Structure Learning in Bayesian Networks Using Asexual Reproduction Optimization

Ali Reza Khanteymoori, Mohammad Bagher Menhaj, and Mohammad Mehdi Homayounpour

A new structure learning approach for Bayesian networks based on asexual reproduction optimization (ARO) is proposed in this paper. ARO can be considered an evolutionary-based algorithm that mathematically models the budding mechanism of asexual reproduction. In ARO, a parent produces a bud through a reproduction operator; thereafter, the parent and its bud compete to survive according to a performance index obtained from the underlying objective function of the optimization problem: This leads to the fitter individual. The convergence measure of ARO is analyzed. The proposed method is applied to real-world and benchmark applications, while its effectiveness is demonstrated through computer simulations. Results of simulations show that ARO outperforms genetic algorithm (GA) because ARO results in a good structure and fast convergence rate in comparison with GA.

Keywords: Bayesian networks, structure learning, evolutionary algorithms, genetic algorithms.

I. Introduction

In the last few years, Bayesian networks (BNs) have become a popular way of modeling probabilistic relationships among a set of variables for a given domain [1], [2]. The BN is a graphical model that denotes joint probabilistic distribution among variables of interest based on the variables' probabilistic relationships. The structure of the BN is represented as a directed acyclic graph (DAG). In a DAG, each node represents a variable that takes on a value over both continuous and discrete data sets of a domain and is connected with its parent's nodes. Each arc represents the conditional dependency between the two nodes so connected. With the development of large-scale database systems, the BN has become a popular knowledge representational scheme for probabilistic knowledge in data mining and knowledge discovery [1]-[4].

In building the BN, which represents the conditional dependencies in a database of cases, the problem of searching for the structure of the BN is both important and difficult. Although sometimes experts can create good Bayesian networks from their own experience, it can be a very hard task for large domains. Therefore, many methods have been investigated to automate the creation of Bayesian networks using cases collected from past experience [5]-[11]. The automated creation of a Bayesian network can be separated into two tasks: i) structure learning, which consists of creating the structure of the Bayesian network from the collected data, and ii) parameter learning, which consists of calculating the numerical parameters for a given structure.

This work will focus on the structure learning problem. Although other methods have been described for such learning, the method considered in this paper relies on searching the

Manuscript received Feb. 28, 2010; revised Aug. 9, 2010; accepted Aug. 23, 2010.

Ali Reza Khanteymoori (phone: +98 21 64542747, email: Khanteymoori@gmail.com) and Mohammad Mehdi Homayounpour (email: homayoun@aut.ac.ir) are with the Computer Engineering Department, Amirkabir University of Technology, Tehran, Iran.

Mohammad Bagher Menhaj (email: Menhaj@aut.ac.ir) is with the Department of Electrical Engineering, Amirkabir University of Technology, Tehran, Iran.

doi:10.4218/etrij.11.0110.0114

structure which best fits the collected data. This method can be seen as an optimization problem: having a formula quality measure which gives the fitness of the structures, the structure that maximizes that formula must be found using a search procedure.

The number of possible structures is huge and it has been proved that the search is NP-hard [12]. Therefore, heuristic search procedures have been tried.

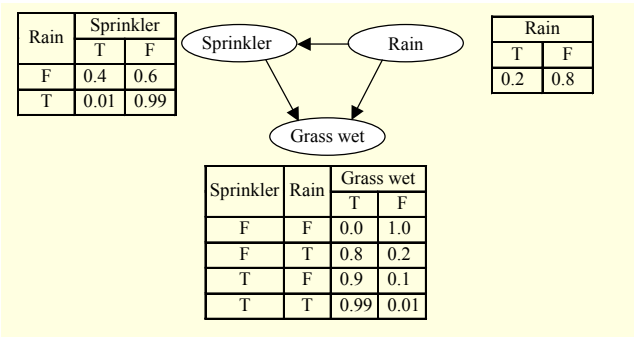
In this paper, a new structure learning method of the BNs based on an asexual-reproduction-optimization genetic algorithm (GA) is proposed. This algorithm is a structural learning method based on a free optimization algorithm inspired by one astounding biological phenomenon, asexual reproduction; hence, we entitle it asexual reproduction optimization (ARO). ARO is an individual-based algorithm which intelligently guides the search process and it can reach to the global optimum in an astonishing time, possessing advantages of both population-based and individual-based algorithms. Meanwhile, it escapes from local optima by adaptive exploration and exploitation as inspired by the biological model. In the proposed method, BN structure is represented as chromosomes, and it learns the topologies of the BN nodes.

This paper is organized as follows. In section II, a brief introduction to BNs is given. Section III introduces structure learning of the BN. ARO is completely described in section IV. In section V, our proposed method for BN structure learning is introduced. Section VI shows the convergence measures of the ARO algorithm. In section VII, the proposed method is applied to real-world and benchmark problems, and the results and analysis are shown. Finally, some conclusions are drawn in section VIII.

II. Bayesian Networks

During the last decade, BNs (and probabilistic graphical models in general) have become very popular in artificial intelligence [1], [2]. A BN provides a means of encoding the dependencies between a set of random variables, where the random variables and dependencies are represented as the nodes and edges of a directed acyclic graph. Missing edges (which imply conditional independence) are exploited in order to factor the joint distribution of all random variables into a set of simpler probability distributions.

A BN expresses a joint probability distribution over a set of random variables, and consists of a set of random variables $X_1,..., X_n$, a directed acyclic graph in which each variable appears once. The immediate predecessors of a variable X_i are referred to as its parents, with values $Parents(X_i)$. The joint probability distribution is factored as



Rain	Sprinkler	
	T	F
F	0.4	0.6
T	0.01	0.99

Rain	
T	F
0.2	0.8

Sprinkler	Rain	Grass wet	
		T	F
F	F	0.0	1.0
F	T	0.8	0.2
T	F	0.9	0.1
T	T	0.99	0.01

Fig. 1. Simple Bayesian network [13].

$$P(X_1 = x_1,..., X_n = x_n) = \prod_{i=1}^{n} p(X_i = x_i \mid Parents(x_i)). \quad (1)$$

Figure 1 shows a simple BN. When the variables are discrete, a tabular representation is used for conditional probabilities. For real-valued observations, Gaussian mixtures can be used. The topology of a BN gives direct information about the dependency relationships between the variables involved. In particular, it represents which variables are conditionally independent given another variable.

