Proceedings of the 1997 Winter Simulation Conference
ed. S. Andradóttir, K. J. Healy, D. H. Withers, and B. L. Nelson

GENETIC ALGORITHMS WITH CLUSTER ANALYSIS FOR PRODUCTION SIMULATION

Robert Entriken
Siegfried Vössner

Department of Engineering Economic Systems & Operations Research
Stanford University
Stanford, CA 94305-4023, U.S.A.

ABSTRACT

This paper describes the application of a Genetic Algorithm to production simulation. The simulation is treated as a detailed, stochastic, multi-modal function that describes a performance statistic. Our aim is to optimize (or at least improve) the performance of the system. In our experiments, we modeled a real-world production line for printed circuit boards that has many products and must often be retooled or reconfigured. Since the product line is always changing, with half of the products turning over within a year, the job of configuring and fine tuning the production line is never ending. Our experiments show that a Genetic Algorithm when attached to the simulation model can provide excellent support for this process. This combination can be used to obtain quick and stable results that do indeed indicate the direction to improved production.

1 GENETIC ALGORITHMS

1.1 Principles

Unlike classical Operations Research techniques, Genetic Algorithms (GAs) (Goldberg 1989; Holland 1975) use the mechanics of natural selection ("survival of the fittest") (Darwin 1859) and genetics that were originally inspired by biological structures and their evolution.

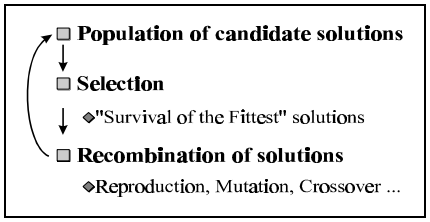

Figure 1: Scheme

Genetic Algorithms process many candidate solutions (a population) at one time, in parallel. In production simulation, a solution would be a certain line configuration, or priority system. The best solutions survive a competition based on line performance, and go on in the selection process, and out of these solutions, new ones are created by recombination.

One important point is that GAs do not operate directly on the variables (phenotype); similar to biology there is an encoded representation (genotype), that allows easy recombination of candidate solutions, regardless of the types of the variables (integer, real, binary, character strings).

1.2 Where and Why to Use Genetic Algorithms

Genetic Algorithms have shown their advantages in dealing with the highly non-linear search spaces that result from noisy and multimodal functions. Furthermore, they are, to a high extent, problem independent.

In our case, the simulation performance measure is itself a random value. For this reason, we must simulate the production line many times, and compute a sample mean. It is the sample mean that is being optimized. This statistic is inherently noisy, and when dealing with decisions like when and how long to take a lunch break, the response of the function can be arbitrarily "lumpy," so the Genetic Algorithm provides a very elegant match in such a difficult situation.

1.3 Cluster Analysis as a Convergence Criterion

In order to analyze the output of an optimization algorithm and to determine when to terminate the algorithm, it is important to know the *convergence state*, the distance to the best or an acceptable solution. Due to the complex structure of GAs it is difficult to predict the convergence state with a mathematical model or even to measure it. At the moment there are some approaches (Davis and Principe 1991; Goldberg et al. 1992; Miller

and Goldberg 1995) that try to formulate an analytical model for convergence by focusing on selected GA operators.

Applying GAs to "real world," discrete-event simulation, we unfortunately cannot use these approaches. An alternative method described in (Vössner and Braunstingl 1996a) localizes several optima at an early stage in the optimization process. Working on the principle that Genetic Algorithms prefer to accumulate their individuals around local optima, the method searches for these accumulations, i.e. clusters, in a population.

A graph of these clusters can give insight as to the convergence state of a GA in general. The following is only a short summary of this method.

1.3.1 Selection of a Generation

The first step is to chose an appropriate generation to analyze from a GA run, which we do by taking note of the average fitness of the population as the algorithm proceeds. The average fitness of all generations of a GA run yields function shapes similar to the ones shown in Figure 2.

