Partial Abductive Inference in Bayesian Belief Networks—An Evolutionary Computation Approach by Using Problem-Specific Genetic Operators

Luis M. de Campos, José A. Gámez, and Serafín Moral

***Abstract*—Abductive inference in Bayesian belief networks (BBNs) is intended as the process of generating the K most probable configurations given observed evidence. When we are interested only in a subset of the network's variables, this problem is called partial abductive inference. Both problems are NP-hard and so exact computation is not always possible. In this paper, a genetic algorithm is used to perform partial abductive inference in BBNs. The main contribution is the introduction of new genetic operators designed specifically for this problem. By using these genetic operators, we try to take advantage of the calculations previously carried out, when a new individual is evaluated. The algorithm is tested using a widely used Bayesian network and a randomly generated one and then compared with a previous genetic algorithm based on classical genetic operators. From the experimental results, we conclude that the new genetic operators preserve the accuracy of the previous algorithm, and also reduce the number of operations performed during the evaluation of individuals. The performance of the genetic algorithm is, thus, improved.**

***Index Terms*—Abductive inference, bayesian belief networks, evolutionary computation, genetic operators, most probable explanation, probabilistic reasoning.**

I. Introduction

PROBABILISTIC methods were discarded for some time as a tool for dealing with uncertain reasoning because they required too complex a specification and computation. Nevertheless, with the appearance of probabilistic network models (mainly Bayesian and Markov networks [1], [2]), probability has enjoyed a spectacular revival, being nowadays one of the most accepted and used measures of uncertainty.

Bayesian belief networks (BBNs) are used frequently as the kernel of a probabilistic expert system because they provide an efficient representation of the joint probability distribution and allow calculation of probabilities by means of local computation, i.e., probabilistic computations are carried out over the initial pieces of information instead of using a global distribution.

In this paper, we are interested in a particular type of inference, known as abductive reasoning or diagnostic reasoning. In fact, it is in the field of diagnosis where abductive reasoning has its clearest application [3]–[5], although other applications exist in natural language understanding [6],[7], vision [8], legal reasoning [9], plan recognition [10],[11], planning [12], and learning [13].

Abduction [14] is defined as the process of generating a plausible explanation for a given set of observations or facts. In the context of probabilistic reasoning, abductive inference corresponds to finding the maximum *a posteriori* probability state of the system's variables, given some evidence (observed variables). It is well known that abductive reasoning in BBNs is a NP-hard problem [15] and this fact has motivated the development of approximate algorithms. As abductive inference in BBNs can be viewed as a combinatorial optimization problem, the use of optimization meta heuristics arises as a good choice to solve it in an approximate way.

In [16], an evolutionary computation approach was employed to tackle the problem of partial abductive inference in BBNs. Concretely, the kind of evolutionary algorithms employed was genetic algorithm (GA). GAs [17] are population-based algorithms inspired by the mechanics of natural selection and natural genetics, i.e., survival of the fittest, and have been applied widely to many difficult optimization problems. In this paper, we focus on the design of specific crossover and mutation operators for the problem of partial abductive inference, with the aim of improving the performance of the GA presented in [16]. To do so, we establish a double goal.

1) As the majority of the computational effort in our problem is devoted to evaluating individuals (probabilistic propagation), we aim to reduce the number of calculations carried out when an individual is evaluated by reusing some of the calculations carried out during the evaluation of previous similar individuals.
2) The GA with new operators should have a degree of accuracy similar to that obtained when the classical operators are used. After all, more important than getting a quick answer is obtaining a good answer.

The rest of the paper is organized as follows. In Section II, we revise the problem of abductive inference in BBNs. (Some introductory material about BBN's and inference in BBN's is given in Appendix I, while knowledge about GAs is assumed.) In Section III, we briefly review the previous work related to GAs and abductive inference in BBN's, providing a more detailed study of the algorithm presented in [16]. In Section IV, we describe the new genetic operators. Section V describes the experimentation carried out in order to test the behavior of the new operators, while the results are shown in Appendix II. Finally, Section VI contains the concluding remarks.

Manuscript received December 10, 1999; revised January 3, 2001 and May 29, 2001. This work was supported by the Spanish Comisión Interministerial de Ciencia y Tecnología (CICYT) under Project TIC97–1135-CO4–01.

L. M. de Campos and S. Moral are with the Departamento de Ciencias de la Computación e I.A., Universidad de Granada, 18071 Granada, Spain.

J. A. Gámez is with the Departamento de Informática, Universidad de Castilla-La Mancha, 02071 Albacete, Spain.

Publisher Item Identifier S 1089-778X(02)02974-0.

II. Preliminaries

In this section, we briefly revise the problem of abductive inference in Bayesian networks. To follow this section, some knowledge about propagation in Bayesian networks is assumed (see Appendix I for details and notation).

A. Abductive Inference

Abductive inference in BBNs, also known as belief revision [18] or the most probable explanation (MPE) problem [19] is defined as the problem of finding the MPE of observed evidence. In the context of BBNs, an explanation for a set of observations $X_O = x_O$ is a configuration of states for the network variables $x_{\mathcal{U}}$ such that $x_{\mathcal{U}}$ is consistent with x_O, i.e., $x_{\mathcal{U}}^{\downarrow X_O} = x_O$. In fact, the explanation is $x_{\mathcal{U}}^{\downarrow X_U \setminus X_O}$ because the values taken by the variables in X_O are previously known. Given the large number of possible explanations and since we are interested in the best explanation, our goal will be to obtain the MPE.

Thus, abductive inference in BBNs [1] corresponds to finding the maximum a posteriori probability state of the network, given the observed variables (the evidence). In a more formal way, if X_O is the set of observed variables and X_U is the set of unobserved variables, we aim to obtain the configuration x_U^* of X_U such that

$$x_U^* = \arg\max_{x_U} P(x_U|x_O) \tag{1}$$

where $X_O = x_O$ is the observed evidence. Usually, x_U^* is known as the MPE.

Dawid [20] has shown that the MPE can be found using probability propagation methods, but replacing summation by maximum in the marginalization operator (due to the distributive property of maximum with respect to multiplication). Therefore, the process of searching for the MPE has the same complexity as probabilities propagation. However, in general we are interested not only in the MPE, but in the K MPEs. For example, in a diagnostic problem, we could probably be more confident in the diagnosis knowing the set of most probable diagnoses because we could pay attention to the similarities and differences of the states taken by the variables in the set of top explanations.

Nilsson [21] has shown that only the *upward* phase of the propagation algorithm is necessary in order to perform abductive inference over a join tree. However, he has also proved that by using Dawid's algorithm, only the three most probable configurations can be identified directly, but in general the fourth cannot be found directly. So, in order to obtain the K MPEs ($K > 3$), more complex methods have to be used [22],[23].

B. Partial Abductive Inference

Sometimes we are interested in obtaining the K MPEs only for a subset of the network's variables called the explanation set [24]. This problem is known as partial abductive inference and we think that, in practical applications, this is more useful than the classical abductive inference problem. In fact, in system diagnosis, we can select as the explanation set those variables representing diseases in a medical diagnosis problem, the variables representing critical components (starter, battery, alternator) in a car diagnosis problem, etc.

Now, if we denote the explanation set by $X_E \subset X_U$, then we aim to obtain the configuration x_E^* of X_E such that

$$x_E^* = \arg\max_{x_E} P(x_E|x_O) = \arg\max_{x_E} \sum_{x_R} P(x_E, x_R|x_O) \tag{2}$$

where $X_R = X_U \setminus X_E$. In general, x_E^* is not equal to the projection of the configuration x_U^* over X_E, so we need to obtain x_E^* directly (2).

The process of finding the MPE x_E^* is more complex than that of finding x_U^* because not all join trees obtained from the original BBN are valid. In fact, because summation and maximum have to be used simultaneously and these operations do not show a commutative behavior, the variables of X_E must form a subtree of the complete join tree. The construction of the join tree is based on the triangulation of an undirected graph. In partial abductive inference [25], in order to obtain a valid join tree, instead of searching for arbitrary deletion sequences, we can only consider sequences in which the variables in X_R come before the variables in X_E. In [25], it is shown that the size of the obtained join tree grows significantly[1] in relation to the size of the join tree obtained without restrictions and so the propagation algorithm for partial abductive inference will be less efficient than propagation algorithms for (total) abductive inference.

III. Abductive Inference and Genetic Algorithms—Previous Works

GAs have been previously used to address NP-hard problems related to BBNs, such as triangulation of graphs [26], imprecise probabilities propagation [27], estimation of a causal ordering for the variables [28],[29], and learning [30]. Given the success of these applications, the NP-hardness of the abductive inference problem and the fact that abductive inference in BBNs can be defined as a combinatorial optimization problem, several authors have used GAs to approximate a solution (Rojas-Guzman and Kramer [31],[32], Gelsema [33]). Below, we describe some relevant points of these algorithms.

In [31] and [32], a chromosome of the population is represented as a copy of the graph included in the BBN, but in which every variable has been instantiated to one of its possible states. This representation makes it possible to implement the crossover operator as the interchange of a subgraph with the center in the variable X_i, X_i being randomly selected for each crossover. In Gelsema's algorithm [33], a chromosome is a configuration of the unobserved variables, i.e., a string of integers. In this case, crossover is implemented as the classical one-point crossover. It is worth noting that Gelsema uses the *a priori* probabilities of the BBN and the observed evidence to generate the initial population, so that the search starts in promising regions of the search space.

The algorithms presented [31]–[33] have in common the use of the same procedure in order to calculate the fitness of an indi-

[1]As an example, consider a BBN with seven variables $\{X_1, \ldots, X_6, Y\}$, such that there is a link $Y \rightarrow X_i$ for each variable X_i. If all the variables can take ten different states, then the size of the *optimum* join tree obtained in order to apply probabilities propagation or (total) abductive inference is 600, while the size of the join tree obtained for partial abductive inference taking $X_E = \{X_1, \ldots, X_6\}$ is 10^7.

vidual. As $P(x_U|x_O)$ is proportional to $P(x_U, x_O)$, this value can be used as the *fitness* for x_U. The calculation of this value can be easily carried out because the state of all the variables is known ($X_U \cup X_R = X_{\mathcal{U}}$) and so the chain rule (9) can be applied. Therefore, to evaluate a chromosome x_U, it is necessary to perform $|X_{\mathcal{U}}|$ multiplications.

Although the previous algorithms are designed to deal with the problem of total abductive inference and not with the partial one, in [33] the task of approximating the best set of (partial) explanations is attempted by integrating the appropriate members occurring in the final population of the GA. As Gelsema [33] points out, this problem is not a trivial matter, and in fact Gelsema's method only finds the best explanation in a small percentage of the runs, not being ranked as the best in most of them [16].

De Campos *et al.* [16] have applied GAs to the problem of partial abductive inference, but approaching the problem directly. The rest of this section is devoted to the review of this algorithm, as it is the basis of the present paper.

1) Representation of the Population: A chromosome or individual of the population is a configuration of states for the variables in the explanation set X_E, i.e., a string of integers of length $|X_E|$. Notice that in this case it is not useful to represent a chromosome as a graph because we are working only with a subset of the variables in the graph and its associated subgraph will usually be a set of disconnected graphs.

2) Evaluation Function: In partial abductive inference, $P(x_E, x_O)$ can be used as the fitness for a chromosome x_E. However, as

$$P(x_E, x_O) = \sum_{x_R \in \Omega_{X_R}} P(x_E, x_R, x_O)$$

it is necessary to use the chain rule $|\Omega_{X_R}|$ times to evaluate a chromosome. For example, if we have a network with $|X_{\mathcal{U}}| = 50$ bivalued variables $|X_E| = 15$, $|X_R| = 30$, and $|X_O| = 5$, then the number of operations to evaluate a chromosome x_E is bounded by $50 \cdot 2^{30}$ multiplications and 2^{30} summations. Clearly, this is computationally intractable given the large number of individuals to be evaluated in the execution of a GA. Because of this, de Campos *et al.* [16] propose to evaluate a chromosome by means of a probabilistic propagation.

As we can see in Algorithm 1 (Appendix I-B), the probability $P(x_O)$ can be calculated by summing in the clique root after the upward phase. When we are going to evaluate a chromosome, the state taken by the variables in the explanation set is known, so we can treat those variables as evidence and calculate $P(x_E, x_O)$ by using the first three steps of Algorithm 1. Algorithm 2 shows the pseudocode of the evaluation function used in [16].

Algorithm 2: Evaluation Function
Input: The join tree $\mathcal{T} = \{C_1, \ldots, C_t\}$ with the evidence $X_O = x_O$ previously instantiated.
The configuration x_E to be evaluated.
Output: $P(x_E, x_O)$.

1. Incorporate $X_E = x_E$ to $\mathcal{T}$ as evidence.
2. for $i \leftarrow t$ downto 2 do
 - if $\mathrm{ch}(C_i) \neq \emptyset$ then
 $M^{i-\mathrm{fa}(i)} \leftarrow \sum_{C_i \setminus S_{i,\mathrm{fa}(i)}} \left\{ \psi(C_i) \otimes \left(\bigotimes_{C_k \in \mathrm{ch}(C_i)} M^{k-i} \right) \right\}$
 else $M^{i-\mathrm{fa}(i)} \leftarrow \sum_{C_i \setminus S_{i,\mathrm{fa}(i)}} \psi(C_i)$
 - Send $M^{i-\mathrm{fa}(i)}$
3. $\psi'(C_1) \leftarrow \psi(C_1) \otimes \left(\bigotimes_{C_k \in \mathrm{ch}(C_1)} M^{k-1} \right)$
4. $P(x_E, x_O) \leftarrow \sum_{C_1} \psi'(c_1)$.

Therefore, to evaluate a chromosome, only the first phase of probabilities propagation needs to be carried out. Furthermore, given the type of inference to be performed (probabilities propagation), the join tree over which the propagation will be carried out is obtained without any constraint and so its size is significantly smaller than the (mostly prohibitive) size of the join tree used for exact partial abduction. Moreover, in [16], the authors propose three operations in order to improve the efficiency of the evaluation function: 1) the join tree is precomputed (pruned) for each explanation set; 2) the way in which the marginalization (summation) is peformed is modified in order to avoid having to instantiate the chromosome in the join tree and so it is not necessary to reload the initial potentials when a new chromosome has to be evaluated; and 3) a hash table is used to store the fitness of the chromosomes previously evaluated, making it unnecessary to repeat the propagation. Although in this paper we also take advantage of these improvements, for the sake of simplicity, we focus our discussion on the evaluation function as it appears in Algorithm 2. For more details, see [16].

3) Generation of the Initial Population: Half of the initial population is generated randomly (the search starts with points in all the search space) and the other half is generated by simulation, using a procedure inspired in the idea of Gelsema and is based on Henrion's probabilistic logic sampling [34] (the search starts with points in promising regions).

4) Transition From One Population $P(t)$ to the Next $P(t+1)$: To obtain a new generation, a procedure similar to the modified GA (modGA) proposed by Michalewicz [35] is used. This GA falls into the category of preservative, generational, and elitist selection and has similar theoretical properties as the classical GA. The main modification with respect to the classical GA is that in modGA the classical selection step is not performed, but rather r distinct chromosomes (usually those that fit best) are selected from $P(t)$ to be copied to $P(t+1)$.

Other models of GAs (classical and steady state) were considered at the beginning of our experimentation. Although all the models worked quite well, when the goal was to search for the best explanation, we decided to use modGA because its structure seemed to fit best the problem of searching for the K MPEs, perhaps due to the fact that modGA maintains a subpopulation containing the best r individuals found during the search.

In [16] the parameters used were those cited below. Although the majority of them have been maintained for this paper, we indicate those that have been modified.

1) Select the best 50% chromosomes from $P(t)$ and copy them to $P(t+1)$. In this way, the population diversity is ensured and the premature convergence problem is avoided.

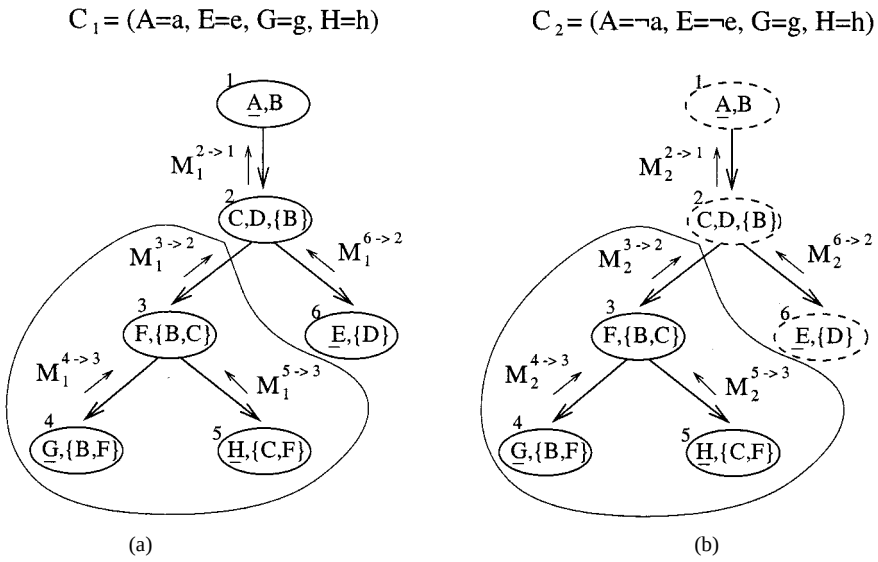

Fig. 1. (a) Messages sent during the evaluation of c_1. (b) Messages sent during the evaluation of c_2.

