ON THE USE OF POLYTREES IN EVOLUTIONARY OPTIMIZATION

Marta Soto[1], Alberto Ochoa y Roberto Santana, Institute of Cybernetics, Mathematics and Physics, Cuba

ABSTRACT
Bayesian networks, are usefull tools for the representation of non-linear interactions among variables. Recently, they have been combined with evolutionary methods to form a new class of optimization algorithms: the Factorized Distribution Algorithms (FDAs). FDAs have been proved to be significantly better than their genetic ancestors. They learn and sample distributions instead of using crossover and mutation operators. Most of the members of the FDAs that have been designed, learn general Bayesian networks. However, in this work we study a FDA that learns polytrees, which are single connected directed graphs.

Key words: Bayesian networks, evolutionary algorithms, FDAs.

MSC: 90B15, 90C35.

RESUMEN
Las redes Bayesianas son herramientas muy útiles para la representación de las interacciones no lineales entre las variables. Recientemente estas han sido combinadas con los métodos evolutivos para formar una nueva clase de algoritmos de optimización: los Algoritmos con Distribución Factorizada (FDA). Se ha probado que los FDA son mejores que sus antecesores, los Algoritmos Genéticos, en problemas donde existe una fuerte interacción entre las variables. Estos algoritmos aprenden y simulan distribuciones probabilísticas, en lugar de usar operadores de cruzamiento y mutación. La mayoría de los FDA diseñados hasta el momento aprenden redes Bayesianas generales. Sin embargo, en este trabajo nosotros estudiamos un FDA que aprende poliárboles, los cuales son grafos dirigidos simplemente conectados.

Palabras clave: redes Bayesianas, algoritmos evolutivos, FDA**.**

INTRODUCTION

Recently a new class of evolutionary algorithms has emerged. These developments are tractable versions of the Estimation Distribution Algorithms (EDAs) [Mühlenbein and Paaβ, 1996]. However, EDAs have only theoretical importance because the estimation and sampling problems are too difficult if we do not restrict the type of distribution to deal with.

At each generation, the tractable subclass, the so called Factorized Distribution Algorithms (FDAs), uses factorizations of the joint distribution of the best points found so far. This information is used to construct a model from which new points are efficiently sampled.

FDAs use results of Bayesian networks research [Mühlenbein **et al**., 1999]. Some recent contributions in this fiel are [Mühlenbein and Mahnig, 1999, Ochoa **et al**., 1999, Etxeberria and Larrañaga, 1999, Soto and Ochoa, 2000]. All these algorithms have performed very well on a wide range of Genetic Algorithms-hard functions. This points out to the fact that the failure of recombination lies on its inability to deal with high order nonlinear interactions.

Most of the members of the FDAs class that have been designed, learn general Bayesian networks. However, in this work we study a FDA that learns polytrees. Our choice is motivated by the fact that we know efficient procedures for learning trees, a special class of polytrees. Also, because under certain restrictions, we can show low cost methods for learning general polytrees.

The outline of the paper is as follows. Section 2 gives a brief introduction on Bayesian networks and Factorized Distribution Algorithms. Section 3 presents a Factorized Distribution Algorithms based on polytrees. The following section presents the experimental results. The main conclusions of our research are given in section 5.

[1]{mrosa,ochoa,rsantana}@cidet.icmf.inf.cu, {mrosa.cu,aa8ar,rsantana.cu}@yahoo.com

2. BACKGROUND

The research on the Factorized Distribution Algorithms is based on two major fields:Graphical Models and Evolutionary Computation.

2.1. Bayesian networks

A Bayesian network is a directed acyclic graph containing nodes, representing the variables, and arcs, representing probabilistic dependencies among nodes.

Let $X = \{X_1,\dots,X_n\}$ denotes the set of random variables. For any node X_i and set of parents π_{X_i} the Bayesian network specifies a conditional probability distribution $P(X_i \mid \pi_{X_i})$. The full joint probability distribution specified by a Bayesian network can be calculated by taking the product of the conditional probabilities:

$$P(X_1,\dots,X_n) = \Pi\ P(X \mid \pi_{X_i})$$

There are different types of Bayesian networks according to their degree of complexity. Some examples are chains, trees, polytrees, simple graphs and general multiple connected networks. The first three are single connected graphs, which means they have no undirected cycles. Simple graphs are multiple connected networks, where the common parents of any variable are always marginally independent between each other. Single connected graphs are specific cases of simple graphs. A different factorization of a joint distribution corresponds to each of these structures.

