Computer Science Department Technical Report
University of California
Los Angeles, CA 90024-1596

SELECTION IN MASSIVELY PARALLEL GENETIC ALGORITHMS

Robert J. Collins
David R. Jefferson

July 1991
CSD-910052

Selection in Massively Parallel Genetic Algorithms*

Robert J. Collins
David R. Jefferson
Artificial Life Laboratory
Department of Computer Science
University of California, Los Angeles
Los Angeles, CA 90024

June 7, 1991

Abstract

The availability of massively parallel computers makes it possible to apply genetic algorithms to large populations and very complex applications. Among these applications are studies of natural evolution in the emerging field of artificial life, which place special demands on the genetic algorithm. In this paper, we characterize the difference between panmictic and local selection/mating schemes in terms of diversity of alleles, diversity of genotypes, the inbreeding coefficient, and the speed and robustness of the genetic algorithm. Based on these metrics, local mating appears to not only be superior to panmictic for artificial evolutionary simulations, but also for more traditional applications of genetic algorithms.

*To appear in *Proceedings of the Fourth International Conference on Genetic Algorithms.* Morgan Kaufmann, (in press).

1 Introduction

The availability of powerful supercomputers such as the Connection Machine [14] means that genetic algorithms are now applied to larger and more difficult optimization problems (e.g. [4], where the search space consists of 2^{25590} points). Some of our recent artificial life work [15, 5, 3, 4] has involved massively parallel genetic algorithms characterized by large populations, enormous search spaces, and fitness functions that change through time.

These simulated evolution applications place special demands on the genetic algorithm. The simulations generally attempt to model the evolution of populations of tens of thousands of artificial organisms in a simulated environment over a period of thousands of generations. The ecosystem in which the fitness of each individual is evaluated can potentially include both direct and indirect interactions with other members of the population, members of coevolving populations, the background environment, etc. In addition, the environment and selection criteria may change both during a generation and over a period of many generations (and may be different in different parts of the simulated world). Such applications require a genetic algorithm that is able to simultaneously explore a wide range of genotypes and can maintain enough genetic diversity to respond to changing conditions.

Genetic algorithms that use panmictic selection and mating (where any individual can potentially mate with any other) typically converge on a single peak of multimodal functions, even when several solutions of equal quality exist [8]. Genetic convergence is a serious problem when the adaptive landscape is constantly changing as it does in both natural and artificial ecosystems. Crowding, sharing, and restrictive mating are modifications to panmictic selection schemes that have been proposed to deal with the problem of convergence, and thus allow the population to simultaneously contain individuals on more than one peak in the adaptive landscape [7, 10, 8]. These modificatior re motivated by the natural phenomena of niches, species, and assortative mati ut they make use of global knowledge of the population, phenotypic distance me res, and global selection and mating, and thus are not well suited for parallel implementation. Rather than attempting to directly implement these natural phenomena, we exploit the fact that they are emergent properties of local mating.

In this paper, we introduce several metrics for evolving populations, and use them to characterize the differences between local and panmictic selection/mating schemes. Although our target applications involve the evolution of artificial organisms, for computational convenience we have performed this study using the optimization of graph partitions as the evolutionary task.

The results of this empirical study indicate that local mating is more appropriate for artificial evolution than the panmictic mating schemes that are usually used in genetic algorithms. In addition, local mating appears to be superior to panmictic mating, even when considering traditional applications. Local mating (a) finds optimal solutions in faster; (b) typically finds multiple optimal solutions in the same run; and (c) is much more robust. These results are encouraging, but are based on a single optimization problem, a single recombination rate, a single mutation rate, one local mating algorithm, and large populations. Further investigation seems both

appropriate and necessary.

2 Graph Partitioning

The graph partitioning problem [1] is to find two subsets V_0 and V_1 of the set V of vertices of a fixed graph G such that $V = V_0 \cup V_1$, $V_0 \cap V_1 = \emptyset$, $|V_0| = |V_1|$ (which assumes that $|V|$ is even), and the cut size (the number of edges with one endpoint in V_0 and the other in V_1) is minimized. We represent a partition of G by a string S of $|V|$ bits, such that vertex $i \in V_{S[i]}$ (the value of each bit indicates to which subset the corresponding vertex belongs).

