Factorized Distribution Algorithms For Functions With Unitation Constraints

Roberto Santana
Institute of Cybernetics,
Mathematics and Physics.
Calle 15, e/C y D, Vedado,
C-Habana, Cuba
rsantana@cidet.icmf.inf.cu

Alberto Ochoa
Institute of Cybernetics,
Mathematics and Physics.
Calle 15, e/C y D, Vedado,
C-Habana, Cuba
ochoa@cidet.icmf.inf.cu

Marta R. Soto
Institute of Cybernetics,
Mathematics and Physics.
Calle 15, e/C y D, Vedado,
C-Habana, Cuba
mrosa@cidet.icmf.inf.cu

Abstract

In this paper we present a Factorized Distribution Algorithm for the optimization of constrained problems where the constraints are expressed in terms of the unitation values of the function. The algorithm uses information about the structure of the problem to conduct the search in the space of feasible solutions. Thus, we present a number of ways FDAs can incorporate application domain knowledge into the search. In the paper we also illustrate, using empirical results, a number of hypotheses that could explain the behavior of FDAs for this class of constrained problems.

1 Introduction

The appearance of Estimation Distribution Algorithms (EDAs) [8], and their tractable subclass, the Factorized Distribution Algorithms (FDAs) [7], brought out solutions for some of the difficulties experienced by the Genetic Algorithms (GAs), like the linkage problem. FDAs can cope with these difficulties by using probability distributions instead of the traditional genetic operators. The success of FDAs in common optimization problems has set the path for considering its application to the constrained ones.

In constrained problems the search for the optimal solution is intricate due to the existence of infeasible points that impose additional restrictions on an efficient search process. Main constraint handling techniques in GAs include: the use of penalty functions that punishes the non feasible solutions, the application of genetic operators that guarantee no illegal solutions will be generated during the search, and the application of repairing methods.

There are few reports on the application of FDAs to constrained problems. In [7] it is shown that a FDA that uses a factorization based on the structure of an additive function can optimize this function in the presence of constraints, if the structure of the function is compatible with the structure of the constraints. This type of FDAs will accomplish the optimization of the function by generating only legal points. The Constraint Univariate Marginal Distribution Algorithm (CUMDA) was introduced in [15] as an alternative for the solution of constrained problems where the constraints are expressed in terms of the unitation values of the function (functions with unitation constraints). Like the Univariate Marginal Distribution Algorithm [6], CUMDA uses an univariate approximation of the joint probability distribution and it generates only legal solutions. For the problems under consideration, CUMDA was shown to be superior[15] to GAs that use heuristic genetic operators [14] to keep the search in the region of feasible solutions.

In [9] we show that a FDA that considers the fulfilling of constraints during the sampling step is able to handle certain class of constraints that do not have to be compatible with the structure of the function. No experimental results were shown.

Our objective in this paper is threefold: To introduce the Constraint Factorized Distribution Algorithm (CFDA) for the optimization of problems with unitation constraints; to present a number of ways FDAs can incorporate application domain knowledge into the search and to advance a number of hypotheses that could explain the behavior of FDAs for this class of problem.

The outline of the paper is as follows. In section 2 we introduce the CFDA. In section 3 we present a set of hypotheses that try to explain the behavior of the CFDAs. These hypotheses are illustrated using empirical data. Section 4 describes two theoretical problems that are treated using the CFDA, we expose how previous information can be used to determine the factorizations and preliminary experimental results are shown. Finally the conclusions of our work are presented in section 5.

2 Constraint FDAs

2.1 Notation

$\widetilde{X} = \{x_1, ..., x_n\}$, $\mathbf{B} = \{0, 1\}$ $X := \mathbf{B}^{|\widetilde{X}|}$, $x \in X$ is a binary vector. $P = \{x^1, ..., x^N\}$ is a set of binary vectors or population of size N. We define the unitation function $u : X \rightarrow N$, $u(x) = \sum_{i=1}^{n} x_i$. $u(x)$ is the unitation value of x. A unitation set Ω is a set of unitation values, $\Omega = \{u_1, u_2, ..., u_n\}$.

A function of unitation is a function whose value depends only on the number of ones in an input string. The

function values of the strings with the same number of ones are equal. Deceptive functions are defined as a sum of more elementary deceptive functions f_k of k variables.

$$f(x) = \sum_{j=1}^{l} f_k(s_j), \quad (1)$$

where s_j are non-overlapping substrings of x containing k elements.

Throughout this paper, when we refer to deceptive functions we mean the class of deceptive functions whose elementary deceptive functions are also functions of unitation.