One interesting property of simple graphs is that, under certain restrictions, they can be recovered using only zero and first order conditional independence test. This means, that the learning algorithm needs only up to third order marginal distributions.

We call Bayesian networks of bounded complexity (BN-BC), those networks that can be recovered from data using only first, second and third order marginal distributions. Therefore, multiple connected networks with a maximum of two parents are also included.

The question of reliability and computational cost, both in term of space and time, may have in optimization a more critical role than they had in previous fields of application of graphical models. BN-BC are important for optimization because they can be recovered from data using simple and efficient learning procedures.

There are some motivations for studying the performance of the FDAs that use BN-BC (FDA-BC). Firstly, it is known the good performance of the Univariate Marginal Distribution Algorithm (UMDA) [Mühlenbein, 1997] in many non-linear problems. Secondly, BN-BC are also good candidates for combination and mixture of models. In this work we deal with an important subclass of the BN-BC: the polytrees.

In a polytree there are only three possible ways of connecting three variables:

$$X \rightarrow Z \rightarrow Y \qquad\qquad X \leftarrow Z \rightarrow Y \qquad\qquad X \rightarrow Z \leftarrow Y$$

Among them, only the third type allows to pass information between X and Y when Z is instantiated. That is, only the third type makes X and Y dependent conditionally on Z. This structure is often called head-to-head connection or pattern.

Any polytree factorization can be written as:

$$p(X) = \Pi p\left(X_i \mid X_{j1(i)}, X_{j2(i)},\dots,Xx_{jr(i)}\right)$$

where $\{X_{j1(i)}, X_{j2(i)},\dots,X_{jr(i)}\}$ is the set of direct parents (possibly empty) of variable X_i. A particular property of polytrees is that the parents of any variable are mutually independent. Hence, for all I we have:

$$p(\pi_{x_i}) = \prod_{j=1}^{r} p(x_{ij})$$

where $p(\pi_{x_i})$ is the joint distribution of the parent set.

2.2. The Factorized Distribution Algorithms

Factorized Distribution Algorithms (see Figure 1) are population based search methods that combine results from research in Graphical Models and Evolutionary Computation. They are considered to be the tractable subclass of Estimation Distribution Algorithms [Mühlenbein and Paaβ, 1996]. The optimization of a function begins with the generation of a random population of points. Then some of the points are selected based on their function's values. The next and crucial step, is the construction of a graphical model approximation of the empirical distribution of the selected points. Sampling from this distribution model gives us the next population.

step 0	$t \leftarrow 1$, Generate N points randomly.
step 1	Get a selected set S with M points ($M < N$), according to the selection method.
step 2	Learn a graphical model approximation of the empirical distribution of the selected points S, $p^s(x,t)$.
step 3	Generate N points according to $p(x, t+1) = p^s(x,t)$.
step 4	$t \leftarrow t+1$. If the termination criteria are not met go to **step 1**.

Figure 1. Scheme of FDAs.

3. FDAs BASED ON POLYTREES

The Polytree Approximation Distribution Algorithm (PADA) is a FDA. Thus is shares the general structure given in the Figure 1 [Soto **et al**., 1999].

PADA learns polytree structures, (trees are special cases of polytrees). Unfortunately, it is not at all clear that learning a polytree is any easier than learning the structure of a general Bayesian network. Moreover, even if we restrict the maximum number of parents to 2, this problem remains to be NP-hard. However, this does not mean that good approximation algorithms can not be created. Following this line of thinking, this paper explores two different ways of implementing the learning component of PADA.

The first method [Etxeberria and Larrañaga, 1999] is based on the minimization of a scoring metric. We call this algorithm SPADA. This algorithm can also be used to learn trees or low complexity Bayesian networks controlling the number of parents.

Another possibility is to follow the ideas developed in [Campos, 1998]. In that work was introduced a graded dependence relation that measures the degree of dependence between two variables. The basic idea of the algorithm is to preserve those edges representing the strongest dependency relations, but with the restriction that the resultant structure must be singly connected. To do this, the algorithm uses a Maximum Weight Spanning Tree (MWST) to recover the skeleton (the undirected graph that supports the polytree). The edges are weighted with the dependency relation mentioned above. It has been proven elsewhere that the above procedure is optimal for trees if the Kullback-Lieber entropy is used as a dependency measure. This means that it is possible to recover the best tree that approximates the data. In [Chow and Liu, 1968] is presented the Chow-Liu's algorithm for learning trees. After the skeleton is obtained, this algorithm selects a node as a root and directs the remainder edges according to this selection. Despite which node is selected as a root we obtain an unique tree factorization.

