Evolutionary Algorithms for Dynamic Optimization Problems: An approach using Evolutionary Theory and the Incident Edge Model.

Roberto Santana, Alberto Ochoa, Marta R. Soto
* Center of Mathematics and Theoretical Physics, ICIMAF,
Calle 15 e/ C y D, Vedado, Havana, Cuba
Rsantana@cidet.icmf.inf.cu

Abstract

In this paper we analyze the use of Evolutionary Algorithms (EA's) for Dynamic Optimization Problems (DOP). We show how research on Evolutionary Theory can throw light to the question of characterizing dynamic problems. We present arguments in favor of using fitness function models as benchmark for the study of DOP. As an example we present the Incident Edge Model, and discuss how different kinds of dynamics for problems defined on graphs can be translated to the model. Finally we apply an EA, the Constraint Univariate Marginal Distribution Algorithm, for the problem of finding the spanning trees of graphs that change through time. We show that efficient EA's should be able to employ information about changes in the environment to guide the search.

Research done in the field of evolutionary theory can elucidate the way in which evolutionary algorithms must deal with different dynamic optimization problems. In evolutionary theory, evolutionary scenarios have been divided from the viewpoint of a single species that adapts in an environment. Three classes have been defined[9] :

1) Evolution in a constant environment
2) Evolution in a variable environment without feedback to the environment
3) Evolution with feedback to the environment

We will use an analogy between the behavior of an EA's for a given problem and the evolution of a specie in an environment. The type of problem (constant of dynamic) determines if the EA is evolving in a constant or dynamic environment. From this analogy, and using the previous classification of evolutionary scenarios, we can group EA's according to the type of evolution they exhibit. We will consider that changes in the environment can be manifested in the EA in one of the following ways:

4) Changes in the parameters of the EA
5) Changes in the correspondence genotype-phenotype
6) Changes in the correspondence phenotype-fitness function.

Theses changes are not always due to the dynamic characteristics of the problem (e.g. Functions with dynamic parameters have been used for the solution of constant problems). Nevertheless in this paper we will consider that when the previous changes are present in the evolution is because they have been provoked by some time dependent characteristic of the problem.

Those EA's for which the correspondence genotype-phenotype-fitness function is the same along the evolution are related to the first class of evolutionary scenarios. We will group in the second class those EA's for which there exist changes in the relationship genotype-phenotype, phenotype-fitness function, or both, during the evolution, and the EA do not influence the environment. This is the class of EA's we are interested to deal with in this paper. In the third class would be those EA's able to display an internal representation of the environment, and to interact with it in both directions (e.g. Classifiers Systems (CS's)).

One difficult task for evolutionary optimization approaches to DOP is defining appropriated benchmarks. In Evolutionary Theory have been proposed the oscillating NK models of fitness landscapes for the study of environmental changes. We will present a similar approach for EA's. For the Classical Kauffman's N-K model [2] the total fitness contribution of the string $x=(x_1,...,x_n)$ is defined as an averaged sum of fitness contributions. In the NK oscillating models the fitness contributions of the substrings ($f_i(s_i)$) is time dependent. Although the family of oscillating NK-landscapes can be used to get a first impression of how evolution in a slowly changing environment behave, it seems to be of little help as a good benchmark for the analysis of EA's for DOP. For this purpose we use instead the Incident Edge Model introduced in [8].

The Incident Edge Model is defined from a simple graph G=<V,E>. To each edge belonging to |E| we map a binary variable x_i in the string $x=(x_1,...,x_{|E|})$. The total fitness contribution f(x) is the sum of n subfunctions defined on the substrings of x and another function v(x) defined on x.

$$f(x) = u(x) + v(x)$$

$$f(x) = \sum_{i=1}^{n} u_i(s_i) + v(x)$$

The distinctive characteristic of the Incident Edge Model is the way the substrings s_i are selected: Each substring s_i includes those variables mapping the edges incident to the vertex i in G. As a consequence the length of the substring s_i is equal to the degree of vertex v_i in the graph. Subfunctions u_i are conveniently chosen. The decomposition of f(x) in the terms u(x) and v(x) reflects the case, commonly found in problems defined on graphs, when the fitness of a solution can be determined by considering first some necessary conditions to be fulfilled by the subsolutions, and then analyzing a measure of the quality of the complete solution. We will call any instance of the IEM as an incident edge function.

Compared to random fitness models, the IEM allows a clearer understanding of the relationship between the structure of the problem and the resultant fitness landscape. For DOP it is possible to represent dynamic changes in the problem using the different components of the model. In this paper we present an example where substrings s_i vary along the evolution. Time dependent functions u(x) and v(x) can be easily constructed.

In [7] we have defined an incident edge function to find the minimum weighted spanning tree of a graph. A **connected grap**h is a graph that contains a chain for each pair x,y of distinct vertices[1]. G is called **tree** if it is connected and contains no cycle. A graph T=(V', E') is called **spanning tree** of G if V'=V, E' E, and T is a tree. If we associate to every edge of the graph a weight, the minimum weighted spanning tree problem consists of finding the spanning tree whose edges' weight sum is minimal. There exist several randomized efficient algorithms to solve this problem, this fact makes of it a good benchmark for studying the behavior of EA's for this problem. Nevertheless here we focused on a different question, we extend the constant problem to that of finding the spanning tree of a graph that changes in time. In every time step, p edges are removed from the graph and other p are added, the graph remains always connected.

