different proportions of noisy data. **a** 0 % of noisy data, **b** 5 % of noisy data, **c** 10 % of noisy data, **d** 15 % of noisy data, **e** 20 % of noisy data, **f** 25 % of noisy data

the Alarm network with different proportions of noisy data. From Fig. 3, we can conclude that the K2 score for each data set decreases as the noise increases and that the three SI-based algorithms nearly achieve the best K2 score values for each data set with or without noise. TS-B also obtains the best K2 score values on some small data sets without noise and slightly lower K2 score values on other data sets. HCST has the worst performance on the K2 score for all data sets. Although the K2 values are different for different data sets, the advantage and disadvantage comparisons of the five algorithms on the scoring performance are consistent. Figure 4 shows that the three SI-based algorithms achieve excellent performance on structural differences for all data sets, and the structural differences decrease as the sample capacity increases. Both TS-B and HCST also comply with the change trend from the Alarm-1000 to Alarm-4000 without noise. However, once noise is introduced into the data set, the change in the structural differences presents large fluctuation, which shows the instability of the two algorithms. As shown in Fig. 5, although there are some fluctuations for TS-B, there is a linear relationship between the execution time and the sample capacity for the five algorithms. The line for HCST has the smallest slope, and the line for ACO-B has the

largest slope. More importantly, the linear relationship is not influenced by noise. From Fig. 6, we can see that there is no great difference between the three SI algorithms according to the log-loss metric. Relatively strictly speaking, ABC-B performs the best, followed by ACO-B and then BFO-B. When there is no noise, TS-B performs very well and can match the SI algorithms on some data sets. However, it cannot match them when there is noise. Compared with the first four algorithms, HCST always has significant gaps on all data sets.

What are the reasons for good or bad performance in BNSL tasks? HCST starts with an empty graph without an arc and then iteratively uses greedy searching operators to construct an available solution. The biggest disadvantage is that it is easy to fall into a local optimum; thus, it acquires the worst results for the K2 score, structural difference, and log-loss with and without noise in the data set. However, it has the best time performance because it adopts a quick and simple searching way. Although TS-B also uses an individual to perform local searches, it combines the tabu strategy and aspiration criteria, so the search has a certain randomness and may escape from a local optimum. Thus it can acquire better K2 scores, structural differences, and log-loss than HCST and may even be comparable with the three SI algorithms



**Fig. 5** Execution time comparisons for five algorithms on Alarm network with different proportions of noisy data. **a** 0% of noisy data, **b** 5% of noisy data, **c** 10% of noisy data, **d** 15% of noisy data, **e** 20% of noisy data, **f** 25% of noisy data