The process of building the BN can be separated into two tasks: structure learning and parameter learning. Structure learning is a search for an appropriate structure for the BN such that the BN accommodates the given set of samples. Parameter learning is computation of the conditional probabilities for the given BN structure such that the output of the BN approximates the distribution of the given set of samples. The most popular parameter learning method is the expectation maximization algorithm [6]. In this paper, the focus is structure learning of the BN and building an appropriate BN structure such that the BN structure accommodates the given set of samples.

III. Structural Learning

Consider the problem of analyzing the distribution over some set X of random variables $X_1,\ldots,X_n$, each of which takes values in some domain $Val(X_i)$. For simplicity, we focus on the case where the variables are discrete-valued; however, our approach extends easily to the continuous case. Our input is a fully observed data set $D = \{x[1],..., x[M]\}$, where each $x[m]$ is a complete assignment to the variables $X_1,\ldots,X_n$ in $Val(X_1,\ldots, X_n)$. Our goal is to find a network structure G that is a good predictor for the data. The most common approach to this task is to define it as an optimization problem. We define a scoring function $score(G:D)$, which evaluates different networks relative to the data D. We then need to solve the combinatorial

optimization problem of finding the network that achieves the highest score. For the remainder of this discussion, we take the training set D to be fixed.

Several scoring functions have been proposed. The most common scoring functions are the Bayesian information criteria (BIC), minimum description length (MDL) score, and the Bayesian Dirichlet (BDe) score [14]. The details of these scores are not relevant for our discussion. The most important property is that the scores are decomposable, that is, that each $score(G)$ is the sum of scores associated with individual families (where a family is a node and its parents):

$$score(G) = \sum_{i=1}^{n} score(X_i, Pa_G(X_i)). \quad (2)$$

Given a scoring function, our task is to find

$$\arg\max_G score(G). \quad (3)$$

This task is a hard combinatorial problem. Several of its specific instantiations have been shown to be NP-hard, even when the maximum number of parents per node is at most two [12]. The key intuition behind this result is that, due to the global acyclicity constraint, the choice of parent set for one node imposes constraints to the possible parent sets for the other nodes.

Robinson showed that $r(n)$, the number of possible structures for BN having n nodes, is given by the recurrence formula as follows [15]:

$$r(n) = \sum_{i=1}^{n} (-1)^{i+1} \binom{n}{i} 2^{i(n-i)} r(n-i) = n^{2^{O(n)}}. \quad (4)$$

Because of the super-exponential size of the search space, an exhaustive search for the best structure is impossible. Many heuristic methods have been proposed for BN structure determination. We are here more specifically interested in score-based methods, primarily greedy search (GS) [15] and minimum-weight spanning tree algorithms [16]. A GS is carried out in DAG spaces where the interest of each structure located near the current structure is assessed by means of a BIC/MDL type measurement or a Bayesian score like BDe [17].

In order to learn a BN from data through heuristic search, three elements must be defined: i) the search space, that is, what hypotheses will be considered during learning, ii) the search heuristic, that is, what criterion will be employed for evaluating different hypotheses, and iii) the search algorithm, that is, how we will search for the best hypothesis. In a structure learning task, the search space is constructed from the set of all DAGs containing (as nodes) the random variables at issue, by learning the conditional probability tables for each one of those DAGs.

IV. Asexual Reproduction Optimization

1. Biological Foundations of ARO

Asexual reproduction is a method of reproduction where a 1N (chromosome number) cell produces two to four cells with the same chromosome number. This can be done by binary fission of a motile stage or a non-motile stage. In many armoured dinokonts, the original cell divides along predetermined sutures and then each half produces a new half with new thin plates [18]. Asexual reproduction involves only one parent passing on the genetic information to their offspring. This sharing of genetic information makes the offspring identical to the parent [19].

Many species reproduce successfully in the absence of sex [20]. Asexual reproduction is common among organisms like bacteria, rapidly reproducing to generate large populations. In these large populations, mutation can provide considerable genetic variations, so sex may be less important in producing genetic diversity within the population [21].

There are different types of asexual reproduction, such as binary fission [22], asexual spore production [23], plants asexual reproduction [24], and budding [25]. In binary fission, only single-celled organisms reproduce. The cell duplicates its contents, including its nucleus and other organelles, and then splits into two cells with each one being identical, for example, bacteria, amoeba, and algae [26]. In asexual spore production, spores are similar to seeds, but are produced by the division of cells of the parent, not by the union of two cells. One parent may produce many spores. Each of these, for example, fungi, green algae, moulds, and ferns, will grow into a new individual, identical to its parent. Many spores are produced to ensure that at least some of the individual organisms will survive. Zoospores can also be produced by some fungi and green algae. They move using tail-like flagella [23].

A plant continues to grow throughout its life. The rapidly growing tips of roots and stems contain specialized reproductive cells called meristem. At a certain time, these cells will be diverted into cells that make up roots, stems, and leaves. If parts of the plant are damaged, the meristem cells make repairs. Clones can be made from cuttings of a plant because the meristem cells can specialize to reproduce the different parts needed to make a new plant. Asexual reproduction can produce many plants very quickly. This is an advantage in places where the environment does not change very much. A good example of this would be bacteria. By building a large population of organisms very quickly, the species is able to thrive. The great disadvantage is that when the environment changes, all of the organisms will die if they do not have the ability to adapt to the change [24]. Eventually in the budding

Table 1. Symbols and their meanings.

Symbol	Description	Symbol	Description
S	Search space	K	Maximum iterate number
L	Length of chromosome	Ω	State space of X
n	Number of variables	Φ	State space of Y
g	Number of selected bits	X^*	Optimal solution
X_{k_t}	t-th bud produced by k-th parent	Y_k	k-th parent
$P_{Y_k X_{k_m}}$	Transforming probability	$Pa_{X_{k-1_m} Y_k}$	Acceptance probability

mathematically modeled by ARO, the parent organism produces a bud (a smaller version of itself), which eventually detaches itself from the parent and becomes a self-sufficient individual while remaining identical to the parent. Coral also reproduce in this way but do not detach themselves. Other examples would be hydra, yeast, and sea sponges [27].

Asexual reproduction is a significant adaptation to specific environments and biological conditions where the cost of sexual reproduction to a species is considerable [28]. Asexuality can be seen as a complicating factor of the question, "Why sex?" which has been asked by scientists throughout the history of biology [29]. According to [29], the question should be directed primarily to the genetic consequences of sex. Awareness of the genetic consequences of reproductive modes forms the basis of the background to the evolutionary approach.