2) 35% of the new population is obtained by crossover. The crossover operator used is the classical two-point crossover and the two children obtained are copied to $P(t+1)$. In the original work [16], the chromosomes are selected with a probability proportional to their fitness, but in this paper, we will use a probability of selection based on the position in which individuals are *ranked* according to their fitness. Our experimentation indicates that this type of selection improves the behavior of our algorithm, especially with respect to the variability of the outcomes.
3) 15% of the new population is obtained by mutation. Mutation is carried out by selecting a chromosome from $P(t)$ and modifying one of its components, the resulting chromosome is copied to $P(t+1)$. We, thus, apply genetic operators on whole individuals as opposed to individual bits (classical mutation). As Michalewicz [35] points out, this would provide an uniform treatment of all operators used in the GA. The *parents* for mutation are selected from $P(t)$ with a probability based on their rank, except for the *best* chromosome, which is always selected as a parent (thus, the area in the proximity of the best chromosome is explored). In the original work, parents for mutation were selected randomly, but we have changed this for the same reasons as in the crossover.

Notice that in $P(t+1)$, only half of the population is new and so only those chromosomes are candidates to be evaluated in each generation. This fact is important in our problem because of the evaluation function complexity. When a new chromosome is evaluated, it is tested to see whether it should be included in $K\text{best}$, an array which contains the K best individuals obtained from the beginning up to the current generation. The numbers 50, 35, and 15 were selected by experimentation in [16], although the behavior of the algorithm seems not to be too sensitive to small variations around these numbers. However, a drastic reduction in the number of individuals to be mutated could degrade (in general) the behavior of the algorithm, especially with respect to searching the K MPEs.

5) Stopping Criterion: The algorithm stops when a fixed number of iterations (ng) has been carried out. The probability of the chromosomes stored in $K\text{best}$ is then divided by $p(x_O)$ in order to obtain $p(x_E|x_O)$.

As in this paper, we are going to experiment with different population sizes and different ways of initializing the population, we have modified the stopping criterion in the following way: the algorithm stops when the probability mass of the explanations included in $K\text{best}$ does not improve in ten generations.

IV. Design of the New Genetic Operators

The main disadvantage of the GA presented in [16] [from now on called GA with classical operators (GACO)] is the need to perform a complete upward propagation each time a new chromosome has to be evaluated. In this section, we introduce new genetic operators to avoid this disadvantage. The new operators are based on the following idea: when a new chromosome, obtained by mutation or crossover has to be evaluated, we can take advantage of some of the calculations carried out during the evaluation of their parents' chromosomes. The modification of GACO to include the new genetic operators will be called GA with specific operators (GASO). Of course, the evaluation of a new chromosome in GASO will be faster than in GACO, although the amount of memory needed will be greater too because we have to store the relevant information created during the evaluation of a chromosome. Let us motivate the new operators with an example.

Example 1: Consider the join tree in Fig. 11 (see Appendix I) and the explanation set $X_E = \{A, E, G, H\}$. The messages required to evaluate the configurations (chromosomes) $c_1 = (A = a, E = e, G = g, H = h)$ and $c_2 = (A = \neg a, E = \neg e, G = g, H = h)$ are shown in Fig. 1(a) and (b), respectively. The variables inside brackets in each cluster are the separator set with its father and we will refer to the variables outside the brackets as the *residual set*. We have also underlined the variables of the explanation set in the residual sets (notice that a

variable can be included in the separator set of several clusters, but only in one of them as residual.[2])

As variables G and H take the same value in both configurations c_1 and c_2, it is clear that the messages $M^{4\to3}, M^{5\to3}$, and $M^{3\to2}$ will be the same in the evaluation of both configurations. Therefore, if we store the calculated messages for c_1 and we use that information for the evaluation of c_2, we only need to operate in clusters C_1, C_2, and C_6 [depicted in dashed line in Fig. 1(b)].

In fact, in the previous example, we have a subtree with the same evaluation for both configurations (remarked in Fig. 1). This gives us the following idea: we can associate a join tree to each chromosome of the population and implement the crossover and mutation between trees and not between chromosomes. In this way, we can avoid the calculations corresponding to the interchanged subtrees.

The relevant information generated when a chromosome is being evaluated is the set of messages sent among the clusters. This is good because the size required to store the messages is much lower[3] than the size required to store the clique potentials. Therefore, in GASO, a chromosome is represented by a string of integers (as in GACO) plus (a vector containing) the messages sent during its evaluation. As C_1 never computes its message, the vector `messages` will be defined from 2 to $|\mathcal{T}|$, so `messages(c)[i]` will represent the message sent by C_i during the evaluation of chromosome c.

Algorithm 3 shows the pseudocode of the modified evaluation function, in which the `messages` vector is considered.

```
Algorithm 3: Modified Evaluation Function
Input: The join tree T = {C_1,...,C_t} with the
 evidence X_O = x_O previously instantiated.
The chromosome c to be evaluated and its asso-
 ciated vector messages(c).
Output: the fitness for c, P(c, x_O).
Auxiliary variable: change
1. Incorporate X_E = c to T as evidence.
2. change ← false
3. for i ← t downto 2 do
 3.1 if messages(c)[i] = NIL then
  - if ch(C_i) ≠ ∅ then
    M^{i-fa(i)} ← Σ_{C_i \ S_{i,fa(i)}} { ψ(C_i) ⊗ ( ⊗_{C_k ∈ ch(C_i)} M^{k-i} ) }
  else M^{i-fa(i)} ← Σ_{C_i \ S_{i,fa(i)}} ψ(C_i)
  - messages(c)[i] ← M^{i-fa(i)}
  - Send M^{i-fa(i)}
  - change ← true
4. if (change = true) then
   ψ'(C_1) ← ψ(C_1) ⊗ ( ⊗_{C_k ∈ ch(C_1)} M^{k-1} )
5. P(x_E, x_O) ← Σ_{C_1} ψ'(c_1)
```

The structure of GASO will be the same as that described previously for GACO, except the necessary changes in the evaluation function in order to adapt it to the new genetic operators. Before introducing more specifically the mutation and crossover operators, it should be pointed out that the initial population is evaluated using the evaluation function presented in Algorithm 2 because, at the beginning, there is no information to reuse.

[2]The cluster in which the variable is summed out (marginalized).

[3]In our experiments, the messages take up about the 20%-25% of the size required to store the clique potentials.

A. Mutation

We shall denote by S_i the separator set of C_i with respect to its father and by R_i the residual set of C_i, i.e., $R_i = C_i \setminus S_i$. Let us define the following set:

$$C_R = \{C_i \in \mathcal{T} \mid R_i \cap X_E \neq \emptyset\}. \tag{3}$$

As the relevant information about a variable X_j is obtained when this variable is summed out (marginalized) and marginalization is carried out in the cluster which contains the variable X_j in its residual set, then C_R contains all the clusters that concern us with respect to the explanation set X_E. The proposed mutation operator is shown in Algorithm 4, where we have used $anc(C_i)$ as the set containing all the clusters in the path between $\mathrm{fa}(C_i)$ and the root C_1.

```
Algorithm 4: Mutation
Input: The chromosome c to be mutated and its
 associated vector messages(c).
Output: The mutated chromosome c' and its asso-
 ciated vector messages(c').
1. Copy c to c' and messages(c) to messages(c').
2. Select randomly a cluster C_i ∈ C_R.
3. Select randomly a variable X_j ∈ R_i ∩ X_E.
4. Mutate the variable X_j in c'.
5. for all C_k ∈ anc(C_i) ∪ {C_i} do
 messages(c')[k] ← NIL
```

Therefore, to evaluate the obtained chromosome c', we only have to carry out new calculations in the cluster containing the mutated variable and in all their ancestors in the join tree. The remaining clusters can reuse the messages calculated for c because they are not affected by the modified variable X_j. For example, if we consider the join tree depicted in Fig. 11 and the explanation set $X_E = \{A, E, G, H\}$, then $C_R = \{C_1, C_4, C_5, C_6\}$. Fig. 2(a) shows the messages sent during the evaluation of chromosome c_1. If the cluster C_4 is selected in Step 2 of Algorithm 4, then it is clear that variable G will be mutated and the state of the join tree to be used for evaluating chromosome c_1' is depicted in Fig. 2(b) (where empty messages are the messages to be calculated and dashed clusters represent the clusters where new computations have to be carried out).

Notice that in the previous example we have considered one of the worst cases because the selected cluster is a leaf in the join tree. For example, if in step two of Algorithm 4 we select cluster C_1 with the result that variable A is mutated, then all the previous messages are valid and the only new computation is performed in cluster C_1 (Step 5 of Algorithm 3).

B. Crossover

In Example 1, we have seen how the variables (more properly, the clusters of C_R that contain those variables) of the explanation set whose state was changed could be isolated in a subtree. This gives us the following idea: if we select a cluster

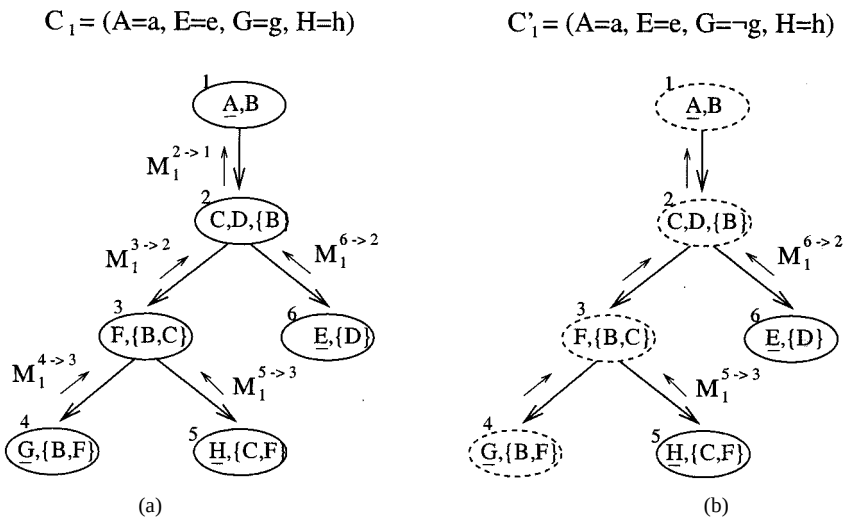

Fig. 2. (a) Messages sent during the evaluation of c_1. (b) State of the join tree after mutating variable G and before the evaluation of c'_1.

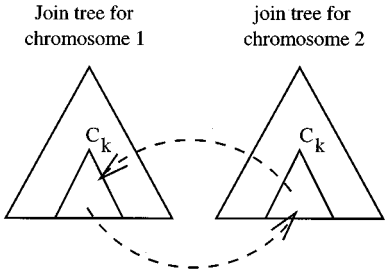

Fig. 3. Structure of the crossover operator.

C_k of $\mathcal{T}$ and we interchange between two chromosomes c_1 and c_2, the subtree whose root is $C_k(\mathcal{T}_k)$, then to evaluate the new chromosomes c'_1 and c'_2 (obtained by interchanging the state of the explanation set variables contained in the residual set of the clusters in $C_R \cap \mathcal{T}_k$), we need only to operate on the clusters belonging to $anc(G_k)$. Fig. 3 shows the structure of this crossover and Algorithm 5 shows the pseudocode of the crossover operator.

Algorithm 5: Crossover
Input: The parent chromosomes c_1 and c_2 and their associated vectors messages(c_1) and messages(c_2).
Output: The obtained chromosomes c'_1 and c'_2, and their associated vectors messages(c'_1) and messages(c'_2).
1. Copy c_1 to c'_1 and messages (c_1) to messages (c'_1).
2. Copy c_2 to c'_2 and messages(c_2) to messages(c'_2).
3. Select (randomly) a cluster $C_k \in \mathcal{T}$.
4. Interchange between c'_1 and c'_2, the state taken by the explanation set variables belonging to the residual set of the clusters contained in $\mathcal{T}_k \cap C_R$.
5. for all $C_i \in anc(C_K)$ do
-messages(c'_1)[i] $\leftarrow NIL$
-messages(c'_2)[i] $\leftarrow NIL$

Fig. 4 shows the messages sent during the evaluation of chromosomes $c_1 = (A = a, E = e, G = g, H = h)$ and $c_2 = (A = \neg a, E = \neg e, G = \neg g, H = \neg h)$. If we select cluster C_3 as the crossover point, then the state of variables G and H is interchanged between c_1 and c_2, obtaining the two children $c'_1 = (A = a, E = e, G = \neg g, H = \neg h)$ and $c'_2 = (A = \neg a, E = \neg e, G = g, H = h)$. Fig. 5 shows the state of the join tree before evaluating the new chromosomes and we can see that only one message and two clusters are involved in new computations, so the evaluation of the two children will be faster.

Remember that if a new chromosome c (obtained by mutation or crossover) has been evaluated previously, we do not recalculate it again, recovering its fitness from the hash table. Then, the vector messages(c) will contain some noncalculated messages. This is no problem for Algorithms 4 and 5, which are able to deal with empty of null (NIL) messages. The only consequence is that (in general) the number of new computations required to evaluate an offspring whose father had NIL messages will be greater. For example, in Fig. 6(a) we have the messages sent during the evaluation of c_1 (messages are represented as white-headed arrows) and in Fig. 6(b) we have the content of the vector messages for chromosome c_2 (messages are represented as blackheaded arrows and absence of arrows denotes NIL messages). If C_5 is selected as the crossover point, Fig. 6(c) and (d) shows the state of the join tree before evaluating the two children c'_1 and c'_2. It is clear that the state of both join trees is correct for the application of Algorithm 3, but as a consequence of the NIL messages in c_2, the number of new computations to be carried out in c'_2 is greater than those caused directly by taking C_5 as the crossover point.

1) Selecting the Crossover Point: Step 3 of Algorithm 5 (crossover) is

Select (randomly) a cluster $C_k \in \mathcal{T}$.

We may now ask the following two questions.

1) Are all the clusters in $\mathcal{T}$ good candidates to be selected as the crossover point?
2) Is random selection the best way to select the crossover point?

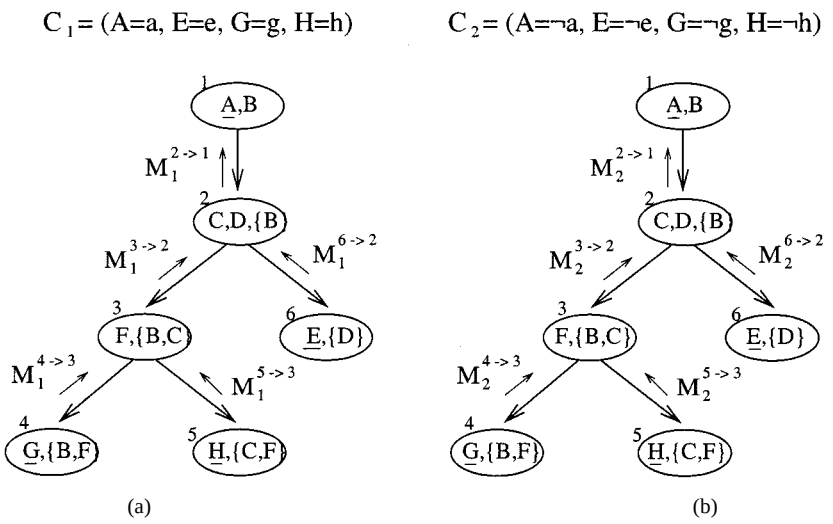

Fig. 4. Messages sent during the evaluation of (a) c_1 and (b) c_2.

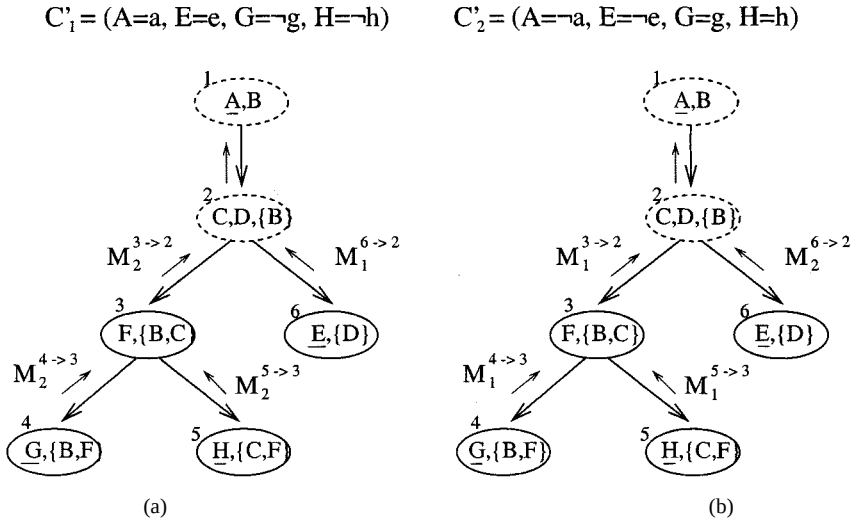

Fig. 5. State of the join tree after performing the crossover and before the evaluation of (a) c_1' and (b) c_2', when C_3 has been selected as crossover point.

