Structure Learning of Bayesian Networks by Genetic Algorithms: A Performance Analysis of Control Parameters

Pedro Larrañaga, Mikel Poza, Yosu Yurramendi, Roberto H. Murga, and Cindy M.H. Kuijpers

Abstract—We present a new approach to structure learning in the field of Bayesian networks: We tackle the problem of the search for the best Bayesian network structure, given a database of cases, using the genetic algorithm philosophy for searching among alternative structures. We start by assuming an ordering between the nodes of the network structures. This assumption is necessary to guarantee that the networks that are created by the genetic algorithms are legal Bayesian network structures. Next, we release the ordering assumption by using a "repair operator" which converts illegal structures into legal ones. We present empirical results and analyze them statistically. The best results are obtained with an elitist genetic algorithm that contains a local optimizer.

Index Terms—Bayesian network, genetic algorithm, structure learning, combinatorial optimization, performance analysis.

1 Introduction

BAYESIAN networks (BNs) have become popular over the last few years within the AI probability and uncertainty community as a method of reasoning under uncertainty. From an informal perspective, BNs are directed acyclic graphs (DAGs), where the nodes are random variables and where the arcs specify the independence assumptions between these variables. After construction, a BN constitutes an efficient device for performing probabilistic inference.

The problem of searching the BN that best reflects the dependence relations in a database of cases is a difficult one because of the large number of possible DAG structures, given even a small number of nodes to connect. In this paper, we present a method for solving this problem of the structure learning of BNs from a database of cases based on genetic algorithms.

The structure of the paper is as follows. In the next section, we introduce the Bayesian networks and we describe the problem of the search of such a network from a database of cases. A brief introduction on genetic algorithms is given in Section 3. In Section 4, we show how genetic algorithms can be used for tackling the problem of the structure learning of BNs. We describe two different approaches, namely with and without assuming an ordering between the nodes of the network. In the latter approach, the offspring constructed by the genetic algorithm are not necessarily BN structures, they may have to be repaired. In both approaches we use a Bayesian approach to measure the fitness of the structures. Empirical results obtained with simulations of the ASIA and ALARM networks are presented in Section 5. Finally, in Section 6, we conclude the work and give some directions for future research.

• *The authors are with the Department of Computer Science and Artificial Intelligence, University of the Basque Country, P.O. Box 649, E-20080 San Sebastián, Spain.*
E-mail: {ccplamup, ccpyumej, ccbmugar, ccbkukuc}@si.ehu.es.

Manuscript received Mar. 16, 1995; revised Mar. 27, 1996. Recommended for acceptance by H.R. Keshavan.
For information on obtaining reprints of this article, please send e-mail to: transpami@computer.org, and reference IEEECS Log Number P96041.

2 Bayesian Networks and Structure Learning

Bayesian networks and associated schemes constitute a probabilistic framework for reasoning under uncertainty that in recent years has gained popularity in the community of artificial intelligence [1], [2], [3].

From an informal perspective, Bayesian networks are directed acyclic graphs (DAGs), where the nodes are random variables, and the arcs specify the independence assumptions that must be held between the random variables.

To specify the probability distribution of a BN, one must give prior probabilities for all root nodes (nodes with no predecessors) and conditional probabilities for all other nodes, given all possible combinations of their direct predecessors. These numbers in conjunction with the DAG, specify the BN completely. The joint probability of any particular instantiation of all n variables in a BN can be calculated as follows:

$$P(x_1, \ldots, x_n) = \prod_{i=1}^{n} P(x_i \mid \pi_i)$$

where x_i represents the instantiation of the variable X_i and π_i represents the instantiation of the parents of X_i.

Once the network is constructed it constitutes an efficient device to perform probabilistic inference. Nevertheless, the problem of building such a network remains. The structure and conditional probabilities necessary for characterizing the network can be provided either externally by experts or from direct empirical observations. The learning task in a BN can be separated into two subtasks, *structure learning*, that is to identify the topology of the network, and *parameter learning*, the numerical parameters (conditional probabilities) for a given network topology.

0162-8828/96$05.00 ©1996 IEEE

Our work focuses upon structure learning rather than upon parameter learning. However, for complete BN construction it is also necessary to estimate the parameters. Previous research in structure learning of BNs has already been carried out. Some authors [4], [5], [6] have worked on inducing the structure of trees or polytrees from a database of cases. The more relevant works on structure learning on multiply connected networks have been developed in [7], [8], [9], [10], [11], [12], [13], [14], [15], [16], [17], [18], [19], [20], [21], [22], [23], [24], [25], [26].

A frequently used procedure for BN network structure construction from data is the K2 algorithm of Cooper and Herskovits [9]. This algorithm (see Fig. 1) searches, given a database D for the BN structure B_S^* with maximal $P(B_S,D)$, where $P(B_S,D)$ is as described in the following theorem proved in [9].

Algorithm K2

INPUT: A set of n nodes, an ordering on the nodes, an upper bound u on the number of parents a node may have, and a database D containing m cases.
OUTPUT: For each node, a printout of the parents of the node.

```
BE GIN K2
   FOR i := 1 TO n DO
      BEG IN
         Π_i := 0;
         P_old := g(i, Π_i):
         OKToProceed := TRUE
         WHI LE OKToProceed AND | Π_i | < u DO
            BEG IN
               Let Z be the node in Pred(X_i) – Π_i that
                maximizes g(i, Π_i ∪ {Z});
               P_new := g(i, Π_i ∪ {Z});
               IF P_new > P_old THEN
                  BEG IN
                     P_old := P_new;
                     Π_i := Π_i ∪ {Z}
                  END
               ELSE OKToProceed := FALSE;
            END;
         WRITE('Node:,' X_i, 'Parents of this node:,' Π_i)
      END;
END K2.
```

Fig. 1. The K2 algorithm.

THEOREM. *Let Z be a set of n discrete variables, where a variable X_i in Z has r_i possible value assignments: $(v_{i1}, \ldots, v_{ir_i})$. Let D be a database of m cases, where each case contains a value assignment for each variable in Z. Let B_S denote a belief network structure containing just the variables in Z, and B_P the conditional probabilities. Each variable X_i in B_S* has a set of *parents, which are represented with a list of variables Π_i. Let w_{ij} denote the jth unique instantiation of Π_i relative to D. Suppose there are q_i such unique instantiations of Π_i. Define N_{ijk} to be the number of cases in D in which variable X_i has the value v_{ik} and Π_i is instantiated as w_{ij}. Let $N_{ij} = \sum_{k=1}^{r_i} N_{ijk}$. If given a BN model, the cases occur independently and the density function $f(B_P \mid B_S)$ is uniform, then it follows that*

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

The K2 algorithm assumes that an ordering on the variables is available and that, a priori, all structures are equally likely. It searches, for every node, the set of parent nodes that maximizes the following function:

$$g(i,\Pi_i) = \prod_{j=1}^{q_i}\frac{(r_i-1)!}{(N_{ij}+r_i-1)!}\prod_{k=1}^{r_i}N_{ijk}!.$$

K2 is a *greedy* heuristic. It starts by assuming that a node lacks parents, after which in every step it adds incrementally that parent whose addition most increases the probability of the resulting structure. K2 stops adding parents to the nodes when the addition of a single parent can not increase the probability. Obviously, this approach does not guarantee to obtain the structure with the highest probability.

A possible improvement of K2 could be the determination of the best combination of at most u parent nodes in which case the number of searches to be carried out for a node j would increase from $\prod_{i=1}^{u}(n-j-i)$ to $\sum_{i=1}^{u}\binom{n-j-1}{i}$.

In Section 4, we present a genetic search algorithm for BN structures that for the evaluation of these structures uses the same metric as K2. We start by maintaining the same ordering restiction on the variables as K2, after which this restriction is released.

3 Genetic Algorithms

Recently five approaches of heuristic search have emerged for solutions to combinatorial complex problems: evolutionary algorithms, neural networks, simulated annealing, tabu search, and target analysis. The first two—evolutionary algorithms and neural networks—are inspired by principles derived from biological sciences; and simulated annealing derives from physical science, notably the second law of thermodynamics. Tabu search and target analysis stem from the general tenets of intelligent problem-solving.

Evolutionary algorithms are probabilistic search algorithms which simulate natural evolution. They were proposed about 30 years ago [27], [28]. Their application to combinatorial optimization problems, however, has only recently become an actual research topic. Roughly speaking three different types of evolutionary algorithms exist: genetic algorithms [29], [30], [31], evolutionary programming [32], and evolution strategies [33]. In this paper we consider the genetic algorithms (GAs). GAs are search algorithms based on the mechanics of natural selection and natural genetics. They combine survival of the fittest among string

structures with a structured yet randomized information exchange to form a search algorithm that under certain conditions evolves to the optimum with probability arbitrarily close to 1 [34], [35], [36], [37], [38].

In GAs the search space of a problem is represented as a collection of individuals. The individuals are represented by character strings, which are often referred to as chromosomes. The purpose of the use of a GA is to find the individual from the search space with the best "genetic material." The quality of an individual is measured with an objective function. The part of the search space to be examined is called the population.

Roughly, a GA works as follows: First, the initial population is chosen, and the quality of each of its individuals is determined. Next, in every iteration parents are selected from the population. These parents produce children, which are added to the population. For all newly created individuals of the resulting population a probability near zero exits that they "mutate," i.e., they change their hereditary distinctions. After that, some individuals are removed from the population according to a selection criterion in order to reduce the population to its initial size. One iteration of the algorithm is referred to as a generation.

The operators which define the child production process and the mutation process are called the crossover operator and the mutation operator respectively. Both operators are applied with different probabilities named the crossover probability and the mutation probability. Mutation and crossover play different roles in the GA. Mutation is needed to explore new states and helps the algorithm to avoid local optima. Crossover should increase the average quality of the population. By choosing adequate crossover and mutation operators as well as an appropriate reduction mechanism, the probability that the GA results in a near-optimal solution in a reasonable number of iterations increases.

