The Hamming Metric in Genetic Algorithms and Its Application to Two Network Problems

William G. Frederick
Robert L. Sedlmeyer
Curt M. White

Indiana University - Purdue University Fort Wayne
Computer Science Department

Abstract

Using the Hamming metric to compute distances between chromosomes values as bits, we introduce two real-valued measures in genetic algorithms. These measures, **genetic drift** and **clustering**, give insight into the behavioral effects of mutation, convergence and mating (using simple crossover) during generation cycles. The genetic drift and clustering measures will then be applied to two common computer network problems: the optimal placement of a multi-station access unit (Steiner Point variation problem) and path loss (edge integrity of a graph).

Introduction and Definitions

Genetic algorithms are a class of search algorithms based on the mechanics of natural biological selection [7]. A data structure is created which represents an individual trial in the search space of all possible feasible solutions. Each individual in the population has a fitness value which is an evaluation of the worth of that individual as a solution.

In one of our applications, we use chromosomes of bit strings for our population members to determine placement, interconnectedness, and the total length of connections in a multi-station access unit. We reward population members with the smaller total length of connections by giving them a higher fitness value and select them for breeding more often. Offspring are created by applying (probabilistic) chromosomal crossover to two members (parents) during a breeding (generation). Offspring from such a union may be further modified by random mutations of bits.

A common difficulty with genetic algorithms is fine tuning the algorithm to prevent **premature convergence**, that is, convergence to local minima or maxima without adequate exploration of the search space. The art of fine tuning genetic algorithms in unique applications is explored in [3], while a fuller description of genetic algorithms can be found in Goldberg's book [7].

ACM-SAC '93/2/93/IN, USA
© 1993 ACM 0-89791-568-2/93/0002/0126...$1.50

We assume that the genetic algorithm **population** consists of individual **chromosomes** (bitmaps): 0 = false, 1 = true. We use the following parameters to tune our genetic algorithms:

Name	Meaning
Popsize	size of the population, number of chromosomes
N	uniform length of the chromosomes in the GA population
HamDist	Hamming distance between two chromosomes

The **Hamming distance (HamDist)** between two individuals in such a population is the number of corresponding bit locations where they differ:

$$HamDist(P_1, P_2) = \sum_{k=1}^{N} P_1[k] \; xor \; P_2[k]$$

where $P_1[k]$ and $P_2[k]$ is the kth bit value of the first and second member (respectively) of the pair.

Using the Hamming metric, we define **genetic drift as the average** distance of all individuals in the current generation from the best (most fit) population member found to date, similar to that found in [4].

$$Genetic\ Drift = \frac{\sum_{j=1}^{Popsize} HamDist(P_j, Best)}{Popsize}$$

Clustering is the average pair-wise distance of all individuals in the current generation.

$$Clustering = \frac{\sum_{k=1}^{Popsize-1} \sum_{j=k+1}^{Popsize} HamDist(P_k, P_j)}{\binom{Popsize}{2}}$$

The denominator is the number of pairs of individuals in a population of size Popsize.

In both formulas, P_i is the ith individual (chromosome) of the

population. The computational complexity of genetic drift is O(Popsize) and that of clustering is O(Popsize^2).

We propose the Hamming metric be used to measure the population behavior during generational cycles in genetic algorithms. Such a consideration of distance between members of a genetic algorithm population can yield a great deal of information about algorithm behavior. In addition, we use a simple, but effective paradigm by considering individual chromosomes as vertices on the binary N-cube [8], where N is the length of the chromosome. However, some caution needs to be exercised with this metric and subsequent measures based on it. A small value in Hamming distance (syntactic information) does not always imply a small variation in their fitness interpretation (semantic information).

Mathematical Analysis

If the population consists of all the 2^N vertices on the Binary N-Cube (N = N), genetic drift and clustering can be computed analytically. For sake of brevity, we state results without proof:

Genetic Drift In the N-Cube
The average Hamming distance between a given point in the Binary N-Cube and any other point is $N * 2^{N-1} / (2^N - 1)$. As N gets large, this value is asymptotic to N / 2.

Clustering in the N-Cube
The average Hamming distance between all possible pairs of points in the Binary N-Cube is $N * 2^{N-1} / (2^N - 1)$. This is the same as Genetic Drift. Similarly, as N gets large, this value is asymptotic to N / 2.

Consequently, we will use N/2 to standardize the Hamming distance in different genetic algorithmic applications.