The class of constrained problems we address in this paper is the optimization of a function $f(x)$ where candidate solutions x are forced to satisfy the constraint $u(x) \in \Omega_{problem}$. $\Omega_{problem}$ is an initial set of unitation values.

2.2 CFDA

The strategy used by the Constraint FDA (CFDA) was partially introduced in [9]. No experimental results were presented. The CFDA addresses the same class of problems that CUMDA and it is based on the FDA introduced in [7]. This algorithm was applied to additively decomposed functions for which, using the running intersection property [5], a factorization of the probability based on residuals (b_i) and separators (c_i) can be obtained. More formally:

Given a set of sets of variables $S = s_1, \ldots, s_l$, for $i = 1, 2, \ldots, l$ the sets d_i, b_i, and c_i are defined as:

$$d_i = \cup_{j=1}^{i} s_j, \qquad b_i = s_i \backslash d_{i-1}, \; c_i = s_i \cap d_{i-1} \quad (2)$$

the s_i are sorted following the order imposed by the junction tree in the generation of points, s_1 corresponds to the root of the tree, d_0 is set to 0.

CFDA also employs a junction tree based on a given factorization of the problem. Additionally, for each vector to be generated using the CFDA, the assignments of values for the subset of variables X_{b_i} have to fulfill the following two conditions:

1) $u(X_{s_i}) \leq b$.

2) $u(X_{s_i}) + (n - |s_i|) \geq a$.

The conditions establish that after generating variables X_{b_i}, the accumulated value of unitation $u(X_{s_i})$ should not exceed the constraint b, and it must still be possible for the vector X to reach the constrained value a.

To enable the CFDA to deal with these constraints only the generation step was changed. The CFDA restricts the set of possible assignments of X_{b_i} according to their unitation values. The marginal probabilities of those assignments that violate the constraints are redistributed among the 'feasible' ones, proportionally to their own marginal probabilities. In this way, the relative proportions of marginal probabilities corresponding to 'legal' configurations are not altered, only their absolute values are changed. Figure 1 shows the pseudocode of the CFDA.

CFDA

- **STEP 0:** Set $t \Leftarrow 0$. Generate $N \gg 0$ points randomly.
- **STEP 1:** Selection of promising points.
- **STEP 2:** Compute the conditional probabilities $p^s(\Pi_{b_i} x | \Pi_{c_i} x, t)$.
- **STEP 3:** For each individual to be generated:

  For each factor i of the junction tree

  a) Identify the set of 'feasible' configurations of X_{b_i}, those that fulfill

  $u(X_{s_i}) \leq b$ and $u(X_{s_i}) + (n - |s_i|) \geq a$.

  b) Redistribute the probabilities of the infeasible configurations among the feasible ones, proportionally to the probabilities of the latter.

  c) Assign to X_{b_i} a 'feasible' configuration sampled from the new probabilities.
- **STEP 4:** If the termination criteria are met, FINISH.
- **STEP 5:** Add the best point of the previous generation to the generated points.
- **STEP 6:** Set $t \Leftarrow t + 1$. Go to STEP 1.

Figure 1: Constraint Factorized Distribution Algorithm

From a different perspective the sampling step as it is implemented in the CFDA, can be seen as a process where constraints are dynamically imposed on each factor based on the previous assignments to variables, and on the necessity of enforcing consistency with future assignments. The fundamental difference with constraint propagation in constraint networks [3] is that the probabilistic table of the junction tree allows a non deterministic and biased selection of the configurations for each factor.

3 Dependencies in constrained domains

There are a number of implicit assumptions in the way the CFDA has been conceived. In this section we present and illustrate these assumptions. We will work on a theoretical framework where FDAs are analyzed in terms of the multivariate probabilities determined by its components.

Let $P^g(X, t)$ denote the multivariate joint distribu-

tion of X at the generation t. $P^s(X,t)$ denotes the probability after selection. $P^a(X,t)$ is the factorized approximation given by the model chosen to approximate $P^s(X,t)$. These multivariate probabilities are analyzed using a number of statistical measures that are employed in order to detect and quantify the interactions between variables, and to measure the distance between two different distributions.

These measures are: The mutual information, and the Kullback-Liebler measure of divergence.

Let $p_{i,j}(x_i, x_j)$ denote the bivariate marginals corresponding to binary variables x_i and x_j and let $I(x_i, x_j)$ denote the mutual information between variables x_i and x_j.

$$I(x_i, x_j) = \sum_{x_i, x_j} p_{i,j}(x_i, x_j) \cdot \log \frac{p_{i,j}(x_i, x_j)}{p_i(x_i) \cdot p_j(xj)} \qquad (3)$$

We use a common distance metric between probabilistic models. This metric is the Kullback-Liebler divergence $D(r||q)$.

$$D(r||q) = \sum\nolimits_i r(x_i) \cdot \log \frac{r(x_i)}{q(x_i)} \qquad (4)$$

This divergence is always non negative with equality only in the case when both distributions are identical.

