Factorized Distribution Algorithms: Selection without selected population

Roberto Santana [*]

Abstract

In this paper we investigate the problem of an efficient implementation of the selection step in Factorized Distribution Algorithms. We demonstrate that while in Genetic Algorithms the selection operator needs the creation of a selected population, in Factorized Distribution Algorithms this is not always the case.

Key words: Estimation distribution algorithms, Factorized distribution algorithms, selection, Genetic Algorithms
MSC 2000: 68T05, 05C35.

1 Introduction

Selection is a corner stone of a number of population based search methods like Genetic Algorithms (GAs) [4, 3], and Estimation Distribution Algorithms (EDAs) [10]. These methods are non deterministic heuristic search strategies, commonly used for optimization. Their main characteristics are: They use a population of individuals or solutions, instead of a single point, to conduct the search. In every iteration (usually called generation) a subset of individuals is selected, and by applying some operators, a new population is created. In this way the algorithm iterates (evolves) until a stop condition is satisfied.

In GAs the recombination and mutation operators are applied to the selected set of individuals to obtain the new population. Other algorithms are characterized by the use of probabilistic modeling of the information contained in the selected set. In this paper we will use the term Estimation Distribution Algorithm to refer to any evolutionary algorithm that uses the estimation of probability distributions instead of the genetic operators. Although not all the proposals fit well in the EDAs scheme, this conceptual framework has been used before [5] as a model to study the algorithms under analysis. Algorithm 1 shows the steps of an EDA.

One efficient way of estimating a probability distribution is by means of factorizations. A probability distribution is factorized when it can be computed as the product of a small number of factors. A subclass of EDAs groups the algorithms that use factorizations of the probability distribution. In this paper we call this subclass Factorized Distribution Algorithms (FDAs) [9]. In

[*]Institute of Cybernetics, Mathematics, and Physics (ICIMAF) Calle 15, entre C y D, Vedado, C-Habana, Cuba. rsantana@cidet.icmf.inf.cu

Algorithm 1: **EDAs**

```
1  Set $t \Leftarrow 0$. Generate $N$ individuals randomly.
2  do {
3      Select a set $S$ of $k \leq N$ individuals according to a selection method.
4      Calculate a probabilistic model of $S$.
5      Generate N new individuals sampling from the distribution represented in the model.
6      $t \Leftarrow t + 1$
7  } until Termination criteria are met
```

the literature the term FDA is frequently used to name a particular type of Factorized Distribution Algorithms. Our definition covers it, and other algorithms that use factorizations. FDAs belong to the EDAs class as well as other evolutionary algorithms where the estimation of the distribution is achieved by other means. In this paper we show that the selection step in a FDA can be omitted when this step involves the calculation of the selection probabilities of all the individuals in the current population. The possibility of changing the way selection is accomplished represents an important difference with GAs.

Let $X = (X_1, \ldots, X_n)$ be a tuple of random variables, and $X \in B^n$ where B^n is the finite n-dimensional binary space. Throughout this paper x will denote a value of the individual X, and x_i the value of X_i, the i-th component of X.

The paper is organized as follows: In the next section we review the main types of selections used by GAs. In section 3 we discuss the way selection is implemented in FDAs. We present a modification to the traditional way of doing selection. In section 4 we conduct a number of experiments to evaluate the impact of the proposed changes in the behavior of some FDAs. Finally, section 5 describes the relationship between selection and the other components of the FDAs, and presents the conclusions of our work.

2 Selection in GAs

In our analysis we have used the classification of the selection methods proposed by Sastry and Goldberg [12]. Selection methods are classified in two classes: (1) Proportional schemes, and (2) Ordinal based schemes. Proportional schemes select an individual based on its relative fitness value compared to others. Ordinal schemes select an individual based on its ranking in the population.

In Proportional schemes selection is usually accomplished in two steps. First, the selection probabilities of the individuals in the current population are determined, then new individuals are sampled from these probabilities. These individuals form the selected set that serves as a mating pool. Examples of these Proportional schemes are the Proportional [3] and Boltzmann [8] selection. In Ordinal based schemes, selection probabilities are not explicitly calculated. Instead, some procedure is used to select the individuals straight from the population. One example is the Tournament selection, where s individuals are randomly chosen from the population, and the best individual from this group is included in the selected set[1]. This process is repeated until the selected set has been filled.

