Evolutionary Algorithms on a Self-Organized Dynamic Lattice

Paul Halpern*
Department of Physics
Philadelphia College of Pharmacy and Science
600 S. 43rd St., Philadelphia, Pa. 19104

March 25, 1997

Abstract

We investigate a new optimization model, one which places the Evolution Strategy approach of Rechenberg and Schwefel on a changing lattice of site connections – namely a structurally dynamic cellular automaton. After seeding 1000-site lattices with real values, one per site, we observe these site values' behavior over time, subject to evolutionary survival of the fittest. In addition to the standard Darwinian optimization processes of selection, breeding, mutation and replacement, we add the feature of a self-selective neighborhood structure. Each generation, the link structure of the lattice is organized by the sites themselves,

*Supported by the German Fulbright Commission and the Institute for Theoretical Physics, Humboldt University Berlin

based upon fitness criterea. In our model, we have found, for various parameters, an overall tendency to greater average fitness. We have discovered, as well, for particular mutation rates, evidence of phase transitions in lattice link structure.

1 Introduction

In recent decades a number of optimization techniques have been developed that exploit Darwinian notions of natural selection in an effort to arrive most efficiently at the fittest solution to a problem. Falling under the general heading of Evolutionary Algorithms, these include the Genetic Algorithm approach (Holland, 1975) and Evolution Strategies schemes (Rechenberg, 1973; Schwefel, 1975; Schwefel, 1995). These methods have achieved considerable success in providing speedy pathways to optimal results.

Typically, such an algorithm begins (Bäck, Hoffmeister and Schwefel, 1991) by defining an ensemble of vectors $\{x\} : x \in M \subseteq R^n$, where M constitutes the set of all considered solutions to a problem (in the case of Genetic Algorithms, these vectors are generally strings of binary numbers) the subset M, and a fitness function $f(x)$ representing the worth of a given particular solution. Then the goal of the algorithm is to find the global optimum, x^*, defined (in the case of maximizing fitness) such that:

$$\forall x \in M : f(x) \leq f(x^*) = f^* \tag{1}$$

Because $min(f(x)) = max(-f(x))$, this definition is general, and applies to minimization as well as maximimzation problems.

Optimization occurs in each of these models through specific processes of selection, recombination, mutation and replacement (though in some models not all of these steps are undertaken). In most cases selection is made by choosing elements for recombination from the entire available set of vectors, namely M.

Mühlenbein has pointed out, in the case of Genetic Algorithms, that the fact that for each generation the search space consists of the entire population pool introduces some danger of premature convergence to a sub-optimal solution (Mühlenbein, 1991). This difficulty arises in situations in which a population evolves toward a smaller peak in the fitness landscape, becomes more and more uniform and consequently fails to notice the largest peak. To avoid this situation, he suggests the introduction of a spatial population structure. Spatially isolated populations, he feels, would have the opportunity of developing toward different peaks. This would greatly improve the chances that a segment of the population would sample, and eventually hone in on, the fittest solution to the problem (the highest peak of the landscape).

Mühlenbein's Parallel Genetic Algorithm model (Mühlenbein, 1989; Mühlenbein, 1991) provides a means of allowing for spatial isolation. In his system, diversification is achieved by restricting fitness and mating to neighborhoods called *demes*. This name, borrowed from population genetics, refers to geographically distinct communities in which organisms develop along disparate lines, rather than mixing with the general pool. *Demes* evolve as separate sub-populations, occassionally exchanging genetic material through the process of migration. In this manner Mühlenbein draws upon Wright's notion (Wright, 1932) that evolution proceeds best when small sub-populations are isolated and left free to stumble upon the fittest peak.

We take a somewhat different approach to the issue of the spatial of population. We place our Evolutionary Algorithm within the structure of a flexible lattice of self-selective connections, namely a Topological Cellular Automaton. The advantage we hope to gain is to allow each organism the choice of communities at any given time step. The choice of neighborhoods in our model is therefore dynamic, rather than fixed, determined exclusively by the participants.

A Topological Cellular Automaton (Ilachinski and Halpern, 1987) is a variation of the standard Cellular Automaton model (Wolfram, 1986) in which link connections between lattice sites are added or removed each generation based upon the values of neighboring sites. Thus the metric structure of these models is entirely self-organized. A wide range of long-term behavior has been found for various initial configurations and connection rules, including growth and decay in link connectivity, as well as periodicity and approach to stable fractional dimensionality (Halpern and Caltagirone, 1990) Applications of Topological Cellular Automata are diverse, including models of phase transformations in physical and chemical systems. (Rosé, 1993; Rosé, Hempel and Schimansky-Geier, 1994).