With respect to the first question, it is clear that the root cannot be selected as the crossover point because, in that case, the entire chromosome is interchanged and so the operation does not constitute a crossover. However, the root is not the only cluster to be avoided as a crossover point. For example, in the join tree shown in Fig. 7, if we select C_7 as the crossover point, the only information to interchange is that corresponding to variable E. However, if we select C_6 as the crossover point, the interchanged information is exactly the same, but the subtree to interchange is greater and, therefore, the number of new computations to evaluate the children is smaller. From this example, we can conclude that it not all the clusters should be regarded as crossover points.

We shall now formalize the previous idea.

Definition 1: The amount of interchanged information when cluster C_k is selected as the crossover point (denoted as $I(C_k)$) is defined as the number of explanation set variables contained in the residual set of the clusters in $\mathcal{T}_k$. More formally

$$I(C_k) = \left| \left\{ X_i \mid X_i \in X_E \cap \left(\bigcup_{C_j \in \mathcal{T}_k} R_j \right) \right\} \right|. \quad (4)$$

Column two of Table I shows the amount of interchanged information when a cluster is selected as crossover point for the join tree in Fig. 7.

Definition 2: All the clusters in the join tree are *valid* as crossover points except those included in the following two categories:

1) the root of the join tree;
2) any cluster C_i such that $I(C_i) = I(\mathrm{fa}(C_i))$.

The fourth column of Table I shows the clusters that are considered as valid crossover points for the join tree in Fig. 7.

At this point, we have answered the first question, but what about the way a cluster is selected as the crossover point? In the rest of this discussion, we only consider the clusters that are valid crossover points. In Table I, we can see that the majority of the crossover points have a small value of $I(\cdot)$, so if all the crossover points have the same probability (random selection) of being selected as the crossover point, then the convergence of the GA could be slowed down. Therefore, in order to give priority to the crossover points with higher value of $I(\cdot)$, we can select a crossover point with probability proportional to $I(\cdot)$.

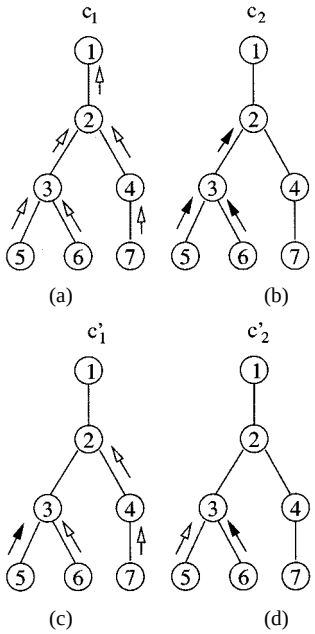

Fig. 6. Example of crossover with NIL messages. (a) Parent 1. (b) Parent 2. (c) Child 1. (d) Child 2.

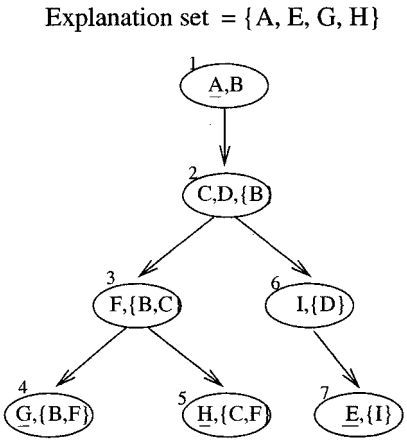

Fig. 7. Join tree.

However, to have a very high value of $I(\cdot)$ is not good either because, in that case, the children obtained after applying the crossover will also be very similar to their parents. In our example, if $I(C_k) = 3$ the children are equal to their parents except in the state of a variable (the same as for $I(C_k) = 1$). To correct this situation, we define $I'(\cdot)$ as

$$I'(C_k) = \min(I(C_k), |X_E| - I(C_k)). \tag{5}$$

Therefore, we will use $I'(\cdot)$ instead of $I(\cdot)$ to implement the proportional selection

$$p_{sel}(C_i) = \frac{I'(C_i)}{\sum_{C_j \text{ valid}} I'(C_j)}. \tag{6}$$

In order to smooth the differences among the probabilities of selection, we can apply logarithms

$$p_{sel}(C_i) = \frac{1 + log(I'(C_i))}{\sum_{C_j \text{ valid}} 1 + log(I'(C_j))}. \tag{7}$$

TABLE I
VALUE OF $I(C_k)$ FOR EACH CLUSTER C_K OF FIG. 7

				Probability of selection		
C_i	$I(C_i)$	$I'(C_i)$	Valid	random	prop.	1+log()
C_1	4	0	no			
C_2	3	1	yes	0.2	1/6	0.189
C_3	2	2	yes	0.2	2/6	0.244
C_4	1	1	yes	0.2	1/6	0.189
C_5	1	1	yes	0.2	1/6	0.189
C_6	1	1	yes	0.2	1/6	0.189
C_7	1	1	no			

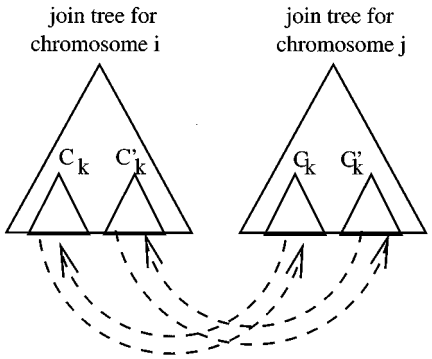

Fig. 8. Structure of the two-point crossover operator.

The third column of Table I shows the values of $I'(\cdot)$ and the last three columns of the table show the probability of selection for three different selection criteria: random, proportional, and log-proportional.

C. Two-Point Crossover

In order to increase the diversity of possible crossovers, we introduce the two-point crossover. This operator is a direct adaptation of the crossover defined in Section IV-B, but selecting two clusters as crossover points and interchanging the two subtrees. The structure of the two-point crossover is shown in Fig. 8.

If m denotes the number of valid crossover points in a join tree with respect to an explanation set X_E, then the number of different crossovers is:

1) m, with one crossover point;
2) $\leq m(m-1)/2$, with two crossover points.

In this way, we have increased the number of possible crossovers significantly; for example, if $m = 10$, we have passed from ten to an upper limit of 45. This number is an upper limit because some crossover points are descendants (in the join tree) of other crossover points and if this situation occurs during the crossover, we are acting in the same way as in the case of one-point crossover.

The GASO algorithm with the two-point crossover operator will be denoted as GASO2.

TABLE II
SOME CHARACTERISTICS OF THE NETWORKS USED IN THE EXPERIMENTS

Network	nodes	arcs	states	min	max	mean
Alarm	37	46	$\{2,3,4\}$	2	108	20.3
random100	100	128	2	2	64	6.54

TABLE III
DESCRIPTION OF THE EXPERIMENTS

#exp.	$\|X_E\|$	network	X_E	$\|\Omega_{X_E}\|$
1	18	Alarm	pseudo-random	143,327,232
2	19	Alarm	pseudo-random	214,990,848
3	20	Alarm	pseudo-random	382,205,952
4	30	random100	pseudo-random	1,073,741,824

V. EXPERIMENTAL EVALUATION

To evaluate our algorithms we have carried out four experiments, described in Sections V-A. In Section V-B, the performance measures used to compare the different algorithms are defined. Finally, in Section V-C, the experimental results are analyzed and some conclusions are formulated. Due to the great amount of generated data and to make possible a continuous reading of the paper, tables and figures are placed in Appendix II.

A. Description of the Experiments

Three experiments have been carried out over the well-known *Alarm* network[4] [38] and the other over an artificially generated Bayesian network: *random100*. The network *random100* has been generated by allowing a maximum of five parents for each variable and by using the following procedure[5] in order to generate the probabilities: two uniform random numbers x and y are generated and the probability of the two values (marginal for root nodes and conditional for the rest) of a variable are determined by normalizing x^5 and y^5, which gives rise to extreme probabilities.

Table II shows some information about these networks, where min, max, and mean make reference to the size of the probability table attached to each node and states makes reference to the number of possible values that each variable can take.

Table III shows a brief description of each experiment. The column $|X_E|$ informs us of the number of variables included in the explanation set, while the column X_E informs us of the way these variables were selected as the explanation set. In all the experiments, the variables to be included in the explanation set were selected in a pseudorandom way, i.e., several sets containing $|X_E|$ variables were randomly generated and the most difficult one to be solved by exact computation was chosen. The difficulty of a problem was measured as a function of the time and space needed to solve the problem exactly. To solve the problem exactly, we have used software implemented in Java and running on an Intel Pentium III (600 MHz) with 384 MB of RAM, Linux operating system, together with the JDK 1.2 virtual machine. The time needed to solve Experiments 1, 2, and 3 was between 1 and 1.5 h, while solving a total abductive inference problem using this software takes less than 0.5 s. For Experiment 4, we have not been able to solve the problem exactly because of memory requirements, i.e., the "out of memory error" was obtained as response. This error is due to the enormous size of the join tree obtained from the *random100* network by means of a compilation constrained by the selected explanation set. In these networks, total abductive inference requires less than 9 s. Notice that this fact does not imply that all the problems with these explanation set sizes are equally hard to solve, because the complexity of the problem depends on: 1) the selected explanation set and 2) the topology of the network. However, the cases considered here are examples of problems in which exact computation is not suitable.

In all the experiments, five variables have been selected as evidence, being instantiated to their *a priori* less probable state. In the four experiments, we have taken $K = 50$, i.e., we look for the 50 MPEs. Taking into account the value of K, two different population sizes have been considered: 100 ($2K$) and 200 ($4K$). Two ways of creating the initial population have been considered: random and $1/2$ random $+ 1/2$ heuristic (as in [16]). Unlike the stopping criterion considered in [16], i.e., a fixed number of generations, here the algorithms stop when $mass50'$ (see the next section) does not improve in ten generations.

The four experiments have been solved by the GACO algorithm and by six versions of the GASO algorithm (GASO1r, GASO1p, GASO1l, GAS2r, GASO2p, and GASO2l), where the number indicates if we are using one or two cliques as crossover points, and the letter denotes the way in which the crossover points are selected: random (r), proportional (p), or logarithmic (l).

B. Performance Measures

The data we have collected during the execution of the algorithms is related to the following.

1) The probability mass of the K MPEs found. Thus, $\mathrm{mass1}, \mathrm{mass10}, \mathrm{mass25}$, and $\mathrm{mass50}$ represent the probability mass of the first 1, 10, 25, and 50 MPEs found by the exact algorithm and $\mathrm{mass1}'$, $\mathrm{mass10}'$, $\mathrm{mass25}'$, and $\mathrm{mass50}'$ represent the probability mass of the first 1, 10, 25, and 50 MPEs found by the proposed algorithms. For Experiments 1, 2, and 3, we present the percentage of probability mass obtained with respect to the exact algorithm ($\%\mathrm{mass}X' = \mathrm{mass}X' * 100/\mathrm{mass}X$). For Experiment 4, because of the absence of exact results, we directly present $\mathrm{mass}X'$.
2) The number of generations performed by the GA.

[4]The *Alarm* Bayesian network constitutes a classical problem for the testing of several types of algorithms (learning, propagation, etc.) in the Uncertainty in Artificial Intelligence (UAI) community; as an example, we can cite two papers [36],[37] of the most recent conference on this topic (UAI'2000). This network has also been used to test previous abductive inference algorithms [32].

[5]This process of generating Bayesian networks was used by Cano *et al.* [39] in order to obtain very complex problems.

3) The number of different[6] evaluated individuals during the execution of the GA.
4) The number of additions and multiplications carried out during the evaluation of the individuals (propagation). Notice that these are the basic parameters to be considered as far as the goal of this paper is concerned because we try to improve the evaluation of individuals and the operations involved in the evaluation (probabilistic propagation) are combination (multiplication) and marginalization (addition).

In the tables, the best result in each major column (e.g, average for %mass1) is in boldface. All the algorithms have been run 50 times over each experiment and, therefore, the average (A) and the standard deviation (SD) are shown. Moreover, a statistical study[7] has been done in order to ascertain whether there are significant differences among the algorithms used. The *Tukey (Studentized) test* with a 0.05 level of significance has been used for the analysis of two samples: GACO versus GASO1r, GACO versus GASO1l, etc. When significant differences are found, they are shown in the average column of the tables (see Appendix II-A) by:

1) a plus sign ($+$) for significant difference in favor of GASO;
2) a minus sign ($-$) for significant difference in favor of GACO.

In fact, statistical tests are performed not only between the GASO algorithms and GACO, but also between every pair of GASO algorithms: GASO1r versus GASO1p, GASO1r versus GASO1, etc. The information produced by these tests is summarized with the help of the Student–Newman–Keuls (SNK) multiple range test (with 0.05 level of significance). This method works in the following way:

1) order the means $\mu_1 \leq \cdots \leq \mu_t$;
2) call compare$(1, t)$.

Compare(i, j): Compare μ_i and μ_j by using a critical value determined by the significance level of the test, the degrees of freedom from the analysis of variance, and the number of means in the range of means being tested. If the range is not significant, no further testing is done and the set of considered means is declared homogenous. Otherwise, call compare$(i, j-1)$ and compare$(i+1, j)$.

The method first ranks the means (averages) from smallest to largest and then looks for homogenous intervals of values. This is done by a recursive procedure. It tests the difference between the smallest and largest means of a sequence. If it is not significant, then all the sequence is considered homogenous. In other case, two new intervals are generated for test: one removing the smallest value of the sequence and the other by removing the largest value. A more detailed description can be found, e.g., in [40]. The output of this test is shown in a compact way by means of diagrams like the one displayed in Fig. 9. On it, the intervals with no significant differences are connected with horizontal lines. The meaning of this diagram can be interpreted as the following.

[6]Notice that due to the use of a hash table, no propagation is carried out when an individual is revisited.

[7]The BMDP program has been used to perform the statistical analysis of the experimental results.

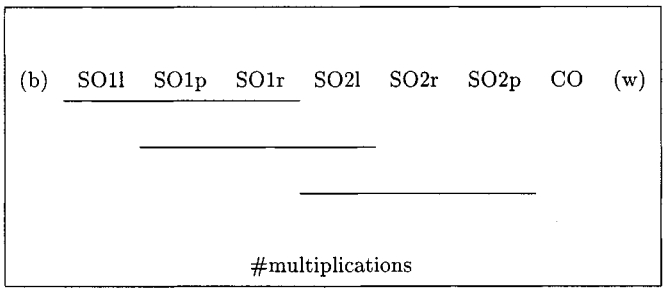

Fig. 9. Example of the output produced by the multiple test. Notice that in this kind of diagram, GASO has been abbreviated to SO and GACO to CO.

1) The algorithms are ordered from best average (b) (left) to worst average (w) (right).
2) Four groups are established {GASO1l, GASO1p, GASO1r}, {GASO1p, GASO1r, GASO21}, {GASO2l, GASO2r, GASO2p}, and {GACO}, indicating that neither of the average differences between the algorithms included in each group are significant.
3) Since no line connects the {GACO} group with any other group, it differs significantly from all the other algorithms.
4) Since no line connects GASO1l with the {GASO2l, GASO2r, GASO2p} group, these groups differ significantly from each other.
5) A similar conclusion to the previous one can be obtained for {GASO1l, GASO1p} with respect to {GASO2r, GASO2p}.

The obtained diagrams are shown in Appendix II-B. When there is no diagram for some of the studied parameters (i.e., mass1), it means that there is a line grouping all the algorithms, i.e., there is no significant difference between them.

In order to analyze the different options in relation with the population (size and initialization), statistical tests have been carried out among the four combinations: R1, H1, R1, H2, where "R" is random population, "H" is 1/2 heuristic population, "1" is population of size 100, and "2" is population of size 200. The analysis has been carried out from the results obtained by GASO1p and GASO2p. We have selected proportional selection of crossover points because it seems to be the medium case. The diagrams obtained are shown in Appendix II-B.

C. Analysis of the Experimental Results

From the experimental results obtained and with the help of the statistical analysis carried out, the two main conclusions obtained are the following.

1) The accuracy of the GA with the new proposed operators is similar to the accuracy of the GA in which the classical operators are used. In fact, there is no significant difference with respect to mass1 and mass10 in any of the experiments carried out. The statistical analysis has revealed that significant differences exist with respect to mass25 and mass50 in a few cases, but sometimes in favor of GACO (9) and sometimes in favor of GASO (15).

2) The way in which the individuals are evaluatd has been improved. In fact, in all the series,[8] significant differences in favor of GASO versus GACO have been found with respect to the number of multiplications and additions carried out during the evaluation of individuals.

As these conclusions coincide with our goals, we think that the new genetic operators proposed constitute a considerable improvement to the problem of tackling partial abductive inference in BBNs by GAs. Moreover, from the experimentation and the statistical analysis, the following more specific conclusions can also be obtained.