[1]Tournament selection can be done with or without replacement of the selected individual.

When Proportional schemes are applied, different sampling algorithms can be used to sample from the selection probabilities. The most known example is the Roulette Wheel (RW) selection [3], generally used for Proportional selection. In RW selection a biased roulette wheel is created where each current individual in the population has a roulette wheel slot sized in proportion to its fitness. Every time the wheel is spinned, a copy of the selected individual is included in the selected set. RW selection is also a clear example of a bad sampling method, for small populations it has a high variance.

To solve this problem other types of sampling methods, like the Stochastic Universal Sampling (SUS) [1], have been proposed. SUS consists of simultaneously selecting N individuals by locating N equally spaced pointers in the roulette. After only one spin, we select the individuals corresponding to the slots where the N pointers are located.

In table 1 an example of the application of the Proportional selection is shown. From a population of 7 individuals the selection probabilities are calculated based on the individuals' fitness. The expected count shown in column 4 is the expected number of copies of each individual (n^j), calculated as the product of the selection probabilities ($p(x^j)$) by the number of generated individuals.

$$n^j = p(x^j) \cdot N \tag{1}$$

Possible outcomes of the RW and SUS methods are presented in columns 5 and 6. Due to the finite size of the population none of the actual counts corresponds to the expected counts.

x	f(x)	$p^s(x^j)$	n^j	n^j_{RW}	n^j_{SUS}
10100	2	0.1	0.7	1	1
10110	2	0.1	0.7	0	1
11000	2	0.1	0.7	0	0
10011	3	0.15	1.05	1	1
10101	3	0.15	1.05	1	1
10111	4	0.2	1.4	2	1
11011	4	0.2	1.4	2	2

Table 1: Different steps of the Proportional selection method

3 Selection in FDAs

As there is no crossover operator in FDAs, the selected set is not used as a mating pool. These algorithms use the selected individuals to construct a probabilistic model that captures the interdependencies between the variables. Considering the complexity of the probabilistic models they use, FDAs can be classified in two classes: (1) Algorithms that make a parametric learning of the probabilities, and (2) Algorithms where a structural learning of the model is done. In parametric, as well as in structural learning, the only relevant information extracted from the data is initially represented in the marginal probabilities of the variables. Structural learning finds a graphic representation of the interactions among the variables using these marginals. Parametric and structural learning are also known as model fitting and model selection. The interested reader is referred to [5] for a review of probabilistic graphical modeling and EDAs.

3.1 Proportional Schemes

Traditional implementations of FDAs that use Proportional schemes determine firstly the set of selected solutions, and calculate then the marginal probabilities from this set. We analytically show that avoiding the creation of the selected set is possible. To present our case we use as an example of FDAs the Univariate Marginal Distribution Algorithm (UMDA) [10]. The UMDA uses a very simple probabilistic model that assumes all the variables are independent. The probability of an individual x is estimated as $p(x) = \prod_{i=1}^{n} p_i^s(X_i = x_i, t)$, where $p_i^s(X_i = x_i, t)$ are the univariate probabilities calculated from the selected population. In the case of the binary problems we treat in this paper, we will use $p_i^s(x_i, t)$ as a shortcut for $p_i^s(X_i = 1, t)$.

Vars./Univ.	p_i	p_i^{sRW}	p_i^{sSUS}	$\hat{p}_i^s$
$x1$	1.0	1.0	1.0	1.0
$x2$	0.286	0.286	0.286	0.3
$x3$	0.571	0.571	0.571	0.55
$x4$	0.571	0.714	0.714	0.65
$x5$	0.571	0.857	0.714	0.70

Table 2: Univariate Probabilities

In table 2 we show the univariate probabilities corresponding to the selected populations presented in table 1. Column 2 shows the univariate marginals of the initial population, calculated assuming that there is only one copy of the 7 individuals. Columns 3 and 4 show the univariate marginals calculated from the populations obtained using RW and SUS (columns 5 and 6 in table 1). The last column corresponds to the univariate probabilities calculated from the expected counts, we call them expected univariate probabilities $\hat{p}_i^s(x_i, t)$.