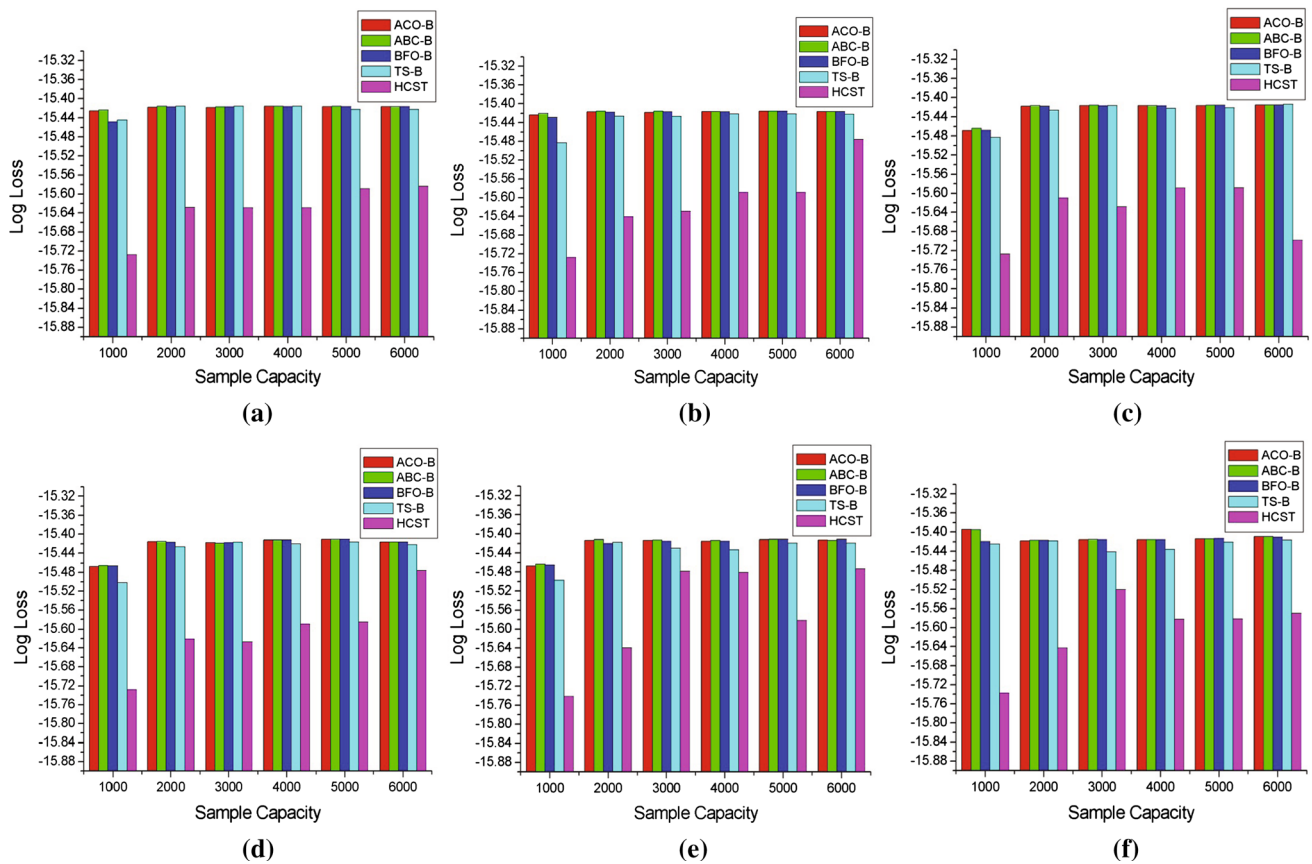
when there is no noise. But it costs more time than HCST due to that searching through a larger search space requires more iterations. To make matters worse, TS-B relies heavily on the parameter of tabu length, which is a fatal disadvantage and gives it poor control and adaptive abilities when there is noise. ACO-B, ABC-B, and BFO-B are three SI-based methods that use a population of individuals to carry out searches. This kind of algorithm can usually determine the global optimal solutions by collaborations and communications among individuals in the population. In particular, they have self-learning ability and can resist noise through useful information delivery and feedback, so they always have access to high-quality solutions regardless of whether there is noise. But they always take longer running time than the single individual-based search methods, since they must complete optimization of the population at each iteration.

### 5 Conclusions

As an important model in artificial intelligence, a BN can effectively represent uncertain knowledge in the real world. Along with the development of new technologies in machine learning and data mining, learning a BN structure from data has received considerable attention over the past decade.

Especially in recent years, SI-based algorithms have been successfully applied to learn BN structures and have shown promising results with good accuracy, robustness, and speed. In this paper, we classified the popular SI algorithms for BNSL into five paradigms and selected three representative algorithms, ACO-B, ABC-B, and BFO-B, to exhibit in detail the working principles of SI methods for BNSL. For the three algorithms, we discussed their common characteristics, compared their basic principles, and conducted experiments to validate the roles of important mechanisms using derived algorithms. To show better robustness of SI methods, we tested the three SI algorithms and two non-SI algorithms on data sets with noisy data. The experimental results demonstrated the following conclusions: (1) For ACO-B, the information transmission, i.e., pheromone transmission and updating, is the most essential mechanism. For ABC-B, the information transmission and the global search, i.e., inductive pheromone transmission and updating and the scout bees' ability of exploring the whole search space, are two key mechanisms. For BFO-B, the local optimization, i.e., chemotaxis toward better solutions by swimming and tumbling, is a determinant mechanism. These results show that different SI methods mainly rely on different specific mechanisms since the biological phenomena, respectively, simulated are different. (2) Among the three SI methods,





**Fig. 6** Log-loss comparisons for five algorithms on Alarm network with different proportions of noisy data. **a** 0 % of noisy data, **b** 5 % of noisy data, **c** 10 % of noisy data, **d** 15 % of noisy data, **e** 20 % of noisy data, **f** 25 % of noisy data