Sexual reproduction usually involves two individual organisms. The offspring that are produced from this union have a mix of characteristics, half from one parent and the other half from the other parent. It should be considered that most of evolutionary algorithms model sexual reproduction. However, sexual reproduction does not always involve male and female parents. They can have specialized gametes (reproductive cells whose one role is to join with another gamete during reproduction). Many organisms are capable of both sexual and asexual reproduction, like some moulds, such as Rhizopus, which produce spores. They can also produce zygospores, enabling them to reproduce sexually as well.

2. Main Framework of ARO

Before a description of the ARO algorithm, we define all the meanings of the necessary symbols in Table 1.

As discussed in the previous section, there are various kinds of asexual reproduction. The proposed algorithm is inspired by

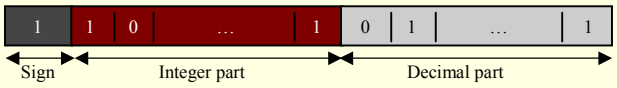


Fig. 2. Chromosome with 3 parts: sign, integer, and decimal.

```
Begin
    t=1;
    % Parent Initialization between lower (L) and upper bound (U)
    P=Initialize(L, U);
    % Fitness of P is calculated
    Fitness_P=fit(P);
    %Stopping Criteria
    While stopping conditions are not met
        % P reproduces a Bud
        Bud(t)=Reproduce(P);
        % Fitness of Bud(t) is calculated
        Fitness_Bud(t)=fit(Bud(t));
        If Fitness_Bud(t) is better than Fitness_P
            % Bud(t) is replaced with P
            P=Bud(t);
        Else
            % Bud(t) is discarded
            clear Bud(t);
        end
        t=t+1;
    End
end
```

Fig. 3. Pseudo code for ARO.

the budding method of asexual reproduction. Each individual is illustrated by a binary string like the binary representation in evolutionary algorithms. A decision variable vector $X = (x_1, x_2, ..., x_n); X \in \Re^n$ is called an individual in ARO, and each variable is considered the chromosome made by a number of bits called genes. Therefore, a chromosome with length L is considered such that the first bit represents the sign of the individual. The next l_1 bits show the integer part while the last l_2 bits present the decimal part of the chromosome. As a result, $L = l_1 + l_2 + 1$, and the length of an individual becomes $n \times L$. Figure 2 illustrates an ARO chromosome.

We assume that each solution in the search space S is an organism in its environment. In addition, it is supposed that there are limited resources in the environment such that only the most deserving individual can survive. To start the algorithm, an individual is randomly initiated in the distinctive domain of S. Thereafter, the individual reproduces an offspring, a bud, by a particular operator called a reproduction mechanism. The parent and its offspring compete to survive according to a performance index or a fitness function. If the bud wins the competition, its parent will be discarded. Therefore, the bud replaces its parent and becomes the new parent. If the parent triumphs, the bud will be thrown away.

The algorithm repeats steps illustrated in Fig. 3 until the stopping criteria are satisfied.

It is obvious that the choice of an appropriate reproduction operator is very crucial. While ARO only applies one operator, most evolutionary algorithms use the number of operators to explore the search space and to exploit available information according to the traditional control theory. In order to reproduce, a substring which has g bits, g~*Uniform*[1, L] in each chromosome is randomly chosen. Afterward bits of the substring mutate such that in any selected gene, 1 is replaced by 0 and vice versa.

In fact, this substring named *larva* is a mutated form of its parent. According to optimization theory, even though both exploration and exploitation mechanisms are an indispensable, mutated form of its parent, only larva explores the search space. In order to enhance the algorithm optimization ability, an exploitation mechanism is appended so the larva and its parent probably share their information by *merging*; consequently, a bud is generated similar to its biological model. On the other hand, during mutation, crossover implicitly occurred. Figure 4 shows the reproduction mechanism.

After it is produced, the bud's fitness is evaluated according to the performance index. As illustrated in Fig. 3, bud fitness is compared with its parent's fitness. Finally, only the one with the most merit is capable of subsisting to reproduce.

3. ARO Adaptive Search Ability

In the first ARO simulation, the reproduction operator was implemented as follows. After the larva was produced by its parent, a random number uniformly distributed in [0, 1] is generated. If this number is less than 0.5, the bit will be selected from the parent. Otherwise it will be chosen from the larva till bud is completed. It means that merging is definitely performed. The number of bits going to be altered, that is, g, is a random number. When g is large, more exploration is expected and vice versa, while the exploitation applied is done based on the aforementioned procedure. This means that the amount of exploration is merely controlled by g. Similarly, the exploitation rate is handled by the probability of merging the larva with its parent. Consequently, we may employ a procedure to adaptively set the rate of exploration and exploitation. As a consequence, when the rate of exploration g is large, the exploitation rate decreases and vice versa. To achieve this goal, a function p calculating the probability of merging is defined. Therefore, ARO can adaptively shrink or widen the search space. Note that

$$p = \frac{1}{1 + Ln(g)}, \tag{5}$$

where Ln is the natural logarithm function, and it is obvious that when g increases, p decreases and vice versa.

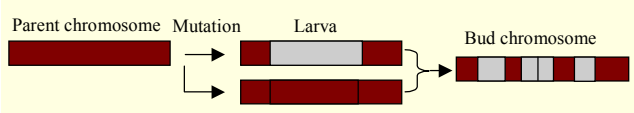


Fig. 4. Reproduction mechanism generating bud chromosome.

4. ARO Strength and Weakness Points

ARO has several advantages distinguishing it from other algorithms. First of all, as we discussed, ARO is an individual-based algorithm; hence, despite population-based algorithms taking a lot of time to evolve, ARO consumes a little energy resulting in a remarkable fast convergence time. This property of ARO makes it very appropriate for real time applications, especially for real time control and filtering, signal processing, and data mining.

Secondly, due to the lack of an optimum parameters setting, the existing algorithms, such as GAs, particle swarm optimization (PSO), classic genetic algorithm, tabu search, and simulated annealing might get in trouble. For example, consider the PSO algorithm. If essential parameters of PSO are not well set, it cannot work properly [30]. In contrast, ARO does not require any parameter tuning.

Furthermore, a selection mechanism is not necessary in ARO. Selection mechanisms have been defended for more than two decades [31]. However, choosing a poor selection mechanism, such as Roulette-Wheel, causes problems like premature convergence and selective pressure.

Finally, in contrast with a number of bio-inspired algorithms applicable for some particular optimization problems like ACO but limited to discrete optimization problems [32], ARO is a model free algorithm which can be applied to a variety of optimization problems. Therefore, it can be exploited for graph optimization, combinatorial optimization, constrained optimization, integer optimization, and much more.

In contrast, a predictable weakness of ARO is perhaps its adaptation with the turbulent environment and genetic diversity in its corresponding biological model. The aforementioned problems can be resolved by implementing particular reproduction operators as we did to explore throughout the search space in this paper.