1) GASO Versus GACO:

1) There is no series in which a significant difference unfavoralbe to GASO2l or GASO2r has been found with respect to mass(1, 10, 25, 50).
2) The differences found with respect to additions are, roughly speaking, that GACO requires between 1.7 and 2.4 more additions than GASO. In the case of multiplications, GACO requires, roughly speaking, between 1.3 and 1.8 more multiplications than GASO. Notice that the reduction ratios are different even for the experiments carried out over the same network, which is due to the fact that savings depend on the topology of the precomputated join tree, and on the way variables in the explanation set are distributed over the join tree.
3) With respect to the number of generations carried out, we can see that significant differences unfavorable to GASO1r with respect to GACO can be observed in some cases (one in Experiments 2 and 4, and three in Experiment 3). By contrast, significant differences in favor of GASO2l with respect to GACO can be observed in 12 of the 16 series.
4) The behavior of the two approaches seems to be similar with respect to the number of different evaluated individuals, except in experiment 4, where several times significant differences have been found in favor of the GASO approach. The four cases in which significant differences are observed in favor of GACO are always with respect to the random selection of crossover points.
5) The extra amount of computer memory required by the GASO approach can be calculated by using the following expression:

$$\text{Extra Memory} = \text{PopSize} \times (\alpha \times \text{JoinTreeSize}) \qquad (8)$$

where α is the percentage of space required to store the messages with respect to the space required to store the cliques in join tree. In our experiments α has taken values in the interval [0.2, 0.27]. Therefore, if a double needs four bytes to be represented, then the computer memory necessary to store the join tree in Experiments 1, 2, and 3 is about 5 kB, while 18 kB are required to store the join tree used in Experiment 4. However, if the GASO approach is applied and population size is 200, then about 150 kB, 156 kB, 125 kB and 745 kB are required to store the `messages` vector in Experiments 1, 2, 3, and 4, respectively. As we can see, the amount of memory required has increased with respect to the GACO approach, although these memory requirements are clearly affordable in today's personal computers.

2) Different GASO Approaches: For this analysis, we focus on the best group determined by the SNK test. By best group, we mean the group in which the algorithm with the best average is included. Notice that this does not mean that there are significant differences between all the algorithms included in this group and those that are excluded, but there is (at least) a significant difference between the algorithm with the best average and those that are excluded from the best group.

1) *Accuracy:* The accuracy exhibited by the six GASO algorithms is similar. In fact, in all the series except one, the six algorithms are included in the same group by the statistical analysis carried out. In the remaining one, there is a significant difference in favor of the two-point crossover approach with respect to the one-point approach.
2) *Additions and multiplications:* In this case, it seems that when significant differences exist (12 of the 16 series), they favor the one-point approach (except GASO1r, which is excluded in four of the 12 series). The explanation of this fact lies (in our opinion) in the greater portion of the join tree that has to be reevaluated when the two-point crossover approach is used.
3) *Generations:* From the analysis, it can be observed that GASO2p and GASO2l are always included in the best group. Although GASO1l and GASO2r are sometimes included in the best group, it can be deduced that GASO2p and GASO2l are the algorithms needing a small number of generations before the stopping criterion is met.
4) *Individuals:* In 11 of the 16 series, all the algorithms are included in the same group. In four of the five remaining series, GASO1r is excluded from the best group. Therefore, in this case, the six GASO algorithms seem to have a similar behavior, the random selection being a bit more unstable.

3) Different Population Options:

1) *Accuracy:* Three different groups can be considered here.
   a) Experiment 2: This problem seems to be the easiest one considered here, and no significant differences between the four combinations are found.
   b) *Experiments 1 and 3:* In these cases, the worst choice is R1, which is always excluded from the best group. The best option seems to be H2 because it is always included in the best group and also has the best average. However, H1 and R2 seem to be competitive with H2, being included in the best group most often.
   c) *Experiment 4:* This case is strikingly different from the previous one because the heuristic initialization of the population yields worse results than a whole random initialization.

Therefore, it seems that the introduction of promising individuals in the initial population contributes to focus the search to promising regions of the search space, although in some situations (Experiment 4) these regions can be local optima, being too difficult for the algorithm to escape from them. Although in this paper we have always used the same percentage (50%) as in [16], it might

[8]Sixteen series have been carried out: four experiments by four population combinations.

be a good idea to reduce the percentage of heuristic individuals introduced in the initial population.

2) *Generations:* With respect to this parameter, it is clear that the presence of heuristic individuals focuses the search; hence, $H2$ and $H1$ need a smaller number of generations than $R2$ and $R1$ to meet the stopping criterion.
3) *Individuals, additions, and multiplications:* For these parameters, the four combinations are usually classified in separate groups. Taking into account the averages, the order in which they are ranked is $H1 < R1 < H2 < R2$.Therefore, $H1$ is the choice that consumes fewer resources. Taking into account this fact and also that we are trying to solve an inference problem (so a quick response is usually important), together with the conclusion previously obtained from the *accuracy* parameters (introduction of less than 50% of heuristic individuals), it seems that the best choice would be $H1$ if we need a quick answer and $H2$ if more time is available.

VI. Concluding Remarks

The problem of performing partial abductive inference in BBNs has been studied. We have improved a previous GA (GACO [16]) by introducing new specific genetic operators, which take into account the way the chromosomes are being evaluated. With the introduction of these new operators, the process of evaluating new individuals (the most time consuming process in our GA) requires less computation and so the resulting GA performs faster. Moreover, from the experimental study carried out, we can conclude that the accuracy of the new GA (GASO) is similar to the one obtained by the previously known GA (GACO).

One disadvantage of the new operators with respect to the classical genetic operators used in GACO is the extra amount of computer memory required in GASO, but nowadays, this fact does not constitute a major problem, while any gain in response time is very much appreciated.

In our future work, we plan to study other ways of evaluating a chromosome (approximate computation) and the use of other optimization techniques such as simulated annealing[41] or tabu search[42], [43].

Appendix I
Propagation in Bayesian Networks

In this appendix, we introduce the Bayesian networks formalism (Appendix I-A) and describe how the basic inference task (probabilities or evidence propagation) is carried out (Appendix I-B).

A. Bayesian Networks

A BBN [1], [44] is a directed acyclic graph where each node represents a random variable and the topology of the graph defines a set of conditional independence properties. These properties can be identified using a graphical criterion called d-separation (see [1]). The quantitative part of the model is given by a probability distribution for each node conditioned to its parents. For example, Fig. 10 shows a Bayesian network with eight variables.

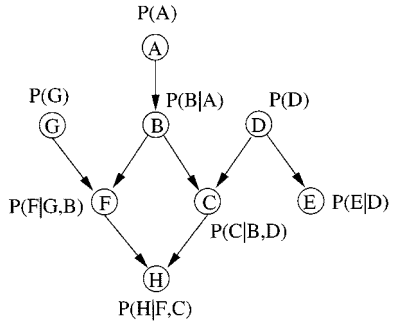

Fig. 10. Bayesian network.

Before proceeding, we define the following notation. A lower case subscript indicates a single variable (e.g., X_i). An upper case subscript indicates a set of variables (e.g., X_I). For some particular problems, the propositional variables are denoted by capital letters without subscript $A, B, C, \ldots$. The state taken by a variable X_i is denoted by x_i and the configuration of states taken by a set of variables X_D is denoted by x_D. That is, capital letters are reserved for variables and sets of variables and lowercase letters are reserved for states and configurations of states. The set of different states that a variable X_i can take is denoted by Ω_{X_i} and the set of different configurations that a subset of variables X_D can take is denoted by Ω_{X_D}.

If $X_{\mathcal{U}} = \{X_1, \ldots, X_n\}$ is the set of variables in the network, then using the independence relationships encoded in the graph, the joint probability can be factorized as

$$P(X_{\mathcal{U}}) = \prod_{X_i \in X_{\mathcal{U}}} P(X_i | pa(X_i)) \tag{9}$$

where $pa(X_i)$ contains the parents of X_i in the graph. This equation is known as the chain rule and allows us to represent the joint probability distribution efficiently. For example, for the network in Fig. 10, the number of values to be stored in order to represent the joint probability distribution is 256 if each variable can take two different states and 6561 if the number of different states is three. However, using the chain rule, the number of values to be stored is 38 for the two states per variable case and 109 for the three states per variable case.

B. Probabilities Propagation

The main type of inference in Bayesian networks is known as probabilities propagation or evidence propagation. This problem consists of obtaining the probability of a variable X_i given some observations (the evidence $X_O = x_O$). For example, what is the probability of having the flu knowing that the patient has a cough and temperature?

In general, we are interested in obtaining this *a posteriori* probability for all the unobserved variables, so, if $X_O = x_O$ is the observed evidence, the goal is to obtain $P(X_i|x_O)$ for every $X_i \in X_{\mathcal{U}} \setminus X_O$, where the backslash denotes the set difference operation.

To compute $p(x_i|x_O)$, it is enough to compute $p(x_i, x_O)$ for every $x_i \in \Omega_{X_i}$ as the former is proportional to the later. In fact, we have $p(x_i|x_O) = p(x_i, x_O)/p(x_O)$ and $p(x_O) = \sum_{x_i} p(x_i, x_O)$. To compute $p(x_i, x_O)$, we cannot apply the chain rule (9), as this expression provides a factorization of the joint probability distribution for all the variables in

the problem and here we need the probability distribution for variables $X_{O\cup\{i\}}$. If $X_{R_i} = X_{\mathcal{U}} \setminus X_{O\cup\{i\}}$, then we have

$$p(x_i, x_O) = \sum_{x_{R_i}} p(x_{R_i}, x_i, x_O). \tag{10}$$

The chain rule can be applied to compute each probability $p(x_{R_i}, x_i, x_O)$, but we should apply it $|\Omega_{X_{R_i}}| = \prod_{j\in R_i} |\Omega_{X_j}|$ times, adding the results afterwards. This approach is clearly unfeasible even for moderate sets of probability distributions. In the last few years, many algorithms [1], [45]–[48] have been proposed to solve this problem (in an exact way) by taking advantage of the conditional (in)dependences among variables given by the structure of the graph. These algorithms are called propagation algorithms because computations are performed locally and information is shared among the nodes in a network by means of messages that are sent (propagated) across this network. Although the propagation problem is NP-hard [49] in the worst case, the existing algorithms work efficiently for moderate size networks.

Nowadays, the most frequently used propagation algorithms are based in the transformation (compilation) of the Bayesian network in a secondary structure called join tree or junction tree in which the calculations are carried out. A join tree is a tree whose nodes are clusters of variables and in which the following two properties hold.

1) For each variable X_i in the Bayesian network, there is at least one cluster C_j in the tree containing the set of variables $\{X_i\} \cup pa(X_i)$.
2) *Running intersection property:* If C_i and C_j are two clusters in the join tree, then the variables in $C_i \cap C_j$ are contained in every cluster along the path between C_i and C_j.

Property 1 is necessary in order to establish a potential representation of the joint probability distribution. Each cluster C_j in the tree has associated a potential function $\psi(C_j) : \Omega_{C_j} \rightarrow \mathbb{R}$. These potentials are initialized in the following way.

1) For all clusters C_j in the join tree, do $\psi(c_j) = 1.0, \forall c_j \in \Omega_{C_j}$.
2) For all variables X_i in the network, select one (and only one) cluster C_j, such that $\{X_i \cup pa(X_i)\} \subseteq C_j$, and do $\psi(C_j) = \psi(C_j) \otimes P(X_i|pa(X_i))$, where $\otimes$ represents point-by-point multiplication. (In this context, the operation $\otimes$ is known as *combination*).

After this initialization of the clusters, if $\mathcal{T}$ denotes the set of clusters in the join tree, then the following expression represents a factorization of the joint probability distribution:

$$P(X_{\mathcal{U}}) = \prod_{C_j\in\mathcal{T}} \psi(C_j). \tag{11}$$

Running intersection property is necessary in order to ensure that computations can be carried out in a local manner, i.e., to ensure consistency in the message passing scheme (see [44] for details).

A join tree can be viewed as an undirected graph or as a directed graph if we select a cluster as the root. Fig. 11 shows a join tree for the network in Fig. 10, where cluster 1 has been selected as the root.

Associated with each edge of the join tree is a separator, which is the set of variables in the intersection of the two clusters at the endpoints of the edge. We use $S_{i,j}$ to denote the separator between C_i and C_j.

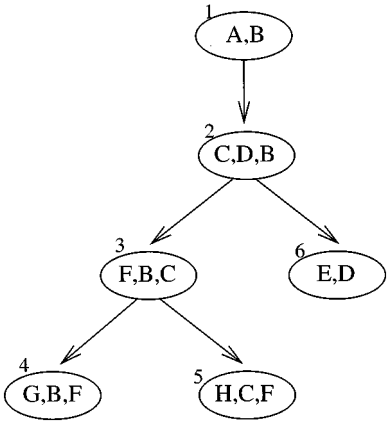

Fig. 11. Multiple test for Experiment 1 with random population of size 100.

Depending on the way the inference is executed over the join tree, we have several architectures: Lauritzen–Spiegelhalter [46], Shafer–Shenoy [48], and HUGIN [50]. In this paper, we focus on the Shafer–Shenoy architecture because it is the simplest to explain and understand. Furthermore, for the propagation tasks here required (abductive inference and likelihood computation), the three architectures are equivalent.

In the Shafer–Shenoy architecture, there are two messages in the separator $S_{i,j}$ between each pair of adjacent clusters C_i and C_j, one in each direction. $M^{i\rightarrow j}$ will denote the message from C_i to C_j. The values of the messages are defined recursively according to the following expression:

$$M^{i\rightarrow j} = \sum_{C_i\setminus S_{i,j}} \left(\psi(C_i) \otimes \left(\bigotimes_{C_k\in ne(j), k\neq j} M^{k\rightarrow i} \right) \right) \tag{12}$$

where $ne(j)$ is the set of neighboring clusters of C_j.

The process of reducing a potential to a subset of variables of the original set is called *marginalization*. In this algorithm, the marginalization is carried out by a summation over the variables of the original set that are not in the subset.

Note that the message going from C_i to C_j depends on the potential in cluster $C_i(\psi(C_i))$ as well as all the messages arriving at C_i from a cluster different from $C_j(M^{k\rightarrow i}, k \neq j)$. The propagation algorithm consists in an orderly computation of messages in order to ensure that C_i sends a message to C_j when all the necessary elements to compute it are available, i.e., C_i has collected information (messages) from all its neighbors except C_j.

During the propagation, the messages flow in two phases: upward (from leaves to top) and downward (from top to leaves). After the two phases the *a posteriori* probability of each variable can be calculated. Algorithm 1 shows the pseudocode of the propagation algorithm.

1) The evidence $X_O = x_O$ is incorporated to the join tree in the following way (by $x^{\downarrow Y}$ we denote the configuration obtained from x after removing the literals corresponding to the variables not in Y)

$$\begin{aligned} &\forall C_i \in \mathcal{T} \text{ such that } C_i \cap X_O \neq \emptyset, \\ &\forall c_i \in \Omega_{C_i} \\ &\psi(c_i) = \begin{cases} \psi(c_i) & \text{if } c_i^{\downarrow C_i\cap X_O} = x_O^{\downarrow C_i\cap X_O} \\ 0 & \text{in other case.} \end{cases} \end{aligned} \tag{13}$$

2) The message passing scheme is controlled by means of an iterative procedure, which uses a topological ordering of the clusters in the join tree, i.e., if $C_i \rightarrow C_j$ is an edge in the tree, then $i < j$. The direction of the edges is established choosing C_1 as the root. In the paper, we suppose that C_1 is always the root and that $C_1, \ldots, C_t$ is a topological ordering.
3) In the algorithm, $\mathrm{ch}(C_i)$ denotes the set of children of C_i in $\mathcal{T}$, $\mathrm{fa}(C_i)$ denotes the father of C_i in $\mathcal{T}$, and $\mathrm{fa}(i)$ denotes the index of $\mathrm{fa}(C_i)$.

```
Algorithm 1: Probabilities Propagation in a Join Tree
Input: The join tree T = {C_1, ..., C_t}. The evidence X_O = x_O.
Output: P(X_i|x_O) for all unobserved variable X_i.
1. Incorporate evidence X_O = x_O to T.
2. Upward phase
 2.1 for i ← t downto 2 do
  - Calculate and send M^{i→fa(i)}
 2.2 ψ'(C_1) ← ψ(C_1) ⊗ (⊗_{C_k ∈ ch(C_1)} M^{k→1})
 3. P(x_O) ← Σ_{C_1} ψ'(c_1) If P(x_O) = 0 exit.
4. Downward phase
 4.1 for i ← 2 to t do
  - Calculate and send M^{fa(i)→i}
5. for all X_i ∉ X_O do
 5.1 Select a cluster C_j such that X_i ∈ C_j
 5.2 ψ(X_i) ← Σ_{C_j \ X_i} ψ(C_i)
 5.3 P(X_i|x_O) ← ψ(X_i)/P(x_O)
```

In Step 3, if $P(x_O) = 0$, the evidence is impossible, so there is no reason to continue with the calculations. Algorithm 1 assumes the existence of observed evidence; otherwise, Steps 1, 3, and 5.3 are not executed.