$$\hat{p}_i^s(x_i, t) = \frac{\sum_{j=1}^{N}(n^j \cdot x_i^j)}{\sum_{j=1}^{N} n^j} \tag{2}$$

Substituting equation (1) in (2), and considering $\sum_j p^s(x^j) = 1$:

$$\begin{aligned}\hat{p}_i^s(x_i, t) &= \frac{N \sum_{j=1}^{N}(p^s(x^j) \cdot x_i^j)}{N \cdot \sum_{j=1}^{N} p^s(x^j)} \\ &= \sum_{j=1}^{N}(p^s(x^j) \cdot x_i^j) \\ &= p_i^s(x_i) \end{aligned} \tag{3}$$

Notice in table 2 that the univariate probabilities of the selected population obtained using SUS are a better approximation of the expected univariate probabilities, than those calculated from the selected population obtained using RW selection. Nevertheless, both approximations have a departure from $\hat{p}_i^s(x_i, t)$. This difference is due to the sample methods used, and to the fact of using a small population. Equation (2) shows that for calculating the expected univariate probabilities there is no need of applying any sampling method, they can be calculated from the expected counts. Even more, as it is observed in equation (3), no even the expected counts are needed.

When Proportional schemes are applied to FDAs it is possible to avoid the step of selecting a set of individuals. The marginal probabilities can be calculated straight from the selection probability distribution determined by the selection method.

If parametric learning is used the learning process finishes after the marginals have been calculated. There is no need to create a selected population, but if a good sampling method like SUS were applied to create the selected population, the obtained marginals would be close to the expected ones, just as in the example described in tables 1 and 2. In the case of structural learning, the only difference is that the marginals are also used for learning the structure of the probabilistic model.

3.2 Ordinal Based Schemes

For Ordinal based schemes, for which selection probabilities are not calculated, the model learning process necessarily has to be done using a selected set of individuals, instead of the selection probabilities. This is the case of Tournament selection. However, in these cases it can still be convenient learning the model from a probability distribution. An algorithm for the case of Ordinal based schemes would have the following steps:

1. Create the selected population.
2. Create a 'compact' population from the selected population where all the individuals are different among each other.
3. Associate to each individual in the compact population its probability in the selected population.
4. Learn the probabilistic model from the probabilities associated to the individuals in the compact population.

In table 3 an example of how to construct a compact population is shown. It is more efficient to do the model learning step when there is only one copy of each individual. Furthermore, by calculating the frequency of each individual in the selected population we can control early convergence of the FDA, and stop the algorithm when a few numbers of individuals dominate the selected population.

4 Experiments

In section 3.1 we have analyticly shown that FDAs do not always need to create a selected population. Thus, the goal of our experiments is not to investigate the validity of this result. Instead, we are interested to find out which is the influence of using a (as shown before redundant) selected population in the behavior of the FDAs that work with Proportional schemes. The goal is to measure the difference between the two selection procedures: when a selected population was used

Selected Pop.	Compact Pop.	Prob.
10100	10100	0.3
10110	10110	0.1
10100	00100	0.2
00100	01001	0.1
01001	11000	0.2
11000	01010	0.1
01010		
11000		
10100		
00100		

Table 3: Construction of a compact population

(SP), or when this was not the case (No SP). We will be concerned with the maximization of a function $f : X \rightarrow R^{\geq 0}$.

An important point is that the results shown below are by no means the best that can be achieved using the FDAs presented. Results with Truncation selection can be much better than those achieved with Proportional selection. On the other hand, the performance of Boltzmann selection can be considerably improved when an adaptive Boltzmann selection schedule is incorporated to the FDA [6]. Similarly, results of the comparison between the different FDAs must not be taken as conclusive.