For general polytrees, once we have constructed the skeleton a procedure tries to direct the edges of the skeleton by using the following scheme: in a head to head pattern $\alpha \rightarrow \gamma \leftarrow \beta$, the instantiation of the head node γ should normally increase the degree of dependency between the variables α and β, whereas in a non head to head pattern (such as $\alpha \leftarrow \gamma \rightarrow \beta$, $\alpha \rightarrow \gamma \rightarrow \beta$ or $\alpha \leftarrow \gamma \leftarrow \beta$), the instantiation of the middle node γ should produce the opposite effect, decreasing the degree of dependency between α and β. So the idea is to compare the degree of dependency betweenα and β after the instantiation of γ, with the degree of dependency between α and β before the instantiation of γ.

Definition: For each subgraph $\alpha - \gamma - \beta \in$ skeleton,

$$\text{if } \mathrm{Dep}(\alpha,\beta \mid \gamma) > \mathrm{Dep}(\alpha,\beta \mid \varnothing) \quad \text{then} \quad \alpha \rightarrow \gamma \leftarrow \beta.$$

The edges that were not oriented after the above test, are directed at random without introducing new head to head connections and minimizing the number of parents.

The learning algorithms of general Bayesian networks have exponential complexity in the number of parents. The method presented above, uses only up to third order distribution. Therefore, it makes sense to explore bounded sampling approximations. Given an ancestral ordering of all the variables (parents go before children), the method samples X_i using $P(X_i \mid \pi_{X_i})$. One obvious problem of this method, is that it will require an exponential number of points in the number of parents, to estimate the conditional probabilities correctly. In PADA we can use several approximations. For example:

Let us assume, without loss of generality, that $\pi_{X_0} = \{ X_1, X_2, X_r \}$. Hence, once the parents are instantiated, the PLS method would sample X_0 with probability $p(x_0 \mid \pi_{X_0})$.

We set,

$$\boxed{M(x_0 = 1) = p(x_0 = 1|x_1x_2) + \ldots + p(x_0 = 1|x_rx_{r-1})}$$

$$\boxed{M(x_0 = 0) = p(x_0 = 0|x_1x_2) + \ldots + p(x_0 = 0|x_rx_{r-1})}$$

Then, we use the PLS method, but sample from:

$$\tilde{p}(x_0 = 1) = \frac{M(x_0 = 1)}{M(x_0 = 0) + M(x_0 = 1)}$$

instead of $p(x_0 \mid \pi_{X_0})$.

4. EXPERIMENTS

4.1. Test functions

In the experiments we explore the following functions:

1. The OneMax or unitation function:

$$\text{OneMax}(x) = \sum_{i=1}^{n} x_i$$

OneMax has (n + 1) different fitness values.

2. The Cuban function (F_{c_5}): this overlapped function was introduced in [Mühlenbein **et al.**, 1999]. It was designed to show that the global solution need not to be composed of locally optimal solutions. The values of the second best optima are almost as large as the global optimum.

$$F^3_{cuban1}(x)\begin{cases} 0.595 & \text{for} \quad x = 000 \\ 0.200 & \text{for} \quad x = 001 \\ 0.595 & \text{for} \quad x = 010 \\ 0.100 & \text{for} \quad x = 011 \\ 1.000 & \text{for} \quad x = 100 \\ 0.050 & \text{for} \quad x = 101 \\ 0.090 & \text{for} \quad x = 110 \\ 0.150 & \text{for} \quad x = 111 \end{cases}$$

$$F^5_{cuban1}(x)\begin{cases} 4 * F^3(x_1, x_2, x_3) & \text{if} \quad x_2 = x_4 \\ & \text{and} \quad x_3 = x_5 \\ 0 & \quad \text{otherwise} \end{cases}$$

$$F^5_{cuban2}(x)\begin{cases} u(x) & \text{for} \quad x_5 = 0 \\ 0 & \text{for} \quad x_1 = 0, x_5 = 1 \\ u(x) - 2 & \text{for} \quad x_1 = 1, x_5 = 1 \end{cases}$$

$$F_{c5}(x) = F^5_{cuban1}(s_0) + \sum_{j=0}^{m}\left(F^5_{cuban2}(s_{2j+1}) + F^5_{cuban1}(s_{2j+2})\right)$$

where $s_i = x_{4i}x_{4i+1}x_{4i+2}x_{4i+3}x_{4i+4}$ and $n = 4(2m + 1) + 1$.