The pseudocode of an abstract genetic algorithm (AGA), is shown in Fig. 2.

Definitions and details of operators related to GAs are defined in Appendix A.

begin AGA
  Make initial population at random
  WHILE NOT stop DO
    BEGIN
    Select parents from the population.
    Produce children from the selected parents.
    Mutate the individuals.
    Extend the population by adding the children to it.
    Reduce the extended population.
    END
  Output the best individual found.
end AGA

Fig. 2. The pseudocode of the abstract genetic algorithm.

4 Genetic Algorithms in the Structure Learning of Bayesian Networks

4.1 Notation and Representation

Our approach on structure learning in the framework of Bayesian networks is based on genetic algorithms. Denoting with D the set of BN structures for a fixed domain with n variables, and the alphabet S being {0,1 }, a Bayesian network structure can be represented by an $n \times n$ connectivity matrix C, where its elements, c_{ij}, verify:

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

In our genetic approach, we represent an individual of the population by string:

$$c_{11}c_{21} \ldots c_{n1}c_{12}c_{22} \ldots c_{n2} \ldots c_{1n}c_{2n} \ldots c_{nn}.$$

With this representation in mind, we will show how the crossover and mutation operators work, by using simple examples.

EXAMPLE 1. Consider a domain of three variables on which the two BN structures of Fig. 3 are defined. The connectivity matrices that correspond to the network structures are, respectively,

$$\begin{pmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 1 & 1 & 0 \end{pmatrix} \text{ and } \begin{pmatrix} 0 & 0 & 1 \\ 0 & 0 & 1 \\ 0 & 0 & 0 \end{pmatrix}.$$

Using the above described representation, the networks are represented by the strings: 001001000 and 000000110. Suppose now that the two network structures are selected to crossover and that the crossover point is chosen between the sixth and the seventh bit. This gives the offspring strings 001001110 and 000000000. Hence, the created offspring structures are the ones presented in Fig. 4.

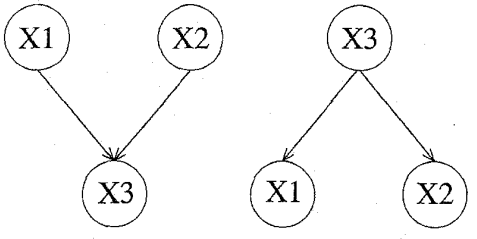


Fig. 3. The parent structures of Example 1.

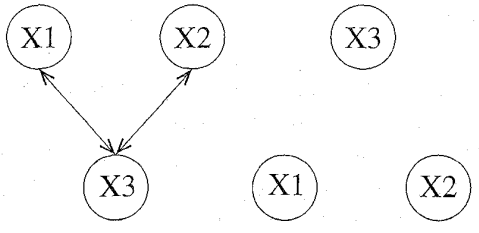


Fig. 4. Crossover does not always result in legal structures.

We see that the offspring structures do not correspond to DAGs. We say that the one point crossover operator is not a closed operator.

EXAMPLE 2. Consider the DAG of Fig. 5a. It is represented by the string 010001000. Suppose that the seventh bit is altered by mutation. This gives the string 010001100, which corresponds with to cyclic graph of Fig. 5b. We observe that the mutation operator is not a closed operator either.

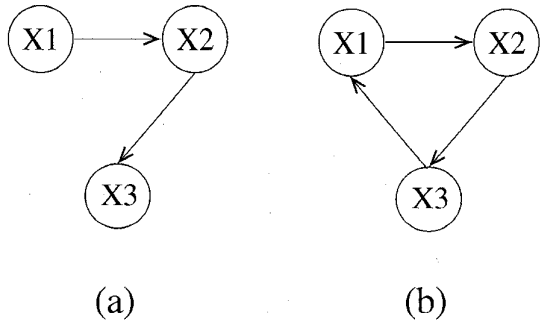


Fig. 5. Mutation is not a closed operator.

4.2 With Ordering Between the Nodes

In the previously mentioned methods to tackle the structure learning of BN structures, most often an ordering between the variables of the BN is assumed. This means that a node X_i can only have node X_j as a parent node if in the ordering node X_j comes before node X_i. It is easy to verify that in case an ordering is assumed, the connectivity matrices of the network structures are triangulated and that therefore the genetic operators are closed operators. In this case the string's length used to represent a BN structure with n nodes is $\binom{n}{2}$, instead of n^2 of the general case. Under the ordering assumption the cardinality of the search space is given by the formula $2^{\binom{n}{2}}$. In [39], the authors demonstrate that for a domain of 10 nodes, which means a search space of approximately 35×10^{12} different structures, the genetic approach is able to find the optimum structure with an average number of structures evaluated that is smaller than 4,000. In [40], two different types of algorithms are compared using simulations of the ALARM network. See Table 1 for the cardinality of the search space for different values of n, where n is the number of nodes in the BN.

TABLE 1
THE CARDINALITY OF THE SEARCH SPACE

number of nodes	With ordering	Without ordering
5	1.024e03	2.928e04
8	2.684e08	7.837e11
10	3.518e13	4.175e18
15	4.056e31	2.377e41
20	1.569e57	2.344e72
25	2.037e90	2.659e111
30	8.872e130	2.714e158
35	1.296e179	2.118e213
37	3.061e200	3.008e237
40	6.359e234	1.124e276

4.3 Without Ordering Between the Nodes

If we do not assume ordering between the nodes, the genetic operators are not closed operators. Then the cardinality of the search space (see Table 1) is given by the formula [41]:

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

To assure the closeness of the genetic operators we introduce a *repair operator*, which converts structures that are not a DAG into a DAG. This repair operator is inserted in the algorithm, just after the mutation operator. The objective of the repair operator is to transform the child structures that do not verify the DAG conditions into DAGs, by randomly eliminating the edges that invalidate the DAG conditions.

4.4 The Algorithm

In this section, we describe the characteristics of the algorithms to be used. The initial population of λ individuals is generated at random. Due to the huge magnitude of the search space, the individuals are created subjected to the restriction that a node never has more than four parent nodes. Höffgen [42] proves that even subject to this restriction, the problem of model search of BN structures is NP-hard.

The *objective function* to be used to evaluate the quality of a structure, is based on the formula of Cooper and Herskovits, described in Section 2, expressed in terms of the natural logarithm. Therefore, our aim is to find the structure with the highest joint probability.

Each individual is selected to be a parent with a probability proportional to the rank of its objective function. If we denote by I_t^j the jth individual of the population at time t, and by $\text{rank}(g(I_t^j))$ the rank of its objective function, then the probability $p_{j,t}$ that individual I_t^j is selected to be a parent is equal to

$$p_{j,t} = \frac{\text{rank}\left(g\left(I_t^j\right)\right)}{\lambda(\lambda+1)/2}.$$

The purpose of this transformation is to avoid the *premature convergence* of the algorithm caused by *superindividuals*, individuals which, because of their extremely large fitness with respect to the rest of the population, would be selected almost always.

In the offspring production process, two parent BN structures are recombined by means of 1-point crossover. The mutation of the offspring structures consists of the probabilistic alteration of the bits that represent their connectivity matrices. This alteration is performed with a probability near to zero. As we already remarked in Section 4.2, crossover and mutation only result in legal DAGs in case an ordering between the nodes is assumed.

After applying crossover and mutation, the created structures do not necessarily fulfill the restriction that the nodes all have at most four parents. To maintain this restriction, in a first approach, we select q parents at random, $(0 \le q \le 4)$, for every node from the parent nodes resulting from crossover and mutation. This approach, however, will give poor results. Therefore, we try a second approach in which we hybridize the genetic algorithm, with a local optimizer. This

optimizer selects the best subset of at most four parent nodes for each node in a network structure. The process of generating child structures and the application of the local optimizer, is repeated in every iteration of the algorithm.

Once the offspring are converted into DAGs in which the nodes never have more than four parents, they are added to the population, after which this population is reduced to its original size. This reduction is carried out following two different criteria to which we will refer to as the *elitist (of degree λ)* reduction criterion and the *simple* reduction criterion. Using the former criterion, the population in the next iteration consists of the λ best structures (among parents and offspring) in the current iteration. With the latter criterion, the children in the current iteration constitute the population in the next iteration.

We decide to *stop* the algorithms when either 10,000 structures have been evaluated or when in 1,000 successive evaluations, the value of the objective function of the best structure corresponds with the average value of the objective function.

The algorithm uses the following parameters:

Population size λ, in the next section we will present results of experiments carried out with $\lambda = 10$ and $\lambda = 50$.

Crossover probability p_c, we choose $p_c = 0.5$ and $p_c = 0.9$.

Mutation rate p_m, we will consider $p_m = 0.1$ and $p_m = 0.01$.

Reduction criterion, we use the simple reduction criterion as well as the elitist reduction criterion.

Ordering restriction, experiments are done with and without assuming ordering between the nodes. The absence of the ordering assumption implies the necessity of the repair operator.

Hybridization, we carry out experiments with and without local optimizer. When the local optimizer is not used the excess parent nodes are deleted at random. If the local optimizer is used, for each node the best subset of at most four parents is chosen—the subset which maximizes the posterior probability—from the set of its parent nodes.

5 Results of the Experiments

5.1 Introduction

In this section we present the empirical results obtained. The different steps to evaluate the behavior of the genetic algorithms considered, have been the following:

- Step 1: Determinate a BN (structure + conditional probabilities) and simulate it, obtaining a database of cases D, which must reflect the conditional independence relations between the variables.
- Step 2: Using the approach based on genetic algorithms try to obtain the BN structure B_s^*, which maximizes the probability $P(D \mid B_S)$.
- Step 3: Evaluate the fitness of the solutions found.