4.1 Functions Used in the Experiments

Most of the functions used in our experiments belong to the class of Additive Decomposable Functions (ADFs), for which the value of the function is the sum of the evaluation of a number of sub-functions in subsets of variables. They represent an interesting class of functions where possible interactions among the variables are reduced to a subset of them, and thus are used to simulate problems that can be decomposed in smaller subproblems. The functions presented have served before as a test bed for evolutionary algorithms. The interested reader is referred to [5] for an account of the performance of other FDAs for these functions.

The simplest additive function is $OneMax$ where each sub-function is evaluated in only one variable.

Function $OneMax$:

$$OneMax(x) = \sum_{i=1}^{n} x_i \tag{4}$$

Functions of unitation are used to define the sub-functions comprised by an ADF. 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. Thus, functions of unitation can be defined in terms of the unitation of the individual (u). The $f_{3deceptive}$ function is defined as a sum of the more elementary unitation function f_{dec}^{3}.

Function $f_{3deceptive}$:

$$f_{3deceptive}(x) = \sum_{i=1}^{i=\frac{n}{3}} f_{dec}^3(x_{3i-2}, x_{3i-1}, x_{3i})$$

where
Function f_{dec}^3:

$$f_{dec}^3(u) = \begin{cases} 0.9 & for \quad u = 0 \\ 0.8 & for \quad u = 1 \\ 0.0 & for \quad u = 2 \\ 1.0 & for \quad u = 3 \end{cases} \tag{5}$$

Function general deceptive of order k, f_{decK}:

$$f_{decK}(u,k) = \begin{cases} k-1 & for \quad u = 0 \\ k-2 & \quad\quad u = 1 \\ & \dots \\ k-i-1 & \quad\quad u = i \\ & \dots \\ k \cdot n & for \quad u = k \end{cases} \tag{6}$$

Function $f_{deceptivek}$:

$$f_{deceptivek}(x) = \sum_{i=1}^{i=\frac{n}{k}} f_{decK}(x_{ki-k+1}, \dots, x_{ki}) \tag{7}$$

Function *Checkerboard*:
The goal of the chekerboard problem is to create a checkerboard pattern of 0's and 1's in an NxN grid. Only the primary four directions are considered in the evaluation. For each position in an (N-2)x(N-2) grid centered in an NxN grid, 1 is added for each of the four neighbors that are set to the opposite value. The maximum evaluation for the function is $4(N-4)(N-4)$.
Function *symmetric*:

$$symmetric(u) = \begin{cases} u & if \;\; 2 \cdot u < n \\ n-u & otherwise \end{cases} \tag{8}$$

4.2 FDAs Used in the Experiments

The FDAs that were chosen for the experiments are briefly described below. Our choice was due to the possibility of incorporating the proposed changes to the available implementations of these algorithms.

1. The UMDA, which is a FDA with a simple probabilistic model that assumes variables are independent. UMDA generates new solutions by only preserving the proportions of the values of a variable, independently of the values for the remaining variables. This approach can work well even for problems where variables are not completely independent.

2. A tree based FDA (Tree-FDA), this algorithm is similar to the MIMICs version presented in [2], where a tree shaped network approximates the joint distribution. It has been used before in the optimization of binary problems, as well as of functions defined on integers.

3. A mixture of trees FDA (MT-FDA) [11], where a mixtures of trees is used to represent the distribution. Mixtures of trees can represent, condensed in just one model, different patterns of interactions among the variables of the problem.

4.3 Numerical Results

n	N	SP			No SP		
		s	eval.	$\bar{f}$	s	eval.	$\bar{f}$
12	12	46	45.2	11.39	50	43.7	11.39
12	24	98	73.8	11.98	97	69.8	11.97
12	30	96	81.6	11.95	100	86.3	12.00
36	36	18	214.9	34.31	26	232.5	34.59
36	80	85	409.9	35.83	85	397.9	35.82
36	100	91	466.6	35.91	98	483.9	35.98
100	500	47	3886.8	98.95	48	3826.7	98.76
100	800	70	5810.9	99.56	77	5697.8	99.60
100	1000	81	6920.0	99.77	85	6994.0	99.93

Table 4: Comparison of the two selection procedures for the Boltzmann selection using the UMDA in the optimization of the $OneMax$ function

Every experiment consists of 100 runs of the algorithm for the given parameters. The number of times the algorithm found the optimum (s), the average number of functions evaluations (eval.), and the average fitness of the function ($\bar{f}$) are calculated from these experiments. As two different criteria for comparison between SP and No SP we take the number of times the optimum was reached in the 100 runs, and the average fitness. When for one of these criteria the algorithms are tied, the algorithm with the minimum number of average function evaluations is considered as the winner.