The crucial factor determining the complexity of the computations is the size of the clusters. Each potential defined on cluster C_i needs $\prod_{k \in C_i} |\Omega_{X_k}|$ values. So, the number of operations to send a message from C_i to C_j is of this order of magnitude. The number of messages can always be made linear in the number of variables. The problem is that $\prod_{k \in C_i} |\Omega_{X_k}|$ is exponential in the size of C_i. So, the propagation is feasible only if we are capable of obtaining a join tree such that each cluster has a reduced number ofvariables. This, in general, mainly depends on the topology of the original network, but there are situations in which this never happens. Even the problem of obtaining an optimal join tree (minimum cluster size) is equivalent to obtaining an optimal triangulation sequence of an undirected graph and this problem is known to be NP-hard [51]

APPENDIX II
EXPERIMENTAL RESULTS

A. Output of the Algorithms

TABLE IV
RESULTS FOR EXPERIMENT 1 WITH RANDOM INITIAL POPULATION OF SIZE 100

	*%mass*1		*%mass*10		*%mass*25		*%mass*50	
Alg.	*A*	*SD*	*A*	*SD*	*A*	*SD*	*A*	*SD*
GACO	83.27	29.55	69.52	25.11	61.61	24.20	58.08	23.87
GASO1r	83.02	30.26	69.93	27.21	62.49	27.08	59.17	27.00
GASO1p	85.47	25.28	72.76	23.46	65.30	24.30	62.00	24.82
GASO1l	78.41	37.27	69.46	34.81	63.91	34.49	61.42	34.40
GASO2r	83.23	24.12	72.86	22.69	66.20	23.04	62.92	23.26
GASO2p	82.75	26.91	73.01	25.64	66.66	25.93	63.48	26.02
GASO2l	**88.18**	17.37	**78.80**	18.62	**71.94**	20.10	**68.42**	20.61
	#generations		*#individuals*		*#additions*		*#multiplications*	
Alg.	*A*	*SD*	*A*	*SD*	*A*	*SD*	*A*	*SD*
GACO	104.5	38.4	1,989.9	489.2	1.72e+06	4.22e+05	2.16e+06	5.31e+05
GASO1r	101.0	34.0	1,862.1	412.3	(+)6.56e+05	1.47e+05	(+)1.13e+06	2.62e+05
GASO1p	96.6	28.3	1,859.3	389.8	(+)6.41e+05	1.36e+05	(+)1.11e+06	2.42e+05
GASO1l	97.9	34.3	1,898.7	426.8	(+)6.38e+05	1.46e+05	(+)1.10e+06	2.6e+05
GASO2r	94.9	31.1	1,954.3	418.8	(+)7.54e+05	1.6e+05	(+)1.32e+06	2.86e+05
GASO2p	90.5	34.6	1,894.7	470.3	(+)7.19e+05	1.77e+05	(+)1.26e+06	3.18e+05
GASO2l	(+)85.4	20.1	1,837.9	283.5	(+)6.84e+05	1.03e+05	(+)1.20e+06	1.86e+05

TABLE V
RESULTS FOR EXPERIMENT 1 WITH 1/2 HEURISTIC INITIAL POPULATION OF SIZE 100

	$\%mass1$		$\%mass10$		$\%mass25$		$\%mass50$	
Alg.	*A*	*SD*	*A*	*SD*	*A*	*SD*	*A*	*SD*
GACO	**100.00**	0.00	89.61	13.03	83.34	19.32	80.57	21.43
GASO1r	**100.00**	0.00	87.29	12.92	79.62	19.07	76.29	21.09
GASO1p	**100.00**	0.00	89.96	12.40	84.01	18.77	81.32	20.94
GASO1l	**100.00**	0.00	**92.70**	10.75	**87.54**	16.32	**84.98**	18.33
GASO2r	**100.00**	0.00	84.70	13.81	76.07	20.22	72.60	22.06
GASO2p	**100.00**	0.00	86.32	13.70	78.46	20.00	75.17	21.86
GASO2l	**100.00**	0.00	88.21	13.25	80.88	19.21	77.66	21.02
	#generations		*#individuals*		*#additions*		*#multiplications*	
Alg.	*A*	*SD*	*A*	*SD*	*A*	*SD*	*A*	*SD*
GACO	82.1	22.0	1,318.0	243.6	1.14e+06	2.1e+05	1.43e+06	2.64e+05
GASO1r	77.5	16.1	1,229.9	172.7	(+)4.73e+05	6.5e+04	(+)7.97e+05	1.15e+05
GASO1p	73.7	19.2	1,235.9	211.6	(+)4.64e+05	7.67e+04	(+)7.83e+05	1.36e+05
GASO1l	77.0	23.2	1,287.4	231.5	(+)4.75e+05	8.62e+04	(+)8.03e+05	1.51e+05
GASO2r	73.5	17.4	1,257.2	228.1	(+)5.19e+05	8.54e+04	(+)8.84e+05	1.53e+05
GASO2p	(+)68.3	18.0	1,216.0	212.9	(+)4.95e+05	8.19e+04	(+)8.43e+05	1.46e+05
GASO2l	(+)70.3	21.6	1,264.0	259.1	(+)5.05e+05	9.74e+04	(+)8.63e+05	1.74e+05

TABLE VI
RESULTS FOR EXPERIMENT 1 WITH RANDOM INITIAL POPULATION OF SIZE 200

	$\%mass1$		$\%mass10$		$\%mass25$		$\%mass50$	
Alg.	*A*	*SD*	*A*	*SD*	*A*	*SD*	*A*	*SD*
GACO	92.60	11.41	83.19	14.74	77.36	18.65	74.65	20.43
GASO1r	90.15	17.19	84.84	18.93	81.33	21.12	79.44	22.29
GASO1p	93.59	10.92	90.68	13.49	88.29	15.55	(+)86.81	16.85
GASO1l	91.15	20.89	88.20	22.11	85.76	23.25	84.34	23.96
GASO2r	94.08	10.64	89.99	14.82	86.90	18.03	85.25	19.62
GASO2p	**95.07**	9.96	**91.50**	13.81	(+)**88.80**	16.86	(+)**87.32**	18.39
GASO2l	92.60	11.41	88.38	15.01	85.31	17.81	83.64	19.29
	#generations		*#individuals*		*#additions*		*#multiplications*	
Alg.	*A*	*SD*	*A*	*SD*	*A*	*SD*	*A*	*SD*
GACO	91.3	32.3	3,849.5	900.6	3.32e+06	7.77e+05	4.18e+06	9.77e+05
GASO1r	87.8	27.5	3,628.8	719.8	(+)1.27e+06	2.59e+05	(+)2.2e+06	4.62e+05
GASO1p	82.3	22.7	3,636.4	611.0	(+)1.23e+06	2.14e+05	(+)2.14e+06	3.81e+05
GASO1l	78.9	20.6	3,541.5	522.7	(+)1.18e+06	1.82e+05	(+)2.04e+06	3.24e+05
GASO2r	77.0	27.4	3,677.5	784.7	(+)1.42e+06	3.01e+05	(+)2.49e+06	5.38e+05
GASO2p	78.3	22.2	3,805.9	621.4	(+)1.43e+06	2.36e+05	(+)2.52e+06	4.24e+05
GASO2l	(+)73.6	20.0	3,669.6	589.9	(+)1.35e+06	2.19e+05	(+)2.38e+06	3.93e+05

TABLE VII
RESULTS FOR EXPERIMENT 1 WITH 1/2 HEURISTIC INITIAL POPULATION OF SIZE 200

	%mass1		%mass10		%mass25		%mass50	
Alg.	*A*	*SD*	*A*	*SD*	*A*	*SD*	*A*	*SD*
GACO	**100.00**	0.00	97.31	7.56	95.51	11.66	94.62	13.24
GASO1r	**100.00**	0.00	96.69	8.16	94.00	12.46	92.59	14.13
GASO1p	**100.00**	0.00	96.18	8.71	93.57	13.40	92.33	15.21
GASO1l	**100.00**	0.00	**97.75**	6.86	**96.10**	10.65	**95.22**	12.16
GASO2r	**100.00**	0.00	95.35	9.68	92.13	14.65	90.56	16.45
GASO2p	**100.00**	0.00	94.30	9.89	90.18	15.14	88.23	17.13
GASO2l	**100.00**	0.00	97.18	6.81	94.40	10.77	92.84	12.52
	#generations		*#individuals*		*#additions*		*#multiplications*	
Alg.	*A*	*SD*	*A*	*SD*	*A*	*SD*	*A*	*SD*
GACO	67.0	17.5	2,451.1	387.7	2.12e+06	3.35e+05	2.66e+06	4.21e+05
GASO1r	64.0	14.8	2,323.7	339.8	(+)8.85e+05	1.27e+05	(+)1.49e+06	2.24e+05
GASO1p	62.7	13.7	2,370.7	347.8	(+)8.79e+05	1.23e+05	(+)1.49e+06	2.18e+05
GASO1l	58.3	13.7	2,314.3	339.9	(+)8.42e+05	1.19e+05	(+)1.42e+06	2.11e+05
GASO2r	62.7	17.6	2,493.1	468.5	(+)1.02e+06	1.83e+05	(+)1.75e+06	3.28e+05
GASO2p	60.0	16.7	2,433.9	445.6	(+)9.85e+05	1.71e+05	(+)1.69e+06	3.06e+05
GASO2l	63.5	17.3	2,565.1	438.8	(+)1.02e+06	1.69e+05	(+)1.74e+06	3.02e+05

TABLE VIII
RESULTS FOR EXPERIMENT 2 WITH RANDOM INITIAL POPULATION OF SIZE 100

	%mass1		%mass10		%mass25		%mass50	
Alg.	*A*	*SD*	*A*	*SD*	*A*	*SD*	*A*	*SD*
GACO	96.36	18.00	96.44	17.33	96.42	16.64	94.77	15.81
GASO1r	**100.00**	0.00	**100.00**	0.03	99.80	0.61	97.99	1.25
GASO1p	98.18	12.84	98.26	12.29	98.14	11.69	96.41	10.90
GASO1l	98.18	12.88	98.25	12.36	98.14	11.79	96.51	11.07
GASO2r	**100.00**	0.00	**100.00**	0.00	**99.94**	0.22	98.25	0.85
GASO2p	98.01	14.04	98.02	14.03	97.94	14.00	96.15	13.78
GASO2l	**100.00**	0.00	**100.00**	0.00	99.90	0.27	**98.34**	1.08
	#generations		*#individuals*		*#additions*		*#multiplications*	
Alg.	*A*	*SD*	*A*	*SD*	*A*	*SD*	*A*	*SD*
GACO	89.4	19.3	1,754.8	286.5	1.57e+06	2.56e+05	2.03e+06	3.31e+05
GASO1r	(-)104.4	22.9	1,803.6	267.4	(+)6.27e+05	8.96e+04	(+)1.1e+06	1.63e+05
GASO1p	101.6	30.9	1,810.2	374.8	(+)6.06e+05	1.21e+05	(+)1.07e+06	2.2e+05
GASO1l	97.7	21.6	1,855.8	281.7	(+)5.97e+05	8.86e+04	(+)1.05e+06	1.62e+05
GASO2r	85.0	21.4	1,756.5	254.6	(+)6.74e+05	9.33e+04	(+)1.2e+06	1.7e+05
GASO2p	80.1	17.5	1,700.4	247.3	(+)6.34e+05	8.43e+04	(+)1.13e+06	1.55e+05
GASO2l	82.1	23.6	1,771.6	331.1	(+)6.37e+05	1.14e+05	(+)1.13e+06	2.09e+05

TABLE IX
RESULTS FOR EXPERIMENT 2 WITH 1/2 HEURISTIC INITIAL POPULATION OF SIZE 100

	%mass1		*%mass10*		*%mass25*		*%mass50*	
Alg.	*A*	*SD*	*A*	*SD*	*A*	*SD*	*A*	*SD*
GACO	**100.00**	0.00	**100.00**	0.00	99.92	0.39	**98.28**	0.80
GASO1r	**100.00**	0.00	**100.00**	0.00	99.87	0.59	(-)97.73	1.22
GASO1p	**100.00**	0.00	**100.00**	0.00	99.90	0.23	(-)97.66	0.78
GASO1l	**100.00**	0.00	**100.00**	0.00	99.84	0.52	(-)97.76	1.17
GASO2r	**100.00**	0.00	**100.00**	0.00	99.98	0.13	98.04	0.55
GASO2p	**100.00**	0.00	**100.00**	0.03	99.94	0.39	98.20	0.68
GASO2l	**100.00**	0.00	**100.00**	0.00	**99.99**	0.06	98.21	0.46
	#generations		*#individuals*		*#additions*		*#multiplications*	
Alg.	*A*	*SD*	*A*	*SD*	*A*	*SD*	*A*	*SD*
GACO	57.0	12.6	886.7	124.4	7.93e+05	1.11e+05	1.02e+06	1.44e+05
GASO1r	59.0	12.6	865.0	121.8	(+)3.47e+05	4.39e+04	(+)5.83e+05	7.91e+04
GASO1p	56.6	10.3	872.5	106.7	(+)3.4e+05	3.64e+04	(+)5.72e+05	6.57e+04
GASO1l	59.7	12.7	919.9	130.9	(+)3.49e+05	4.47e+04	(+)5.88e+05	8.08e+04
GASO2r	(+)46.5	9.0	(+)788.7	101.8	(+)3.4e+05	3.84e+04	(+)5.74e+05	6.99e+04
GASO2p	(+)47.6	10.1	(+)810.6	108.3	(+)3.42e+05	4.08e+04	(+)5.79e+05	7.43e+04
GASO2l	(+)46.3	8.9	(+)801.5	104.3	(+)3.32e+05	3.79e+04	(+)5.59e+05	6.9e+04

TABLE X
RESULTS FOR EXPERIMENT 2 WITH RANDOM INITIAL POPULATION OF SIZE 200

	%mass1		*%mass10*		*%mass25*		*%mass50*	
Alg.	*A*	*SD*	*A*	*SD*	*A*	*SD*	*A*	*SD*
GACO	**100.00**	0.00	**100.00**	0.00	**100.00**	0.00	**99.30**	0.54
GASO1r	**100.00**	0.00	**100.00**	0.00	99.97	0.09	99.01	0.73
GASO1p	**100.00**	0.00	**100.00**	0.00	99.97	0.14	98.97	0.74
GASO1l	**100.00**	0.00	**100.00**	0.00	99.99	0.11	99.27	0.69
GASO2r	**100.00**	0.00	**100.00**	0.00	99.96	0.16	99.12	0.70
GASO2p	**100.00**	0.00	**100.00**	0.00	**100.00**	0.01	99.22	0.55
GASO2l	**100.00**	0.00	**100.00**	0.00	99.97	0.16	99.17	0.72
	#generations		*#individuals*		*#additions*		*#multiplications*	
Alg.	*A*	*SD*	*A*	*SD*	*A*	*SD*	*A*	*SD*
GACO	73.4	11.6	3,337.6	346.2	2.98e+06	3.09e+05	3.85e+06	4e+05
GASO1r	80.1	15.8	3,215.6	392.2	(+)1.1e+06	1.31e+05	(+)1.94e+06	2.38e+05
GASO1p	76.3	15.2	3,252.5	405.9	(+)1.07e+06	1.3e+05	(+)1.88e+06	2.36e+05
GASO1l	73.5	12.8	3,283.5	374.0	(+)1.04e+06	1.12e+05	(+)1.82e+06	2.05e+05
GASO2r	72.2	17.8	3,372.7	450.4	(+)1.29e+06	1.64e+05	(+)2.3e+06	2.99e+05
GASO2p	65.9	11.3	3,241.5	356.5	(+)1.2e+06	1.25e+05	(+)2.14e+06	2.31e+05
GASO2l	(+)62.8	12.9	3,188.3	431.0	(+)1.14e+06	1.44e+05	(+)2.02e+06	2.65e+05

TABLE XI
RESULTS FOR EXPERIMENT 2 WITH 1/2 HEURISTIC INITIAL POPULATION OF SIZE 200

	%mass1		%mass10		%mass25		%mass50	
Alg.	*A*	*SD*	*A*	*SD*	*A*	*SD*	*A*	*SD*
GACO	**100.00**	0.00	**100.00**	0.00	99.99	0.06	**98.92**	0.62
GASO1r	**100.00**	0.00	**100.00**	0.00	99.98	0.13	98.71	0.76
GASO1p	**100.00**	0.00	**100.00**	0.03	99.89	0.54	(-)98.52	0.81
GASO1l	**100.00**	0.00	**100.00**	0.00	99.93	0.38	(-)98.53	0.69
GASO2r	**100.00**	0.00	**100.00**	0.00	**100.00**	0.00	98.81	0.57
GASO2p	**100.00**	0.00	**100.00**	0.00	**100.00**	0.00	98.83	0.60
GASO2l	**100.00**	0.00	**100.00**	0.00	**100.00**	0.00	98.59	0.47
	#generations		*#individuals*		*#additions*		*#multiplications*	
Alg.	*A*	*SD*	*A*	*SD*	*A*	*SD*	*A*	*SD*
GACO	44.0	9.0	1,592.5	235.9	1.42e+06	2.11e+05	1.84e+06	2.72e+05
GASO1r	47.6	11.6	1,575.9	249.5	(+)6.32e+05	8.81e+04	(+)1.06e+06	1.59e+05
GASO1p	43.7	8.5	1,546.1	204.3	(+)6.04e+05	6.86e+04	(+)1.01e+06	1.25e+05
GASO1l	44.2	10.2	1,592.8	229.7	(+)6.03e+05	7.56e+04	(+)1.01e+06	1.36e+05
GASO2r	40.9	8.5	1,546.4	217.1	(+)6.66e+05	8.05e+04	(+)1.13e+06	1.47e+05
GASO2p	39.7	11.0	1,535.8	268.1	(+)6.48e+05	9.83e+04	(+)1.09e+06	1.79e+05
GASO2l	(+)36.3	8.1	1,466.9	209.8	(+)6.07e+05	7.6e+04	(+)1.02e+06	1.38e+05