Fig. 6 shows these steps.

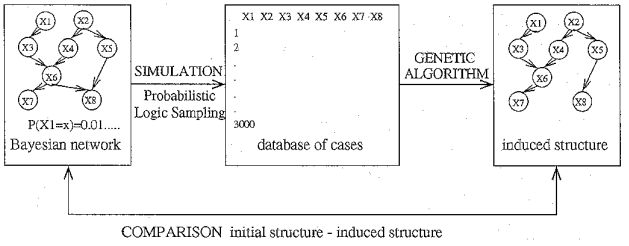


Fig. 6. The evaluation of the proposed method for structure learning from a database of cases.

The BNs used in the experiments are the ASIA and the ALARM networks. The ASIA network, introduced by Lauritzen and Spiegelhalter [43] to illustrate their method of propagation of evidence, considers a small piece of fictitious qualitative medical knowledge. Fig. 7 presents the structure of the ASIA network. Several techniques exist for simulating BNs, we used *probabilistic logic sampling* [44], with which we generated a database of 3000 cases. The ALARM network, see Fig. 8, was constructed by Beinlinch et al. [45] as a prototype to model potential anesthesia problems in the operating room. We will use the 3000 first cases of the database that was generated from it by Herskovits [46]. For both database we consider different subsets consisting of the first 500, 1000, 2000, and 3000 cases. The evaluations of the initial structures for the different databases can be seen in Table 2.

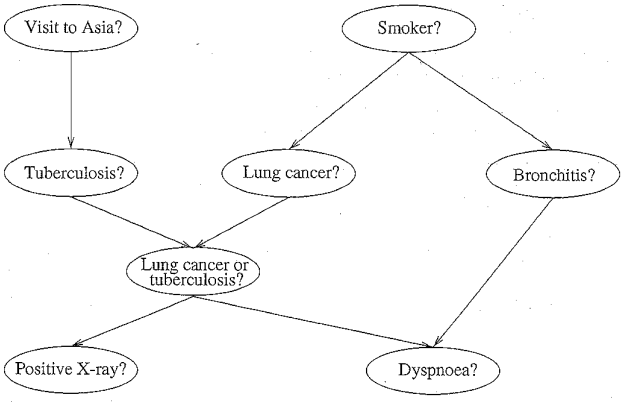


Fig. 7. The structure of the ASIA network.

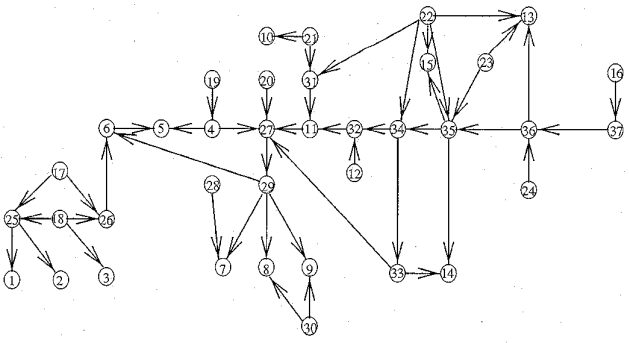


Fig. 8. The structure of the ALARM network.

TABLE 2
THE EVALUATIONS OF THE INITIAL STRUCTURES

	log $P(D\|B_s)$	
number of cases	ASIA	ALARM
500	-5.4856e02	-2.6461e03
1000	-1.0800e03	-5.0345e03
2000	-2.1541e03	-9.7291e03
3000	-3.2437e03	-1.4412e04

5.2 Parameters

As objective function to maximize by the genetic algorithm, the formula of Cooper and Herskovits is used. For each database of cases, for each of the 10 runs done with each of the 64 different parameter combinations, four values related to the behavior of the algorithm are considered:

- the *average objective function value* (*AOF*), average of the objective values of the 10 runs.
- the *best objective function value* (*BOF*), obtained through the evolution of the GA.
- the *average Hamming distance* (*AHD*), average of the Hamming distance between the best BN structure found in each search, BN_f, and the initial BN structure $BN_{initial}$.
- the *average number of evaluations* (*ANE*) performed before BN_f was found.

Tables 3-18 present the same structure. The first block of numeric information, contains results related with $p_m = 0.01$, and $p_c = 0.5$. In this block the first row corresponds to AOF, the second row to BOF (both expressed in terms of $-\log P(D \mid B_S)$), and the two last correspond to AHD and ANE, respectively. The three following blocks contain the information for the combination of parameters: $p_m = 0.01$, $p_c = 0.9$; $p_m = 0.10$, $p_c = 0.5$, and $p_m = 0.10$, $p_c = 0.9$, respectively.

Only for the elitist algorithms we have considered the average number of evaluated structures, since these algorithms are the only ones that guarantee their convergence. In Fig. 9, we show how the different types of genetic algorithms—with or without local optimizer, simple, or elitist—typically evolve.

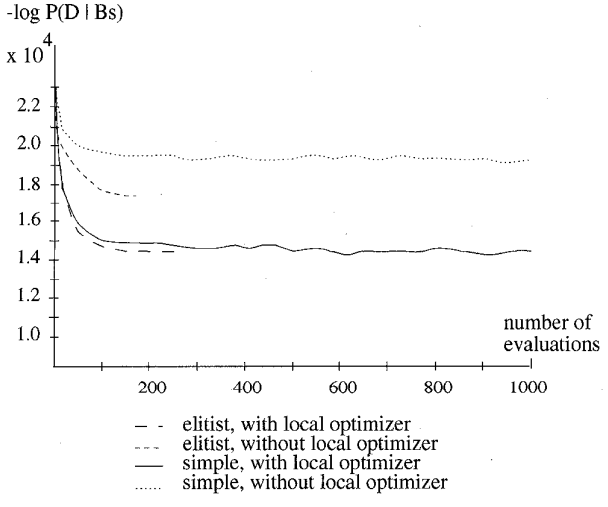


Fig. 9. A comparison of the evolution of the best individual found with the different types of algorithm (based on results obtained with 3,000 cases of the ALARM network).

5.3 Results Obtained with Order Restriction

5.3.1 The ASIA Network

The results corresponding to the 500, 1000, 2000, and 3000 cases databases are represented in the Tables 3, 4, 5, and 6, respectively. Noticeable is that of all genetic algorithms that follow the elitist reduction criterion, independent of the other parameters, the average results are better than the evaluation of the initial network structure (compare Table 2 and Tables 3, 4, 5, 6). For all databases, the Kruskal-Wallis test [47] shows that statistically significant differences exist in the evaluation function and in the Hamming distance, with respect to the hybridization, the reduction criterion and the mutation probability. In the average number of evaluations statistically significant differences are found with respect to the hybridization, the population size and the crossover operator. Moreover, the robustness of the hybrid algorithms which incorporate the local optimizer can be observed, as well as the bad results of the no-hybrid algorithms which use the simple selection criterion. As the database size increases, the evaluation function increases and the evaluation function corresponding to the network structure, induced by the algorithm, adjusts to the evaluation of the initial structure. With respect to the average Hamming distance between the initial network structure and the best structures obtained in the last iteration of the algorithm, the worst results are obtained when no local optimizer is incorporated, the simple reduction criterion is used and the mutation rate is high. The rest of the algorithms have a quite stable behavior, obtaining AHD values of roughly 4, 4, 1, and 0 for the databases with, respectively, 500, 1000, 2000, and 3000 cases.

Evidently, the algorithms that use a population of 10 individuals converge faster than the ones that have a population of 50 individuals, the ones with a population of 10 carry out about half of the evaluations of the ones with a population of 50. Moreover, the hybrid algorithms with a low mutation rate have a relatively high number of evaluated structures.

5.3.2 The ALARM Network

Tables 7, 8, 9, and 10 represent the results found with the databases of 500, 1000, 2000, and 3000 cases, respectively. All the hybrid algorithms that use the elitist reduction criterion give, independent of the other parameters, better average results than the evaluation that corresponds to the initial structure. Also the hybrid algorithms that are not elitist in combination with a low mutation rate sometimes are able to obtain, on average, superior results compared to the evaluation of the initial structure.

With relation to the statistical significance of the analyzed parameters, we observe, for all databases, the same significances as we found with the ASIA network and statistically significant differences in the average number of evaluations with respect to the mutation probability. In this case, probably because of the increasing dimension of the search space, only the hybrid algorithms that use elitist reduction maintain a low variability in its behavior.

Like with the ASIA network, increasing the database size results in structures the evaluations of which are better approximations of the evaluation of the initial structure.