In order to incorporate this problem statement into a genetic algorithm, we use the fitness function defined by Ackley [1], which has a penalty that is quadratic in the imbalance of a partition:

$$f(x) = -cutsize(x) - 0.1(Z(x) - O(x))^2$$

where x is an arbitrary bit string, $Z(x)$ is the number of 0's in x, and $O(x)$ is the number of 1's in x. The cutsize and penalty term are given negative signs to make this a function maximization problem (with maximum value of 0).

There is little structure in small random graphs, so good partitions are relatively easy to find by simple hill climbing methods [1]. To make the problem harder, a "clumpy" or *multilevel* graph is used (a clump is a strongly connected group of vertices). We adopt Ackley's multilevel graphs, where a clump consists of four fully connected vertices, and each graph consists of two identical, disconnected pieces (so the optimal partition has fitness 0). Each of the connected components of the graph connects its clumps in a hypercube. For example, a 64 vertex multilevel graph consists of 16 clumps. Each of the two connected components is cube with a clump in each of the 8 corners.

We place the vertices within each clump and each connected component consecutively on the string S. This means that there are two optimal solutions to the multilevel graph partitioning problem: $\frac{|V|}{2}$ 0's followed by $\frac{|V|}{2}$ 1's, and the bitwise complement.

To the genetic algorithm, the clumps act as short, low–order and highly fit schemata. thus the search is quickly focused on partitions that do not cut clumps. In order to continue the search for an optimal partition, it is necessary to move clumps across the partition via recombination. Recombination is necessary because moving a clump across the cut one vertex at a time (i.e. by point mutations) results in a dramatic decrease in fitness. The result is that the multilevel graph partitioning problem is difficult for genetic algorithms, unless convergence can be avoided (recombination requires diversity in order to have an impact, and convergence implies little diversity).

3 Evolution Metrics

In this section, we introduce four metrics that we use to characterize evolving populations. These particular metrics were chosen in order to measure and highlight the

differences in the evolutionary dynamics of structured (locally mating) and panmictic (globally mating) populations.

3.1 Diversity of Alleles

We measure the allele diversity of a population by comparing the observed allele frequencies to the expected allele frequencies of a maximally diverse population. In this paper, we consider loci with exactly 2 alleles, so the maximum diversity occurs when each allele frequency is 0.5. We define the diversity of alleles of a population at locus (bit position) i as

$$D_i = 1 - 4(0.5 - observed\_freq_i)^2$$

where $observed\_freq_i$ is the frequency of the 0 allele at locus i. D_i can range from 0, which indicates complete fixation (convergence), to 1 which indicates the maximum possible genetic diversity. D, the genetic diversity of the population for the entire genome S, is

$$D = \frac{\sum D_i}{|S|}$$

which also ranges from 0 (fixation) to 1 (maximal diversity).

3.2 Diversity of Genotypes

We measure genotypic diversity by choosing a random sample (without replacement) of 10 loci and count the number of unique genotypes (with respect to these loci) represented in the population. A new set of loci is sampled each generation. This provides a measure of the breadth of the genetic search.

3.3 Inbreeding Coefficient

We define inbreeding to be the mating of two individuals who are more similar to each other than would be expected if mates were chosen randomly. The primary effect of inbreeding is to decrease the frequency of heterozygous genotypes [13]. A (diploid) genotype is heterozygous at a locus if the two haplotypes contain different alleles at that locus. The inbreeding coefficient F is calculated by comparing the actual proportion of heterozygous genotypes in the population with the proportion that would occur under random mating. Heterozygosity (and thus the inbreeding coefficient) is defined in terms of diploid organisms, but with few exceptions genetic algorithms use haploid strings. Only during sexual reproduction are our individuals diploid, so this is when we measure F.

Let H be the observed proportion of heterozygous genotypes, and H_0 be the expected heterozygosity in a randomly mating population with the same allele frequencies. The standard form for F is

$$F = \frac{H_0 - H}{H_0}$$

In this paper, there are two alleles (0 and 1) at each locus, so (from the Hardy-Weinberg principle) $H_0 = 2p(1-p)$, where p is the frequency of the 0 allele.

3.4 Speed and Robustness

We measure the speed of evolution achieved by a genetic algorithm in two ways.

- Number of generations required to discover an acceptable solution. This measure allows implementation independent comparisons.

- Computational time required to discover an acceptable solution. This measure takes into account the varying computational costs of the reproduction portion of the genetic algorithm.