3.1 Dependencies in the selected set

Our first assumption is:

- Dependencies among variables related by the function structure can be preserved in the regions determined by the constraints.

We illustrate this assumption by showing how the dependencies among variables evolve when proportional selection is used for a given function. This analysis is done for different unitation regions. In general we are interested in investigating how bivariate interactions change for the class of non-overlapping unitation functions, with constraints enforced by the values of unitation of the solutions.

The function used in our simulations was the following:

Function $f_{3deceptive}$:

$$f_{3deceptive}(X) = \sum_{i=1}^{i=4} f^3_{dec}(X_{3i-2}, X_{3i-1}, X_{3i}) \qquad (5)$$

where the fitness contribution from blocks with unitation u are denoted as α_u. For f^3_{dec} these parameters are $\alpha_0 = 0.9$, $\alpha_1 = 0.8$, $\alpha_2 = 0$, $\alpha_3 = 1$. As the parameters of the function $f_{3deceptive}$ we have $n = 12$, $k = 3$.

If we concentrate on pairwise interactions it is easy to realize that in non overlapping deceptive functions there exist only two types of interactions: Interactions between pairs of variables that belong to different definition sets, and interactions between pairs of variables that belong to the same definition set.

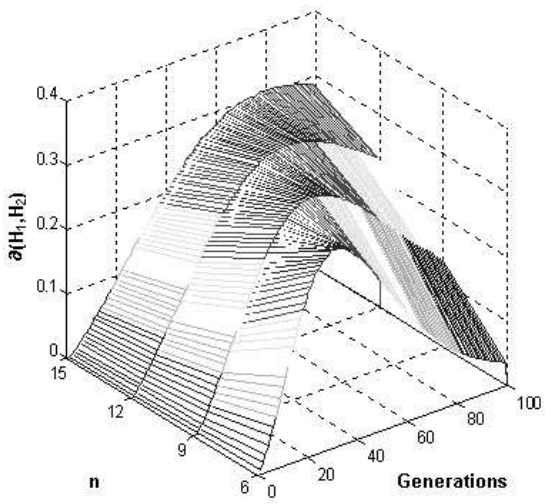

Figure 2: Difference between the mutual information of the two classes for the regions determined by the unitation constraints.

In order to compare the interactions we will calculate the mutual information between pairs of variables that belong to both classes. The two classes of pairwise interactions between variables ($class \in \{1, 2\}$) are: $class = 1$, when variables belong to the same definition set; $class = 2$, when they are located in different definition sets. $H_{class}(x_i, x_j)$ denotes a schema where variables (x_i, x_j) belong to the class $class$, their values are fixed, and the rest of the variables are free.

We start by calculating the fitness contribution of the schema $H_{class}(x_i, x_j)$ $(f(H_{class}(x_i, x_j)))$ for the four possible values of $\{x_i, x_j\}$. For the proportional selection the probability of the schema $H_{class}(x_i, x_j)$ after selection can be calculated. It is also possible to calculate the fitness contribution and the corresponding bivariate probability of selection for each class in the constrained regions defined by a set of unitation values Ω. The mutual information is calculated using these probabilities.

Figure 2 shows the difference in the mutual information between the two classes i.e. $\partial(H_1, H_2) = MI(H_1(x_i, x_j)) - MI(H_2(x_i, x_j))$ for the regions Ω_u defined as $\Omega_u = [u, 12]$, where $u \in [0, 11]$. The optimum is contained in all the 11 regions. $\partial(H_1, H_2)$ must be positive if the interactions between variables that belong to the same definition set are stronger than interactions between variables that belong to different unitation sets. In the figure, $\partial(H_1, H_2)$ is always positive, furthermore, for the experiments conducted

$MI(H_1(x_i, x_j)) >> MI(H_2(x_i, x_j))$.

What this example and the experiments tell us is that there are functions for which the correspondence between the structure of the function and the interactions among variables holds when only feasible points are considered. This fact is important because FDAs that use up to second order dependencies construct a probabilistic model where variables with the strongest pairwise dependencies are joined with an edge in the corresponding graphical model. One approach we intend to follow in the future is to characterize a subclass of deceptive functions, for which there exists a correspondence between the structure of the function and the kind of interactions that arise among variables in the regions determined by the unitation constraints.