TABLE 3
RESULTS WITH THE ASIA NETWORK WITH ORDER RESTRICTION (500 CASES)

with local optimizer				without local optimizer			
elitist		simple		elitist		simple	
$\lambda = 10$	$\lambda = 50$	$\lambda = 10$	$\lambda = 50$	$\lambda = 10$	$\lambda = 50$	$\lambda = 10$	$\lambda = 50$
545.7	545.7	551.4	545.7	546.2	545.9	545.9	546.1
545.7	545.7	545.7	545.7	545.7	545.7	545.7	545.7
4.0	4.0	3.7	4.0	5.5	4.4	4.5	4.5
283	475	—	—	558	825	—	—
545.7	545.7	545.7	545.7	545.9	545.9	545.9	545.7
545.7	545.7	545.7	545.7	545.7	545.7	545.7	545.7
4.0	4.0	4.0	4.0	4.6	4.6	4.7	4.1
261	585	—	—	527	1170	—	—
545.7	545.7	545.7	545.7	545.9	545.7	552.0	547.8
545.7	545.7	545.7	545.7	545.7	545.7	547.6	547.3
4.0	4.0	4.0	4.0	4.6	4.4	8.5	5.6
100	350	—	—	495	838	—	—
545.7	545.7	545.7	545.7	545.7	545.7	562.3	552.6
545.7	545.7	545.7	545.7	545.7	545.7	553.5	549.3
4.0	4.0	4.0	4.0	4.0	4.1	8.8	7.5
162	608	—	—	806	1800	—	—

TABLE 4
RESULTS WITH THE ASIA NETWORK WITH ORDER RESTRICTION (1,000 CASES)

with local optimizer				without local optimizer			
elitist		simple		elitist		simple	
$\lambda = 10$	$\lambda = 50$	$\lambda = 10$	$\lambda = 50$	$\lambda = 10$	$\lambda = 50$	$\lambda = 10$	$\lambda = 50$
1076.1	1076.1	1076.1	1076.1	1076.1	1076.1	1076.1	1076.3
1076.1	1076.1	1076.1	1076.1	1076.1	1076.1	1076.1	1076.1
4.0	4.0	4.0	4.0	4.6	5.2	5.8	5.0
240	413	—	—	328	750	—	—
1076.1	1076.1	1076.1	1076.1	1076.1	1076.1	1076.1	1076.1
1076.1	1076.1	1076.1	1076.1	1076.1	1076.1	1076.1	1076.1
4.0	4.0	4.0	4.0	5.2	4.3	4.0	5.3
270	608	—	—	581	1193	—	—
1076.1	1076.1	1076.4	1076.1	1076.1	1076.1	1083.9	1078.7
1076.1	1076.1	1076.1	1076.1	1076.1	1076.1	1078.3	1077.2
4.0	4.0	3.5	4.0	4.9	5.0	9.0	6.3
110	375	—	—	450	975	—	—
1076.1	1076.1	1077.7	1076.1	1076.1	1076.1	1093.1	1083.1
1076.1	1076.1	1076.1	1076.1	1076.1	1076.1	1080.9	1077.1
4.0	4.0	3.7	4.0	4.6	4.7	10.8	8.1
144	652	—	—	680	1800	—	—

TABLE 5
RESULTS WITH THE ASIA NETWORK WITH ORDER RESTRICTION (2,000 CASES)

with local optimizer				without local optimizer			
elitist		simple		elitist		simple	
$\lambda = 10$	$\lambda = 50$	$\lambda = 10$	$\lambda = 50$	$\lambda = 10$	$\lambda = 50$	$\lambda = 10$	$\lambda = 50$
2154.0	2154.0	2154.0	2154.0	2154.0	2154.0	2154.1	2154.1
2154.0	2154.0	2154.0	2154.0	2154.0	2154.0	2154.0	2154.0
1.0	1.0	1.0	1.0	1.0	1.0	1.1	1.2
193	388	—	—	495	788	—	—
2154.0	2154.0	2154.0	2154.0	2154.0	2154.0	2154.1	2154.0
2154.0	2154.0	2154.0	2154.0	2154.0	2154.0	2154.0	2154.0
1.0	1.0	1.0	1.0	1.0	1.0	1.2	1.0
288	608	—	—	504	1283	—	—
2154.0	2154.0	2158.7	2154.0	2154.0	2154.1	2166.1	2156.8
2154.0	2154.0	2154.0	2154.0	2154.0	2154.0	2157.2	2154.1
1.0	1.0	0.8	1.0	1.1	1.2	7.7	4.2
83	363	—	—	550	925	—	—
2154.0	2154.0	2165.1	2154.0	2154.0	2154.0	2177.4	2162.8
2154.0	2154.0	2154.0	2154.0	2154.0	2154.0	2160.5	2157.8
1.0	1.0	1.8	0.9	1.0	1.0	8.5	6.1
1040	473	—	—	743	1778	—	—

TABLE 6
RESULTS WITH THE ASIA NETWORK WITH ORDER RESTRICTION (3,000 CASES)

with local optimizer				without local optimizer			
elitist		simple		elitist		simple	
$\lambda = 10$	$\lambda = 50$	$\lambda = 10$	$\lambda = 50$	$\lambda = 10$	$\lambda = 50$	$\lambda = 10$	$\lambda = 50$
3243.7	3243.7	3243.7	3243.7	3244.5	3243.7	3243.7	3244.1
3243.7	3243.7	3243.7	3243.7	3243.7	3243.7	3243.7	3243.7
0.0	0.0	0.0	0.0	0.4	0.2	0.1	1.5
235	338	—	—	488	788	—	—
3243.7	3243.7	3243.7	3243.7	3243.7	3243.7	3243.7	3243.7
3243.7	3243.7	3243.7	3243.7	3243.7	3243.7	3243.7	3243.7
0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.1
207	450	—	—	527	1215	—	—
3243.7	3243.7	3243.7	3243.7	3243.7	3243.7	3259.2	3248.8
3243.7	3243.7	3243.7	3243.7	3243.7	3243.7	3244.6	3245.8
0.0	0.0	0.5	0.0	0.0	0.8	7.1	3.6
75	250	—	—	520	963	—	—
3243.7	3243.7	3246.02	3243.7	3243.8	3243.7	3307.0	3258.0
3243.7	3243.7	3243.7	3243.7	3243.7	3243.7	3256.2	3252.2
0.0	0.0	0.6	0.0	0.4	0.2	9.4	7.5
126	495	—	—	585	1800	—	—

TABLE 7
RESULTS WITH THE ALARM NETWORK WITH ORDER RESTRICTION (500 CASES)

with local optimizer				without local optimizer			
elitist		simple		elitist		simple	
$\lambda = 10$	$\lambda = 50$	$\lambda = 10$	$\lambda = 50$	$\lambda = 10$	$\lambda = 50$	$\lambda = 10$	$\lambda = 50$
2635.0	2635.0	2635.0	2635.0	2784.2	2807.2	3535.6	3144.7
2635.0	2635.0	2635.0	2635.0	2744.2	2778.9	3406.6	3058.7
12.2	12.0	12.2	12.6	61.2	63.6	105.7	91.5
963	2238	—	—	4790	5000	—	—
2635.0	2635.0	2646.4	2635.0	2746.8	2807.2	3693.4	3407.5
2635.0	2635.0	2635.0	2635.0	2725.9	2778.9	3492.5	3267.1
12.6	12.2	11.0	12.0	54.8	61.8	111.7	103.0
1179	3195	—	—	8932	9000	—	—
2635.6	2635.3	2875.9	2723.8	3648.2	3605.8	3974.7	3853.5
2635.0	2635.0	2783.3	2666.7	3546.3	3423.1	3890.8	3748.5
11.6	11.3	21.5	18.1	111.2	103.7	117.0	110.8
1590	2838	—	—	1073	1563	—	—
2635.2	2635.1	2976.1	2839.3	3531.6	3517.8	3982.2	3948.4
2635.0	2635.0	2864.9	2782.5	3444.0	3438.2	3820.4	3856.9
12.1	11.4	29.9	23.6	102.8	102.3	121.2	113.9
2570	5693	—	—	2979	6638	—	—

TABLE 8
RESULTS WITH THE ALARM NETWORK WITH ORDER RESTRICTION (1,000 CASES)

with local optimizer				without local optimizer			
elitist		simple		elitist		simple	
$\lambda = 10$	$\lambda = 50$	$\lambda = 10$	$\lambda = 50$	$\lambda = 10$	$\lambda = 50$	$\lambda = 10$	$\lambda = 50$
5027.9	5027.9	5028.0	5027.9	5278.8	5423.2	6789.3	6019.5
5027.9	5027.9	5027.9	5027.9	5216.6	5264.8	6570.6	5881.8
4.0	4.0	3.9	4.0	61.2	74.5	112.4	104.2
925	2438	—	—	4925	5000	—	—
5027.9	5027.9	5050.3	5027.9	5213.3	5344.6	7169.3	6481.8
5027.9	5027.9	5027.9	5027.9	5164.3	5294.0	6882.5	6330.9
4.0	4.0	6.3	4.0	54.8	66.7	109.7	95.6
1058	3353	—	—	8919	9000	—	—
5028.4	5030.9	5586.4	5234.0	6854.3	7010.8	7674.5	7515.5
5027.9	5027.9	5361.4	5160.7	6729.0	6800.1	7447.6	7302.5
4.4	4.5	20.2	12.8	107.8	113.7	112.7	112.6
1623	2425	—	—	1285	1675	—	—
5027.9	5027.9	5686.2	5467.7	6891.4	6756.9	7833.7	7614.9
5027.9	5027.9	5515.9	5350.3	6669.4	6504.7	7578.6	7451.2
4.0	4.0	24.9	19.1	111.1	108.0	116.8	113.0
2664	6683	—	—	2043	5490	—	—

TABLE 9
RESULTS WITH THE ALARM NETWORK WITH ORDER RESTRICTIONS (2,000 CASES)

with local optimizer				without local optimizer			
elitist		simple		elitist		simple	
$\lambda = 10$	$\lambda = 50$	$\lambda = 10$	$\lambda = 50$	$\lambda = 10$	$\lambda = 50$	$\lambda = 10$	$\lambda = 50$
9720.9	9720.0	9741.2	9720.0	10,159	10,353	13,238	11,505
9720.0	9720.0	9720.0	9720.0	9995.4	10,202	12,627	11,031
3.1	3.0	3.4	3.0	66.3	74.4	111.8	97.2
848	2125	—	—	4625	5000	—	—
9720.0	9720.0	9757.2	9720.0	9955.6	10,244	13,806	12,658
9720.0	9720.0	9720.0	9720.0	9902.4	10,167	13,297	11,999
3.0	3.0	4.6	3.0	52.7	70.4	115.0	100.6
1215	2993	—	—	8879	9000	—	—
9721.2	9729.5	10,714	10,151	13,326	13,574	15,027	14,599
9720.0	9720.0	10,332	9771.0	12,497	13,100	14,665	14,091
3.2	3.4	20.0	13.7	110.2	111.4	118.5	114.4
1435	2475	—	—	1415	1538	—	—
9721.3	9720.0	10,956	10,623	13,417	13,087	15,461	14,786
9720.0	9720.0	10,504	10,345	13,039	12,838	15,080	14,233
3.1	2.9	24.0	20.5	113.3	107.4	120.2	115.1
2169	6637	—	—	1935	7335	—	—