ABC-B usually achieves the highest K2 score and the least time, BFO-B gets the smallest structural differences on the whole, but it is difficult to tell which algorithm is better on the log-loss values. These results demonstrate that ABC-B is more appropriate for the situation that there are higher requirements for efficiency and the matching degree between a BN structure and a data set. BFO-B is better suited to the case that there is a higher demand for the structural difference between the learned network and the real one. Compared to ABC-B and BFO-B, ACO-B has no significant advantages. (3) Except the time performance, the three SI methods are superior to two non-SI methods on K2 scoring, structural difference, and log-loss values. These results suggest that SI methods have better robustness and global search ability and are more suitable for situations where the data sets contain noisy data and the search spaces are large.

In our future work, we will continue to study the SI methods for BNSL in the following aspects.

(1) *Robustness* This paper probed the learning performances of different SI algorithms with regard to the presence of noise in the data set and found that these algorithms have a certain degree of robustness. However,

many real-world situations are even more complicated. For example, there may be hidden variables in a complex system, and data sets are usually missing data. Moreover, some questions remain to be answered, such as whether learning performances are sensitive to the values of the parameters involved in the random search processes and the relationship between the performances and the parameters. Thus, a development that can effectively model the relationships among system variables and tolerate many uncertain factors to a great extent will be critical for the SI-based algorithms to learn a BN structure from data.

(2) *Distributed and parallel mechanisms* For a complex system with a large number of random variables, the superexponential search space is a major difficulty in BNSL. Although the learning algorithms based on SI do not perform brute-force searches, population-based searching is still time-consuming. To speed up the SI-based learning algorithms, it would be desirable to leverage their natural parallel processing mechanism.

(3) *Parameter modeling* At present, most of studies just like this paper determine the parameter values by hand tuning and set a fixed value for each parameter,

which may seriously affect the performance, because this method adopts the assumption that the parameters are independent, which may not hold in most cases. Even if the parameters in SI algorithms are irrelevant, the potential relationships among these parameters may be reflected by the evolution of the solutions. Thus, it would be necessary for us to further reveal the inherent relationships between different parameters and to realize the simultaneous optimization of parameters.

(4) *Researching on more SI methods* This paper only did in-depth study of three SI methods for BNSL. In fact, there are some other SI algorithms for BNSL, such as PSO, AIS. To fully show the superiorities of SI methods for BNSL, it is necessary to take a further research on more SI methods for BNSL in the future.

**Acknowledgements** We would like to thank the anonymous referees for their many valuable suggestions and comments. We thank the authors of the corresponding algorithms for sharing the binary executable systems. This work is partly supported by the NSFC Research Program (61375059, 61332016), the National “973” Key Basic Research Program of China (2014CB744601), the Specialized Research Fund for the Doctoral Program of Higher Education (20121103110031), and the Beijing Municipal Education Research Plan key Project (Beijing Municipal Fund Class B) (KZ201410005004).