One of the motivations for our inclusion of dynamically changing neighborhoods is to model the tendency for animals to choose mates from neighboring groups, but not from their own immediate families. A well-studied example of this is the propensity for song-birds to select partners with similar, but not identical songs. (Baker and Cunningham, 1985) Following this strategy, they hope to avoid mating with family members, while increasing their odds of finding fit neighbors with whom to procreate. As this principle works in biological systems to increase species fitness, we hope that it will similarly serve well for computational models.

2 Lattice-based Evolution Strategies

We now present the dynamics of our model for Evolution Strategies on a dynamic lattice. Our algorithm can be described as follows:

STEP 0: Define a representation of the problem as a vector space and an associated fitness function mapping each vector onto a number.

STEP 1: Create an initial lattice configuration by randomly connecting a set of N sites with links. The connectivity between sites is encoded in the connection matrix C_{ij}, where i and j each range from 1 to N. Each matrix element is equal to 1 if and only if the two sites i and j are directly connected; otherwise it is equal to 0.

STEP 2: Seed each site i with a single vector of length L, encoded by x_{im}, where m ranges from 1 to L.

STEP 3: Each site surveys its own neighborhood (sites with which it is linked). It ranks each of its nearest neighbors according to fitness. It also ranks each of its next-nearest neighbors according to fitness.

STEP 4: Based upon a decoupling parameter set initially, each site disconnects with a percentage of its least fit nearest neighbors, and forms connection with a percentage of its fittest next-nearest neighbors. In other words, each site forms a new neighborhood based upon fitness criterea.

STEP 5: Each site randomly mates with one of its nearest neighbors. This mating takes place according to the procedure of intermediate recombination. (Bäck, 1996), namely, each offspring consists of the vector average of its two parents.

STEP 6: With small probability (chosen as one of the initial conditions) each offspring has the possibility of becoming mutated. This mutation consists of a small random increment added to or subtracted from the vector. Periodic boundary conditions are applied here. If the mutation is such that it would take the vector out of its designated range, then a suitable correction is either added or subtracted until the vector lies once again within the appropriate span of values.

STEP 7: The set of offspring replaces the least fit members of the general population.

STEP 8: Repeat steps 3 through 7, for a fixed number of time steps, or until a suitable convergence criterion is met.

The following parameters are set initially before each run begins:

N: Size of lattice

L: Length of each vector

M: Initial number of links per site

C: Connection fraction – Fraction of fittest next-nearest neighbors with which each site must connect each time step.

D: Disconnect fraction – Fraction of least fit neighbors with which each site must disconnect each time step.

P: Probability of mutation – Chance that a given offspring is subject to mutation

T: Total number of time steps

For our trial runs on DEC workstations we used lattices of N= 1000 sites each. For the purpose of simplicity, we chose vectors of unit length, with values ranging from 0 to 1.

We utilized a fitness function with a single peak:

$$f(x) = 1 - x^2 \tag{2}$$

We set the amount of mutation to range from -.05 to +.05, for any given time step. Periodic boundary conditions were established such that if a mutation threatened to bring a vector's value outside the range of 0 to 1 then a unit value was either added or subtracted as appropriate. The mutation rate, fixed for an entire run, was generally set to be P= .001. This was varied in some runs, however, for the sake of comparison.

The decoupling and coupling criterea were generally set at D = C = 0.1. (10% of the least fit nearest neighbors were disconnected, and 10% of the fittest next-nearest neighbors were disconnected per site at any given time step.) Once again, though, these rates were altered for particular runs for the sake of comparison.

We ran this algorithm for T= 1000 time steps per run, with 10 different randomly chosen initial configurations.

3 Site Value Behavior and Connectivity

As in the case of (non-lattice based) Evolution Strategies and Parallel Genetic Algorithms, we found that our model served well to optimize the chosen fitness function. For our default initial parameters, listed above, we found that our algorithm brought the average fitness to its maximum value within 200 time steps. Varying the parameters from their initial values seemed in most cases to have a deleterious effect. For example, as the disconnect fraction D was increased, the time until the model reached its fitness peak grew longer (see fig. 1).

We speculate that the reason for this change in performance stems from the fact that, for higher values of D, neighborhood connections are broken down too quickly. For optimum performance, one needs to build up links first before breaking them down. Increasing the mutation probability much above 0.01 also seemed to hinder, rather than help, the model. (see fig. 2).

Of more interest than the fitness growth itself is the connection between model performance and lattice connectivity. We found that the fastest hillclimbing took place when lattice connectivity increased, then decreased, and finally levelled off. The first stage, of increasing connections, allows for the establishment of the fittest possible communities. But then, once these fit neighborhoods are established, connections can be severed with the poorer performers.

We found two different sorts of approaches toward high connectivity. The first sort, for low mutation probabilities, is monotonic growth over time (see fig. 3).

The other type of behavior, of more interest to those studying complex systems, is that of a catastrophic phase transition between lower and higher degrees of connectivity. We have found this sort of behavior to occur when P is set to be approximately 0.05 (and within a small window of this value). There is a sharp increase in number of links per site between generations 350 and 450. (See fig. 4).