TABLE 10
RESULTS WITH THE ALARM NETWORK WITH ORDER RESTRICTION (3,000 CASES)

with local optimizer				without local optimizer			
elitist		simple		elitist		simple	
$\lambda = 10$	$\lambda = 50$	$\lambda = 10$	$\lambda = 50$	$\lambda = 10$	$\lambda = 50$	$\lambda = 10$	$\lambda = 50$
14,411	14,404	14,405	14,404	14,824	15,325	19,362	17,080
14,404	14,404	14,404	14,404	14,641	15,009	18,731	16,452
1.2	1.0	1.1	1.0	56.4	72.8	111.9	95.5
798	2013	—	—	4930	5000	—	—
14,404	14,404	14,485	14,404	14,757	15,055	20,692	18,926
14,404	14,404	14,404	14,404	14,671	14,863	20,184	18,065
1.0	1.0	3.7	1.0	57.4	70.6	116.3	105.9
1112	2903	—	—	8946	9000	—	—
14,407	14,412	15,903	15,108	19,915	20,016	22,593	21,723
14,404	14,404	15,303	14,737	19,198	19,473	21,760	21,332
1.3	1.7	20.4	12.3	110.1	112.4	119.0	111.8
1595	2763	—	—	1260	1888	—	—
14,404	14,404	16,666	15,729	20,018	19,680	22,853	22,247
14,404	14,404	16,035	15,117	19,493	18,995	22,265	21,957
1.0	1.0	25.0	20.1	108.8	112.7	120.4	115.6
2399	5805	—	—	2268	4950	—	—

TABLE 11
RESULTS WITH THE ASIA NETWORK WITHOUT ORDER RESTRICTION (500 CASES)

with local optimizer				without local optimizer			
elitist		simple		elitist		simple	
$\lambda = 10$	$\lambda = 50$	$\lambda = 10$	$\lambda = 50$	$\lambda = 10$	$\lambda = 50$	$\lambda = 10$	$\lambda = 50$
546.6	545.9	546.4	545.3	547.4	546.0	546.6	545.8
545.4	544.9	545.5	544.9	545.5	544.9	544.9	544.9
10.7	10.6	12.9	7.4	9.9	11.9	12.2	9.8
290	950	—	—	470	1275	—	—
545.9	545.8	545.8	545.6	546.3	545.8	547.5	545.5
544.9	544.9	544.9	544.9	544.9	545.2	545.5	544.9
10.0	10.6	9.0	9.2	9.6	10.3	9.8	9.5
396	1350	—	—	657	1935	—	—
545.8	545.7	550.2	545.1	548.5	546.6	565.0	553.9
544.9	545.4	545.5	544.9	546.2	545.2	553.5	549.5
9.4	9.8	9.8	5.7	13.9	11.6	15.6	14.1
250	925	—	—	395	1975	—	—
545.6	545.3	558.4	547.8	547.4	546.2	574.9	565.8
544.9	544.9	548.3	545.5	546.2	544.9	564.1	555.9
9.1	6.8	13.0	10.4	13.1	9.5	14.9	16.1
324	1890	—	—	1008	4905	—	—

TABLE 12
RESULTS WITH THE ASIA NETWORK WITHOUT ORDER RESTRICTION (1,000 CASES)

with local optimizer				without local optimizer			
elitist		simple		elitist		simple	
$\lambda = 10$	$\lambda = 50$	$\lambda = 10$	$\lambda = 50$	$\lambda = 10$	$\lambda = 50$	$\lambda = 10$	$\lambda = 50$
1077.3	1075.6	1075.8	1075.0	1077.2	1075.9	1076.2	1075.3
1074.7	1074.7	1074.7	1074.7	1075.5	1074.7	1074.7	1074.7
14.1	11.5	12.1	9.6	13.7	13.2	11.8	11.6
350	1125	—	—	470	1550	—	—
1076.0	1075.6	1075.6	1075.3	1076.1	1075.1	1077.0	1075.1
1074.7	1074.7	1074.7	1074.7	1074.7	1074.7	1075.1	1074.7
11.7	12.9	10.6	11.2	12.6	10.5	9.5	10.8
540	1575	—	—	846	2250	—	—
1075.7	1074.9	1083.5	1075.5	1081.0	1076.0	1113.7	1088.1
1074.7	1074.7	1077.1	1074.7	1075.9	1074.7	1088.9	1083.4
11.1	9.0	12.3	7.3	15.5	11.6	16.3	14.0
230	975	—	—	430	2250	—	—
1075.2	1074.7	1093.0	1082.2	1076.8	1075.4	1125.6	1105.8
1074.7	1074.7	1076.2	1077.2	1074.7	1074.7	1099.7	1098.9
9.4	8.0	10.1	11.8	12.2	10.9	13.9	14.2
540	2115	—	—	1179	5580	—	—

TABLE 13
RESULTS WITH THE ASIA NETWORK WITHOUT ORDER RESTRICTION (2,000 CASES)

with local optimizer				without local optimizer			
elitist		simple		elitist		simple	
$\lambda = 10$	$\lambda = 50$	$\lambda = 10$	$\lambda = 50$	$\lambda = 10$	$\lambda = 50$	$\lambda = 10$	$\lambda = 50$
2153.9	2153.0	2153.6	2153.0	2156.4	2153.4	2154.3	2152.9
2152.5	2152.5	2152.5	2152.5	2153.7	2152.5	2153.0	2152.5
12.1	10.0	11.5	8.8	10.7	11.5	10.5	10.6
360	1175	—	—	475	1400	—	—
2153.6	2153.3	2153.0	2152.7	2153.9	2153.1	2155.4	2152.9
2152.5	2152.5	2152.5	2152.5	2152.5	2152.5	2154.3	2152.5
11.0	9.5	8.3	7.8	11.4	10.6	10.2	9.4
495	1485	—	—	792	2745	—	—
2153.9	2152.6	2192.0	2153.0	2156.7	2153.8	2225.4	2170.6
2152.5	2152.5	2155.3	2152.5	2154.6	2152.5	2174.7	2160.7
10.8	8.1	10.4	8.8	11.9	8.5	16.5	12.5
315	1075	—	—	460	1800	—	—
2153.1	2152.5	2202.4	2166.0	2155.5	2152.8	2291.9	2212.6
2152.5	2152.5	2167.0	2153.5	2152.5	2152.5	2228.7	2182.4
11.3	7.0	11.4	8.3	10.7	10.2	16.3	15.2
504	1935	—	—	1116	5445	—	—

TABLE 14
RESULTS WITH THE ASIA NETWORK WITHOUT ORDER RESTRICTIONS (3,000 CASES)

with local optimizer				without local optimizer			
elitist		simple		elitist		simple	
$\lambda = 10$	$\lambda = 50$	$\lambda = 10$	$\lambda = 50$	$\lambda = 10$	$\lambda = 50$	$\lambda = 10$	$\lambda = 50$
3247.2	3242.5	3243.2	3242.0	3245.6	3243.9	3245.9	3242.6
3243.0	3241.6	3241.6	3241.6	3242.9	3241.6	3241.6	3241.6
12.7	8.8	8.5	8.4	11.1	10.9	10.2	10.2
390	1150	—	—	540	1800	—	—
3244.3	3242.2	3243.0	3242.1	3244.8	3242.3	3252.5	3252.7
3242.1	3241.6	3241.6	3241.6	3242.2	3241.6	3242.5	3241.6
11.1	7.0	8.1	9.6	11.6	8.8	10.8	10.2
648	1755	—	—	639	2520	—	—
3242.6	3241.7	3270.2	3242.8	3247.6	3243.6	3372.7	3267.7
3241.6	3241.6	3247.4	3242.1	3242.7	3242.4	3275.0	3248.0
9.8	6.7	12.6	6.6	11.5	11.4	15.3	14.2
340	1150	—	—	535	2350	—	—
3242.3	3241.7	3340.3	3265.8	3244.2	3243.1	3408.5	3315.0
3241.6	3241.6	3272.6	3243.0	3246.9	3242.0	3265.5	3252.5
7.4	6.7	10.8	9.3	8.9	10.0	15.9	15.8
495	1800	—	—	1197	4860	—	—

TABLE 15
RESULTS WITH THE ALARM NETWORK WITHOUT ORDER RESTRICTION (500 CASES)

with local optimizer				without local optimizer			
elitist		simple		elitist		simple	
$\lambda = 10$	$\lambda = 50$	$\lambda = 10$	$\lambda = 50$	$\lambda = 10$	$\lambda = 50$	$\lambda = 10$	$\lambda = 50$
2663.2	2645.5	2712.3	2640.9	3357.9	3125.3	3788.1	3491.7
2642.1	2634.1	2661.9	2632.6	3181.4	3013.9	3713.7	3386.1
43.6	39.6	42.7	34.4	100.8	95.9	99.4	97.6
1330	5100	—	—	960	5800	—	—
2647.5	2641.0	2864.8	2716.3	3178.2	2997.9	3877.7	3770.0
2636.0	2632.2	2761.9	2671.5	3086.9	2921.6	3780.3	3661.5
38.3	34.0	50.3	40.1	92.5	88.1	100.3	95.8
2358	8640	—	—	3168	17,010	—	—
2822.2	2741.3	3055.1	2947.7	3866.5	3766.1	4043.9	3955.0
2733.2	2708.5	2960.2	2887.0	3821.0	3677.2	3907.5	3887.5
50.8	45.1	56.9	54.0	101.9	100.6	103.6	101.7
610	3800	—	—	360	1750	—	—
2795.7	2684.9	3134.3	3070.2	3789.5	3717.4	4059.4	4006.9
2713.3	2653.9	2990.7	3029.1	3673.4	3624.9	3972.7	3808.0
54.2	40.0	60.4	61.9	98.7	98.5	98.8	102.0
1404	13,500	—	—	810	5850	—	—