V. ARO Algorithm for BN Structure Learning

This section describes how the ARO method can be applied to the structure learning of the BNs. In the ARO, the connectivity matrix is assumed to be upper triangular, and it is

the same as that of the GA method. Thus, the entire solution space is searched for the fittest structure of the BN.

In order to encode all of the information related to the structure learning of the BN into a chromosome and search the entire solution space for the fittest structure, an ARO representation is proposed.

The BN structure with n variables is represented by an $n \times n$ connectivity matrix $C = (c_{i,j})$, where

$$C(i,j) = \begin{cases} 1, \\ 0, \end{cases} \tag{6}$$

and each individual of the population is encoded as a chromosome:

$$c_{1,1}c_{1,2}\ldots c_{1,n}c_{2,1}c_{2,2}\ldots c_{2,n}\ldots c_{n,1}c_{n,2}\ldots c_{n,n}. \tag{7}$$

With this representation, the plain mutation operator and reproduction system would produce illegal BN structures. In the previous method, to overcome this problem, the connectivity matrix was assumed to be upper triangular and the connectivity matrix,

$$\begin{pmatrix} 0 & c_{1,2} & c_{1,3} & \ldots & c_{1,n-1} & c_{1,n} \\ 0 & 0 & c_{2,3} & \ldots & c_{2,n-1} & c_{2,n} \\ & & & \vdots & & \\ 0 & 0 & 0 & \ldots & 0 & c_{n-1,n} \\ 0 & 0 & 0 & \ldots & 0 & 0 \end{pmatrix} \tag{8}$$

was encoded as a chromosome:

$$X = c_{1,2}\ldots c_{1,n}c_{2,3}\ldots c_{2,n}\ldots c_{n-2,n-1}c_{n-2,n}c_{n-1,n}. \tag{9}$$

In other words, the ordering among the variables of the BN was fixed and the node A_i was allowed to have another node A_j as a parent node only if the node A_j came before the node A_i in the ordering. This scheme restricts the values of $c_{i,j}$ $(i \geq j)$ to 0 and narrows the search space.

VI. ARO Convergence Analysis

In this section, the ARO convergence is analyzed. Firstly, *state* and *state space* are defined:

State. A state is said to exist when the algorithm inaugurates the search from an initial parent denoted by X_0, consequently two sequences $\{X_0, X_1, X_2, \ldots\}$ and $\{Y_0, Y_1, Y_2, \ldots\}$ are created. As stated before, ARO starts with an initial parent and frequently generates new buds using the reproduction operator till one of these buds excels the parent. In other words, acquiring a better objective function value, the bud replaces the parent and in so doing, becomes a new parent. In the search process, a finite sequence $\{X_{k_1}, X_{k_2}, \ldots, X_{k_m}\}$ is used for all buds generated by exploiting the k-th parent. It means that X_{k_t} is the t-th bud produced by using the k-th parent. Similarly, in the finite sequence $\{Y_0, Y_1, \ldots, Y_K\}$, Y_k is the k-th parent generated by the algorithm.

State Space. A state space of random variables $\{X_{k_t}, k_t \in K_T\}$ includes all possible values for X_{k_t} denoted by Ω. Also, a state space of random variables $\{Y_k, k \in K\}$ indicated by Φ is all the possible values of Y_k.

We state three conditions that will be shown to be sufficient to guarantee the ARO convergence.

C1. We assume that the state space S is a bounded and closed subset of $\Re^n$, and there exists an X^* such that $\forall X \in S \Rightarrow f(X^*) \leq f(X)$.

C2. For every ε, the set $S_\varepsilon = \{X \in S : |f^* - f(X)| < \varepsilon\}$ is defined as the optimum set, and members of this set are symbolized by X_ε, which is an optimal set with ε value of error. When finding an $X \in X_\varepsilon$, the algorithm stops.

C3. In the proposed algorithm, each offspring is reproduced by using a function H entitled reproduction function, that is, $X_{k_{t+1}} = H(Y_k)$. Reproduction function is delineated by mutation and merging. This function is continuous, and its image is indeed the whole state space. H is able to produce all points of S and adaptively explore and exploit. If the algorithm is stuck in a local optimum, H increases exploration to expand the search space and narrows it by exploitation when needed.

To analyze the ARO convergence, some lemmas are required, which are described as follows. Proofs of these lemmas are shown in [18].

Lemma 1. The stochastic sequence $\{f(Y_k), k \in K\}$ is a non-increasing sequence.

It is required to show whether the sequence $\{X_{k_t}, k_t \in K_T\}$ is finite or not. The following lemma deals with this question and states that the number of required iterations to achieve a local optimum is limited.

Lemma 2. For each $Y_k \notin S_\varepsilon$, the sequence $\{X_{k_t}, k_t \in K_T\}$ will be a finite stochastic sequence.

Lemma 3. The sequence of stochastic variables $\{Y_k, k \in K\}$ is a stochastic sequence modeled by a Markov chain.

Lemma 4. For every $X_0 \in S$ and $\varepsilon > 0$, there exists an integer N such that for any $\delta > 0$,

$$P\{Y_n \notin S_\varepsilon \mid X_0 = \omega_0\} < \delta, \qquad \forall n > N. \tag{10}$$

To ascertain lemma 4, it is adequate to demonstrate that N is a finite integer number. For that reason, a Markov chain is used as illustrated in Fig. 5.

In Fig. 5, the nonintegrated Markov chain illustrates each state, which is defined above. There are two classes of transmission probability, stated as $P_{Y_k X_{k_m}}$, which is the probability of transferring from the current state Y_k, a perturbed

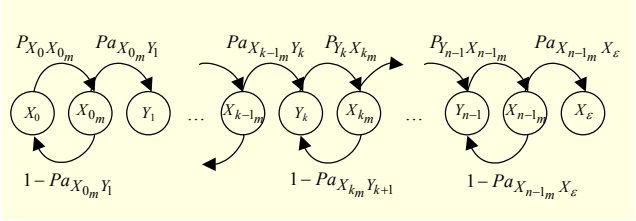


Fig. 5. Markov chain of states generated by ARO.

state X_{k_m} (probability of variation), and $Pa_{X_{k-1_m}Y_k}$, which is the acceptance probability of X_{k-1_m} as a new parent Y_k. Note that this probability is nonzero when $f(X_{k-1_m}) < f(Y_{k-1})$ (probability of acceptance).

Lemma 5. In the aforementioned Markov chain, all Y_k and X_{k_m} represent transient states.