We define the robustness to be the fraction of runs that find at least one acceptable solution. In this paper, we define an acceptable solution as either of the two optimal graph partitions. Since we do not always discover acceptable solutions, we stop such runs at 1000 generations. We report speed in terms of the median run (of all runs, including those that do not find optimal partitions).

4 Selection Schemes

4.1 Local Selection

One of the basic assumptions of Wright's shifting balance theory of evolution is that spatial structure exists in large populations [19, 6]. The structure is in the form of *demes*, or semi-isolated subpopulations, with relatively thorough gene mixing within a deme, but restricted gene flow between demes. One way that demes can form in a continuous population and environment is ***isolation by distance***: the probability that a given parent will produce an offspring at a given location is a fast-declining function of the geographical distance between the parent and offspring locations. Wright's theory of evolution has played a role in the design of several parallel genetic algorithms [17, 11, 16, 4].

To simulate isolation by distance in the selection and mating process, we place the individuals on a toroidal, 1 or 2 dimensional grid, with one organism per grid location. Selection and mating take place locally on this grid, with each individual competing and mating with its nearby neighbors. In our local mating scheme, the two parents of each offspring are the highest scoring individual encountered during two random walks that begin at the offspring location, one parent per walk. The parents are chosen with replacement, so it possible for the same high-scoring individual to be encountered during both random walks, in which case it would act as both parents for the offspring. Deme size (and thus the rate of gene flow) is a function of R, the length of the random walks.

4.2 Stochastic Selection

The most typical panmictic selection strategy that is used in genetic algorithms is known as stochastic selection with replacement (or roulette wheel selection) [9]. We define the probability that individual x is chosen as a parent as

$$P(x) = \frac{f_x}{\sum f_i}$$

The stochastic selection algorithm assumes that all fitness values are non-negative. Because $f_x \leq 0$ for the graph partitioning problem, we adjust the fitness scores for the stochastic selection algorithm

$$f'_x = f_x + |\min f_i|$$

where f_x is the fitness of partition x.

4.3 Linear Rank Selection

Another panmictic selection algorithm is the linear rank method [12]. The linear rank selection algorithm defines the *target sampling rate* (TSR) of an individual x as

$$TSR(x) = Min + (Max - Min)\frac{rank(x)}{N-1}$$

where $rank(x)$ is the index of x when the population is sorted in increasing order based on fitness, and N is the population size. Also, the constraints that $0 \leq TSR(x)$, $\sum TSR(x) = N$, $1 \leq Max \leq 2$, and $Min + Max = 2$ are imposed. The TSR is the number of times an individual should be chosen as a parent for every N sampling operations.

5 Empirical Results

We have compared various genetic algorithms on the multilevel graph partitioning problem. The selection algorithms that we used were local mating in both 1 and 2 dimensional geometries (for $R = \{1, 2, 3, 5, 10, 20, 30\}$), linear rank selection (with $Min = 0.0$ and $Max = 2.0$), and stochastic selection with replacement. The optimization problem is the partitioning of the 64–vertex multilevel graph. The population size N varies over a range from $2^{13} = 8,192$ to $2^{19} = 524,288$ individuals in each generation. In this section, we present only a representative sample of our results, because space constraints prevent us from presenting all of our data here.

The genetic operators include both recombination and mutation. During recombination, crossovers occur with a constant probability of 0.02 between each pair of consecutive bits, so most matings (64%) will result in zero or one crossovers, but some matings may result in many crossovers. In a similar way, point mutations (bit flips) occur with a constant probability of 0.001 per bit, with only 6% of the individuals experiencing one or more mutations.

5.1 Diversity of Alleles

The diversity of alleles over time that is maintained by the four selection algorithms is plotted in Figure 1. Both of the panmictic selection algorithms quickly lose diversity, while both of the local selection schemes maintain a high degree of variation. Although the 1 dimensional local scheme maintains nearly perfect variation, the 2 dimensional algorithm loses a small amount over time. Of the two panmictic selection algorithms, linear ranking loses diversity sooner, and stabilizes at a lower level.