TABLE 16
RESULTS WITH THE ALARM NETWORK WITHOUT ORDER RESTRICTION (1000 CASES)

with local optimizer				without local optimizer			
elitist		simple		elitist		simple	
$\lambda = 10$	$\lambda = 50$	$\lambda = 10$	$\lambda = 50$	$\lambda = 10$	$\lambda = 50$	$\lambda = 10$	$\lambda = 50$
5074.7	4984.1	5185.5	4995.5	6293.1	5789.6	7300.0	6595.1
5057.2	4972.9	5122.7	4983.8	6127.6	5645.6	7245.9	6443.5
42.7	33.3	44.5	22.25	115.3	115.9	98.6	101.2
1750	5700	—	—	940	7500	—	—
5039.5	4933.8	5537.9	5185.5	6041.4	5638.6	7501.4	7199.3
5024.9	4969.7	5489.8	5096.8	5938.7	5425.8	7322.6	6972.4
36.4	31.4	57.1	45.6	95.1	91.6	108.5	95.4
2470	15,480	—	—	3540	2790	—	—
5437.2	5235.8	5991.0	5688.9	7451.0	7199.3	7853.8	7602.0
5285.4	5142.6	5875.3	5575.6	7362.2	6947.2	7775.2	7515.6
65.7	47.8	64.7	49.7	108.3	105.4	92.5	107.3
725	3610	—	—	390	2830	—	—
5286.2	5160.3	5991.0	5940.7	7300.0	7148.9	7803.4	7702.7
5154.3	5093.2	5845.5	5889.9	7189.5	7043.4	7795.7	7560.8
53.1	45.4	59.9	56.1	93.7	110.8	105.4	102.8
1560	11,545	—	—	970	7125	—	—

TABLE 17
RESULTS WITH THE ALARM NETWORK WITHOUT ORDER RESTRICTION (2,000 CASES)

with local optimizer				without local optimizer			
elitist		simple		elitist		simple	
$\lambda = 10$	$\lambda = 50$	$\lambda = 10$	$\lambda = 50$	$\lambda = 10$	$\lambda = 50$	$\lambda = 10$	$\lambda = 50$
9837.4	9737.5	10,131	9750.7	12,245	11,362	14,398	12,917
9745.6	9719.7	9843.4	9718.5	11,890	10,938	14,021	12,402
44.6	31.2	46.4	25.9	108.0	107.9	106.4	103.6
1460	6850	—	—	970	6150	—	—
9765.6	9734.4	11,082	10,072	11,709	10,869	14,795	14,052
9726.5	9723.3	10,747	9932.3	10,917	10,440	14,299	13,907
34.7	28.8	61.3	40.0	102.3	99.4	104.5	98.7
2574	10,710	—	—	2952	21,870	—	—
10,739	10,239	12,048	11,262	14,575	14,125	15,475	14,949
10,499	10,020	11,721	10,990	14,244	13,725	14,871	14,587
63.7	51.9	63.2	60.5	102.1	102.6	101.9	102.1
580	3250	—	—	340	2300	—	—
10,399	10,137	11,840	11,750	14,279	13,986	15,433	15,147
10,147	9951.8	10,958	11,423	13,967	13,715	14,871	14,744
60.5	49.0	64.7	62.7	100.8	106.4	103.3	107.7
1386	9360	—	—	1080	6480	—	—

TABLE 18
RESULTS WITH THE ALARM NETWORK WITHOUT ORDER RESTRICTION (3,000 CASES)

with local optimizer				without local optimizer			
elitist		simple		elitist		simple	
$\lambda = 10$	$\lambda = 50$	$\lambda = 10$	$\lambda = 50$	$\lambda = 10$	$\lambda = 50$	$\lambda = 10$	$\lambda = 50$
14,519	14,444	15,066	14,439	18,378	17,377	21,632	19,384
14,455	14,412	14,737	14,413	17,515	16,525	20,433	18,885
44.5	37.6	43.2	35.5	107.4	115.4	101.5	102.8
1690	6300	—	—	960	4550	—	—
14,446	14,432	16,344	15,026	17,414	16,114	22,016	20,986
14,417	14,412	15,772	14,735	16,360	15,646	20,931	20,687
39.0	31.1	57.3	41.2	110.2	101.1	100.0	102.0
3222	10,710	—	—	3420	24,480	—	—
16,039	15,348	17,834	16,899	21,791	21,130	22,998	22,469
15,286	15,055	16,952	16,089	20,826	20,623	22,312	22,114
64.2	55.5	64.2	59.8	103.8	103.5	101.1	99.9
580	3400	—	—	360	2050	—	—
15,657	14,919	18,164	17,702	21,346	20,910	23,259	22,508
15,223	14,678	17,490	17,492	20,900	20,451	22,201	21,827
61.6	50.8	66.2	66.0	104.7	103.8	103.0	103.3
1296	12,600	—	—	990	6570	—	—

Considering the average Hamming distance, we observe that the hybrid elitist algorithms as well as the simple hybrid algorithms that have a low mutation rate give a good performance. For these algorithms the AHD values are about 12, 4, 3, and 1 for, respectively, the 500, 1000, 2000, and 3000 cases databases.

With respect to the ANE value, we see also here that the algorithms with population size 10 converge faster than the ones with population size 50. We observe that in contrast with the ASIA network, as in the elitist hybrid algorithms, the mutation rate grows, the number of evaluations needed to produce convergence increases, while the no hybrid algorithms show the opposite tendency.

5.4 Results Obtained Without Order Restriction

5.4.1 The ASIA Network

The results obtained with the different simulations of the ASIA network are shown in Tables 11, 12, 13, 14. We observe that as the number of cases increases it becomes more difficult to find better results than the evaluation of the initial structure. For example, for the 500 cases database, we see that except for the algorithms without local optimizer, that use the simple selection criterion, all algorithms give, on average, better values than the initial structure evaluation. For the database of 1,000 cases, this observation can be done for all algorithms except for the ones that use simple reduction and a high mutation probability. With the 2,000 case database, on average, there are better results found than the initial structure evaluation by the elitist algorithms that incorporate the local optimizer, by the algorithms with the local optimizer that use simple reduction and a low mutation probability, and by the elitist algorithms that do not contain the local optimizer and have a large population size. The absence of the local optimizer in combination with simple reduction, low mutation probabilities and large population sizes also give, on average, better results than the evaluation of the initial structure. Finally, for the 3,000 case database, the best results were found using an elitist algorithm with a large population size, by algorithms that incorporate the local optimizer, that use simple reduction and a low mutation rate and by elitist algorithms that contain the local optimizer and that have a high mutation probability.

With regard to the statistical significance of the analyzed parameters, for all databases we find that with respect to the evaluation function, statistically significant differences exist in the use of the local optimizer, the reduction criterion, the population size and the mutation probability. With respect to the Hamming distance statistically significant differences exist in the use of the optimizer and in the population size. Moreover, significant differences are detected with the 500 and the 1,000 case databases in the mutation probability, with the 2,000 case database in the selection criterion.

Concerning the AHD values, we observe that the effect of the parameters is similar to the one described in Section 5.3.1. The worst performance is found with no hybrid algorithms that use simple reduction and have a high mutation rate. For the rest of the algorithms the AHD takes a value of about 10, independent of the size of the database.

With respect to the ANE value, the algorithms with a population size of 50 evaluate about 4 times more structures than the algorithms with a population size of 10. In general the ANE value increases as the mutation rate grows.

5.4.2 The ALARM Network

The results obtained are represented in Tables 15, 16, 17, 18. Like with the ASIA network, we observe that as the simulation size grows it becomes more difficult to find parameter combinations for the genetic algorithm that obtain structures with better evaluations than the initial structure.

For the 500 case database, we observe a good performance of the elitist hybrid algorithms as well of the simple hybrid algorithms that have a low mutation rate. Only two parameter combinations result in better average results than the evaluation corresponding to the ALARM network structure. There are 4 elitist hybrid algorithms with a low mutation rate and one simple hybrid algorithm ($\lambda = 50$, $p_m = 0.01$, and $p_c = 0.5$) that have better evaluations than the initial structure.

For the 1,000 case database, only the elitist hybrid algorithms with population size 50 and a low mutation rate result in a better evaluation than that of the one of the initial network structure. Moreover, the elitist hybrid algorithm that has a population size 10 and a high crossover probability has found a better evaluation than the one of the ALARM network.

With 2,000 cases, the best results were obtained with the elitist hybrid algorithms with $p_m = 0.01$, as well as with the simple hybrid algorithm with $\lambda = 50$, $p_c = 0.5$, and $p_m = 0.01$. None of the parameter combinations give on average a better performance than the initial structure. For some parameter combinations, however, some results were found that were better than the ALARM network evaluation.

For the 3,000 case database, the best results were obtained with the elitist hybrid algorithms that had a low mutation rate, as well as with the simple hybrid algorithm with $\lambda = 50$, $p_m = 0.01$, and $p_c = 0.5$. None of the searches found better results than the ALARM network evaluation, but in both of them we found the same evaluation than that of the initial structure.

With respect to the statistically significant differences in the evaluation function, for all four databases a similar behavior was detected. Significant differences are found with respect to the use of the local optimizer, the reduction criterion, the population size and the mutation rate. In the Hamming distance statistically significant differences are detected, for all the four databases, with respect to the use of the local optimizer and the mutation rate. For the 500 case database, significant differences in the Hamming distances also exist with respect to the reduction criterion. The best values of the average Hamming distance are obtained with elitist hybrid algorithms that have a low mutation rate as well as by simple hybrid ones that have low mutation and crossover rates. In these algorithms the AHD takes a value between 35 and 40.