Genetic Drift and Clustering in Genetic Algorithms

When we compute the genetic drift and clustering measures on a reduced set (Popsize) of vertices from the binary N-cube, we gain some immediate insight. In the initial population, both genetic drift and clustering measure how well we have chosen vertices (generated chromosomes) randomly from all possible vertices. Both population measures should initially be close to N/2 (or 1, when standardized). If initial values differ from this expected value, then our sampling techniques or pseudo-random number generators are suspect.

There is, however, more to be gained from these measures than just insight into randomization of the initial population in a genetic algorithm. Intuitively, genetic drift measures the movement of the population away from, or towards the most fit population member found. As genetic drift increases, the vertices of the N-cube being chosen by the genetic algorithm tend to be farther away from the best vertex found to date. It is this very movement that ensures the success and robustness of these algorithms. If the GA is well tuned, it will proceed to search away from best if allowed to continue. This can help avoid premature convergence to false optimal values.

In topological terms, clustering measures the average diameter of the neighborhood of vertices chosen by this population from the N-cube. Intuitively, this is the "geometric size" of the current generation. If clustering = C, then the population is "beginning to collect" on a 2C dimensional hyperplane slicing through the N cube.

Intuitively, if we consider a genetic algorithm population as a "searchlight" being focused on the binary N-cube, genetic drift measures the movement of that light around the cube and clustering measures the size of the beam being cast.

Two GA Problems for Analysis

To illustrate the efficacy of both measures, we will use two different network problems. The first GA problem is a variation on the classical Steiner Point problem.

Steiner Point Variation Problem
Given a collection of K test points in the plane, and S Steiner points, locate any subset of the S Steiner points in such a way so as to minimize the distance of the selected Steiner points from the given test points and the length of the connections between the Steiner points. This is similar to the problem of optimally locating a multi-station access unit on a local area network.

In the classical Steiner problem, each Steiner point is connected to 3 other test points or another Steiner point. In this variation, there are no such restrictions on Steiner connections. Graph I shows the connections and total distance of such generated by the genetic algorithm using K=8 test points, generated at random in the unit square, and S=2 Steiner points. The GA population consists of chromosomes with 90 bits. A common, unbiased selection algorithm (SUS) [2] was used. Single crossover and mutation were also applied.

The following is the bitmap decoding of the chromosome used in the GA.

Bits	Interpretation
1 - 20	X - coordinate in unit square of 1st Steiner point in floating point binary representation
21 - 40	Y - coordinate of 1st Steiner point
41 - 80	X and Y coordinates of 2nd Steiner point
81 - 88	Bitmap for connections of 1st Steiner point to test points 1 through 8 (0 = not connected, 1 = connected)
89 - 90	Unused

The second Steiner point is assumed to be connected to any test points not already chosen to be connected to by the first Steiner point. The solid line on Graph II is the best total distance (use right hand Y-axis) of all connections between test points and Steiner points found by the algorithm to date. Fitness among these population members is measured by the square of the ratio of the best (smallest) Steiner distance found to each individual's total distance.

Graph II shows the change in genetic drift and clustering during the execution of the genetic algorithm on this problem. The left hand Y-axis is the actual standardized Hamming distance value computed. The X-axis is the generation number. (See section on qualitative analysis).

Edge Integrity of a Graph - IPrime

The second network problem deals with the loss or removal of a path (edge) between two nodes in a network (graph). The edge-integrity of a graph G, denoted I'(G), is defined to be the minimum, taken over all edge sets S of G, of the sum of the number of edges in S and the order of the largest component of G-S.

Edge-integrity has been suggested as a measure of how vulnerable a communications network is to disruption. Fellows and Stueckle [5] have shown that the computation of I'(G) is NP-complete; however, there do exist good algorithms for certain classes of graphs [1]. We are interested in applying genetic algorithms to the computation of I'(G) for Halin graphs. A Halin graph is a tree in which all vertices of degree one are connected to form a cycle. Figure 1 shows our Halin graph.

In this GA implementation, each bit represents an edge of the graph. Thus the chromosome length depends on the number of edges in the graph. An edge set S is defined by non-zero bits. Encoding/decoding procedure for chromosomes in the Iprime calculation uses the following bitmap procedure:

1. Label each edge of the Halin graph, H.
2. Encode an edge-set, s, of H as a binary string, S.
   S(i) = 1 iff edge i is in the edge-set s. For example, if s = { 1, 3, 4 } for a graph with 7 edges, then the chromosome S = 1011000.