Table 4 shows the results of the optimization of function $OneMax$ using the UMDA, when the Boltzmann selection is used with base $b = e$. Results are presented for different number of variables and population sizes. The maximum number of generations was 50. As it can be seen in the table, if we consider the times the optimum was reached No SP was the best 8 times, and SP was the best only once. Considering the average fitness, No SP was better than SP 6 times, and SP was the best the other 3 times. In general, the differences in the results are not significant.

Table 5 shows results of the two selection procedures for Proportional selection, using the Tree-FDA and the MT-FDA. In all the cases the function used was $f_{3deceptive}$, $n = 12$. Column 2 refers to the use of probabilistic priors during the optimization procedure. The use of priors has been proposed in [7] to enhance the performance of FDAs, and to accelerate their convergence. Column 3 refers to the traditional mutation operator used by GAs. In this case we have used a mutation rate of 0.02.

FDA	*Prior*	*Mut*	SP		
			s	eval.	f
Tree-FDA	*no*	*no*	56	940.3	3.95
MT-FDA	*no*	*no*	45	1633.4	3.94
Tree-FDA	*no*	*yes*	72	1606.9	3.97
MT-FDA	*no*	*yes*	67	2053.6	3.97
Tree-FDA	*yes*	*no*	93	2120.6	3.99
MT-FDA	*yes*	*no*	74	2095.0	3.97
FDA	*Prior*	*Mut*	No SP		
Tree-FDA	*no*	*no*	46	820.5	3.94
MT-FDA	*no*	*no*	45	1494.3	3.94
Tree-FDA	*no*	*yes*	68	1789.0	3.97
MT-FDA	*no*	*yes*	55	2225.2	3.95
Tree-FDA	*yes*	*no*	88	2324.0	3.99
MT-FDA	*yes*	*no*	70	2146.6	3.96

Table 5: Comparison of the two selection procedures for the Proportional selection, using the Tree-FDA and the MT-FDA in the optimization of the $f_{3deceptive}$ function under different conditions

Function	n	N	*FDA*	SP			No SP		
				s	eval.	f	s	eval.	f
Checkerboard	25	300	UMDA	93	8739.5	35.87	81	9052.2	35.64
			Tree-FDA	91	6174.9	35.91	93	6517.9	35.93
			MT-FDA	93	10620.3	35.93	84	11548.1	35.83
$f_{deceptive3}$	24	300	UMDA	8	8672.0	22.08	7	7476.0	21.89
			Tree-FDA	73	5993.3	23.67	63	6185.1	23.57
			MT-FDA	53	11007.6	23.41	53	11018.9	23.43
symmetric	24	200	UMDA	99	6334.8	23.99	100	6576.0	24.00
			Tree-FDA	90	4830.1	23.88	89	4680.9	23.88
			MT-FDA	87	8278.9	23.86	86	8590.4	23.84

Table 6: Comparison of the two selection procedures for the Proportional selection using different FDAs and functions

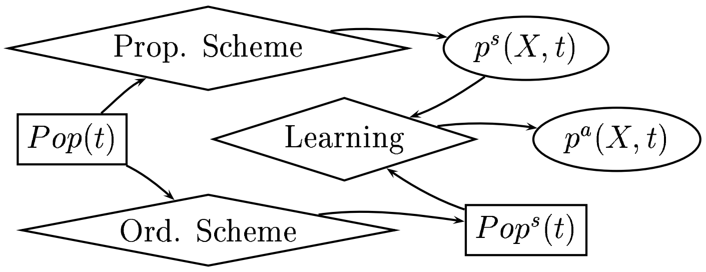

Figure 1: Alternative ways for the selection operator in FDAs. $Pop(t)$: populations at time t; $p^s(X,t)$, $p^a(X,t)$: Join probabilities determined by selection and the graphical model approximations.