An algorithm with population size 10 evaluates about 1/5 of the number of structures that an algorithm with population size 50 evaluates.

6 Conclusions and Further Research

We have presented a method for structure learning of Bayesian networks from a database of cases. The method is based on genetic algorithms. To guarantee the closeness of the genetic operators, we have devised a *repair operator*.

Using simulations of the ASIA and ALARM networks, we carried out a performance analysis on the control parameters of the genetic algorithms (population size, local optimizer, reduction mechanism, probability of crossover, mutation rate). The obtained results indicate that in using genetic algorithms in the structure learning of BNs, it is recommended to use a hybrid algorithm that uses elitist reduction in combination with a not too small population size and a relatively low mutation rate. This is even more true if no ordering restriction between the nodes is assumed.

It would be interesting to experiment with a repair operator that does not break cycles by deleting arcs at random, but by some optimization criterion or with a repair operator based on the fusion of two structures like in [48].

In the future we want to extend the described structure learning approach based on genetic algorithms by trying to find the optimal ordering of the system variables. We think to tackle the search for an optimal ordering with a genetic algorithm that uses genetic operators that were used in the tackling of the Traveling Salesman Problem.

We also plan to adapt the described structure learning approach to dynamic BNs [49]. In other two problems related to Bayesian networks of which we expect that they can be tackled successfully with genetic algorithms are the so-called *optimal decomposition of a BN*, and *the fusion of multiple authors BNs*.

Appendix—Basic Definitions

Definition 1. *a) An instance of an optimization problem is a pair (D, f) where D is the domain of the feasible points, and f is the cost function. The problem is to find a $w \in D$ for which $f(w) \leq f(y)$ for all $y \in D$. b) An optimization problem is a set I of instances of an optimization problem.*

Definition 2. *An encoding of a domain D is a function $e : D \rightarrow S^l$, where S is the alphabet, S^l is the search space, and $l \geq \log_{\|S\|} \| D \|$.*

Thus the encoding of the elements of D is a mapping from the domain D to the strings of length l over S.

Definition 3. *$g(x) = f(e(x))$, the composition of the functions f and e, is the objective function.*

Suppose that λ_t denotes the size of the population of the genetic algorithm at time t. To simplify, we assume that $\lambda_t = \lambda$ for every t. P_t denotes the population at time t. $P_t = \{I_t^1, \ldots, I_t^\lambda\}$, where $I_t^j (j = 1, \ldots, \lambda)$ denotes the jth individual of the population at time t, and $I_t^j = (s_1, \ldots, s_l)$, where $s_w (w = 1, \ldots, l)$ are elements of the alphabet S. PS_λ denotes the set of populations of size λ.

In the following, and without loss of generality, we will not use the t index.

Definition 4. *The global selection function, f_{sel}, selects randomly with reemplacement a collection $y \in PS_\lambda$ from a population $x \in PS_\lambda$:*

$$f_{sel} : (\alpha, x) \rightarrow y,$$

where α is a vector of dimension λ of randomly chosen values.

Definition 5. *The selection function is based on the rank of the objective function if the probability that I^i becomes a parent is proportional to the rank of its evaluation. This means that:*

$$p_{parent}(I^i) \propto r(rank(g(I^i)))$$

where r is a decreasing function of the rank of $g(I^i)$, and rank $(g(I^i)) = d \Leftrightarrow \exists d - 1$ individuals with better evaluation function than I^i.

As an example of this kind of selection function, we have:

$$r\left(rank\left(I^i\right)\right) = \frac{\lambda + 1 - rank\left(g\left(I^i\right)\right)}{\lambda(\lambda+1)/2}$$

DEFINITION 6. *The global production function, f_{prod}, produces offspring $z \in PS_\lambda$ from selected individuals $y \in PS_\lambda$ using a crossover operator:*

$$f_{prod}: (\beta, y) \rightarrow z,$$

where β is a vector of dimension λ of randomly chosen integer values from 1 to λ.

DEFINITION 7. *The production function is one point crossover if the parents $I^i = (s_1, ..., s_l)$ and $I^j = (b_1, ..., b_l)$ produce children $CH^{i,j,1} = (c_1, ..., c_l)$ and $CH^{i,j,2} = (d_1, ..., d_l)$ verifiying:*

$$c_j = \begin{cases} s_j & \text{if } j \le m \\ b_j & \text{if } j > m \end{cases}$$

$$d_j = \begin{cases} b_j & \text{if } j \le m \\ s_j & \text{if } j > m \end{cases}$$

where m is taken from the uniform distribution defined on the interval $[1, l]$.

DEFINITION 8. *The individual mutation function, f_{ind_mut}, applied to individual $I = (s_1, ..., s_l)$, generates another individual $MI = (sm_1, ..., sm_l)$, that is $f_{ind\_mut}(I) = MI$, such that $\forall j \in \{1, ..., l\}$, $P(sm_j = s_j) = 1 - p_m$, where p_m is the mutation probability.*

DEFINITION 9. *The extension function, f_{ext}, creates from two populations $x, z \in PS_\lambda$, a population $n \in PS_{2_\lambda}$:*

$$f_{ext}: (x, z) \rightarrow n.$$

Denoting by N_i with $i = 1, ..., 2\lambda$ the ith individual in n, and by X_k, with $k = 1, ..., \lambda$ the kth individual in x, and by Z_j with $j = 1, ..., \lambda$ the jth individual in z, we have:

$$N_i = \begin{cases} X_i & \text{if } i \le \lambda \\ Z_{i-\lambda} & \text{if } i > \lambda \end{cases}$$

DEFINITION 10. *The global reduction function, f_{red}, converts a population $n \in PS_{2_\lambda}$ to a population $r \in PS_\lambda$*

$$f_{red}: n \rightarrow r.$$

Notice that r denotes the population of individuals at time $t + 1$.

DEFINITION 11. *The reduction function is elitist of degree λ if the population at time t+1 is formed by selecting the best λ individuals—taking into account the objective function—among the λ individuals of the population at time t and the offspring derived from them.*

DEFINITION 12. *The reduction function is simple if the population at time $t + 1$ is formed by the offspring derived from the population at time t. Using the notation introduced in definitions 6, 9, and 10, the reduction function will be simple if and only if $r = z$.*

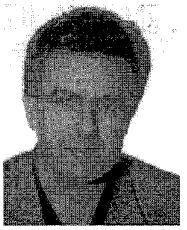
ACKNOWLEDGMENTS

We wish to thank Gregory F. Cooper for providing his simulation of the ALARM Network. We also thank the referees for their work and comments. This work was supported by the Diputación Foral de Gipuzkoa, under grant OF 95/1127, and by grant PI94/78 of the Gobierno Vasco.

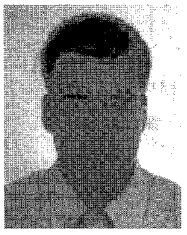
REFERENCES

[1] J. Pearl, *Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference*. San Mateo, Calif.: Morgan Kaufmann, 1988.
[2] R.E. Neapolitan, *Probabilistic Reasoning in Expert Systems. Theory and Algorithms*. John Wiley & Sons, 1990.
[3] F.V. Jensen, "Introduction to Bayesian Networks," Technical Report IR 93-2003, Dept. of Mathematics and Computer Science, Univ. of Aalborg, Denmark, 1993.
[4] C.K. Chow and C.N. Liu, "Approximating Discrete Probability Distributions with Dependence Trees," *IEEE Trans. Information Theory*, vol. 14, pp. 462-467, 1968.
[5] G. Rebane and J. Pearl, "The Recovery of Causal Polytrees from Statistical Data," *Uncertainty in Artificial Intelligence*, vol. 3, pp. 175-182, 1989.
[6] S. Acid, L.M. De Campos, A. Gonzalez, R. Molina, and N. Perez de la Blanca, "Learning with CASTLE," *Symbolic and Quantitative Approaches to Uncertainty*, R. Kruse and P. Siegel, eds., *Lecture Notes in Computer Science 548*. Springer-Verlag, 1991.
[7] R.M. Fung and S.L. Crawford, "CONSTRUCTOR: A System for the Induction of Probabilistic Models," *Proc. AAAI*, pp. 762-769, 1990.
[8] E. Herskovits and G. Cooper, "Kutató: An Entropy-Driven System for Construction of Probabilistic Expert Systems from Databases," Report KSL-90-22, Knowledge Systems Laboratory, Medical Computer Science, Stanford Univ., 1990.
[9] G.F. Cooper and E.A. Herskovits, "A Bayesian Method for the Induction of Probabilistic Networks from Data," *Machine Learning*, vol. 9, no. 4, pp. 309-347, 1992.
[10] R.R. Bouckaert, "Probabilistic Network Construction Using the Minimum Description Length Principle," *Lectures Notes in Computer Science 747, Symbolic and Quantitative Approaches to Reasoning and Uncertainty, ECSQARU '93*, pp. 41-48, 1993.
[11] D. Wedelin, "Efficient Algorithms for Probabilistic Inference Combinatorial Optimization and the Discovery of Causal Structure from Data," doctoral dissertation, Chalmers Univ. of Technology, Göteborg, 1993.
[12] S.L. Lauritzen, B. Thiesson, and D.J. Spiegelhalter, "Diagnostic Systems Created by Model Selection Methods—A Case Study," *Proc. Fourth Int'l Workshop Artificial Intelligence and Statistics*, pp. 93-105, 1993.
[13] C.F. Aliferis and G.F. Cooper, "An Evaluation of an Algorithm for Inductive Learning of Bayesian Belief Networks Using Simulated Data Sets," *Uncertainty in Artificial Intelligence, Proc. 10th Conf.*, pp. 8-14, 1994.
[14] R.R. Bouckaert, "Optimizing Causal Orderings for Generating DAGs from Data," *Uncertainty in Artificial Intelligence, Proc. Eighth Conf.*, pp. 9-16, 1992.
[15] R.R. Bouckaert, "Properties of Bayesian Belief Networks Learning Algorithms," *Proc. Uncertainty in Artificial Intelligence, 10th Ann. Conf.*, pp. 102-109, Washington, D.C., 1994.
[16] D.M. Chickering, D. Geiger, and D. Heckerman, "Learning Bayesian Networks: Search Methods and Experimental Results," *Fifth Int'l Workshop Artificial Intelligence and Statistics*, pp. 112-128, 1995.
[17] D. Heckerman, D. Geiger, and D.M. Chickering, "Learning Bayesian Networks: The Combination of Knowledge and Statistical data," Technical Report MSR-TR-94-09, Microsoft, 1994.
[18] W. Lam and F. Bacchus, "Learning Bayesian Belief Networks. An Approach Based on the MDL Principle," *Computational Intelligence*, vol. 10, no. 4, 1994.
[19] W. Lam and F. Bacchus, "Using Causal Information and Local Measures to Learn Bayesian Networks," *Uncertainty in Artificial Intelligence, Proc. Ninth Conf.*, pp. 243-250, 1993.
[20] D. Madigan, A.E. Raftery, J.C. York, J.M. Bradshaw, and R.G. Almond, "Strategies for Graphical Model Selection," *Proc. Fourth Int'l Workshop Artificial Intelligence and Statistics*, pp. 331-336, 1993.
[21] R. Mechling and M. Valtorta, "PaCCIN: A Parallel Constructor of Markov Networks," *Proc. Fourth Int'l Workshop Artificial Intelligence and Statistics*, pp. 405-410, 1993.
[22] G.M. Provan and M. Singh, "Learning Bayesian Networks Using Feature Selection," *Preliminary Papers Fifth Int'l Workshop Artificial Intelligence and Statistics*, pp. 450-456, Florida, 1995.
[23] G.M. Provan, "Model Selection for Diagnosis and Treatment Using Temporal Influence Diagrams," *Proc. Fourth Int'l Workshop Artificial Intelligence and Statistics*, pp. 469-480, 1995.
[24] M. Singh and M. Valtorta, "An Algorithm for the Construction of Bayesian Network Structures from Data," *Uncertainty in Artificial Intelligence, Proc. Ninth Conf.*, pp. 259-265, 1993.