Theorem 1. State i is recurrent if we have $\eta_i = 1$, and it becomes transient if $\eta_i < 1$ [18].

Now, we proceed as follows:

For any state Y_k, we can easily have

$$\begin{aligned}
&\eta_{Y_k}^{(1)} = 0,\\
&\eta_{Y_k}^{(2)} = P_{Y_kX_{k_m}}\left(1-Pa_{X_{k_m}Y_{k+1}}\right),\\
&\eta_{Y_k}^{(3)} = 0,\\
&\vdots\\
&\eta_{Y_k}^{(n)} = \begin{cases}\left[P_{Y_kX_{k_m}}\left(1-Pa_{X_{k_m}Y_{k+1}}\right)\right]^n, & n\in E,\\ 0, & n\in O.\end{cases}
\end{aligned} \tag{11}$$

We finally obtain

$$\begin{aligned}
\eta_{Y_k} = \sum_{n=1}^{\infty}\eta_{Y_k}^{(n)} &= P_{Y_kX_{k_m}}\left(1-Pa_{X_{k_m}Y_{k+1}}\right)\\
&+\left[P_{Y_kX_{k_m}}\left(1-Pa_{X_{k_m}Y_{k+1}}\right)\right]^2\\
&+\ldots+\left[P_{Y_kX_{k_m}}\left(1-Pa_{X_{k_m}Y_{k+1}}\right)\right]^n+\ldots,
\end{aligned} \tag{12}$$

where η_{Y_k} indicates a geometric series with $P_{Y_kX_{k_m}}\left(1-Pa_{X_{k_m}Y_{k+1}}\right)<1$. As a consequence, the sum of this series is equal to

$$\eta_{Y_k} = \sum_{n=1}^{\infty}\eta_{Y_k}^{(n)} = \frac{P_{Y_kX_{k_m}}\left(1-Pa_{X_{k_m}Y_{k+1}}\right)}{1-P_{Y_kX_{k_m}}\left(1-Pa_{X_{k_m}Y_{k+1}}\right)}. \tag{13}$$

According to lemma 4, $\lim_{t\to M} Pa_{X_{k_m}Y_{k+1}} = 1$. Hence, $\eta_{Y_k} = 0$, implying that Y_k is a transient` state. For X_{k_m}, if $f(X_{k_m}) < f(Y_k)$, then it will be accepted as a new parent or be discarded. Hence, the return probability to this state in the long term equals zero, and the proof is completed.

Lemma 6. In the above mentioned Markov chain, X_ε is a recurrent state.

To conclude this section, the following theorem that theoretically demonstrates the ARO convergence to a global optimum in a finite time based on aforementioned lemmas is stated.

Theorem 2. Let f be a function defined on a subset $S \subset \Re^n$, and let $\{Y_1, Y_2, Y_3, \ldots\}$ be the sequence of states generated by ARO. Assume that conditions C1 to C3 hold. Then, for every initial state X_0, the sequence of function values $\{f(Y_k), k \in K\}$ for finite values of K converges to f^*.

Proof. As stated by the above lemmas, if the algorithm begins from any random solution $X_0 \in S$, after a finite number of iterations as indicated by lemma 2 and lemma 4, it will get to X_ε and will stay there forever due to lemma 5 and lemma 6, which state that all states are transient, excluding X_ε, which is an absorbing state. In other words, after a finite number of iterations, the algorithm converges to X_ε. The proof of theorem 2 is consequently accomplished. □

VII. Simulations

In this section, the proposed method is applied to two real world problems, and the validity of this method is demonstrated through computer simulation.

1. Car Diagnosis Problems

The car diagnosis problem introduced by Norsys is a simple example of a belief network. The reason why a car does not move is presumed, based on spark plugs, headlights, main fuse, and so on [33]. Eighteen nodes are used in this BN, and all nodes of the network take discrete values. Some of them can take on three discrete states, and the others can take on two states. A database of two thousand cases is utilized to train the BN. The database was generated from Netica tool [33]. Figure 6 shows the structure of the car diagnosis problem depicted by Netica from which the sample cases were collected. Note that, Fig. 6 depicts the desired BN structure, and the goal of the structure learning for BN is to obtain the structures which are close to this desired one.

2. ALARM Network

A logical alarm reduction mechanism (ALARM) is a medical diagnostic system for patient monitoring. It is a complex belief network with eight diagnoses, sixteen findings, and thirteen intermediate variables [34]. A database of two thousand cases is utilized to train the BN. As in the previous example, the database is generated from Netica tool [34].

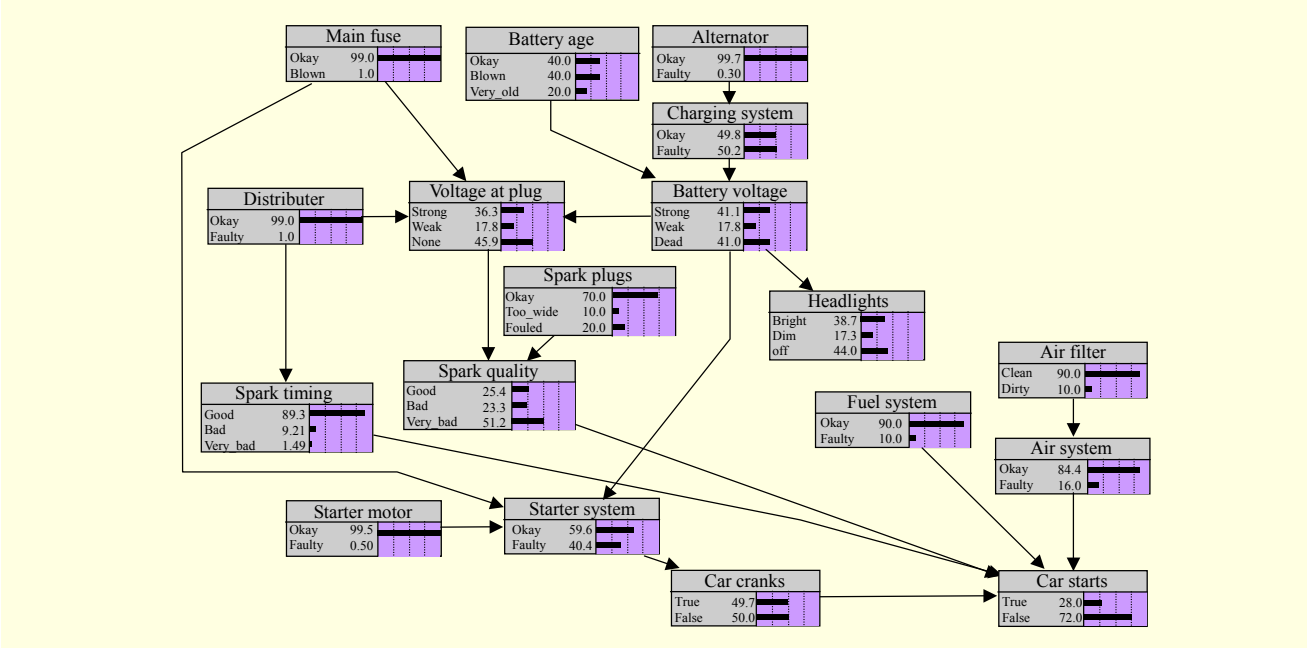


Fig. 6. Structure of car diagnosis problem network.