#### Compliance with ethical standards

**Conflict of interest** The authors declare that they have no competing interests.

## References

- Alcobó JR (2004) Incremental hill-climbing search applied to Bayesian network structure learning. In: Proceedings of the 15th European conference on machine learning, Pisa, Italy
- Aouay S, Jamoussi S, Ayed YB (2013) Particle swarm optimization based method for Bayesian network structure learning. In: Proceedings of the 5th international conference on modeling, simulation and applied optimization (ICMSAO'13), pp 1–6
- Bang-Jensen J, Gutin GZ (2008) Digraphs: theory, algorithms and applications. Springer, Berlin
- Beinlich IA, Suermondt HJ, Chavez RM, Cooper GF (1989) The alarm monitoring system: a case study with two probabilistic inference techniques for belief networks. AIME, Lecture notes in medical informatics, vol 38, pp 247–256
- Bielza C, Larrañaga P (2014) Bayesian networks in neuroscience: a survey. *Front Comput Neurosci* 8:131
- Buntine W (1996) A guide to the literature on learning probabilistic networks from data. *IEEE Trans Knowl Data Eng* 8(2):195–210
- Castro PAD, Von Zuben FJ (2005) An immune-inspired approach to Bayesian networks. In: Proceedings of the 5th international conference on hybrid intelligent systems (HIS'05), IEEE, pp 6–9
- Cheng J, Bell DA, Liu W (1997) Learning belief networks from data: an information theory based approach. In: Proceedings of the 6th international conference on information and knowledge management (CIKM'97), pp 325–331
- Chickering DM, Geiger D, Heckerman D (1994) Learning Bayesian networks is NP-Hard. Technical Report MSR-TR-94-17, Microsoft Research
- Cooper GF, Herskovits E (1992) A Bayesian method for the introduction of probabilistic networks from data. *Mach Learn* 9(4):309–347
- Daly R, Shen Q, Aitken S (2011) Learning Bayesian networks: approaches and issues. *Knowl Eng Rev* 26(02):99–157
- Daly R, Shen Q (2009) Learning Bayesian network equivalence classes with ant colony optimization. *J Artif Intell Res* 35(1):391–447
- De Campos LM, Puerta JM (2001) Stochastic local and distributed search algorithms for learning belief networks. In: Proceedings of third international symposium on adaptive systems: evolutionary computation and probabilistic graphical model, pp 109–115
- De Campos LM, Fernández-Luna JM, Gámez JA, Puerta JM (2002) Ant colony optimization for learning Bayesian networks. *Int J Approx Reason* 31(3):291–311
- De Campos LM, Gámez JA, Puerta JM (2008) Learning Bayesian networks by ant colony optimisation: searching in two different spaces. *Mathw Soft Comput* 9(3):251–268
- De Campos LM, Huete JF (2000) A new approach for learning belief networks using independence criteria. *Int J Approx Reason* 24(1):11–37
- De CT Gomes L, De Sousa JS, Bezerra GB, De Castro LN, Von Zuben FJ (2003) Copt-aiNet and the gene ordering problem. In: Second Brazilian workshop on bioinformatics
- Dorigo M, Maniezzo VA, Colomi A (1996) The ant system: optimization by a colony of cooperating agents. *IEEE Trans Syst Man Cybern B* 26(1):29–41
- Eberhart RC, Shi Y, Kennedy J (2001) Swarm intelligence. Morgan Kaufmann, Los Altos
- Heckerman DE, Geiger D, Chickering DM (1995) Learning Bayesian networks: the combination of knowledge and statistical data. *Mach Learn* 20(3):197–243
- Heckerman D (1998) A tutorial on learning Bayesian networks: learning in graphical models. Kluwer, Dordrecht
- Heng XC, Qin Z, Tian L, Shao LP (2007) Learning Bayesian network structures with discrete particle swarm optimization algorithm. In: IEEE symposium on foundations of computational intelligence (FOCI'07). IEEE, pp 47–52
- Henrion M (1986) Propagating uncertainty by logic sampling in Bayes networks. In: Proceedings of the AAAI workshop on uncertainty in artificial intelligence, Philadelphia
- Ji JZ, Wei HK, Liu CN, Yin BC (2013) Artificial bee colony algorithm merged with pheromone communication mechanism for the 0–1 multidimensional knapsack problem. *Math Probl Eng* 2013, Article ID 676275
- Ji JZ, Zhang HX, Hu RB, Liu CN (2008) A tabu-search based Bayesian network structure learning algorithm. *Beijing Gongye Daxue Xuebao* 37(8):1274–1280
- Ji JZ, Zhang HX, Hu RB, Liu CN (2009) A Bayesian network learning algorithm based on independence test and ant colony optimization. *Acta Autom Sin* 35(3):281–288
- Ji JZ, Hu RB, Zhang HX, Liu CN (2011) A hybrid method for learning the Bayesian networks based on ant colony optimization. *Appl Soft Comput* 11(4):3373–3384
- Ji JZ, Wei HK, Liu CN (2013) An artificial bee colony algorithm for learning Bayesian networks. *Soft Comput* 6:983–994
- Karaboga D (2005) An idea based on honey bee swarm for numerical optimization. Technical Report TR06, Computer Engineering Department, Erciyes University, Turkey
- Kennedy J, Eberhart R (1995) Particle swarm optimization. In: Proceedings of the 1995 IEEE international conference on neural networks, pp 1942–1948
- Koller D, Friedman N (2009) Probabilistic graphical models: principles and techniques. MIT Press, Cambridge
- Larrañaga P, Karshenas H, Bielza C, Santana R (2013) A review on evolutionary algorithms in Bayesian network learning and inference tasks. *Inf Sci* 233(1):109–125