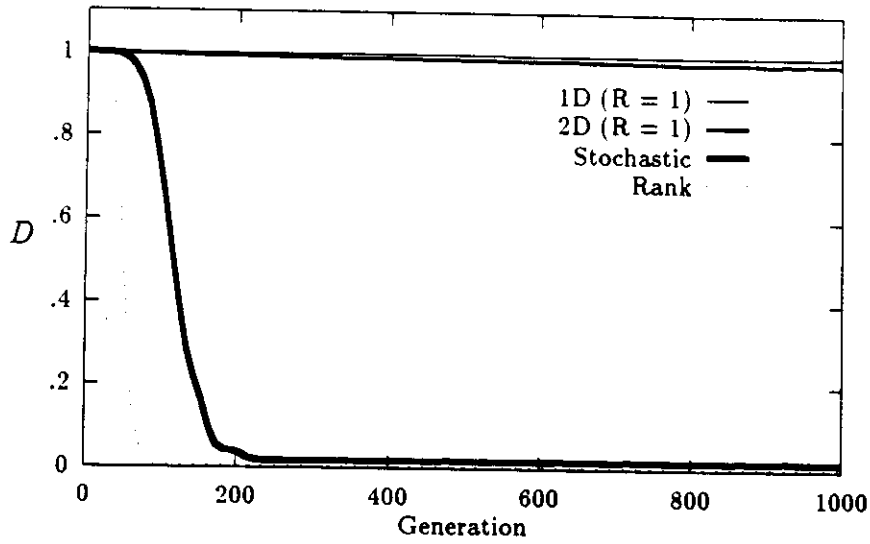

Figure 1: The diversity of alleles D maintained by the four selection algorithms. $N = 2^{16} = 65,536$ and the task is the 64 vertex multilevel graph. Each curve is the average of 7 runs.

5.2 Diversity of Genotypes

The genotypic diversity for the four selection algorithms is plotted in Figure 2. This data is based on a random sample (with a new sample each generation) of 10 of the 64 loci in the genome. In all four cases, the genotypic diversity begins to fall after only a few generations. Both of the local mating schemes maintain a count of about 150 genotypes (out of a possible 1024), while stochastic selection remains around 40, and linear rank selection around 15. With the exception of 1 dimensional local mating, all of the algorithms quickly reach their stable values.

5.3 Inbreeding Coefficient

The inbreeding coefficient for the four selection algorithms is plotted in Figure 3. In the early generations, both of the panmictic selection algorithms show no excess homozygosity (F near 0), while both local algorithms immediately show significant and increasing excess homozygosity. A significant degree of inbreeding is observed in

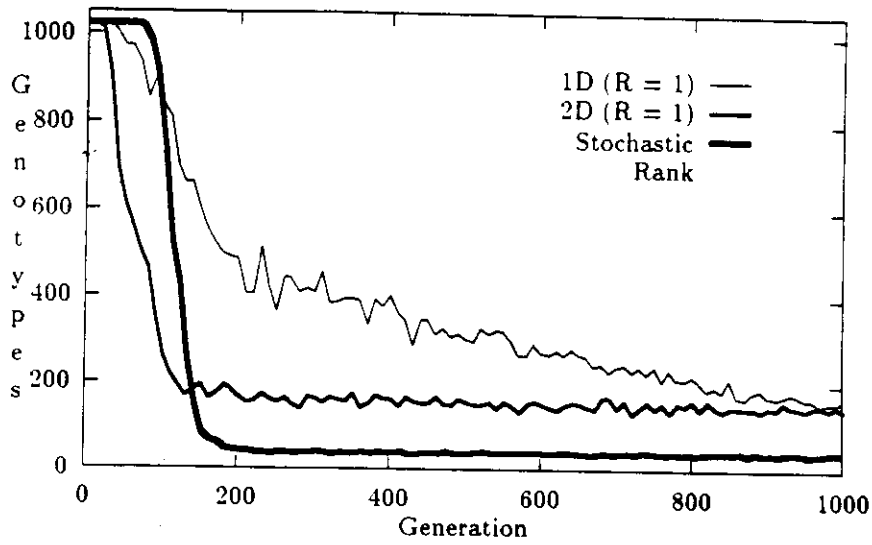

Figure 2: The genotypic diversity for the four selection algorithms, based on sampling 10 loci per generation. $N = 2^{16} = 65,536$ and the task is the 64 vertex multilevel graph. Each curve is the average of 7 runs.

later generations with both panmictic algorithms, and throughout the experiments for both local selection schemes. In later generations, stochastic selection is characterized by a much lower inbreeding coefficient than the others.