3. The Isotorus function: this function is defined on the grid of size $n = m * m$. The peak function $IsoT_1$ is used at the upper left corner of the torus. Let u denotes the number of 1_s in a string.

u	0	1	2	3	4	5
$IsoT_1$	m	0	0	0	0	m - 1
$IsoT_2$	0	0	0	0	0	M^2

$$F_{IsoTorus} = IsoT_1(x_{1-m+n}, x_{1-m+n}, x_1, x_2, x_{1+m}) + \sum_{i=2}^{N} IsoT_2(x_{up}, x_{left}, x_i, x_{right}, x_{down})$$

where x_{up}, etc., are defined as the appropriate neighbors, wrapping around. This function is difficult to optimize. The best and the second best strings have values $m^3 - m + 1$ and $m^3 - m$ respectively.

4.2. Numerical Results

In the following tables N is the population size and n is the number of variables. The average generation when the optimum is found is denoted by G, whereas % represents how many times this optimum is found in 100 runs. The truncation value is denoted by τ. The column P applies to SPADA and to a general Bayesian network (GBN). P denotes the maximum number of parents. PADA and TREE are both implement following the second approach presented in section 3. Note that all these algorithms, including the GBN with 2 parents are considered in this paper to be algorithms with bounded complexity.

4.2.1. Function OneMax

For the OneMax, the performance of the PADA is compared with that of the UMDA.

It is worth noting taht UMDA is a special case of the PADA.It has a graph without arcs, and therefore it uses only univariate marginal probabilities. For the OneMax function UMDA performs better than PADA (see Table 1), but considering that UMDA has knowledge of the exact dependency model, the difference seems to be not so big. For truncation equal to 0.5 the difference in performance is smaller. Of course, we are not considering the learning cost of PADA. More important is the fact that PADA uses the same small set of parameters that UMDA does: population size and selection pressure.

Tabla 1. Numerical results for PADA and UMDA for the OneMax function.

Algorithm	N	n	τ	G	%
PADA	30	30	0.3	6.21	69
UMDA	30	30	0.3	6.74	75
PADA	30	30	0.5	9.69	72
UMDA	30	30	0.5	9.13	81

4.2.2. Function Cuban

We have found that for a class of popular test function FDA-BC performs as good as, or even better that algorithms using more general distributions. Table 2 shows that GBNs depend strongly on the number of parents. We have also made test with 3 and 4 parents; the results agree with what is shown here. However, SPADA seems to be more stable in this respect. For 21 and 37 variables the FDA-BC perform well (the TREE is the bets). For 101 variables even with a population size of 13000 the success rate is below 60 %. However, in these cases, it makes no sense to use a GBN with more parents.

Table 2 Results of Cuban function with 0.1 truncation value.

Cuban function		**TREE**		**SPADA**			**GBN**	
n	N	G	%	P	G	%	G	%
21	1000	2.4	100	2	2.2	100	2.3	100
21	1000	-	-	5	2.1	98	2.6	60
37	3000	4.3	96	2	4.7	82	4.1	88
37	3000			5	4.6	73	4.4	65
101	13000	10.4	55	2	10.4	41	10.4	52
101	13000			5	10.5	55	10.6	59

The results in the Table 3 are better for the GBNs. We have observed that more general distributions perform better with larger truncation values. Besides, better informed models have lower convergence times. However, as the table shows there is no a big difference between the results of the algorithms. Having in mind the low computational cost of the FDA-BC we conclude that they are better for the optimization of this function.

Table 3. Results for Cuban function with 0.1 truncation value.