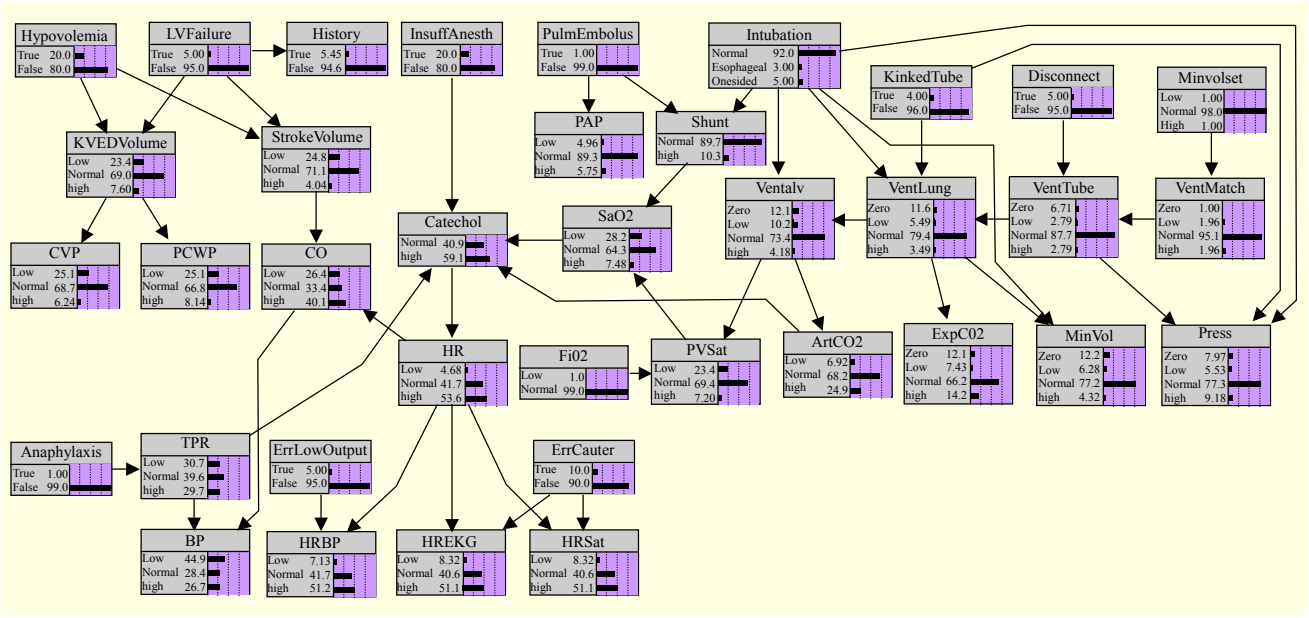


Fig. 7. Structure of ALARM network.

Figure 7 shows the structure of the ALARM network depicted by Netica from which the sample cases are collected. Note that Fig. 7 depicts the desired BN structure, and the goal of the structure learning for BN is to obtain the structures which are close to this desired one.

3. Simulation Results

Using these databases, the BNs were built to represent the probabilistic relationships between the variables. The formula in [10] was utilized to apply the proposed algorithms and evaluate the validity of the structure of the given BN.

The simulation was run ten times for each problem, and the parameters of the simulations were as follows. The population size of ARO was 50, and the mutation rate was 0.05. The formula in theorem 3 was used as an objective function to be maximized. The algorithms were stopped when 5,000 structures had been evaluated, which means that 100

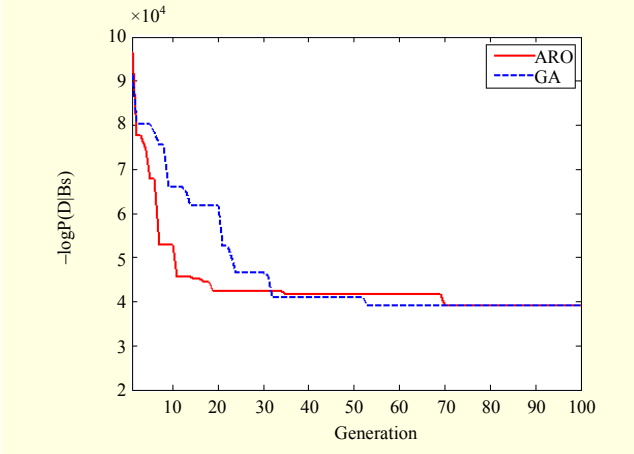


Fig. 8. Performance of ARO and GA for one run (car diagnosis problem).

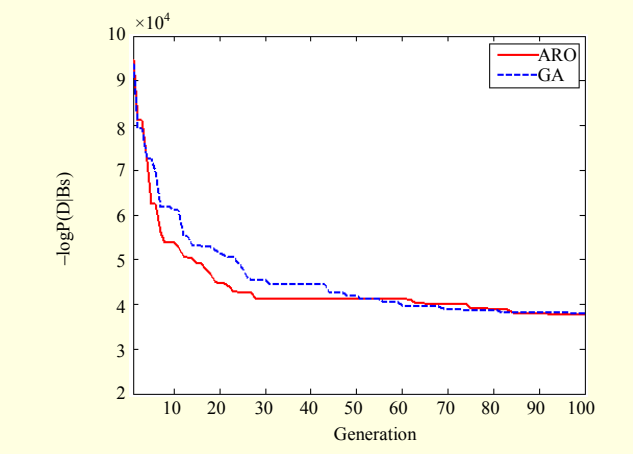


Fig. 10. Mean performance of ARO and GA for 5 runs (Car diagnosis problem).