TABLE XII
RESULTS FOR EXPERIMENT 3 WITH RANDOM INITIAL POPULATION OF SIZE 100

	%mass1		%mass10		%mass25		%mass50	
Alg.	*A*	*SD*	*A*	*SD*	*A*	*SD*	*A*	*SD*
GACO	**79.46**	37.63	**77.52**	37.25	**73.71**	36.86	**70.00**	36.57
GASO1r	68.72	42.34	65.54	40.97	59.40	38.78	54.38	36.91
GASO1p	70.77	41.04	67.64	39.97	62.16	38.40	57.52	37.07
GASO1l	75.23	38.72	72.21	37.85	66.36	36.73	61.48	35.62
GASO2r	69.59	42.86	67.44	42.14	63.44	40.90	60.07	40.08
GASO2p	67.60	42.39	64.78	41.11	59.63	39.17	55.34	37.58
GASO2l	69.64	41.44	66.81	40.40	61.57	38.99	57.43	37.78
	#generations		*#individuals*		*#additions*		*#multiplications*	
Alg.	*A*	*SD*	*A*	*SD*	*A*	*SD*	*A*	*SD*
GACO	101.7	32.7	1,989.4	447.3	1.53e+06	3.45e+05	2.09e+06	4.69e+05
GASO1r	106.3	27.6	1,989.1	351.3	(+)6.75e+05	1.2e+05	(+)1.19e+06	2.17e+05
GASO1p	103.0	26.7	2,011.4	385.3	(+)6.6e+05	1.25e+05	(+)1.16e+06	2.26e+05
GASO1l	108.9	32.0	2,111.3	415.7	(+)6.75e+05	1.37e+05	(+)1.18e+06	2.46e+05
GASO2r	92.0	21.9	2,002.4	323.6	(+)7.55e+05	1.19e+05	(+)1.34e+06	2.17e+05
GASO2p	92.7	27.8	2,015.5	433.5	(+)7.39e+05	1.54e+05	(+)1.31e+06	2.8e+05
GASO2l	(+)82.8	21.6	1,906.2	366.5	(+)6.71e+05	1.27e+05	(+)1.18e+06	2.3e+05

TABLE XIII
RESULTS FOR EXPERIMENT 3 WITH 1/2 HEURISTIC INITIAL POPULATION OF SIZE 100

	%mass1		*%mass10*		*%mass25*		*%mass50*	
Alg.	*A*	*SD*	*A*	*SD*	*A*	*SD*	*A*	*SD*
GACO	99.53	1.88	98.75	4.12	**92.78**	10.06	**86.62**	15.18
GASO1r	98.59	3.05	96.56	7.14	88.18	14.75	81.13	18.96
GASO1p	**99.69**	1.55	**98.88**	4.05	91.06	10.62	83.38	15.70
GASO1l	98.59	3.05	96.42	7.08	86.40	13.69	77.47	17.36
GASO2r	99.06	2.58	97.44	6.18	88.53	12.92	80.23	17.31
GASO2p	99.21	2.38	97.87	5.25	87.32	10.30	77.31	14.65
GASO2l	99.21	2.38	98.11	4.96	90.97	10.66	83.64	15.71
	#generations		*#individuals*		*#additions*		*#multiplications*	
Alg.	*A*	*SD*	*A*	*SD*	*A*	*SD*	*A*	*SD*
GACO	71.7	12.6	1,181.3	141.1	9.11e+05	1.09e+05	1.24e+06	1.48e+05
GASO1r	(-)81.5	19.0	(-)1,291.2	218.1	(+)4.76e+05	7.67e+04	(+)8.25e+05	1.38e+05
GASO1p	73.2	12.6	1,232.1	150.0	(+)4.41e+05	5.16e+04	(+)7.6e+05	9.24e+04
GASO1l	70.7	15.1	1,226.2	183.5	(+)4.29e+05	6.18e+04	(+)7.37e+05	1.1e+05
GASO2r	66.8	17.0	1,207.5	206.9	(+)4.81e+05	7.97e+04	(+)8.38e+05	1.44e+05
GASO2p	(+)62.3	15.1	1,160.9	207.0	(+)4.53e+05	7.68e+04	(+)7.87e+05	1.39e+05
GASO2l	(+)58.6	10.7	1,132.7	144.6	(+)4.29e+05	5.23e+04	(+)7.41e+05	9.45e+04

TABLE XIV
RESULTS FOR EXPERIMENT 3 WITH RANDOM INITIAL POPULATION OF SIZE 200

	%mass1		*%mass10*		*%mass25*		*%mass50*	
Alg.	*A*	*SD*	*A*	*SD*	*A*	*SD*	*A*	*SD*
GACO	**90.50**	26.76	**89.47**	26.99	87.47	27.72	85.64	28.54
GASO1r	**90.50**	26.76	**89.47**	27.00	**87.73**	27.76	**85.95**	28.39
GASO1p	86.37	31.49	84.90	31.57	82.37	32.05	80.03	32.53
GASO1l	86.82	31.24	85.75	31.46	84.04	32.00	82.30	32.35
GASO2r	87.01	31.67	85.91	31.75	83.40	32.02	81.01	32.49
GASO2p	81.47	35.52	79.52	35.27	76.09	35.27	73.28	35.39
GASO2l	77.95	38.21	76.78	38.32	74.89	38.56	73.21	38.46
	#generations		*#individuals*		*#additions*		*#multiplications*	
Alg.	*A*	*SD*	*A*	*SD*	*A*	*SD*	*A*	*SD*
GACO	78.7	15.8	3,629.4	486.0	2.8e+06	3.75e+05	3.81e+06	5.1e+05
GASO1r	(-)98.9	20.9	(-)4,093.8	643.8	(+)1.38e+06	2.13e+05	(+)2.44e+06	3.86e+05
GASO1p	86.4	19.3	3,839.4	585.1	(+)1.24e+06	1.9e+05	(+)2.19e+06	3.44e+05
GASO1l	81.7	18.8	3,833.1	592.3	(+)1.19e+06	1.87e+05	(+)2.08e+06	3.38e+05
GASO2r	84.2	18.7	(-)4,098.5	672.3	(+)1.55e+06	2.47e+05	(+)2.76e+06	4.52e+05
GASO2p	75.1	14.2	3,882.4	636.2	(+)1.42e+06	2.21e+05	(+)2.52e+06	4.04e+05
GASO2l	70.8	16.8	3,768.6	713.9	(+)1.32e+06	2.4e+05	(+)2.33e+06	4.36e+05

TABLE XV
RESULTS FOR EXPERIMENT 3 WITH 1/2 HEURISTIC INITIAL POPULATION OF SIZE 200

	%mass1		%mass10		%mass25		%mass50	
Alg.	*A*	*SD*	*A*	*SD*	*A*	*SD*	*A*	*SD*
GACO	**100.00**	0.00	**99.96**	0.15	**99.14**	3.29	**97.74**	6.50
GASO1r	99.84	1.11	99.56	2.17	96.30	6.80	92.78	12.07
GASO1p	99.84	1.11	99.48	2.16	(-)94.64	7.41	(-)89.35	13.26
GASO1l	**100.00**	0.00	99.88	0.25	97.15	5.78	94.17	10.94
GASO2r	99.84	1.11	99.57	2.17	96.67	6.53	93.33	11.60
GASO2p	99.69	1.55	99.27	3.03	96.34	7.33	93.22	12.30
GASO2l	**100.00**	0.00	99.77	0.30	(-)94.68	6.87	(-)89.24	13.17
	#generations		*#individuals*		*#additions*		*#multiplications*	
Alg.	*A*	*SD*	*A*	*SD*	*A*	*SD*	*A*	*SD*
GACO	59.1	9.3	2,211.9	225.7	1.71e+06	1.74e+05	2.32e+06	2.37e+05
GASO1r	(-)70.4	13.8	(-)2,475.9	336.4	(+)9.09e+05	1.19e+05	(+)1.58e+06	2.14e+05
GASO1p	62.8	13.8	2,364.3	353.9	(+)8.41e+05	1.24e+05	(+)1.45e+06	2.23e+05
GASO1l	60.2	11.3	2,366.7	293.5	(+)8.16e+05	9.97e+04	(+)1.4e+06	1.79e+05
GASO2r	57.1	11.5	2,365.3	325.9	(+)9.42e+05	1.25e+05	(+)1.65e+06	2.27e+05
GASO2p	54.5	8.0	2,342.7	233.6	(+)9.1e+05	8.73e+04	(+)1.58e+06	1.57e+05
GASO2l	(+)51.3	9.9	2,250.2	301.1	(+)8.51e+05	1.1e+05	(+)1.47e+06	1.98e+05

TABLE XVI
RESULTS FOR EXPERIMENT 4 WITH RANDOM INITIAL POPULATION OF SIZE 100

	*mass*1		*mass*10		*mass*25		*mass*50	
Alg.	*A*	*SD*	*A*	*SD*	*A*	*SD*	*A*	*SD*
GACO	0.013527	0.002299	0.086642	0.014464	0.156269	0.026370	0.220818	0.038205
GASO1r	0.013401	0.002418	0.085503	0.016331	0.155129	0.031879	0.221612	0.048476
GASO1p	**0.014197**	0.000000	**0.090888**	0.000194	**0.165407**	0.003262	**0.236975**	0.010271
GASO1l	0.014047	0.001058	0.089835	0.007415	0.163855	0.014881	0.236030	0.023674
GASO2r	0.013551	0.002219	0.086613	0.014808	0.158053	0.028501	0.227447	0.042833
GASO2p	0.014024	0.001226	0.089822	0.007860	0.163969	0.014956	0.235535	0.023781
GASO2l	0.013575	0.002137	0.086725	0.014608	0.158833	0.028884	0.229372	0.044679
	#generations		*#individuals*		*#additions*		*#multiplications*	
Alg.	*A*	*SD*	*A*	*SD*	*A*	*SD*	*A*	*SD*
GACO	104.1	24.3	2,113.7	351.8	7.3e+06	1.22e+06	2.43e+07	4.04e+06
GASO1r	108.0	12.4	(+)1,940.3	191.9	(+)3.58e+06	3.41e+05	(+)1.72e+07	1.68e+06
GASO1p	107.7	25.0	1,958.6	312.7	(+)3.56e+06	5.56e+05	(+)1.72e+07	2.73e+06
GASO1l	93.7	18.0	(+)1,862.8	255.0	(+)3.32e+06	4.43e+05	(+)1.62e+07	2.21e+06
GASO2r	94.5	24.1	1,947.0	313.0	(+)3.81e+06	5.99e+05	(+)1.78e+07	2.83e+06
GASO2p	(+)86.7	19.3	(+)1,868.9	284.1	(+)3.62e+06	5.24e+05	(+)1.69e+07	2.53e+06
GASO2l	(+)86.1	19.1	(+)1,911.2	267.7	(+)3.62e+06	4.93e+05	(+)1.71e+07	2.38e+06

TABLE XVII
RESULTS FOR EXPERIMENT 4 WITH 1/2 HEURISTIC INITIAL POPULATION OF SIZE 100

	*mass*1		*mass*10		*mass*25		*mass*50	
Alg.	*A*	*SD*	*A*	*SD*	*A*	*SD*	*A*	*SD*
GACO	0.009558	0.003669	0.058740	0.025710	0.103597	0.052045	0.144063	0.078483
GASO1r	**0.011054**	0.003731	**0.069072**	0.026295	0.125337	0.054066	0.180078	0.084479
GASO1p	0.010904	0.003752	0.068017	0.026445	0.123264	0.054330	0.177207	0.084693
GASO1l	0.010605	0.003776	0.065908	0.026616	0.119015	0.054699	0.170586	0.085485
GASO2r	**0.011054**	0.003731	**0.069072**	0.026295	**0.125482**	0.054082	**0.180747**	0.084539
GASO2p	0.010605	0.003776	0.065908	0.026616	0.118978	0.054857	0.170810	0.086093
GASO2l	0.010904	0.003752	0.068009	0.026438	0.123293	0.054387	0.177569	0.085227
	#generations		*#individuals*		*#additions*		*#multiplications*	
Alg.	*A*	*SD*	*A*	*SD*	*A*	*SD*	*A*	*SD*
GACO	83.6	31.4	1,469.1	424.6	5.08e+06	1.47e+06	1.69e+07	4.87e+06
GASO1r	99.0	30.9	1,497.2	385.6	(+)2.84e+06	6.89e+05	(+)1.34e+07	3.38e+06
GASO1p	93.2	35.4	1,449.3	421.5	(+)2.73e+06	7.57e+05	(+)1.29e+07	3.7e+06
GASO1l	88.2	35.5	1,418.0	425.3	(+)2.63e+06	7.53e+05	(+)1.26e+07	3.71e+06
GASO2r	87.4	29.5	1,477.9	398.7	(+)2.93e+06	7.55e+05	(+)1.35e+07	3.59e+06
GASO2p	77.7	23.1	1,360.8	325.4	(+)2.69e+06	6.07e+05	(+)1.24e+07	2.91e+06
GASO2l	71.7	25.0	1,328.8	360.0	(+)2.58e+06	6.6e+05	(+)1.2e+07	3.19e+06

TABLE XVIII
RESULTS FOR EXPERIMENT 4 WITH RANDOM INITIAL POPULATION OF SIZE 200

	*mass*1		*mass*10		*mass*25		*mass*50	
Alg.	*A*	*SD*	*A*	*SD*	*A*	*SD*	*A*	*SD*
GACO	0.013874	0.001603	0.088904	0.009871	0.161572	0.018020	0.231215	0.026246
GASO1r	**0.014197**	0.000000	0.091013	0.000232	(+)0.167701	0.003335	(+)0.244295	0.008667
GASO1p	**0.014197**	0.000000	**0.091014**	0.000211	(+) **0.167885**	0.003115	(+)0.245368	0.007798
GASO1l	**0.014197**	0.000000	0.090998	0.000210	(+)0.167568	0.002947	(+)0.244430	0.007055
GASO2r	**0.014197**	0.000000	0.090989	0.000209	(+)0.167580	0.002807	(+)0.244657	0.006378
GASO2p	**0.014197**	0.000000	0.090998	0.000210	(+)0.167883	0.002949	(+)0.245363	0.007077
GASO2l	**0.014197**	0.000000	0.090998	0.000210	(+)0.167820	0.002905	(+) **0.245374**	0.006468
	#generations		*#individuals*		*#additions*		*#multiplications*	
Alg.	*A*	*SD*	*A*	*SD*	*A*	*SD*	*A*	*SD*
GACO	81.2	19.3	3,813.6	522.5	1.32e+07	1.81e+06	4.38e+07	6e+06
GASO1r	(-)93.9	14.3	3,712.5	408.0	(+)6.8e+06	7.11e+05	(+)3.28e+07	3.54e+06
GASO1p	85.2	17.0	3,599.1	457.5	(+)6.48e+06	8.17e+05	(+)3.16e+07	4.01e+06
GASO1l	(+)71.5	12.3	(+)3,360.0	374.0	(+)5.92e+06	6.47e+05	(+)2.92e+07	3.23e+06
GASO2r	76.2	11.8	3,648.0	435.0	(+)7.15e+06	8.24e+05	(+)3.33e+07	3.92e+06
GASO2p	(+)71.5	9.4	(+)3,557.4	322.4	(+)6.9e+06	6.03e+05	(+)3.23e+07	2.89e+06
GASO2l	(+)70.1	16.0	(+)3,557.0	465.7	(+)6.75e+06	8.7e+05	(+)3.19e+07	4.17e+06

TABLE XIX
RESULTS FOR EXPERIMENT 4 WITH 1/2 HEURISTIC INITIAL POPULATION OF SIZE 200

	*mass*1		*mass*10		*mass*25		*mass*50	
Alg.	*A*	*SD*	*A*	*SD*	*A*	*SD*	*A*	*SD*
GACO	0.013149	0.002623	0.083834	0.018484	0.154895	0.037618	0.223703	0.057487
GASO1r	**0.013748**	0.001795	**0.088059**	0.012652	**0.164611**	0.026017	**0.242705**	0.040638
GASO1p	0.013000	0.002771	0.082916	0.019247	0.153979	0.039729	0.225917	0.062266
GASO1l	0.013149	0.002623	0.083839	0.018486	0.155964	0.038024	0.229271	0.059499
GASO2r	0.013299	0.002456	0.084894	0.017312	0.158133	0.035610	0.232641	0.055659
GASO2p	0.012551	0.003131	0.079620	0.022069	0.147282	0.045401	0.215739	0.070999
GASO2l	0.013000	0.002771	0.082785	0.019531	0.153794	0.040174	0.225964	0.062916
	#generations		*#individuals*		*#additions*		*#multiplications*	
Alg.	*A*	*SD*	*A*	*SD*	*A*	*SD*	*A*	*SD*
GACO	88.1	25.8	3,350.3	729.0	1.16e+07	2.52e+06	3.84e+07	8.37e+06
GASO1r	90.1	21.1	(+)2,947.0	495.1	(+)5.56e+06	9.04e+05	(+)2.63e+07	4.38e+06
GASO1p	79.1	22.2	(+)2,718.4	556.7	(+)5.1e+06	1.01e+06	(+)2.42e+07	4.93e+06
GASO1l	76.1	22.8	(+)2,712.0	613.0	(+)5.01e+06	1.07e+06	(+)2.4e+07	5.32e+06
GASO2r	(+)74.5	17.3	(+)2,817.5	496.7	(+)5.62e+06	9.41e+05	(+)2.58e+07	4.48e+06
GASO2p	(+)67.5	17.3	(+)2,645.0	534.5	(+)5.23e+06	9.97e+05	(+)2.41e+07	4.78e+06
GASO2l	(+)65.3	19.3	(+)2,645.2	575.8	(+)5.13e+06	1.07e+06	(+)2.39e+07	5.15e+06

B. Multiple Test Diagrams

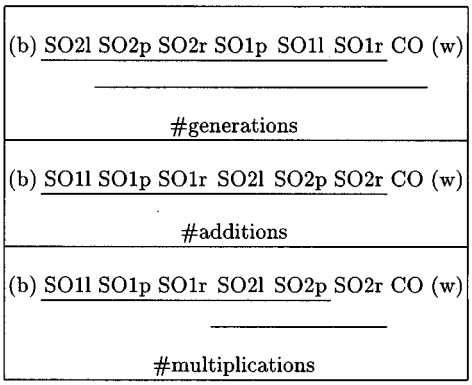

Fig. 12. Multiple test for Experiment 1 with random population of size 100.