5.4 Speed and Robustness

The first speed comparison is based on the number of generations to the first appearance of an optimal partitioning of the 64 vertex graph (Table 1). Of the two panmictic selection algorithms, linear rank selection finds solutions nearly twice as fast as stochastic selection. We also note that both panmictic algorithms do not reliably find optimal solutions when applied to the smaller populations. For both local mating algorithms, longer random walks result in faster evolution, and for the same R value, 2 dimensional local mating is faster than 1 dimensional local mating. With the exception of the most constrained local mating (1 dimensional with $R = 1$), local mating always beats panmictic, and in some cases by about a factor of 7.

Across all of the genetic algorithms, increasing the population size causes only slight speed improvements (in terms of number of generations to an optimal solution). In addition, we found that an optimal solution is either found within a couple hundred generations, or the run times out (across all selection algorithms). We never observed a run that first discovered an optimal solution between generation 200 and 1000.

The second speed comparison is based on the actual time required to find an optimal solution (Table 2). The run-time measurements reported here are based on implementations in C++/CM++ [2] that differ only in the selection/mate choice code. The data was gathered on an 16K processor Connection Machine–2 equipped with 64K bits of memory per processor and 32-bit floating point accelerators, with a

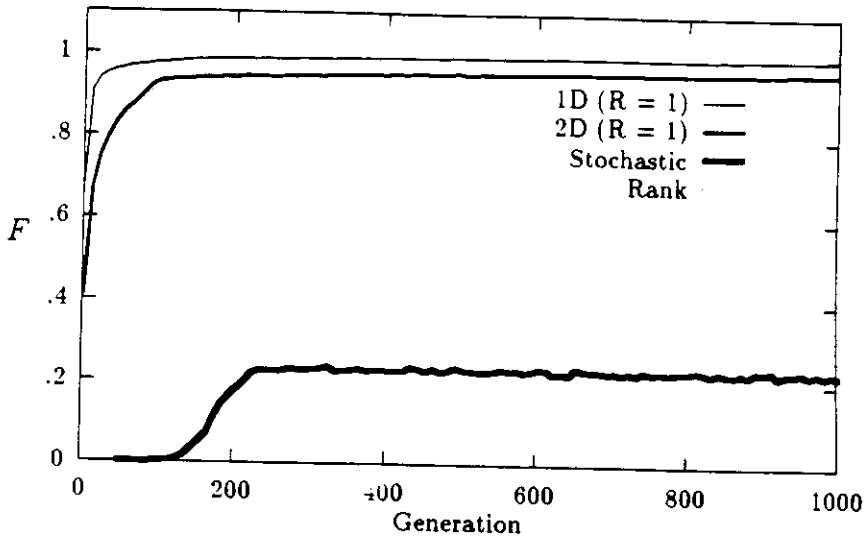

Figure 3: The inbreeding coefficient F for the four selection algorithms. $N = 2^{16} = 65,536$ and the task is the 64 vertex multilevel graph. Each curve is the average of 7 runs.

Algorithm	$\log_2$ Population Size						
	13	14	15	16	17	18	19
Stochastic	†	†	151	137	119	†	136
Linear Rank	†	57	66	52	35	32	31
1D ($R = 1$)	156	148	142	124	126	108	114
1D ($R = 5$)	85	79	59	63	57	49	50
1D ($R = 10$)	56	50	47	50	43	40	41
1D ($R = 20$)	42	40	37	37	34	30	27
1D ($R = 30$)	41	39	32	32	29	26	24
2D ($R = 1$)	48	43	41	42	40	40	38
2D ($R = 5$)	23	20	16	17	16	16	15
2D ($R = 10$)	14	13	13	12	12	11	11
2D ($R = 20$)	13	11	10	11	9	9	9
2D ($R = 30$)	11	8	9	9	8	8	8

Table 1: Generation of first appearance of an optimal partition (median of 11 runs) on the 64 vertex multilevel graph problem. † indicates that the median run did not find an optimal solution within 1000 generations.