The idea of this experiment is to study if there exist significant differences in the behavior of the algorithms when adding a perturbation to the population (in this example by means of priors and mutation). Considering the number of times the optimum was reached, SP was the best in 5 of the 6 experiments. The same happens if the average fitness criterion is considered. We hypothesize that in this case the stochastic noise caused by the use of SUS contributed to infuse a convenient diversity to the generated population.

Table 6 presents the results for functions $f_{deceptive3}$, *Checkerboard*, and *symmetric*. In this experiment a maximum of 60 generations was allowed. Considering the number of times the optimum was reached SP was the best in 7 of the 9 experiments. The picture is less clear if we analyze the average fitness because SP was the best only in 5 of the 9 cases.

5 Conclusions

The main result achieved in this paper is to show that new possibilities of doing selection arise when crossover operators are replaced by probabilistic modeling of the solutions. Basically, we have proven that when using Proportional and Boltzmann selection, no selected population needs to be constructed in order to generate the individuals in the next generation, according to the selection probabilities. In figure 1 we present how Proportional and Ordinal based selection schemes can be inserted in FDAs, and related with their other components.

We have conducted a number of experiments to evaluate what is the influence of using sampling methods like SUS when they are not needed. Our preliminary results show that the use of SUS, although redundant, does not significantly deteriorate the results, and in some cases can improve them. This is probably due to the side effect that the random noise associated to the procedure for sampling the selection probabilities can have in introducing diversity to the search. In this paper it has been shown that the learning of the probabilistic model from the joint selection probability distribution can be done also for Ordinal based schemes. We expect that our results have shed some light to the differences that exist between GAs and FDAs.

References

[1] J. E. Baker. Reducing bias and inefficiency in the selection algorithm. In *Proceedings of the Second International Conference on Genetic Algorithms*, pages 14–21. Lawrence Erlbaum Associates (Hillsdale), 1987.

[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] D. E. Goldberg. *Genetic algorithms in search, optimization, and machine learning.* Addison-Wesley, Reading, MA, 1989.

[4] J. H. Holland. *Adaptation in natural and artificial systems.* University of Michigan Press, Ann Arbor, MI, 1975.

[5] P. Larrañaga and J. A. Lozano. *Estimation Distribution Algorithms. A new tool for Evolutionary Optimization.* Kluwer Academic Publishers, Boston/Dordrecht/London, 2002.

[6] T. Mahnig and H. Mühlenbein. Comparing the adaptive Boltzmann selection schedule SDS to truncation selection. In *Evolutionary Computation and Probabilistic Graphical Models. Proceedings of the Third Symposium on Adaptive Systems (ISAS-2001)*, pages 121–128, Habana, Cuba, March 2001.

[7] T. Mahnig and H. Mühlenbein. Optimal mutation rate using Bayesian priors for Estimation of Distribution Algorithms. In K. Steinhöfel, editor, *Proceedings of the First Symposium on Stochastic Algorithms: Foundations and Applications, SAGA-2001*, volume 2264 of *Lecture Notes in Computer Science*, pages 33–48. Springer, 2001.

[8] H. Mühlenbein and T. Mahnig. Convergence theory and applications of the Factorized Distribution Algorithm. *Journal of Computing and Information Technology*, 7(1):19–32, 1998.

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

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

[11] R. Santana, A. Ochoa, and M. R. Soto. The Mixture of Trees Factorized Distribution Algorithm. In L. Spector, E. Goodman, A. Wu, W. Langdon, H. Voigt, M. Gen, S. Sen, M. Dorigo, S. Pezeshk, M. Garzon, and E. Burke, editors, *Proceedings of the Genetic and Evolutionary Computation Conference GECCO-2001*, pages 543–550, San Francisco, CA, 2001. Morgan Kaufmann Publishers.

[12] K. Sastry and D. E. Goldberg. Modeling tournament selection with replacement using apparent added noise. In *Intelligent Engineering Systems Through Artificial Neural Networks. Proceedings of the Conference ANNIE 2001*, volume 2, pages 129–134, 2001.