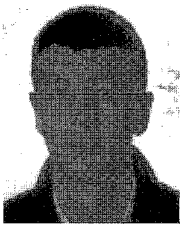
[25] J. Suzuki, "A Construction of Bayesian Networks from Databases Based on an MDL Principle," *Uncertainty in Artificial Intelligence, Proc. Ninth Conf.*, pp. 266-273, 1993.
[26] P. Larrañaga and Y. Yurramendi, "Structure Learning Approaches in Causal Probabilistics Networks," *Symbolic and Quantitative Approaches to Reasoning and Uncertainty*, M. Clarke, R. Kruse, and S. Moral, eds., *Lecture Notes in Computer Science 747*, pp. 227-232. Springer-Verlag, 1993.
[27] H.J. Bremermann, M. Rogson, and S. Salaff, "Search by Evolution," *Biophysics and Cybernetic Systems*, M. Maxfield, A. Callahan, and L.J. Fogel, eds. , pp. 157-167. Washington, D.C.: Spartan Books, 1965.
[28] I. Rechenberg, *Optimierung technischer Systeme nach Prinzipien der biologischen Information*. Stuttgart: Frommann Verlag, 1973 (in German).
[29] J.H. Holland, *Adaptation in Natural and Artificial Systems*. Ann Arbor, Mich.: Univ. of Michigan Press, 1975.
[30] D.E. Goldberg, *Genetic Algorithms in Search, Optimization and Machine Learning*. Reading, Mass.: Addison-Wesley, 1989.
[31] *Handbook of Genetic Algorithms*, L. Davis, ed. New York: Van Nostrand Reinhold, 1991.
[32] L.J. Fogel, "Autonomous Automata," *Ind. Res.*, vol. 4, pp. 14-19, 1962.
[33] H.-P. Schwefel, *Numerische Optimierung von Computer-Modellen mittels der Evolutionsstrategie*. Basel: Birkhäuser, 1977 (in German).
[34] A.E. Eiben, E.H.L. Aarts, and K.M. van Hee, "Global Convergence of Genetic Algorithms: An Infinite Markov Chain Analysis," *Computing Science Notes*, Eindhoven Univ. of Technology, 1990.
[35] C. Janikov and Z. Michalewicz, "On the Convergence Problem in Genetic Algorithms," UNCC technical report, 1990.
[36] R.F. Hartl, *A Global Convergence Proof for a Class of Genetic Algorithms*. Univ. of Technology, Vienna, 1990.
[37] U.K. Chakraborty and D.G. Dastidar, "Using Reliability Analysis to Estimate the Number of Generations to Convergence in Genetic Algorithms," *Information Processing Letters*, vol. 46, no. 4, pp. 199-209, 1993.
[38] G. Rudolph, "Convergence Analysis of Canonical Genetic Algoritms," *IEEE Trans. Neural Networks*, vol. 5, no. 1, pp. 96-101, 1994.
[39] P. Larrañaga and M. Poza, "Structure Learning of Bayesian Networks by Genetics Algorithms," *Studies in Classification, Data Analysis, and Knowledge Organization*, E. Diday, ed., pp. 300-306. Springer-Verlag, 1994.
[40] P. Larrañaga, R.H. Murga, M. Poza, and C.M.H. Kuijpers, "Structure Learning of Bayesian Networks by Hybrid Genetic Algorithms," *Preliminary Papers Fifth Int'l Workshop Artificial Intelligence and Statistics*, pp. 310-316, 1995.
[41] R.W. Robinson, "Counting Unlabeled Acyclic Digraphs," *Lectures Notes in Mathematics 622: Combinatorial Mathematics V*, C.H.C. Little, ed. , pp. 28-43. New York: Springer-Verlag, 1977.
[42] K.Höffgen, "Learning and Robust Learning of Product Distributions," Technical Report 464, Fachbereich Informatik, Universität Dortmund, 1993.
[43] S.L. Lauritzen and D.J. Spiegelhalter, "Local Computations with Probabilities on Graphical Structures and Their Application on Expert Systems," *J. Royal Statistical Soc. B*, vol. 50, no. 2, pp. 157-224, 1988.
[44] M. Henrion, "Propagating Uncertainty in Bayesian Networks by Probabilistic Logic Sampling," *Uncertainty in Artificial Intelligence*, vol. 2, pp. 149-163, 1988.
[45] I.A. Beinlinch, H.J. Suermondt, R.M. Chavez, and G.F. Cooper, "The ALARM Monitoring System: A Case Study with Two Probabilistic Inference Techniques for Belief Networks," *Proc. Second European Conf. Artificial Intelligence in Medicine*, pp. 247-256, 1989.
[46] E.H. Herskovits, "Computer Based Probabilistic-Network Construction," doctoral dissertation, Medical Information Sciences, Stanford Univ., 1991.
[47] *SPSS-X User's Guide*. third edition, 1988.
[48] I. Matzkevich and B. Abramson, "The Topological Fusion of Bayes Nets," *Proc. Eighth Conf. Uncertainty in Artificial Intelligence*, pp. 191-198, 1992.
[49] U. Kjærulff, "A Computational Scheme for Reasoning in Dynamic Probabilistic Networks," *Proc. Eighth Conf. Uncertainty in Artificial Intelligence*, pp. 121-129, 1992.



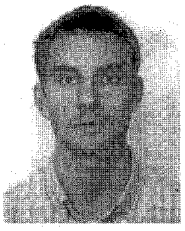
Pedro Larrañaga received the MSc degree in mathematics (mentions: statistics) from the University of Valladolid, Spain, in 1981, and the PhD degree in computer science from the University of the Basque Country in 1995. Since 1985, he has been a lecturer in statistics and artificial intelligence in the Department of Computer Science and Artificial Intelligence at the University of the Basque Country. His current research interests are in the fields of Bayesian networks, combinatorial optimization and data analysis with applications to medicine, molecular biology, cryptoanalysis, and finance.



Mikel Poza received the MSc degree in computer science from the University of the Basque Country in 1992. In 1993, he joined the Department of Computer Science and Artificial Intelligence of the University of the Basque Country, where he is currently working in data analysis.



Yosu Yurramendi received the MSc degree in mathematics (mentions: statistics and operations research) from Universidad Complutense of Madrid, Spain, in 1977, and the PhD degree in statistics (mention: data analysis) from Université Pierre et Marie Curie (Paris VI) in 1984. Since 1979, he has been a lecturer in statistics and artificial intelligence in the Computer Science and Artificial Intelligence Department at the University of the Basque Country. His research interests include data analysis and machine learning.



Roberto H. Murga received the MSc degree in computer science from the University of the Basque Country in 1994. In 1995, he collaborated with the Department of Computer Science and Artificial Intelligence of the University of the Basque Country. He is currently working at Telefonica, a firm in the telecommunications arena.

Cindy M.H. Kuijpers received the MSc degree in applied mathematics from the University of Technology of Eindhoven, the Netherlands, in 1993. She carried out her final project, which was within the field of combinatorial optimization, at the Philips Research Laboratories in Eindhoven, the Netherlands. Her Master's thesis was awarded the VVS-prize 1994 of the Netherlands Society for Statistics and Operations Research (VVS). In 1994, she joined the Department of Computer Science and Artificial Intelligence at the University of the Basque Country, where she is currently working on the use of genetic algorithms for tackling combinatorial optimization problems related to Bayesian networks.