Each member of the population defines an edge set S which yields an estimate of I'(G). A member's fitness is computed by subtracting this estimate from the upper bound of I'(G). The order of the graph defines a simple upper bound. The results of a single run of the GA is shown in Graph III.

Qualitative Analysis of Genetic Drift and Clustering

Both genetic drift (open squares) and clustering (filled triangles) are at the initially expected normalized Hamming distance of 1.0 = N/2. The initial populations of both GAs seem to be well randomized as vertices on the binary N-cube.

Graphs II and III demonstrate the behavior of both genetic drift and clustering. Genetic drift (open squares) has a more "discontinuous" behavior than clustering. These jumps are caused by an abrupt change in the best-to-date Steiner coordinates or Iprime estimate. These graphs (and others) support the intuitive notion that the population is constantly searching for and moving toward a different solution.

There is, initially, a rapid decrease in genetic drift, followed by an almost asymptotic stabilization thereafter. We have observed that whenever a new best value occurs and remains as the best problem solution for several generations, the population tends to "drift" away from this best member, to continue the search. When a new best is found, the population is initially closer to this one.

The clustering measure (closed triangles) is apparently more stable because of the averaging used in its computation. Clustering is not dependent on the measure of distance from best but on the distance between all population members. It would be difficult to change the average pair-wise distance in any population without disrupting many bit values. This is why little behavioral changes occur in either of these measures when crossover probabilities are varied. The effect of single crossover on both these measures is not necessarily negligible [6].

Mutation, however, is much more effective means of controlling both clustering and genetic drift. Clustering (pair-wise distance) seems to approach an asymptotic limit, with minor perturbations.

Seeking some universality of behavior in both of these measures, independent of problem and parameters, we collected data on several runs of the GA algorithm for this variation of the Steiner problem and the Iprime computation. Using the same 8 random test points and Halin graph, we re-ran the algorithms, varying the probability of mutation and crossover. Both genetic drift and clustering have been converted to the normalized Hamming distance.

As we were only interested in the behavior of the two measures under different mutation rates, the probability of crossover was held fixed. Both genetic drift and clustering are affected significantly by changes in mutation.

In fact, if mutation is too high (0.1) both genetic drift and clustering remain near 1.0. The population is spread fairly evenly over the binary N-cube, indicating that the GA is performing less judiciously, more like a random search.

On the other hand, with no mutation at all (0.0), genetic drift and clustering tend to be converge rapidly. Intuitively, this means that the GA population is searching the N-cube somewhat inefficiently--indicative of slow or even premature convergence. Obviously, fine tuning mutation probability is critical for robust search and subsequent (non-premature) convergence of GA algorithms. We hope to eventually provide some guidelines for interactive tuning of this GA parameter based on clustering.

Conclusions and Directions for Further Research

Genetic algorithms hold considerable promise for the complex computations required in network problems. Such problems are, all too often, computationally difficult. Blind application of genetic algorithms to difficult network problems without accompanying measurements of convergence is fraught with error. To that end, we have provided two straight-forward, easily computed measures of GA population behavior to measure stability and convergence of the algorithm. Such measures provide some measure of confidence to the resulting

solutions.

While advocating the introduction of genetic drift and clustering as indicators of convergence, we recognize that we have given rise to additional questions. In particular, could genetic drift or clustering be used to control premature convergence and allow more autonomous behavior of the genetic algorithm through control of population parameters, such as mutation, population size, and crossover? What proof is there of the efficacy of either measure? At the very least, it seems that clustering and genetic drift as measures give new insights into population behavior of genetic algorithms.

References

[1] Bagga Kunwarjit S., Beineke, Lowell W., Lipman, Marc J., Pippert, Raymond E., and Sedlmeyer, Robert L., "A good algorithm for the computation of the edge-integrity of trees," Congressus Numerantium 67 (1988), 225-232.

[2] Baker, James Edward, "Reducing Bias and Inefficiency in the Selection Algorithm", Genetic Algorithms and Their Applications, Proceedings of the Second International Conference on Genetic Algorithms, July, 1987, pp. 14-21.

[3] Davis, Lawrence, Handbook of Genetic Algorithms, Van Nostrand Reinhold, 1991.

[4] De Jong, K.A."An analysis of the behavior of a class of genetic adaptive systems," Dissertation Abstracts International, 36(10), 5140B, University Microfilms No. 76-9381.

[5] Fellows, M.R. and Stueckle, S., "The immersion order, forbidden subgraphs and the complexity of network integrity," J. Combin. Math. Comb. Comput. (to appear).