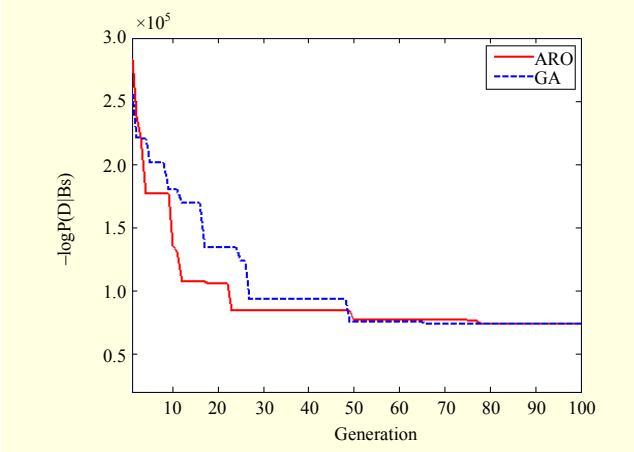


Fig. 9. Performance of ARO and GA for one run (ALARM network).

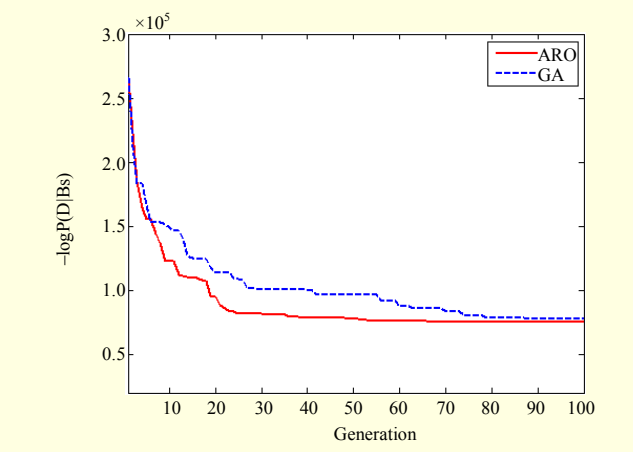


Fig. 11. Mean performance of ARO and GA for 5 runs (ALARM network).

generations have been evaluated.

In this simulation, we assume that there is no prior knowledge about the node ordering or the casual relationship between the nodes. Therefore, we randomly select the order of the nodes in the conventional method. Figures 8 and 9 show the learning result of the first run for the car diagnosis and ALARM problems, respectively. Figures 10 and 11 show the learning result of the five run for them. In these figures, the dashed lines represent the result of the conventional method, and the solid lines represent the result of the proposed method.

As BN attempts to approximate the entire database, the vertical axis approaches zero since the probability of one corresponds to zero on the negative logarithm axis. In the car diagnostic systems, it can be seen that the GA initially outperforms the ARO, but after some iterations, ARO outperforms the GA and continues to outperform the conventional method for fifty generations, as shown in Fig. 10. In this case, ARO shows good performance for less than fifty generations, and it outperforms the GA method.

In the ALARM network, as shown in Fig. 11, ARO initially outperforms the GA and continues to outperform the conventional method for one hundred generations.

Statistical results are summarized in Table 2. In this table, the learned structures are compared with the target structure from which the sample data are taken.

Table 2. Statistical results for car diagnosis problem and ALARM network.

Application	Algorithm	Best	Mean	Variance	Target
Car diagnosis problem	GA	38,101	38,489	161,136	37,821
	ARO	38,005	38,129	16,810	
ALARM network	GA	74,703	77,496	1,290,145	73,401
	ARO	74,624	75,327	499,296	

Table 3. Comparisons with target structure.

Application	Trial time	Missed arcs	Wrongly added arcs
Car diagnosis problem	1	1	1
	2	1	0
	3	0	1
	4	2	0
	5	1	0
ALARM network	1	2	1
	2	3	0
	3	2	1
	4	2	0
	5	1	1

For further evaluation, we compare the structure results from ARO with the target network in the terms of number of arcs correctly added between the same nodes as those in the target network, the number of missed arcs of the target network, or the number of extra arcs added wrongly. The results are depicted in Table 3. As a conclusion, we can say that the resulting network is close to the target network, and it is acceptable because of very few missed and wrongly added arcs.

VIII. Conclusion

In this paper, ARO, a new structure learning approach for BNs, has been proposed. ARO, a new individual-based optimization algorithm, is inspired by asexual reproduction. Fast convergence time to the best solution, no parameter setting, being model free, evolutionary foundations, and biological bases are ARO's advantages making it very appropriate for realtime applications. The ARO convergence was comprehensively analyzed.

The proposed method was applied to two real-world and benchmark problems. Simulations revealed improved performance over the GA method for these case studies, according to the rate of convergence. Indeed the ARO algorithm is faster than GA because ARO has simpler operators.

References

[1] J. Pearl, *Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference*, San Mateo, CA: Morgan Kaufmann, 1988.

[2] R.E. Neapolitan, *Learning Bayesian Networks*, Upper Saddle River, NJ: Prentice Hall, 2004.

[3] S. Russell and P. Norvig, *Artificial Intelligence: A Modern Approach,* 2nd ed., Upper Saddle River, NJ: Prentice Hall , 2003.

[4] F.V. Jensen, "Introduction to Bayesian Networks," Dept. of Mathematics and Computer Science, Univ. of Aalborg, Denmark, Technical Report IR 93–2003, 1993.

[5] M.L. Wong and K.S. Leung, "An Efficient Data Mining Method for Learning Bayesian Networks Using an Evolutionary Algorithm-Based Hybrid Approach," *IEEE Trans. Evolutionary Computing*, vol. 8, no. 4, Aug. 2004, pp. 378-404.

[6] D.M. Chickering, D. Geiger, and D. Heckerman, "Learning Bayesian Networks: Search Methods and Experimental Results," *5th Int. Workshop Artificial Intell. Statistics*, 1995, pp. 112-128.

[7] H. Wang, D. Dash, and M.J. Druzdzel, "A Method for Evaluating Elicitation Schemes for Probabilistic Models," *IEEE Trans. Syst. Man Cybern. B*, vol. 32, no. 1, Feb. 2002, pp. 38-43.

[8] G.M. Provan, "Model Selection for Diagnosis and Treatment Using Temporal Influence Diagrams," *Proc. 4th Int. Workshop Artificial Intell. Statistics*, 1995, pp. 469-480.

[9] P. Larranaga et al., "Structure Learning of Bayesian Network by Genetic Algorithms: A Performance Analysis of Control Parameters," *IEEE Trans. Pattern Anal. Machine Intell.*, vol. 18, no. 9, Sept. 1996, pp. 912-926.

[10] G.F. Cooper and E.A. Herskovits, "A Bayesian Method for the Induction of Probabilistic Networks from Data," *Mach. Learning*, vol. 9, no. 4, 1992, pp. 309-347.