3.2 Approximation operators

The second assumption states that:

- Factorizations based on the function structure can be used to approximate the joint probability distribution of selected points, even if only the feasible region of the search space is considered.

We present an example using the $f_{3deceptive}$ function with the same parameters previously used. We calculate 3 approximations of $P^s(x, 0)$ determined by different factorizations, in different constrained regions defined by the values of unitation. To measure how good is the approximation we will evaluate the Kullback-Liebler distance between $P^s(X, 0)$ and its different approximations in the region of feasible points.

We develop our analysis using the Boltzman selection and the set of all feasible points. Given an additive function like the $f_{3deceptive}$, if the points of the initial population follows a Boltzman Distribution of base u, then for the Boltzman EDA (BEDA) the distribution of points after selection at generation t is given by [7]

$$P^s(x,t) = \frac{w^{f(x)}}{\sum_{j=1}^{2^n} w^{f(x_j)}}, \; w = u.v^t \tag{6}$$

Let $P_F^S(X,t)$ and $P_I^S(X,t)$ denote the probability of selecting points that respectively belong to the feasible and infeasible regions. In principle we would like $P_I^S(X,t) = 0$. Nevertheless, there exists a contradiction between this assumption and the way the Boltzman distribution is defined. The Boltzman distribution is strictly positive for every point. To solve this problem we associate a positive, although very small, value to $P_I^S(X,t)$ (i.e. $P_I^S(X,t) \approx 0$). This assignment of probabilities is equivalent to assigning a negative fitness value to the infeasible points.

Three approximation operators will be considered.

In the UMDA [6] the univariate marginal frequencies for the selected set $p_i(x_i)$ are calculated first. The distribution of points in the selected set is estimated as

$$p_U^a(x_1, ..., x_n) = \prod_{i=1}^{n} p_i^s(x_i) \tag{7}$$

Baluja and Davies [2] present an optimization algorithm that uses a probabilistic model defined on trees. In each generation the algorithm searches the best probability distribution T that is conformal with $P^s(x,t)$. The probability distribution P_T^a that is conformal with a tree model can be defined as:

$$P_T^a(x_1, ..., x_n) = \prod_{i=1}^{n} p(_{x_i|pa(x_i)}) \tag{8}$$

where the parent of the vertex corresponding to variable x_i is represented as $pa(x_i)$, the root vertex has no parent.

The FDA approximation is done in the following way:

$$p_F^a(x_1, ..., x_n) = \prod_{i=1}^{n} p_i^s(x_{b_i}|x_{c_i}) \tag{9}$$

In our simulation the factors of P_F^a were the definition sets of the $f_{3deceptive}$ function. The parameters of P_U^a, P_T^a and P_F^a were calculated straight from $P_I^S(X, 0)$. Figure 3 shows the Kullback-Liebler divergence between the $P_I^S(X, 0)$ and the three factorizations for the regions Ω_u defined as $\Omega_u = [u, 12]$, where $u \in [0, 11]$.

It can be seen that for Ω_0, when the whole search space is considered, the approximation of P_F^a is almost perfect. This fact is consistent with the theory. As the feasible region contracts the divergence increases, up to the region Ω_8 the P_F^a approximation comes closest. However, after that point the situation is reversed.

The example tells us that our hypothesis is not always valid, whether its validity is linked to the dimension of the constrained search space is still an open question.

3.3 Sampling

The third hypothesis is:

- Sampling algorithms can take advantage of the dependency relationships contained in the probability models while simultaneously keeping the search constrained to the space of feasible solutions.

Even if a probabilistic model is calculated from a set of feasible solutions, during the sampling there exists the possibility of generating points from the infeasible region. The main question of sampling in constrained domains is how to distribute the probability corresponding to infeasible points among the feasible ones.

There are a number of alternatives that can be tried by algorithms that generate only legal solutions. The

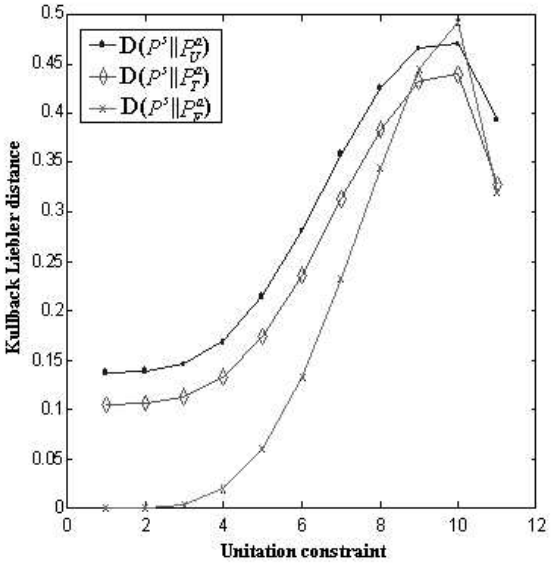

Figure 3: Kullback-Liebler divergence between the probability after selection and three different factorizations.