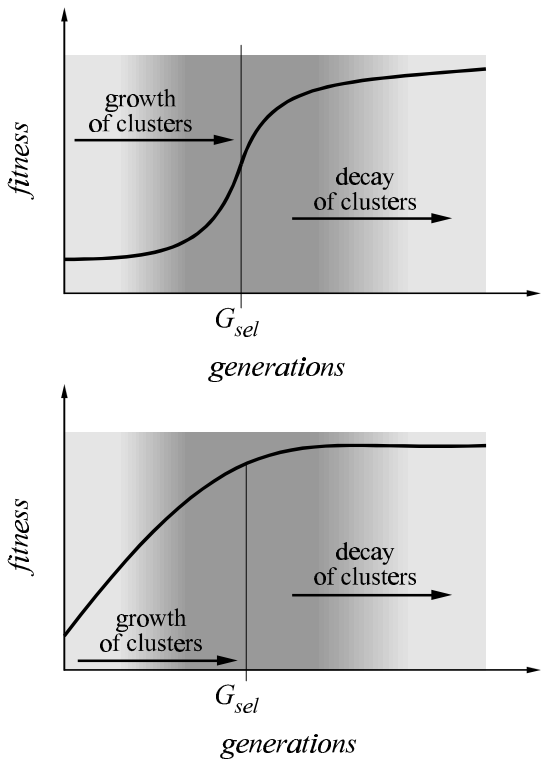

Figure 2: Selection of a Generation for Cluster Analysis

When the individuals are distributed randomly, as in the initial population, only one cluster containing all individuals will be found, and the average fitness of the population will be low. As the algorithm proceeds, individuals begin to move towards areas of higher fitness, more clusters appear, and the average fitness increases rapidly. Then the individuals begin congregating into one or more optima. As the initial clusters start to disappear or split, the average fitness grows more slowly.

The following heuristic has turned out to be very useful for determining an appropriate generation to begin cluster analysis - at least for all our optimization problems. We recommend selecting the population G_{sel}, which comes closest to the point of inflection respectively, to the point of maximum curvature of the curves shown in Figure 2.

1.3.2 Cluster Analysis

For a simpler exposition, all parameter intervals are normalized to the range $0 \leq parameter < 1$, so that the parameter space, formerly a hyper rectangle, is now a hyper cube (with unit edge length and volume). In this normalized parameter space, all of the distances between all individuals are calculated, where the distance d_{ij} between two individuals i and j is taken with the Euclidean metric. Doing this yields a symmetric distance matrix ($n \times n$) where : $d_{ij}=d_{ji}$ for $i, j=1, ..., n$.

Clusters of individuals are defined using the distance matrix. The distances of all individuals are compared with a particular threshold value d. When the distance between two individuals is smaller than the threshold, they lie in the same cluster.

Clusters that are identified by this algorithm can have arbitrary shapes, which is especially useful if the fitness function has an asymmetric gradient around a local optimum. Suppose the optimum is the highest point of a narrow ridge. Then the individuals will accumulate along this ridge, forming a longish, thin cluster. Such a solution is not as robust as a the case of a high plateau.

The clustering algorithm is summarized in the following pseudo code, where we define a cluster to consist of at least 2 individuals (solution points). Given any two individuals i and j, if the distance between them $d_{ij:} < d$, then they should be members of the same cluster.

There are three cases:

1. If they do not belong to a cluster yet, they form a new one.
2. If only one of the two individuals already belongs to a cluster, the other one joins this cluster.
3. If both individuals are members of different clusters, these clusters are combined thus forming a single one.

1.3.3 Cluster Diagram

We decrease the check distance by discrete factors, starting from the hypercube's space diagonal d - where exactly one cluster will be found - down to a distance d_{min} where no clusters any longer exist. At each of these steps, we perform a cluster analysis as described above. In this context, the former definition, that a cluster consists of at least two individuals, is important. Since lone individuals do not count as clusters, no clusters can be found if the check distance is sufficiently small. From this analysis we get a diagram of clusters for different check distances (see Figure 3).

The results of different cluster analyses, ordered by decreasing check distance (*d/1, d/2, d/4,* ...), are arranged in a cluster diagram, where each circle represents a cluster, and the connection from one cluster to the next outward cluster(s) shows how clusters split with decreasing check distance.

The diagram shows the number of clusters, their sizes, distances, history of origin and the average fitness of the corresponding individuals. The numbers within the circles identify them, while the size of the circles is proportional to the number of individuals forming that cluster. The shading of a cluster indicates the average fitness of the individuals within it.

One of the shades is labeled with "avg" in the fitness scale at the bottom of the diagram. If a cluster number is underlined, the average fitness of its elements is greater or equal to the average fitness of all individuals. We call such a cluster "outstanding."