[11] X. Li, X. He, and S. Yuan, "Learning Bayesian Networks Structures from Incomplete Data Based on Extending Evolutionary Programming," *Proc. Int. Conf. Mach. Learning Cybern.*, vol. 4, Aug. 2005, pp. 2039-2043.

[12] D.M. Chickering, "Learning Bayesian Networks is NP-Complete," *Learning from Data, Artificial Intelligence and Statistics*, Springer Verlag, 1996.

[13] http://en.wikipedia.org/wiki/Bayesian_network

[14] D. Heckerman, D. Geiger, and D.M. Chickering, "Learning Bayesian Networks: The Combination of Knowledge and Statistical Data," *Mach. Learning*, vol. 20, 1995, pp. 197-243.

[15] R.W. Robinson, "Counting Unlabeled Acyclic Digraphs," *Combinatorial Mathematics V: Proc. 5th Australian Conf.*, Aug. 24-26, 1976.

[16] D.M. Chickering, "Optimal Structure Identification with Greedy Search," *J. Machine Learning Research*, vol. 3, 2002, pp. 507-554.

[17] C. Chow and C. Liu, "Approximating Discrete Probability Distributions with Dependence Trees," *IEEE Trans. Inf. Theory*, vol. 14, no. 3, 1968, pp. 462-467.

[18] M.A. Faust and R.A. Gulledge, "Identifying Harmful Marine Dinoflagellates," *Smithsonian Inst. Contrib. U.S. Natl. Herb.*, vol. 42, 2002, pp. 1-144.

[19] S. Willcox, N.A. Moltschaniwskyj, and C. Crawford, "Asexual Reproduction in Scyphistomae of Aurelia sp.: Effects of

Temperature and Salinity in an Experimental Study," *J. Experimental Marine Biology Ecology*, vol. 353, no. 1, 2007, pp. 107-114.

[20] C. Zilberberg, A.M. Solé-Cava, and M. Klautau, "The Extent of Asexual Reproduction in Sponges of the Genus Chondrilla (Demospongiae: Chondrosida) from the Caribbean and the Brazilian Coasts," *J. Experimental Marine Biology Ecology*, vol. 336, no. 2, 2006, pp. 211-220.

[21] K. Holmstrom and H.J. Jensen, "Who Runs Fastest in an Adaptive Landscape: Sexual Versus Asexual Reproduction," *Physica A: Statistical Theoretical Physics*, vol. 337, no. 1-2, 2004, pp. 185-195.

[22] W. Song, "Morphogenesis of Cyrtohymena tetracirrata (Ciliophora, Hypotrichia, Oxytrichidae) during Binary Fission," *European J. Protistology*, vol. 40, no. 3, 2004, pp. 245-254.

[23] K. Lee et al., "Light Regulation of Asexual Development in the Rice Blast Fungus, Magnaporthe Oryzae," *Fungal Genetics Biology*, vol. 43, no. 10, 2006, pp. 694-706.

[24] T. Yanagi, T. Yachi and N. Okuda, "Photoperiodic Reaction of Sexual and Asexual Reproduction in Fragaria Chiloensis L. CHI-24-1 Plants Grown at Various Temperatures," *Scientia Horticulturae*, vol. 110, no. 2, 2006, pp. 187-191.

[25] F. Prall and C. Ostwald, "High-Degree Tumor Budding and Podia-Formation in Sporadic Colorectal Carcinomas with K-Ras Gene Mutations," *Hum. Pathol.*, vol. 38, 2007, pp. 1696-702.

[26] A.V. Ramayya et al., "Binary and Ternary Fission Studies with ^{252}Cf," *Progress Particle Nuclear Physics*, vol. 46, no. 1, 2001, pp. 221-229.

[27] S. Stertz et al., "The Intracellular Sites of Early Replication and Budding of SARS-Coronavirus," *Virology*, vol. 361, no. 2, 2007, pp. 304-315.

[28] R.F. Green and D.L.G. Noakes, "Is a Little Bit of Sex as Good as a Lot?" *J. Theoretical Biology*, vol. 174, no. 1, 1995, pp. 87-96.

[29] M.T. Ghiselin, "The Evolution of Sex: A History of Competing Points of View," *The Evolution of Sex: An Examination of Current Ideas*, R.E. Michod and B.R. Levin, Eds., Sunderland: Sinauer Associates Inc., 1988, pp. 7-23.

[30] M. Clerc and J. Kennedy, "The Particle Swarm–Explosion, Stability, and Convergence in a Multidimensional Complex Space," *IEEE Trans. Evolutionary Computation*, vol. 6, 2002, pp. 58-73.

[31] M. Mitchel, *An Introduction to Genetic Algorithms*, MIT Press, 1999.

[32] M. Dorigo and T. Stützle, *Ant Colony Optimization*, India, New Delhi: Prentice-Hall, 2005.

[33] http://www.norsys.com/networklibrary.html

[34] I.A. Beinlinch et al., "The ALARM Monitoring System: A Case Study with Two Probabilistic Inference Techniques for Belief Networks," *Proc. 2nd European Conf. Artificial Intell. Medicine*, 1989, pp. 247-256.

Ali Reza Khanteymoori received his BSc and MSc in computer engineering from Iran University of Science and Technology (IUST), and Amirkabir University of Technology, Tehran, Iran. He is currently a PhD candidate at the Computer Engineering Department, Amirkabir University of Technology, Tehran, Iran. He is member of the faculty of the Engineering Department at Zanjan University. His research interests include graphical models, speech processing, machine learning, and biocomputing.

Mohammad Bagher Menhaj received his PhD from the School of Electrical and Computer Engineering at OSU in 1992. After completing one year with OSU as postdoctoral fellow in 1993, he joined Amirkabir University of Technology, Tehran, Iran. From December 2000 to August 2003, he was with School of Electrical and Computer Engineering and Department of Computer Science at OSU as a visiting faculty member and research scholar. His main research interests are the theory of computational intelligence, learning automata, adaptive filtering and its applications in control, power systems, image processing, pattern recognition, communications, rough set theory, and knowledge discovery.

Mohammad Mehdi Homayounpour received his BSc and MSc in electrical engineering from Amirkabir University of Technology and Khajeh Nasir Toosi University of Technology, Tehran, Iran, respectively, and his PhD degree in electrical engineering from Université de Paris Sud, Paris, France. He is currently an associate professor of the Computer Engineering Department, Amirkabir University of Technology, Tehran, Iran. His research interests include digital signal processing, speech processing, speech recognition, text-to-speech, and speaker recognition.