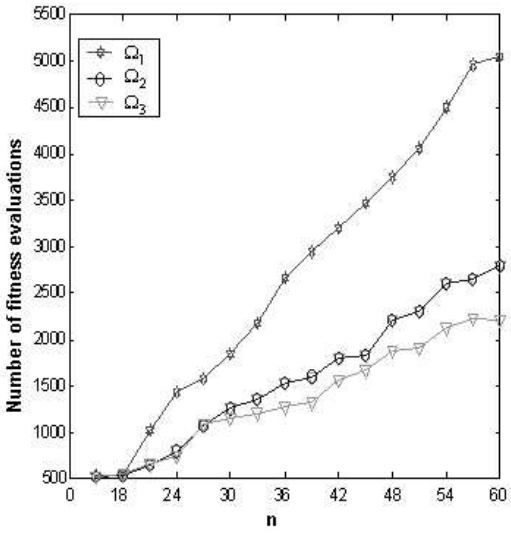

Figure 4: Results of the CFDA for the $f_{3deceptive}$ function in three different unitation regions.

probability of the whole space of infeasible solutions can be uniformly distributed among the feasible points. The probability of infeasible points can be also distributed among the feasible points, associating to each feasible point an additional probability proportional to its current (feasible) probability of being selected. Finally in a distribution according to the distance between the feasible and infeasible region, feasible points that are close in a genotypical distance measure would be favored in the distribution of the infeasible probabilities.

We have considered all these sampling strategies in the framework of the CFDA. Maybe the simplest strategy is to sample from the probabilistic model without regard to the feasibility of the solutions, and repairing the infeasible solutions later. A way to repair infeasible solutions, those for which $u(x) \notin [u, n]$, is to randomly select a value v such that $u(x) + v \geq u$, and randomly select v variables to be set among the $n - u(x)$ variables with value 0. The probability of a feasible solution x of being generated ($P^r(x)$) with repairing will be:

$$P^r(x) = P^a(x) + \sum_y t(y, x) \cdot P^r(y) \qquad (10)$$

where $t(x, y)$ is the probability of making a transition from the infeasible solution y to the feasible x.

Nevertheless, although this is a very simple strategy , the distribution from which v is sampled from is critical to the algorithm, and it determines in fact the transition probabilities $t(y, x)$. Using information from the probability model for sampling v seems to be a better alternative than sampling v from an uniform distribution in $[u - u(x), n - u(x)]$. This is the path we have followed in the design of a sampling strategy for the CFDA.

4 Experimental results

In our experiments we used a set of trial functions comprised from the $f_{3deceptive}$ function and two real problems. The experiments were designed in order to show the behavior of CFDA for these functions. The first experiment was done for the $f_{3deceptive}$ function. The scalability of the CFDA is investigated for this function for $3 \leq n \leq 60$. Three different regions of unitation were considered: $\Omega_1 = [0, n]$, $\Omega_2 = [\frac{n}{3}, n]$ and $\Omega_3 = [\frac{2n}{3}, n]$.

For all problems, the average number of fitness evaluations is calculated. We first determine the population size that allows the algorithm to converge in 90% of 50 trials. For this population size the total number of fitness evaluations is calculated. Parameters of the algorithms were: Truncation selection with $T = 10$. A maximum number of 20 generations.

In figure 4 we show results achieved in the optimization of the $f_{3deceptive}$ function. When only a fraction of the search space is searched the algorithm converges to the optimum more quickly. As expected, the optimum is found with less function evaluations as the dimension of the feasible region reduces. This continuous to be the case when the number of variables scales up.