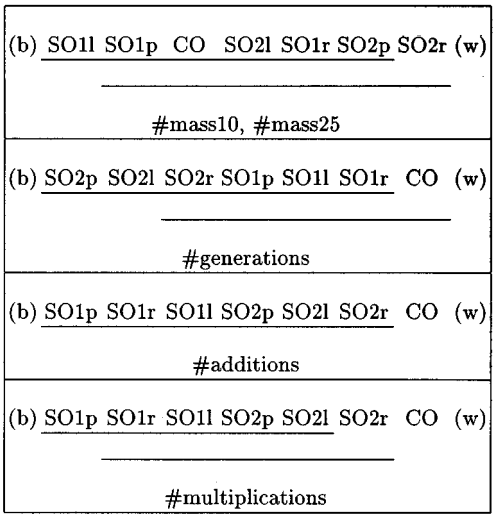

Fig. 13. Multiple test for Experiment 1 with 1/2 heuristic population of size 100.

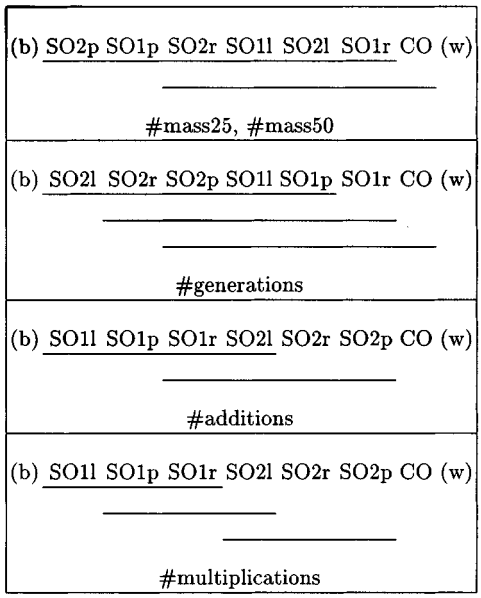

Fig. 14. Multiple test for Experiment 1 with random population of size 200.

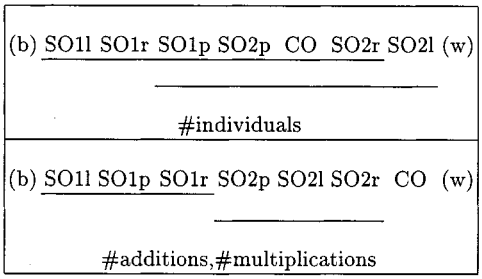

Fig. 15. Multiple test for Experiment 1 with 1/2 heuristic population of size 200.

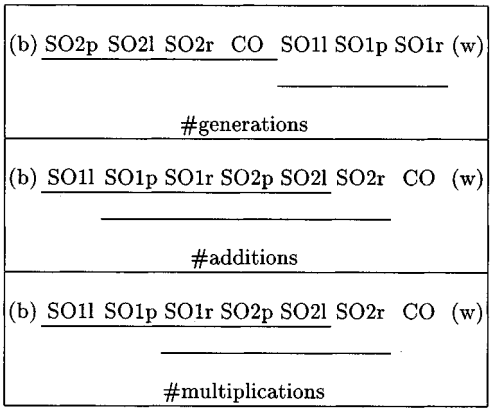

Fig. 16. Multiple test for Experiment 2 with random population of size 100.

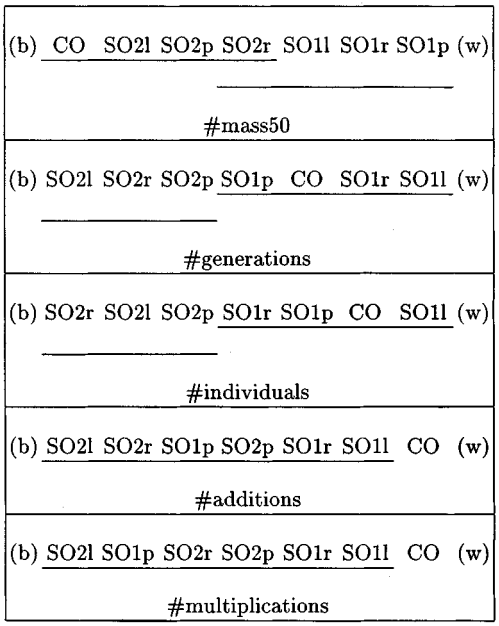

Fig. 17. Multiple test for Experiment 2 with 1/2 heuristic population of size 100.

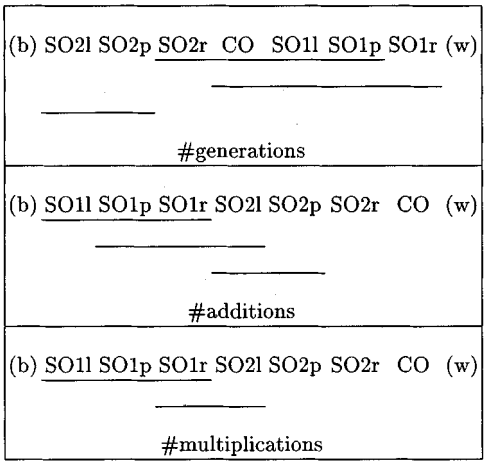

Fig. 18. Multiple test for Experiment 2 with random population of size 200.

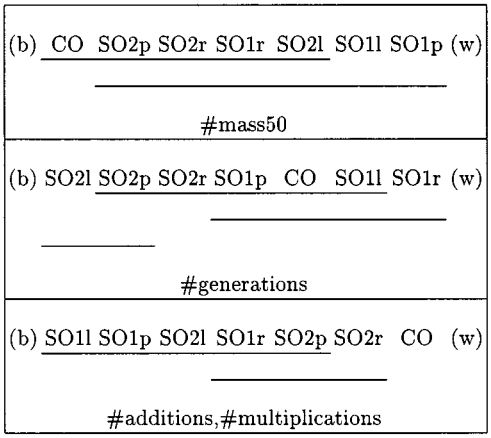

Fig. 19. Multiple test for Experiment 2 with 1/2 heuristic population of size 200.

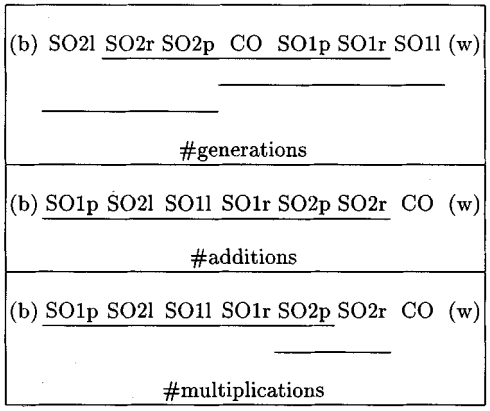

Fig. 20. Multiple test for Experiment 3 with random population of size 100.

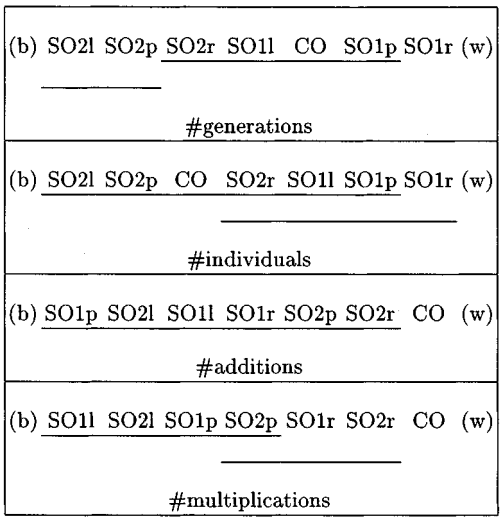

Fig. 21. Multiple test for Experiment 3 with 1/2 heuristic population of size 100.

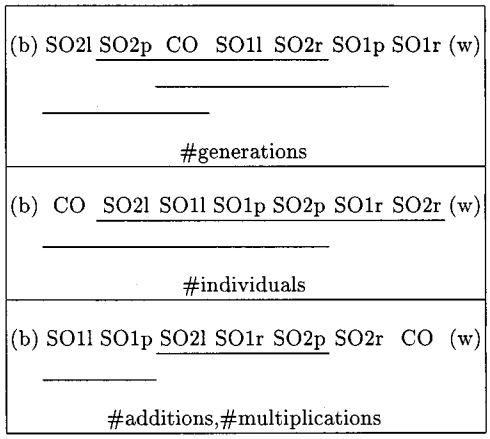

Fig. 22. Multiple test for Experiment 3 with random population of size 200.

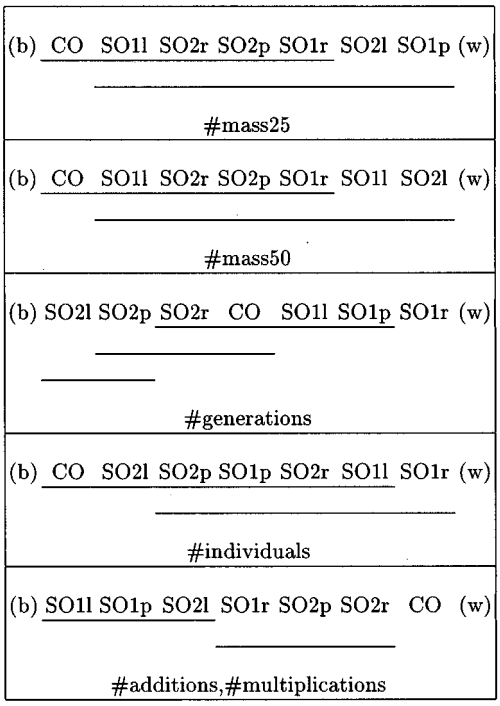

Fig. 23. Multiple test for Experiment 3 with 1/2 heuristic population of size 200.

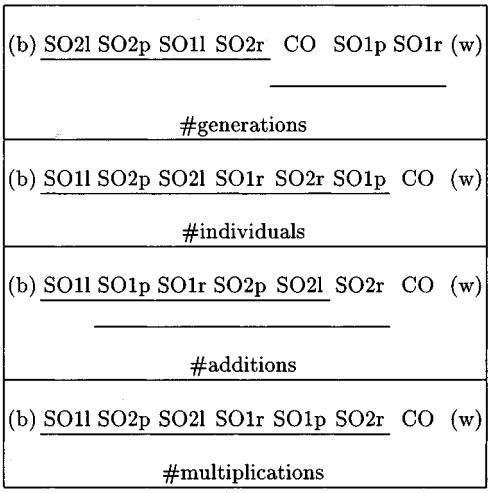

Fig. 24. Multiple test for Experiment 4 with random population of size 100.

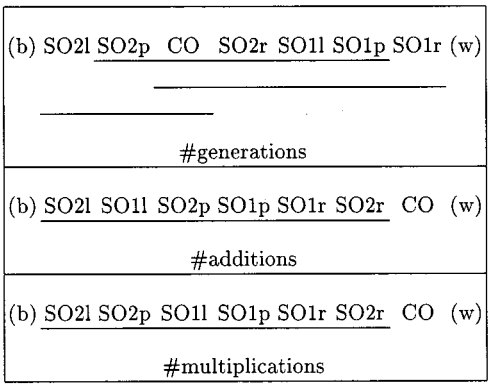

Fig. 25. Multiple test for Experiment 4 with 1/2 heuristic population of size 100.

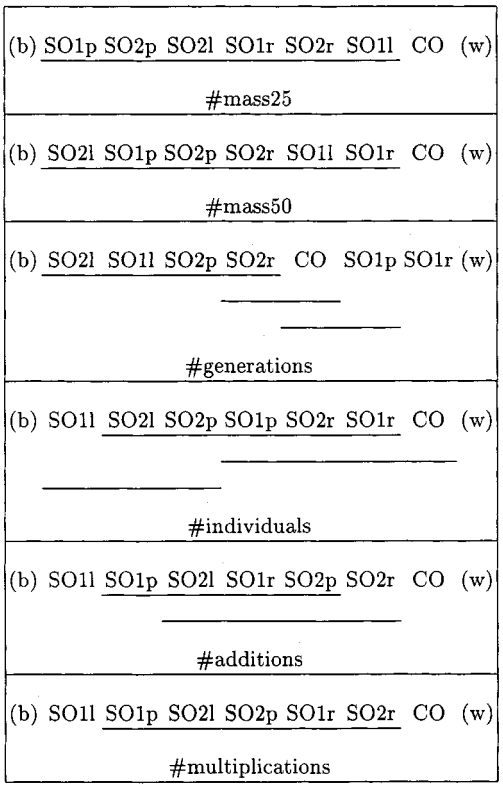

Fig. 26. Multiple test for Experiment 4 with random population of size 200.

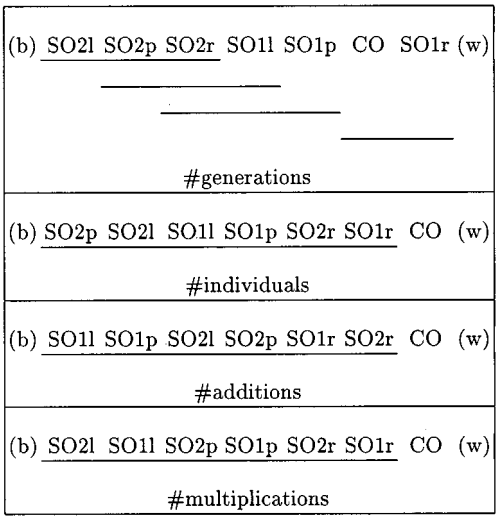

Fig. 27. Multiple test for Experiment 4 with 1/2 heuristic population of size 200.

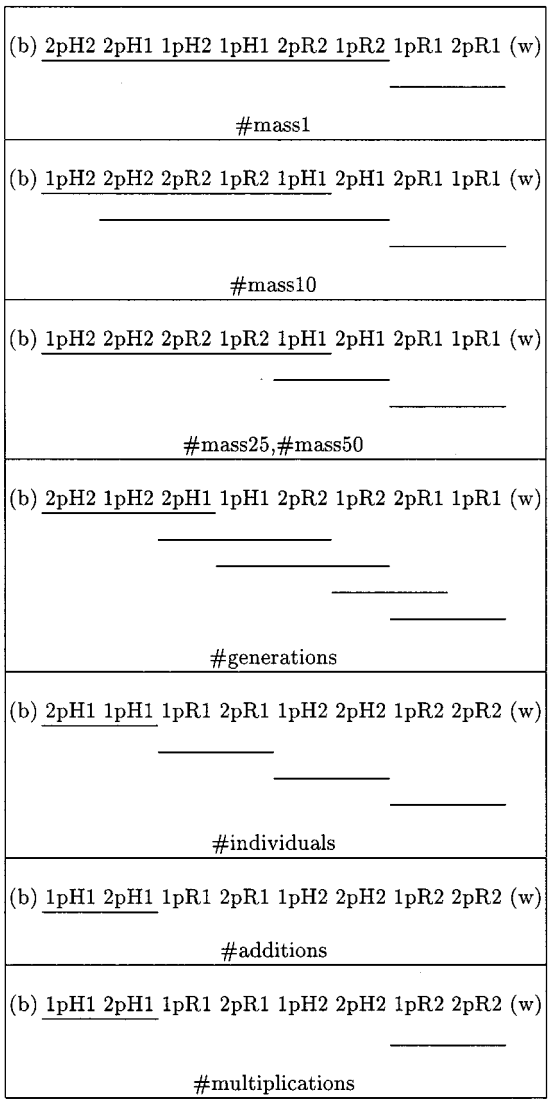

Fig. 28. Multiple test for GASO1p and GASO2p in Experiment 1.