Algorithm	$\log_2$ Population Size					
	14	15	16	17	18	19
Stochastic	† (0.57)	165 (1.09)	285 (2.08)	552 (4.64)	† (9.47)	2652 (19.50)
Linear Rank	40 (0.70)	83 (1.26)	132 (2.54)	200 (5.72)	379 (11.84)	755 (24.34)
1D ($R = 1$)	24 (0.16)	41 (0.29)	70 (0.56)	139 (1.10)	264 (2.44)	540 (4.74)
1D (	15 (0.19)	21 (0.35)	42 (0.67)	81 (1.42)	133 (2.72)	265 (5.29)
1D	13 (0.25)	20 (0.42)	37 (0.74)	65 (1.50)	122 (3.06)	252 (6.15)
1D	16 (0.39)	20 (0.54)	35 (0.95)	72 (2.13)	116 (3.85)	199 (7.38)
1D	16 (0.42)	22 (0.69)	38 (1.18)	71 (2.44)	120 (4.61)	216 (9.00)
2D (	7 (0.16)	12 (0.29)	25 (0.59)	51 (1.28)	98 (2.46)	185 (4.87)
2D ($R = 5$)	4 (0.22)	6 (0.38)	12 (0.73)	26 (1.62)	48 (3.01)	86 (5.72)
2D ($R = 10$)	3 (0.26)	6 (0.47)	10 (0.87)	22 (1.86)	39 (3.54)	75 (6.86)
2D ($R = 20$)	4 (0.39)	7 (0.67)	13 (1.22)	23 (2.58)	43 (4.82)	83 (9.27)
2D ($R = 30$)	4 (0.53)	8 (0.91)	14 (1.60)	26 (3.29)	49 (6.14)	93 (11.67)

Table 2: Computation time (seconds) to the first appearance of an optimal partition (median of 11 runs) on the 64 vertex multilevel graph problem. The time (seconds) per generation is shown in parentheses. † indicates that the median run did not find an optimal solution within 1000 generations.

$\log_2 N$	Linear Rank	Stochastic
13	0.45	0.27
14	0.55	0.36
15	0.64	0.64
16	0.82	0.55
17	1.0	0.55
18	1.0	0.27
19	1.0	0.73
overall	0.78	0.48

Table 3: Fraction of runs finding an optimal solution within 1000 generations for the panmictic selection algorithms.

Sun 4/330 front end running SunOS 4.1.1 and Connection Machine software version 6.0.

Although the run time per generation for stochastic selection is less than linear rank selection, it still requires more than twice as much time to find optimal solutions. For local mating, although long random walks require significant computation, the fastest evolution occurs when R is in the range $5 < R < 20$. Even with long random walks, the local mating algorithms run significantly faster (per generation) than the panmictic algorithms, which accentuates the fact that local mating optimizes in fewer generations.

When we compare the fastest panmictic algorithm that we implemented (linear rank) to our slowest local mating algorithm (1 dimensional with $R = 1$), we find that local mating is faster by about a factor of 2. When compared to the fastest local mating algorithm (2 dimensional with $R = 10$), linear rank is slower by more than an order of magnitude and stochastic selection is slower by about a factor of 25.

We measure the robustness of the various genetic algorithms in terms of the fraction of runs that find one of the two optimal solutions within 1000 generations. None

of the local mating runs, across all population sizes and R values, failed to find an optimal solution. Unlike the local algorithms, the panmictic algorithms are not 100% robust (Table 3). Of the two panmictic algorithms, linear rank selection is more robust.

6 Discussion

Simulations of the evolution of artificial organisms operate on large populations in complex and changing ecosystems. The adaptive landscapes are generally enormous (hundreds or thousands of orders of magnitude larger than the population size) and constantly changing. Artificial life simulations require a genetic algorithm that is resistant to convergence and can simultaneously explore different parts of the adaptive landscape.

Local mating should be resistant to convergence, because each deme can explore different peaks in the adaptive landscape. Small demes allow effective exploration of the adaptive landscape, because temporary reductions in fitness are possible, due to random genetic drift (selection would prevent this in a large population) [18]. When a deme discovers a higher adaptive peak, the new genotype will spread and deliver more genes to the nearby demes. This results in the gradual spread of the favorable alleles and allele combinations throughout the population, by means of interdeme selection.

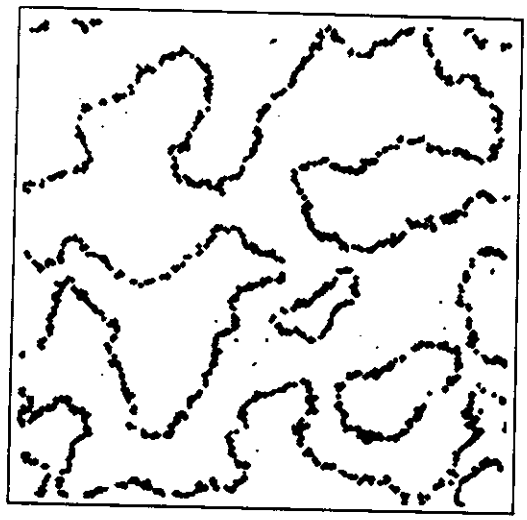

Figure 4: The geographical distribution of fitness scores in generation 150 a run using the 2 dimensional local mating algorithm ($R = 1$) with a population size of $2^{16} = 65,536$ on the 64-vertex multilevel graph partitioning problem. Individuals scoring less than -3 are represented by black pixels.