4.1 3-SAT

The next test problem which we tested our algorithm on was the MAX-SAT, where the goal is to find a true assignment of n Boolean variables that maximizes the number of satisfied clauses of a given formula in a con-

Inst.	min	max	s	FDA		CFDA	
				succ.	gen.	succ.	gen.
434	5	15	3	88	5.20	96	5.95
434	5	15	6	89	6.51	100	5.06
623	5	15	3	95	4.85	93	6.77
623	5	15	6	91	6.36	91	6.23
836	5	15	3	72	7.85	77	8.53
836	5	15	6	55	8.36	90	7.84
7	4	16	3	98	4.36	100	4.08
7	4	16	6	96	4.48	100	3.54
9	4	16	3	53	8.07	58	8.83
9	4	16	6	36	8.19	65	7.09
34	4	16	3	100	4.68	100	3.82
34	4	16	6	100	3.54	100	3.58
984	4	18	3	77	7.74	87	7.23
984	4	18	6	69	7.35	93	7.13
986	2	16	3	97	5.95	87	7.75
986	2	16	6	99	4.86	94	5.38
997	2	16	3	95	7.05	98	6.55
997	2	16	6	87	7.4	98	6.7

Table 1: Comparison between the FDA and the UMDA for the 3-SAT problem

junctive normal form. We use a set of instances (uf20-91) available at http://www.satlib.org/benchm.html. The test-set is sampled from the phase transition region of uniform Random-3-SAT. Random-3-SAT instances sampled from this region tend to be particularly hard for both systematic SAT solvers and Stochastic Local Search (SLS) algorithms. All the problems in uf20-91 are satisfiable, each 3-CNF formula has 20 variables and 91 clauses. We selected 9 instances from the 1000 available.

In our experiments we concentrate our analysis on two aspects related with the behavior of the CFDAs for 3-SAT. How to find an appropriate factorization to the problem and how to incorporate knowledge about the problem by means of unitation constraints.

From the set of clauses a dependency graph is constructed, where two variables are connected by an edge if they appear together in one of the clauses of the formula. In [13] we have shown that for problems defined on graphs it is possible to construct a convenient factorization that fulfills the running intersection property by removing edges from the graph. In that paper experiments were also presented where the success of the FDA was shown to be related to the number of edges extracted from the original dependency graph.

In order to find an appropriate factorization for the 3-SAT problem we have designed a heuristic algorithm that finds a factorization of a graph whose maximum clique has size equal or less than s. s is a parameter of the algorithm that restricts the size of the maximum clique contained in the factorization. The heuristic algorithm tries to remove as few edges as possible, satisfying the running intersection property. The size of the maximum clique in the factorization is a critical parameter for the FDA, population size requirements are exponential to the size of this clique. A factorization is obtained by applying this heuristic algorithm to the dependency graph constructed from the clauses.

The second step is to find the values of unitation.It is relatively easy to conclude that for every clause where all the literals are positive there must be at least one positive literal in the solution, unless at least one of the clause's literals belongs also to a clause with the same characteristic (all literals being positive) analyzed before. A similar conclusion can be reached for clauses where all literals appear negated. In this way we calculate a minimum number of positive and negated literals, which are in fact the values min and $n - max$ of the unitation.

The CFDA is run with the factorization and the constrained values, obtained as explained above. In our experiments we used truncation selection with parameter $T = 0.1$, population size of 500 and a maximum of 30 generations. In all the algorithms a mutation operator with probability 0.01 is added in each generation. This operator looks for adding diversity to the population. The fitness function is the sum of the weights of the clauses that are satisfied. Weight clausing is commonly used in SLS applied to SAT problems because the important improvements this technique can help to achieve [4]. Clauses begin with an equal weight, after a number of solutions have been evaluated clauses increase their weight inversely to the frequency they have been satisfied.

Tables 1 shows a comparison between the CFDA and the FDA. Column 1 specifies the instance used for the experiments, each instance represents in fact a different problem with particular characteristics. Columns 2 and 3 respectively represent minimum and maximum unitation values calculated for the instance. s is the size of the maximum clique for the algorithm that finds the factorization. $succ$ and $gen.$ respectively represent the number of successful runs and the average number of generations.

In the table it can be seen that the CFDA outperforms the FDA in 12 of the 18 experiments, in 3 experiments both algorithms are equally good, and in the rest the FDA performs better. Two of the 3 cases where the CFDA does not perform as well as the FDA correspond to the same instance, this problem seems to be 'deceptive' for the CFDA. Another interesting detail is the relation between s and the behavior of the algorithms. For the FDA, better results are achieved when $s = 3$, these results are reversed when $s = 6$, although the difference is not as great as in the previous case.

The conclusion to draw from these experiments is that the use of constraints by means of the CFDA can increase

the efficiency of the search. The validity of using approximate factorizations is also confirmed. We want to point out that the UMDA can also show a good behavior for this kind of problems when it combines the use of clause weighting with more sophisticated dynamic fitness functions based on the use of the univariate marginal probabilities. Also, preliminary results have been presented that support the idea of using Bayesian FDAs for 3-SAT [10]. Moreover, in their work the authors mention the possibility of using problem information for the Bayesian network learning algorithm, although experiments were not conducted. In general numerous FDAs' approaches to SAT seem feasible, and we feel this topic well worth future research.