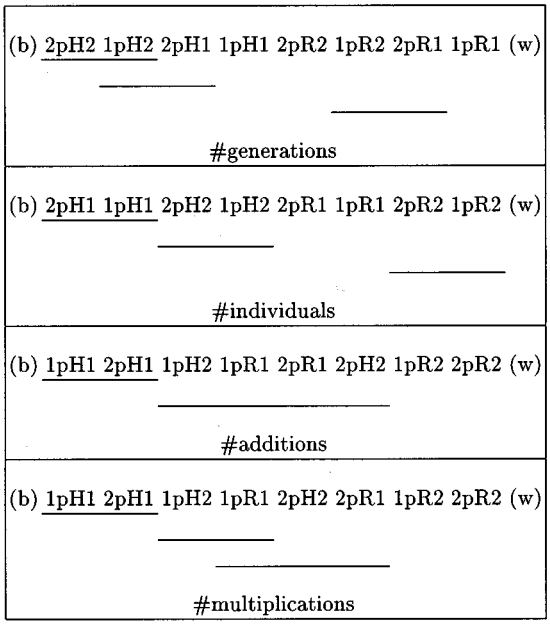

Fig. 29. Multiple test for GASO1p and GASO2p in Experiment 2.

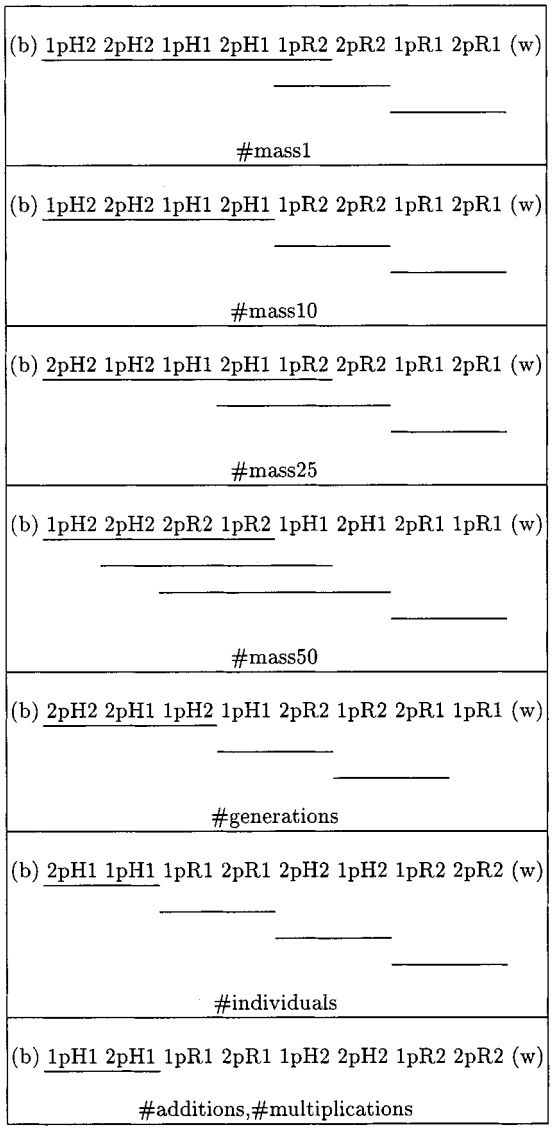

Fig. 30. Multiple test for GASO1p and GASO2p in Experiment 3.

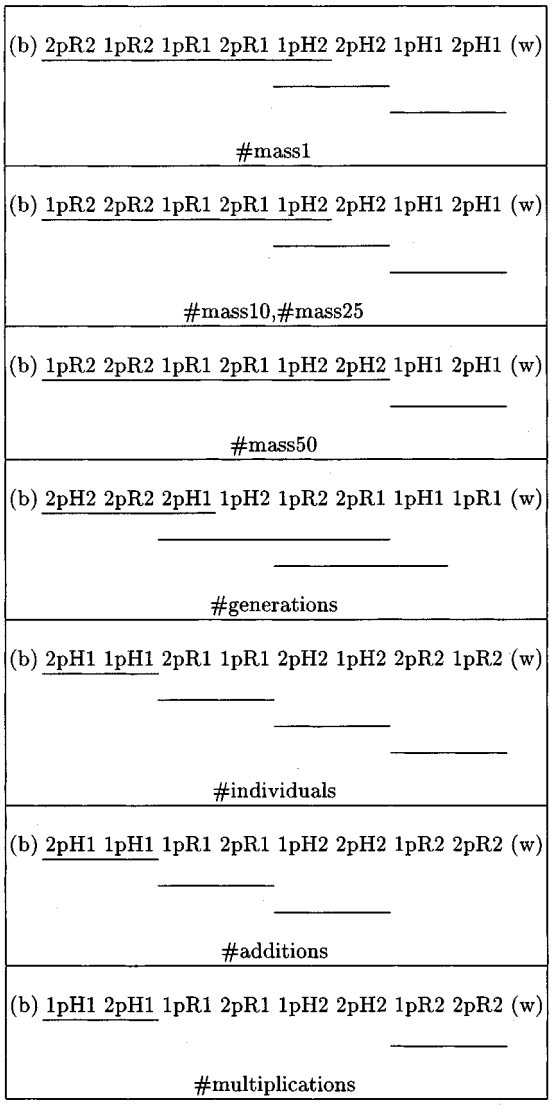

Fig. 31. Multiple test for GASO1p and GASO2p in Experiment 4.

ACKNOWLEDGMENT

The authors would like to thank D. B. Fogel and the anonymous reviewers for their careful reading of this paper and for their constructive and valuable comments, which have helped to improve the quality and the presentation of the paper.

REFERENCES

[1] J. Pearl, *Probabilistic Reasoning in Intelligent Systems*. San Mateo, CA: Morgan Kaufmann, 1988.
[2] E. Castillo, J. M. Gutiérrez, and A. S. Hadi, "Expert systems and probabilistic network models," in *Monographs in Computer Science*, New York: Springer-Verlag, 1997.
[3] Y. Peng and J. A. Reggia, "A probabilistic causal model for diagnostic problem solving. Part one," *IEEE Trans. Syst., Man, Cybern.*, vol. SMC-17, pp. 146–162, Mar. 1987.
[4] ——, "A probabilistic causal model for diagnostic problem solving. Part one," *IEEE Trans. Syst., Man, Cybern.*, vol. SMC-17, pp. 395–406, May 1987.
[5] E. S. Gelsema, "Diagnostic reasoning based on a genetic algorithm operating in a Bayesian belief network," *Pattern Recognit. Lett.*, vol. 17, pp. 1047–1055, Sept. 1996.
[6] E. Charniak and E. McDermott, *Introduction to Artificial Intelligence*. Reading, MA: Addison-Wesley, 1985.
[7] M. E. Stickel, "A Prolog-like inference system for computing minimum-cost abductive explanations in natural language interpretation," AI Center, SRI International, Menlo Park, CA, Tech. Rep. 451, 1988.
[8] U. P. Kumar and U. B. Desai, "Image interpretation using Bayesian networks," *IEEE Trans. Pattern Anal. Machine Intell.*, vol. 18, pp. 74–78, Jan. 1996.
[9] P. Thagard, "Explanatory coherence," *Behav. Brain Sci.*, vol. 12, no. 3, pp. 435–467, 1989.
[10] D. E. Appelt and M. Pollack, "Weighted abduction for plan ascription," *User Model. User-Adapted Interaction*, vol. 2, no. 1–2, 1991.
[11] H. Kautz and J. Allen, "Generalized plan recognition," in *Proc. Nat. Conf. Artificial Intelligence*, Aug. 1986, pp. 32–37.
[12] D. Poole and K. Kanazawa, "A decision-theoretic abductive basis for planning," in *Proc. AAAI Spring Symp. Decision-Theoretic Planning*, Mar. 1994, pp. 232–239.
[13] P. O'Rorke, S. Morris, and D. Schulenberg, "Theory formation by abduction: Initial results of a case study based on the chemical revolution," Dept. Info. Comput. Sci., Univ. California, Irvine, CA, Tech. Rep. ICS-TR-89-25, 1989.
[14] H. E. Pople, "On the mechanization of abductive logic," in *Proc. 3rd Int. Joint Conf. Artificial Intelligence*, 1973, pp. 147–152.
[15] S.E. Shimony, "Finding maps for belief networks is NP-hard," *Artif. Intell.*, vol. 68, no. 2, pp. 399–410, July 1994.

[16] L. M. d. Campos, J. A. Gámez, and S. Moral, "Partial abductive inference in Bayesian belief networks using a genetic algorithm," *Pattern Recognit. Lett.*, vol. 20, no. 11-13, pp. 1211–1217, 1999.
[17] T. Bäck, U. Hammel, and H.-P. Schwefel, "Evolutionary computation: Comments on the history and current state," *IEEE Trans. Evol. Comput.*, vol. 1, pp. 3–17, Apr. 1997.
[18] J. Pearl, "Distributed revision of composite beliefs," *Artif. Intell.*, vol. 33, no. 2, pp. 173–215, Oct. 1987.
[19] B. K. Sy, "Reasoning MPE to multiply connected belief networks using message passing," in *Proc. 11th Natl. Conf. AI*, July 1992, pp. 570–576.
[20] A. P. Dawid, "Applications of a general propagation algorithm for probabilistic expert systems," *Statistics Comput.*, vol. 2, pp. 25–36, 1992.
[21] D. Nilsson, "An algorithm for finding the most probable configurations of discrete variables that are specified in probabilistic expert systems," M.Sc. thesis, Univ. Copenhagen, Copenhagen, Denmark, 1994.
[22] ——, "An efficient algorithm for finding the M most probable configurations in Bayesian networks," *Statistics Comput.*, vol. 8, no. 2, pp. 159–173, June 1998.
[23] B. Seroussi and J. L. Goldmard, "An algorithm directly finding the K most probable configurations in Bayesian networks," *Int. J. Approx. Reason.*, vol. 11, no. 3, pp. 205–233, Oct. 1994.
[24] R. E. Neapolitan, *Probabilistic Reasoning in Expert Systems. Theory and Algorithms*, New York: Wiley, 1990.
[25] L. M. d. Campos, J. A. Gámez, and S. Moral, "On the problem of performing exact partial abductive inference in Bayesian belief networks using junction trees," in *Proc. 8th Int. Conf. Information Processing and Management of Uncertainty in Knowledge-Based Systems*, July 2000, pp. 1270–1277.
[26] P. Larrañaga, C. M. Kuijpers, M. Poza, and R. H. Murga, "Decomposing Bayesian networks: Triangulation of the moral graph with genetic algorithms," *Statistics Comput.*, vol. 7, no. 1, pp. 19–34, 1997.
[27] A. Cano and S. Moral, "A genetic algorithm to approximate convex sets of probabilities," in *Proc. 6th Int. Conf. Information Processing and Management of Uncertainty in Knowledge-Based Systems*, 1996, pp. 847–852.
[28] P. Larrañaga, C. Kuijpers, and R. Murga, "Learning Bayesian network structures by searching for the best ordering with genetic algorithms," *IEEE Trans. Syst., Man, Cybern.*, vol. 26, pp. 487–493, July 1996.
[29] L. M. de Campos and J. F. Huete, "Approximating causal orderings for Bayesian networks using genetic algorithms and simulated annealing," in *Proc. 8th Int. Conf. Information Processing and Management of Uncertainty in Knowledge-Based Systems*, July 2000, pp. 333–340.
[30] P. Larrañaga, M. Poza, Y. Yurramendi, R. Murga, and C. Kuijpers, "Structure learning of Bayesian networks by genetic algorithms. A perfomance analysis of control parameters," *IEEE Trans. Pattern Anal. Machine Intell.*, vol. 18, pp. 912–926, Sept. 1996.
[31] C. Rojas-Guzman and M. A. Kramer, "Galgo: A genetic algorithm decision support tool for complex uncertain systems modeled with Bayesian belief networks," in *Proceedings of the 9th Conference on Uncertainty in Artificial Intelligence*. San Mateo, CA: Morgan Kauffman, 1993, pp. 368–375.
[32] ——, "An evolutionary computing approach to probabilistic reasoning in Bayesian networks," *Evol. Comput.*, vol. 4, no. 1, pp. 57–85, Spring 1996.
[33] E. S. Gelsema, "Abductive reasoning in Bayesian belief networks using a genetic algorithm," *Pattern Recognit. Lett.*, vol. 16, no. 8, pp. 865–871, Aug. 1995.
[34] M. Henrion, "Propagating uncertainty in Bayesian networks by probabilistic logic sampling," in *Uncertainty in Artificial Intelligence 2*, J. Lemmer and L. Kanal, Eds. Amsterdam, The Netherlands: North-Holland, 1988, pp. 149–263.
[35] Z. Michalewicz, *Genetic Algorithms + Data Structures = Evolution Programs*, 3rd ed. New York: Springer-Verlag, 1996.
[36] J. Cheng and M. J. Druzdzel, "Computational investigation of low-discrepancy sequences in simulation algorithms for Bayesian networks," in *Uncertainty in Artificial Intelligence: Proceedings of the Sixteenth Conference (UAI-2000)*. San Mateo, CA: Morgan Kaufmann, 2000, pp. 72–81.
[37] N. Friedman and D. Koller, "Being Bayesian about network structure," in *Uncertainty in Artificial Intelligence: Proceedings of the Sixteenth Conference (UAI-2000)*. San Mateo, CA: Morgan Kaufmann, 2000, pp. 201–210.
[38] I. A. Beinlich, 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," in *Proceedings of the Second European Conference on Artificial Intelligence in Medicine*. New York: Springer-Verlag, 1989, pp. 247–256.
[39] A. Cano, S. Moral, and A. Salmerón, "Penniless propagation in join trees," *Int. J. Intell. Syst.*, vol. 15, no. 11, pp. 1027–1059, Nov. 2000.
[40] J. H. Zar, *Biostatistical Analysis*, 4th ed. Englewood Cliffs, NJ: Pretice-Hall, 1999.
[41] P. J. M. V. Laarhoven and E. H. L. Aarts, *Simulated Annealing*. Amsterdam, The Netherlands: Reidel, 1988.
[42] F. Glover, "Tabu-Search: Part I," *ORSA J. Comput.*, vol. 1, no. 3, pp. 190–206, 1989.
[43] ——, "Tabu Search: Part II," *ORSA J. Comput.*, vol. 2, no. 1, pp. 4–32, 1990.
[44] F. V. Jensen, *An Introduction to Bayesian Networks*. London, U.K.: UCL Press, 1996.
[45] F. V. Jensen, S. L. Lauritzen, and K. G. Olesen, "Bayesian updating in causal probabilistic networks by local computation," *Comput. Statistics Quart.*, vol. 4, pp. 269–282, 1990.
[46] S. L. Lauritzen and D. J. Spiegelhalter, "Local computations with probabilities on graphical structures and their application to expert systems," *J. R. Statistical Soc. Ser. B*, vol. 50, no. 2, pp. 157–224, 1988.
[47] R. D. Shachter, B. D. D'Ambrosio, and B. D. D. Favero, "Symbolic probabilistic inference in belief networks," in *Proc. 8th Nat. Conf. Artificial Intelligence*, July 1990, pp. 126–131.
[48] P. P. Shenoy and G. R. Shafer, "Axioms for probability and belief-function propagation," in *Uncertainty in Artificial Intelligence*, R. Shachter, T. Levitt, L. Kanal, and J. Lemmer, Eds. Amsterdam, The Netherlands: North-Holland, 1990, pp. 169–198.
[49] G. F. Cooper, "Probabilistic inference using belief networks is NP-hard," *Artif. Intell.*, pp. 393–405, 1990.
[50] F. V. Jensen, K. G. Olesen, and S. K. Andersen, "An algebra of Bayesian belief universes for knowledge based systems," *Networks*, vol. 20, pp. 637–659, 1990.
[51] W. X. Wen, "Optimal decomposition of belief networks," in *Uncertainty in Artificial Intelligence 6*, P. Bonnissone, M. Henrion, L. Kanal, and J. Lemmer, Eds. Amsterdam, The Netherlands: North-Holland, 1990, pp. 209–224.

Luis M. de Campos was born in in 1961. He received the M.S. degree in mathematics and the Ph.D. degree from the University of Granada, Spain, in 1984 and 1988, respectively.

He is currently an Associate Professor of Computer Science with the University of Granada, Spain. His current research interests Numerical Representations of Uncertainty, Probabilistic Graphical Models, Machine Learning, and Information Retrieval.

José A. Gámez received the M.S. and Ph.D. degrees in computer science from the University of Granada, Spain, in 1991 and 1998, respectively.

He is currently an Assistant Professor with the Department of Computer Science at the Universitiy of Castilla-La Mancha, Spain. His current research interests include probabilistic reasoning, Bayesian networks, and the application of evolutionary computation to these topics and machine learning.

Serafín Moral received the Ph.D. degree from the University of Granada, Spain, in 1984.

He is a Professor of Computer Science and Artificial Intelligence and a Member of the Research Group on Uncertainty Treatment on Artificial Intelligence at the University of Granada. His current research interests include imprecise probabilities, propagation algorithms in dependence graphs, and uncertain and defeasible reasoning in general.