In our empirical experiments, we have observed that the panmictic selection algorithms become focused towards one of the two optimal solutions in the very early generations, and eventually converge on or near that solution. In fact, we have not

observed a single run in which a panmictic selection algorithm discovered both optimal solutions. In sharp contrast, the local mating algorithms consistently discover both optimal solutions. For all but the smallest population sizes, both optimal solutions are usually discovered many times in any one run by geographically separated demes. Every run with a large, locally mating population resulted in a nearly even mixture of both optimal solutions, and formed genetically homogeneous regions that are separated by boundaries (composed of low-fitness hybrids) that are stable for thousands of generations (Figure 4).

The genetic diversity data (Figure 1) demonstrates quite dramatically how panmictic selection converges towards only one of the solutions, while local selection contains a nearly equal mix of both. (Remember that the two solutions are bitwise complements of each other.) Local selection also maintains much greater diversity of genotypes than panmictic selection (Figure 2). What we expect to see is a cloud of mutants surrounding (in the adaptive landscape) an optimal solution. Because local mating finds and maintains both optimal solutions but panmictic mating finds only one, we would expect local mating to have something like twice the genotypic diversity of panmictic selection. This is clearly not the case—we observe a factor of 4 difference from stochastic selection and a factor of 10 from linear rank selection. We find this difference to be particularly important, because it demonstrates that at equilibrium, local selection explores many more genotypes in each generation.

The inbreeding coefficient results (Figure 3) are quite interesting, because they show two different sources of inbreeding. As we expected, local mating is characterized by a high degree of inbreeding throughout the experiment. The inbreeding is a result of fixation on a particular genotype within each deme due to selection pressure and random genetic drift. The fixation is local—diversity is maintained because each deme fixes on a different genotype.

On the other hand, panmictic selection results in almost no inbreeding until the population begins to converge on a particular solution. When this occurs, we start to observe excess homozygosity, which is due to selection for the optimal genotype. Linear rank selection shows a very high degree of inbreeding, indicating that very few sub–optimal genotypes are chosen as parents for the next generation. In contrast, stochastic selection shows a much lower degree of inbreeding. This shows that the stochastic selection algorithm allows sub–optimal genotypes to be sampled with a higher frequency (which is consistent with the greater allele and genotype diversity).

The diversity and inbreeding data demonstrate the dramatic dynamical differences between local and panmictic mating. It is clear that local mating is more appropriate for artificial evolution, because it maintains a broad genetic search for thousands of generations. If the adaptive landscape or fitness function were changing over time, this diversity would allow the population to exploit the genotypes which suddenly have higher relative fitness values. Local mating would be more appropriate than panmictic mating for artificial life applications, even if local mating results in slower evolution. Fortunately, this is not the case.

The data on the speed of evolution shows that the dynamics of local mating results in a faster and more robust genetic algorithm. Local mating is characterized by faster evolution both in terms of the number of fitness function evaluations and

time required to find an optimal solution. The robustness of local mating is rather surprising: across all population sizes, both 1 and 2 dimensional geometries, and all R values, local mating *never* failed to find an optimal solution, while both of the panmictic algorithms had a significant fraction of their runs "time out" (no optimal solution found within 1000 generations).

The fact genetic algorithms that use local mating can be significantly faster and more robust than traditional genetic algorithms is an important result, and suggests that further investigation is in order. While many important theoretical results have been developed for panmictic mating algorithms, it is not clear that any of these results can be applied directly to local mating. Also, it is not at all clear how local mating will perform with small populations, although the difference in robustness between local and panmictic mating is greatest for the smaller population sizes that we examined. In addition, we have only examined one very simple local mating algorithm, and it is almost certainly not the best that can be (or has been) found.