Cuban function		**TREE**		**SPADA**			**GBN**	
n	N	G	%	P	G	%	G	%
21	1000	3.2	100	2	3.4	100	3.34	100
21	1000	-	-	5	3.8	100	3.5	100
37	3000	7.2	98	2	7.5	97	6.7	99
37	3000			5	7.6	95	6.6	99
101	9000	17.1	90	2	16.9	93	16.9	94
101	9000			5	17.2	90	17	94

4.2.3. Funtion IsoTorus

It was expected that the IsoTorus function should not be easily optimized by PADA or SPADA. This function is a highly overlapped grid function, and therefore its variables interact strongly. However, the result was very interesting. It can be easily optimized even with a tree structure (see Table 4).

Table 4. Results for IsoTorus with 49 variables.
N = 1000 and $\tau = 0.1$.

TREE		**PADA**		**SPADA**			**GBN**	
G	%	G	%	P	G	%	G	%
6.18	75	7.0	75	2	6.57	71	5.4	73
-	-	-	-	5	6.7	56	4.5	6

The success rate for GBNs comes down dramatically from 73 to 6. The reason is simple. Because the data set is so connected, the best model should have many parents (max = 5). Unfortunately this increases the population size requirements, and therefore the success rate for a fixed population size comes down. SPADA uses the same greedy algorithm that the GBN uses, however the constraint of polytrees prevents the unnecessary grow of the network, and the success rate falls down only to 56. A more interesting fact is the following: PADA does not constraint the number of parents. Therefore, it can learn more complex polytrees than SPADA with two parents. However, because PADA uses only up to third order distributions, its success rate is higher.

5. CONCLUSIONS

Most of the members of the FDAs that have been designed, learn general Bayesian networks. However, in this work we study a FDA that learns polytrees: the Polytree Approximation Distribution Algorithm (PADA). We explore two implementations of the learning component of PADA. The first method is based on the minimization of a scoring metric. The second approach uses independence tests between two and three variables. PADA has been investigated in the case of three functions.

The experiments shows that, at least under certain conditions, bounded complexity approximation models perform as good as or even better that algorithms that learn more general distributions.

6. ACKNOWLEDGMENTS

This work was funded by the Ministry of Sciences and Technology of Cuba, under the project Low Cost Evolutionary Algorithms.

REFERENCES

CAMPOS, L.M.D. (1998): "Independency relationship and learning algorithms for singly connected networks", **Journal of Experimental and Theoretical Artificial Intelligence**, 511-549.

CHOW, C.K. and C.N. LIU (1968): **Approximating discrete probability distributions with dependence trees**, IT14, 462-467.

ETXEBERRIA, R. and P. LARRAÑAGA (1999): "Global optimization using Bayesian networks", In **Second Symposium on Artificial Intelligence** (CIMAF-99), 332-339, Habana, Cuba.

MÜHLENBEIN, H. (1997): "The equation for response to selection and its use for prediction", **Evolutionary Computation**, 5(3):303-346.

MÜHLENBEIN, H. and T. MAHNIG (1999): "FDA – a scalable evolutionary algorithm for the optimization of additively decomposed functions", **Evolutionary Computation**, 7(4): 353-376.

MÜHLENBEIN, H.; T. MAHNIG and A. OCHOA (1999): "Schemata, distributions and graphical models in evolutionary optimization", **Journal of Heuristics**, 5(2):213-247.

MÜHLENBEIN, H. and G. PAAβ (1996): "From recombination of genes to the estimation of distributions I. Binary parameters", In Eiben, A., Bäck, T., Shoenauer, M., and Schwefel, H., editors, **Parallel Problem Solving from Nature** – PPSN IV, 178-187, Berlin, Springer Verlag.

OCHOA, A.; M.R. SOTO; R. SANTANA; J.C. MADERA and N. JORGE (1999): "The factorized distribution algorithm and the junction tree: a learning perspective. In Ochoa, A., Soto, M.R. and Santana, R., editors, **Second Symposium on Artificial Intelligence** (CIMAF-99), 368-377, Habana, Cuba.

SOTO, M.R. and A. OCHOA (2000): "A factorized distribution algorithm based on polytrees", In: **Proceedings of the 2000 Congress on Evolutionary Computation** CEC00, 232-237, La Jolla Marriott Hotel La Jolla, California, USA. IEEE Press.

SOTO, M.R.; A. OCHOA, S. ACID and L.M. CAMPOS (1999): "Bayesian evolutionary algorithms based on simplified models", In: **Second Symposium on Artificial Intelligence** (CIMAF-99), 360-367, Habana, Cuba.