4.2 Typical testors

Now we present the typical testor problem. The notation introduced in [1] will be employed.

Let $I = \{k_1, ..., k_m\}$ be the set of the rows of a Boolean matrix M, and $J = \{j_1, ..., j_n\}$ the set of labels of its columns (features). Let $T \subseteq J$, $M_{/T}$ is the matrix obtained from M eliminating all columns not belonging to the set T.

Definition 1 *A set $T \subseteq J$ is a testor of M if no row with only zeros in $M_{/T}$ exists.*

Definition 2 *The feature $j_{i_r} \subseteq T$ is typical with respect to (wrt) T and M if $\exists q$, $q \in \{1, ..., m\}$ such that $a_{k_q i_r} = 1$ and for $s > 1$,$a_{k_q i_r} = 0, \forall p$, $p \in \{1, ..., s\}$ $p \neq r$.*

Definition 3 *A set T has the property of typicality wrt a matrix M if all features in T are typical wrt T and M.*

Definition 4 *A set $T = \{j_{i_1}, ..., j_{i_s}\} \subseteq J$ is denominated typical testor of M if it is a testor and it has the property of typicality wrt M.*

The typical testor problem consists of finding a typical testor of a given matrix M. Testors, and particularly typical testors, have been used in feature selection and supervised classification problems. A recent overview of the concept of testor can be found in [11]. The computation of the set of all typical testors belongs to the NP class of problem. When the dimension of the matrix is small, deterministic algorithms (DA) have been used. These algorithms incorporate different kinds of heuristics to accomplish the search among all possible subsets of column labels of the matrix in a more efficient way. Recently, the UMDA has been used to deal with matrices of larger dimension [1].

In our experiments we used the FDA and the CFDA for finding typical testors. The following representation and fitness function introduced in [1] were used:

Representation: Binary vectors represent the characteristic vectors of the subsets of features. We will represent $M_{/x}$ as the matrix obtained from M eliminating all columns whose corresponding component in x is 0. Each vector $x = (x_1, ..., x_n)$; $x_i \in \{0, 1\}$; $i = 1, ..., n$ is evaluated using the following objective function:

α/u	$[0,8]$	$[1,7]$	$[2,6]$	$[3,5]$	$[4]$	FDA
0.99	95	100	100	100	100	42
0.98	98	100	100	100	100	36
0.97	98	100	99	99	100	12
0.96	97	92	95	95	100	1
0.95	81	79	85	78	100	0
0.94	21	44	46	39	100	2

Table 2: Comparison between the FDA and the CFDA for the typical testor problem

$$f(x) = \alpha \frac{t(x)}{m} + (1 - \alpha) \frac{p(x)}{\sum_{v=1,..,n} x_v}, 0 < \alpha < 1 \qquad (11)$$

where $t(x)$ is the number of rows of $M_{/x}$ t that contain some unitary element. $p(x)$ is the number of typical features in $M_{/x}$. α is a weighting coefficient between the two properties. Notice that for any vector x, $0 \leq f(x) \leq 1$. If the function f reaches the maximal value 1 for a vector x, then it is a typical testor. This happens if, and only if, any row of $M_{/x}$ has at least one unitary element and all the features of the set are typical.

There are a number of reasons that make the typical testor problem very interesting for the analysis of FDAs. First, its generality, many pattern recognition problems can be addressed using the testors theory. Second, test matrices can be constructed in such a way that we know in advance which are the most related features, allowing the creation of a suitable test best for optimization algorithms.

Finally, there is link between the properties evaluated by the typical testor function and the dependency relations that arise between variables. While the testor property can be seen as non deceptive (the union of two testors will also be a testor), the typicality property is (the union of two typical testors is not a typical testor). When there are more than one typical testor in the matrix the typical testor problem is a very deceptive one, subsolutions from each typical testors compete. The parameter alpha can be used to confront this deception. By tuning α, we can also study how sensitive the algorithm is to the strength of interactions between variables.

In our experiments we used different test matrices constructed following the strategies introduced in [12]. We present results for only one Boolean matrix of 256 rows and 16 columns This matrix has 4 testors of unitation 4. Each feature is present in only one of the solutions, so testors do not overlap. The factorization is constructed clustering those variables associated to columns that are similar in the Boolean matrix. For this matrix we perform experiments using the FDA and the CFDA for different values of the parameter α, and in different

unitation regions. We use a population size of 100, a truncation parameter $T = 0.15$ and a maximum of 12 generations.