[6] Frederick, William G., "The Single Crossover Operator and Hamming Distance in Genetic Algorithms," (unpublished).

[7] Goldberg, David Edward, Genetic Algorithms in Search, Optimization, and Machine Learning, Addison-Wesley Publishing Company, 1989.

[8] Harary, Frank, Graph Theory, Addison-Wesley Publishing Company, 1972.

[9] White, Curt M. and Frederick, William G., "Genetic algorithms and a variation on the Steiner Point problem," Proceedings of the Fourth Conference on Neural Networks and Parallel Distributed Processing, Fort Wayne, IN, April 1991.

Caterpillar-Like Halin Graph
IPrime = 22

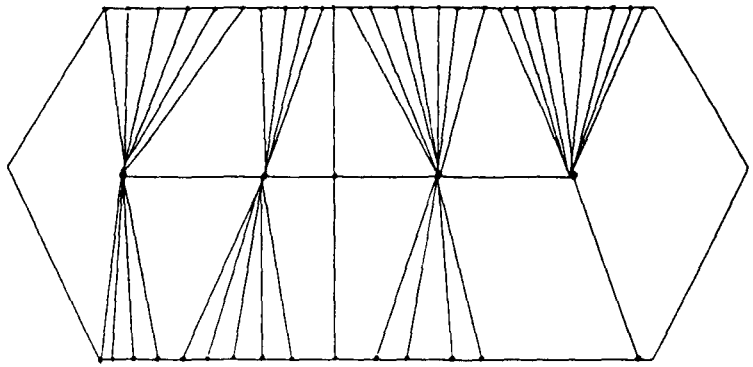
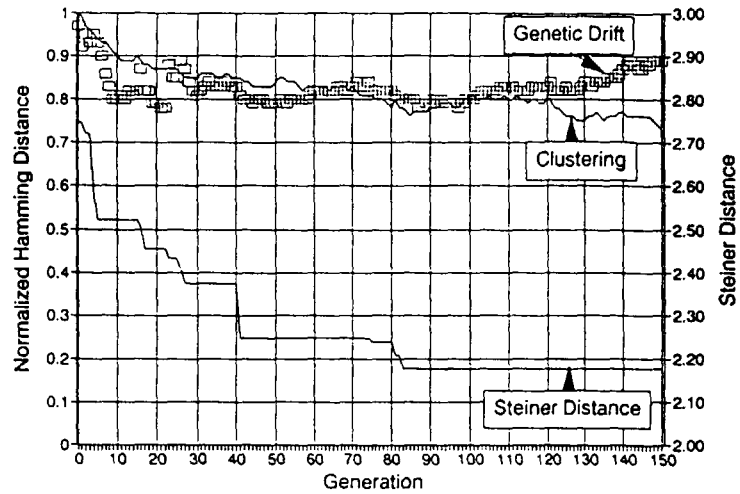
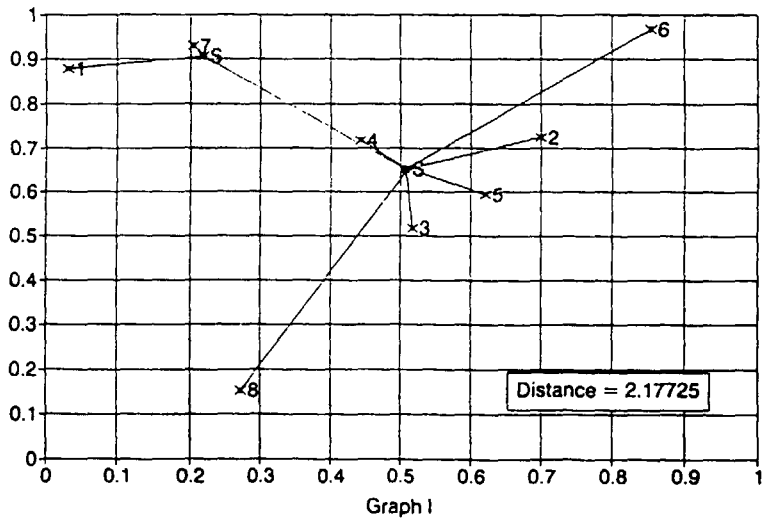


Figure 1

Steiner Point Problem
Graph II



Best Solution to Steiner Point Problem
8 Random Test Points, 2 Steiner Points



Graph I

Halin Graph - IPrime Computation
Graph III