Plotting out the overall link distribution over time, one obtains a sense of why this phase transition occurs. During the early part of the model's evolution, most sites have few links. Therefore, there is still considerable room for growth to occur. (See figures 5 and 6).

As lattice growth continues to occur, the number of offspring produced rises. With this increase, the total number of mutated offspring tends to grow as well. This is especially the case for relatively high values of P (such as those that we studied). Soon, a number ofmutations begin to take place that radically alter the fitness of particular offspring. These offspring suddenly jump in fitness (the periodic boundary conditions make this possible). Because of this increase, suddenly many sites want to be connected to these fitter vectors. The level of connectivity dramatically increases, as many sites assume new connections. In short, a bifurcation occurs in which some of the sites maintain their original level of connectivity, while others increase theirs drastically. This leads to a phase transition in the average connectivity. (See figures 7 and 8).

4 Conclusion

In an earlier version of this model (Halpern, 1996), a lattice-based variation of genetic algorithms, we noted that lattice connectivity can lead to greater optimization. Here we see a similar effect. Placing evolution strategies on a lattice can serve to organize a physical space in addition to a vector space, leading to an overall increase in fitness. Furthermore, it can result in an interesting dynamics, including the possibility of phase transitions.

Possible applications of this model include cases in which flexible connectivity would benefit optimization. One natural arena in which such a procedure might be found useful is in the domain of parallel computing. Several units, working on a particular problem, might use fitness criterea to determine whether or not it would be best to make or sever a connection. Then optimization might proceed through the best networking of these units.

In future studies of this model, we plan to look at a more complex range of fitness functions. We will continue to examine how placing an evolutionary algorithm on a lattice affects the overall performance of the algorithm and the connectivity of the lattice.

Acknowledgements: We wish to thank Werner Ebeling, Helge Rosé and Torsten Asselmeyer for their help, suggestions, and for many useful discussions.

References

[1] Bäck, Thomas, 1996, Evolutionary Algorithms in Theory and Practice, (Oxford University Press, New York).

[2] Bäck, Thomas, Hoffmeister, Frank, and Schwefel, H. -P., 1991, A survey of evolution strategies. In R. K. Belew and L. B. Booker, editors, Proceedings of the Fourth International Conference on Genetic Algorithms and their Applications, (Morgan Kaufmann, San Mateo).

[3] Baker, Myron, and Cunningham, Michael, 1985, The Biology of Bird-Song Dialects, Behavioral and Brain Sciences 8, 85–133.

[4] Halpern, Paul, 1989, Sticks and stones: A guide to structurally dynamic cellular automata, American Journal of Physics 57, 405–408.

[5] Halpern, Paul, 1996, Community-Based Genetic Algorithms, DYNAMIK - EVOLUTION - STRUKTUREN, (Verlag Dr. Koester, Berlin).

[6] Halpern, Paul, and Caltagirone, Gaetano, 1990, Behavior of Topological Cellular Automata, Complex Systems 4, 623–651.

[7] Holland, John H., 1975, Adaptation in Natural and Artificial Systems, (University of Michigan Press, Ann Arbor).

[8] Ilachinski, Andrew, and Halpern, Paul, 1987, Structurally Dynamic Cellular Automata, Complex Systems 1, 503–528.

[9] Mühlenbein, Heinz, 1989, Parallel genetic algorithm, population dynamics and combinatorial optimization. In H. Schaffer, editor, Third International Conference on Genetic Algorithms, (Morgan Kaufmann, San Mateo).

[10] Mühlenbein, Heinz, 1991, Evolution in time and space – the parallel genetic algorithm. In G. Rawlins, editor, Foundations of Genetic Algorithms, (Morgan Kaufmann, San Mateo).

[11] Rechenberg, Ingo, 1973, Evolutionsstrategie: Optimierung technischer Systeme nach Prinzipien der biologischen Evolution, (Frommann-Holzboog, Stuttgart).

[12] Rosé, Helge, 1993, Topologische Zelluläre Automaten, Diplomarbeit, Humboldt University Berlin.

[13] Rosé, Helge, Hempel, Harald, and Schimansky-Geier, Lutz, 1994, Stochastic Dynamics of Catalytic CO Oxidation on Pt(100), Physica A 206, 421–440.

[14] Schwefel, H. -P., 1975, Evolutionstrategie und numerische Optimierung, Dissertation, Technical University of Berlin.

[15] Schwefel, H. -P., 1995, Evolution and Optimum Seeking, Sixth-Generation Computer Technology Series, (Wiley, New York).

[16] Wolfram, Stephen, 1986, Theory and Practice of Cellular Automata, (World Scientific, Singapore).

[17] Wright, S., 1932, The roles of mutation, inbreeding, crossbreeding and selection in evolution, Proceedings of the 6th International Congress on Genetics.