In Table 2 the results of the experiments for the problem are shown. It can be seen how the optimization becomes harder as α decreases. However, when the search is accomplished in the unitation regions the percentage of success increases. When the unitation region is [4] the CFDA always finds the optimum. Even when this region expands CFDA obtains good results outperforming those achieved by the FDA. In general the CFDA can be applied for the typical testor problem if some knowledge is available about the unitation of the solutions, or if we are interested in typical testors with a previously defined number of features.

5 Conclusions

In this paper we have introduced the CFDA. This algorithm can be used for the optimization of problems with constraints based on the values of unitation. We have presented a set of hypotheses that could help to explain the behavior of the algorithm. These assumptions have been illustrated using empirical data. For the experiments conducted we have shown the utility of the algorithm introduced. We have also addressed the question of how to find an appropriate factorization for a given problem. Topics for future work are to develop a theoretical framework that allows us to investigate our assumptions, and the extension of the algorithms to constrained problems defined on integers.

Bibliography

[1] E. Alba-Cabrera, R. Santana, A. Ochoa-Rodriguez, and M. Lazo-Cortes. Finding typical testors by using an evolutionary strategy. In *Proceedings of the Fith Ibero American Symposium on Pattern Recognition*, pages 267–278, Lisbon, Portugal, 2000.

[2] S. Baluja and S. Davies. Using optimal dependency-trees for combinatorial optimization: Learning the structure of the search space. In *Proceedings of the 14th International Conference on Machine Learning*, pages 30–38. Morgan Kaufmann, 1997.

[3] R. Dechter. Constraint networks. In S. C. Shapiro, editor, *Encyclopedia of Artificial Intelligence*, volume 1. Addison-Wesley Publishing Company, 1992. Second Edition.

[4] K. Kask and R. Dechter. GSAT and local consistency. In *Proceedings of the 14th IJCAI*, pages 616–622, Montreal, Canada, 1995.

[5] S. L. Lauritzen. *Graphical Models*. Oxford:Clarendom Press, 1996.

[6] H. Mühlenbein. The equation for response to selection and its use for prediction. *Evolutionary Computation*, 5(3):303–346, 1997.

[7] H. Mühlenbein, T. Mahnig, and A. Ochoa. Schemata, distributions and graphical models in evolutionary optimization. *Journal of Heuristics*, 5(2):213–247, 1999.

[8] H. Mühlenbein and G. Paaß. From recombination of genes to the estimation of distributions I. Binary parameters. In A. Eiben, T. Bäck, M. Shoenauer, and H. Schwefel, editors, *Parallel Problem Solving from Nature - PPSN IV*, pages 178–187, Berlin, 1996. Springer Verlag.

[9] A. Ochoa, M. R. Soto, R. Santana, J. C. Madera, and N. Jorge. The Factorized Distribution Algorithm and the juction tree: A learning perspective. In A. Ochoa, M. R. Soto, and R. Santana, editors, *Proceedings of the Second Symposium on Artificial Intelligence (CIMAF-99)*, pages 368–377, Habana, Cuba, March 1999.

[10] M. Pelikan, D. E. Goldberg, and K. Sastry. Bayesian Optimization Algorithm, decision graphs, and Occam's razor. IlliGAL Report No. 2000020, University of Illinois at Urbana-Champaign, Illinois Genetic Algorithms Laboratory, Urbana, IL, 2000.

[11] J. Ruiz-Shulcloper, M. Lazo-Cortés, and E. Alba-Cabrera. An overview of the concept of testor. *Pattern Recognition Journal*, 34(4):13–21, 2001.

[12] R. Santana and E. Alba. Generating test matrices to evaluate the performance of strategies to search typical testors. Technical Report ICIMAF 2000-130, Institute of Cybernetics, Mathematics and Physics, Havana, Cuba, January 2001.

[13] R. Santana, E. P. de León, and A. Ochoa. The Edge Incident Model. In *Proceedings of the Second Symposium on Artificial Intelligence (CIMAF-99)*, pages 352–359, Habana, Cuba, March 1999.

[14] R. Santana and E. P. de Leon. An evolutionary optimization approach for detecting structures on graphs. In Dagli, Akay, Buczac, Ersoy, and Fernandez, editors, *Smart Engineering System Design: Neural Network, Fuzzy Logic, Rough Sets and Evolutionary Programming*, pages 371–376. ASME press, 1998.

[15] R. Santana and A. Ochoa. A Constraint Univariate Marginal Distribution Algorithm. Technical Report ICIMAF 99-76, CENIA 99-04, Institute of Cybernetics, Mathematics and Physics, Havana, Cuba, 1999.