Acknowledgments

We would like to thank Ernst Mayr, Joe Pemberton, Chuck Taylor, Greg Werner, and Alexis Wieland for their valuable input. We also acknowledge the three anonymous reviewers for their helpful comments. This work is supported in part by W. M. Keck Foundation grant number W880615, University of California Los Alamos National Laboratory award number CNLS/89-427, and University of California Los Alamos National Laboratory award number UC-90-4-A-88. The empirical data was gathered in part on a CM-2 computer at UCLA under the auspices of National Science Foundation Biological Facilities, grant number BBS 87 14206, and a CM-2 computer at the Advanced Computing Laboratory of Los Alamos National Laboratory under the auspices of the U.S. Department of Energy, contract W-7405-ENG-36.

References

[1] David H. Ackley. *Stochastic iterated genetic hillclimbing.* PhD thesis, Department of Computer Science, Carnegie Mellon University, 1987.

[2] Robert J. Collins. CM++: A C++ interface to the Connection Machine. In *Proceedings of the Symposium on Object Oriented Programming Emphasizing Practical Applications.* Marist College, September 1990.

[3] Robert J. Collins and David R. Jefferson. Representations for artificial organisms. In Jean-Arcady Meyer and Stewart Wilson, editors, *Proceedings of Simulation of Adaptive Behavior.* The MIT Press/Bradford Books, 1991.

[4] Robert J. Collins and David R. Jefferson. AntFarm: Towards simulated evolution. In Christopher Langton, J. Doyne Farmer, Steen Rasmussen, and Charles Taylor, editors, *Artificial Life II.* Addison-Wesley Publishing Company, (in press).

[5] Robert J. Collins and David R. Jefferson. An artificial neural representation for artificial organisms. In Reinhard Männer and David E. Goldberg, editors, *Proceedings of Parallel Problem Solving from Nature*. Springer-Verlag, (in press).

[6] James F. Crow. *Basic Concepts in Population, Quantitative, and Evolutionary Genetics*. W. H. Freeman and Company, New York, 1986.

[7] Kenneth A. De Jong. *An analysis of the behavior of a class of genetic adaptive systems*. PhD thesis, Department of Computer and Communication Sciences, University of Michigan, 1975.

[8] Kalyanmoy Deb and David E. Goldberg. An investigation of niche and species formation in genetic function optimization. In *Proceedings of the Third International Conference on Genetic Algorithms* Morgan Kaufmann, June 1989.

[9] David E. Goldberg. *Genetic Algorithms Search, Optimization and Machine Learning*. Addison-Wesley Publishing Company, Inc., Reading, Massachusetts, 1989.

[10] David E. Goldberg and Jon T. Richardson. Genetic algorithms with sharing for multimodal function optimization. In *Genetic algorithms and their applications: Proceedings of the Second International Conference on Genetic Algorithms*. Lawrence Erlbaum Associates, 1987.

[11] Martina Gorges-Schleuter. ASPARAGOS an asynchronous parallel genetic optimization strategy. In *Proceedings of the Third International Conference on Genetic Algorithms*. Morgan Kaufmann, June 1989.

[12] John J. Grefenstette and James E. Baker. How genetic algorithms work: A critical look at implicit parallelism. In *Proceedings of the Third International Conference on Genetic Algorithms*. Morgan Kaufmann, June 1989.

[13] Daniel L. Hartl and Andrew G. Clark. *Principles of Population Genetics*. Sinauer Associates, Inc., Sunderland, Massachusetts, 1989.

[14] W. Daniel Hillis. *The Connection Machine*. The MIT Press, Cambridge, Massachusetts, 1985.

[15] David Jefferson, Robert Collins, Claus Cooper, Michael Dyer, Margot Flowers, Richard Korf, Charles Taylor, and Alan Wang. The Genesys System: Evolution as a theme in artificial life. In Christopher Langton, J. Doyne Farmer, Steen Rasmussen, and Charles Taylor, editors, *Artificial Life II*. Addison-Wesley Publishing Company, (in press).

[16] Bernard Manderick and Piet Spiessens. Fine-grained parallel genetic algorithms. In *Proceedings of the Third International Conference on Genetic Algorithms*. Morgan Kaufmann, June 1989.

[17] Heinz Mühlenbein. Parallel genetic algorithms, population genetics and combinatorial optimization. In *Proceedings of the Third International Conference on Genetic Algorithms*. Morgan Kaufmann, June 1989.

[18] William B. Provine. *Sewall Wright and Evolutionary Biology*. University of Chicago Press, 1986.

[19] Sewall Wright. *Evolution and the Genetics of Populations. Volume 2: The Theory of Gene Frequencies*. University of Chicago Press, 1969.