The coding is as follows: We first associate the i component of the vector to the i edge in E. For the component i, value 1 stands for "select the corresponding edge", whereas bit value 0 stands for "do not select the corresponding edge". Then, each vector will represent a subgraph G' of G such that all the edges of G' are those components of the vector with value 1. As all the spanning trees of a graph with |V| vertices have exactly |V|-1 edges, we can constrain the search to those vectors with exactly |V|-1components set to 1. We simplify the initial problem of finding the minimum weight spanning tree to the problem of finding any spanning tree of the graph, this is equivalent to have a problem where the weights of edges are all the same. Our final function f(x) will consider as separated factors how many vertices are connected and how many edges belong to cycles. We look to maximize the following function correspondent to the IEM:

$f(x) = \sum_{i=1}^{n} u_i(s_i) + v(x)$, where $u_i(s_i) = \frac{1}{|V|}$ if $\sum_{l=1}^{|xj_l|} x_l > 0$, otherwise $u_i(s_i) = 0$. EE is the number of edges that do not belong to cycles and $v(x) = \frac{EE}{|V|-1}$.

Function u(x) rewards 1/|V| to all the vertices that are connected to the tree. When the subgraph represented in the vector is fully connected u(x)=1. Function v(x) rewards those edges that are not in any cycle. When there are cycles v(x) =1. Spanning trees will receive a maximum fitness value of 2. For the problem we are analyzing, there is a change in the relation genotype-phenotype along the evolution. Every time that p edges are removed from the graph, the p variables that mapped these edges will map the new p incoming edges. As a consequence for the genotype x in time t, we will have another phenotype in time t+1, the change in the phenotype causes a change in the evaluation of the function. Nevertheless, depending on the value p, it is possible that good solutions in time t will be still useful in time t+1. Given the additive component of the IEM it is expected that changes in the fitness function will not be dramatic.

For the optimization of function f(x) we will use an EA that belongs to the class of Estimation Distribution Algorithms(EDA)[3]. These algorithms instead of using crossover and mutation operators, estimate the probability distribution of points in the selected set, and uses this information to generate new points. EDA have recently experienced a fast development [3][4][5]. Given the constrained characteristic of our problem we will used the Constraint Univariate Marginal Distribution Algorithm (CUMDA)[6].

CUMDA

Step 1: Set T 1 Generate N>>0 points randomly

Step 2: Select k N points according to a selection method. Compute the marginal frequencies $p^s_i(x_i,t)$ of the selected set.

Step 3: Given S= $\sum_{i=1}^{n} p^s(x_i = 1, t)$, set $q_i = p^s(x_i = 1,t)/S$, For each individual set r of the n variables sampled without replacement with probabilities qi. The remainder variables are set to 0.

Step 4: If the termination is not met, go to Step 2.

Defining the experimental settings for DOP is a very complex task, given the space constraints we face in this paper we will present a compact resume of the preliminary results that have been achieved. Initial experiments were conducted on randomly connected graphs with 25 vertices and 70 edges. We begin the algorithm by searching the spanning tree of the initial graph. A population of feasible solutions is generated, we apply CUMDA until the maximum of the function is reached (i.e. one spanning tree has been found), then we change the graph by removing and adding simultaneously p edges (Change in the graph is reflected in the coding) . From the population were the last optimum is contained we restart the search for the new spanning tree. In this way, the algorithm is forced to track the movement of the optimum every time that there is a change in the graph

In our experiments we corroborated that the number of edges removals (p) is a critical parameter of the problem. The Evolutionary Algorithm is more likely to track the optimum when there is not a big gap between two consecutive graphs . On the other hand, although the population where the last optimum was found is a good starting point to find the next one, it is also important guaranteeing an adequate diversity in this population.

Bibliography

[1] Berge:1985 Claude Berge (1985): Graphs, North-Holland, Mathematical Library, New York.

[2] Kauffman S. (1993) : Origins of Order. Oxford University Press.

[3] Mühlenbein, H. (1998). The equation for response to selection and its use for prediction, Evolutionary Computation 5, pp. 303--346.

[4] Mühlenbein, H.; Mahnig, T., Ochoa A. (1998). Schemata, Distributions and Graphical Models in Evolutionary Optimization, to appear in Journal of Heuristics Vol 5, No. 2. Also at http://set.gmd.de/AS/ga.publineu.html#index.

[5] Pelikan, M.; Goldberg D. E., Cantú-Paz E.(1998). Linkage problem, Distribution estimation and Bayesian Estimation Networks. Illegal Report 98013. November 1998.

[6] Santana R., Ochoa A . (1999). Dealing with constraints with Estimation Distribution Algorithms: The Univariate Case. Proceedings of the II Second Symposium on Artificial Intelligence. ISAS' 99 pp. 378-384
[7] Santana R., Ochoa A. Soto M. R. (1999). Dealings with constraints with Estimation Distribution Algorithms: The Multivariate Case. Research Report ICIMAF. ISSN 0138-8916
[8] Santana R., Ponce de León, E. Ochoa A . (1999). The Incident Edge Model. Proceedings of the II Second Symposium on Artificial Intelligence. ISAS' 99 pp. 352-359
[9] Wilke C. O. (1998) Evolution in time-dependent fitness landscapes. Internal Report No IR-INI 98-09, Nov. 1998. Institut für Neuroinformatik, Ruhr-Universität Bochum. ISSN 0943-2752.