- Li XL (2010) A particle swarm optimization and immune theory-based algorithm for structure learning of Bayesian networks. *Int J Database Theory Appl* 3(2):61–69
- Martens D, Baesens B, Fawcett T (2011) Editorial survey: swarm intelligence for data mining. *Mach Learn* 82(1):1–42
- Mori M, Tsukiyama M, Fukuda T (1993) Immune algorithm with searching diversity and its application to resource allocation problem. *Trans Inst Electron Eng Jpn C* 113:872–878
- Mumford JA, Ramsey JD (2014) Bayesian networks for fMRI: a primer. *NeuroImage* 86:573–582
- Needham CJ, Bradford JR, Bulpitt AJ, Westhead DR (2007) A primer on learning in Bayesian networks for computational biology. *PLoS Comput Biol* 3(8):e129
- Passino KM (2002) Biomimicry of bacterial foraging for distributed optimization and control. *Control Syst* 22(3):52–67
- Pearl J (1988) Reasoning in intelligent systems: networks of plausible inference. Morgan Kaufmann, San Mateo
- Pinto PC, Nägele A, Dejori M, Runkler TA, Sousa JMC (2009) Using a local discovery ant algorithm for Bayesian network structure learning. *IEEE Trans Evol Comput* 13(4):767–779
- Rissanen J (1978) Modeling by shortest data description. *Automatica* 14(5):465–471
- Robinson RW (1977) Counting unlabeled acyclic digraphs. In: Little CHC (ed) *Combinatorial mathematics V*. Springer, Berlin
- Romero C, Ventura S (2010) Educational data mining: a review of the state of the art. *IEEE Trans Syst Man Cybern C Appl Rev* 40(6):601–618
- Rubio-Largo Á, Vega-Rodríguez MA, Gómez-Pulido JA, Sánchez-Pérez JM (2012) A comparative study on multiobjective swarm intelligence for the routing and wavelength assignment problem. *IEEE Trans Syst Man Cybern C Appl Rev* 42(6):1644–1655
- Schwarz G (1978) Estimating the dimension of a model. *Ann Stat* 6(2):461–464
- Shan C, Gong S, McOwan PW (2009) Facial expression recognition based on local binary patterns: a comprehensive study. *Image Visi Comput* 27(6):803–816
- Spiegelhalter DJ, Dawid AP, Lauritzen SL, Cowell RG (1993) Bayesian analysis in expert systems. *Stat Sci* 8(3):219–247
- Suzuki J (1999) Learning Bayesian belief networks based on the minimum description length principle: basic properties. *IEICE Trans Fundam Electron Commun Comput Sci* 82(10):2237–2245
- Tsamardinos I, Brown LE, Aliferis CF (2006) The max–min hill-climbing Bayesian network structure learning algorithm. *Mach Learn* 65(1):31–78
- Tsamardinos I, Aliferis CF, Statnikov A (2003) Scaling-up Bayesian network learning to thousands of variables using local learning technique. Technical Report DSL TR-03-02, Department of Biomedical Informatics, Vanderbilt University
- Wang T, Yang J (2010) A heuristic method for learning Bayesian networks using discrete particle swarm optimization. *Knowl Inf Syst* 24(2):269–281
- Weber P, Medina-Oliva G, Simon C, Iung B (2012) Overview on Bayesian networks applications for dependability, risk analysis and maintenance areas. *Eng Appl Artif Intell* 25(4):671–682
- Wong ML, Lam W, Leung KS (1999) Using evolutionary programming and minimum description length principle for data mining of Bayesian networks. *IEEE Trans Pattern Anal Mach Intell* 21(2):174–178
- Wu Y, McCall J, Corne D (2010) Two novel ant colony optimization approaches for Bayesian network structure learning. In: *Proceedings of the 2010 IEEE congress on evolutionary computation (CEC'10)*, pp 1–7
- Yang CC, Ji JZ, Liu JM, Liu JD, Yin BC (2016) Structural learning of Bayesian networks by bacterial foraging optimization algorithm. *Int J Approx Reason* 69:147–167
- Zhao J, Sun J, Xu W, Zhou D (2009) Structure learning of Bayesian networks based on discrete binary quantum-behaved particle swarm optimization algorithm. In: *Proceedings of the 5th international conference on natural computation (ICNC'09)*, 86–90