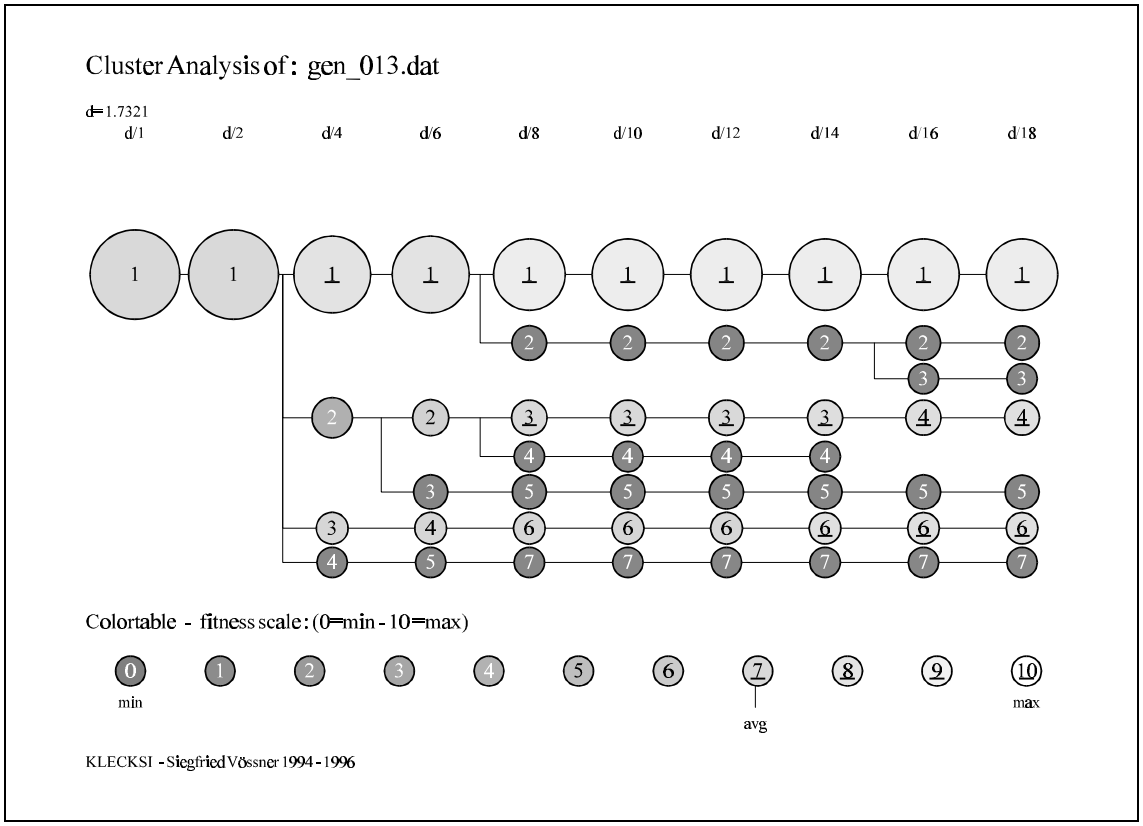

Figure 3: Example Cluster Diagram

2 AN EXAMPLE

Our real-world production line produces a large variety of integrated circuit boards in small lot sizes for Hewlett-Packard Company in Puerto (Rojas 1993). Daily production mixes vary from 6 to 14 different board types per day, total daily production ranges between 135 and 320 boards per day. Half of the products in production at any given time are out of production within a year. Frequent adjustments of various stations (i.e., maximum throughput of a station) are necessary to operate the line at a high level of performance. For this reason, a support system based on computations done with a Genetic Algorithm can be very useful.

2.1 The Production Line

The line, depicted in Figure 4, consists of twenty-one different stations, each consisting of machines that are serially connected or accept work in parallel. There are three different materials-handling systems: electric conveyors for stations one to fifteen, automated guided vehicles from station fifteen to seventeen, and carts and tote boxes from station seventeen onwards. The different boards require different sequences of stations for their production, and set-up times are required for retooling.

The solid arcs in Figure 4 denote the normal production sequence, and the dashed lines show paths for alternate sequences. The first dashed line, labeled "Bottom Loading," is for boards that have chips on both sides. The second, labeled "No Hand Loading," allows some boards to skip the hand loading and wave soldering steps. The last dashed line allows some boards to skip burn-in and packing.

2.2 The Simulation Software

Our simulation is implemented with a commercial production simulation software package called TestSim®/X (Tyecin Systems 1994), and incorporates detailed representations of equipment, operations and maintenance personnel, products, processes, and operating rules for priorities and rework. Equipment is subject to failure and is repaired by a limited pool of

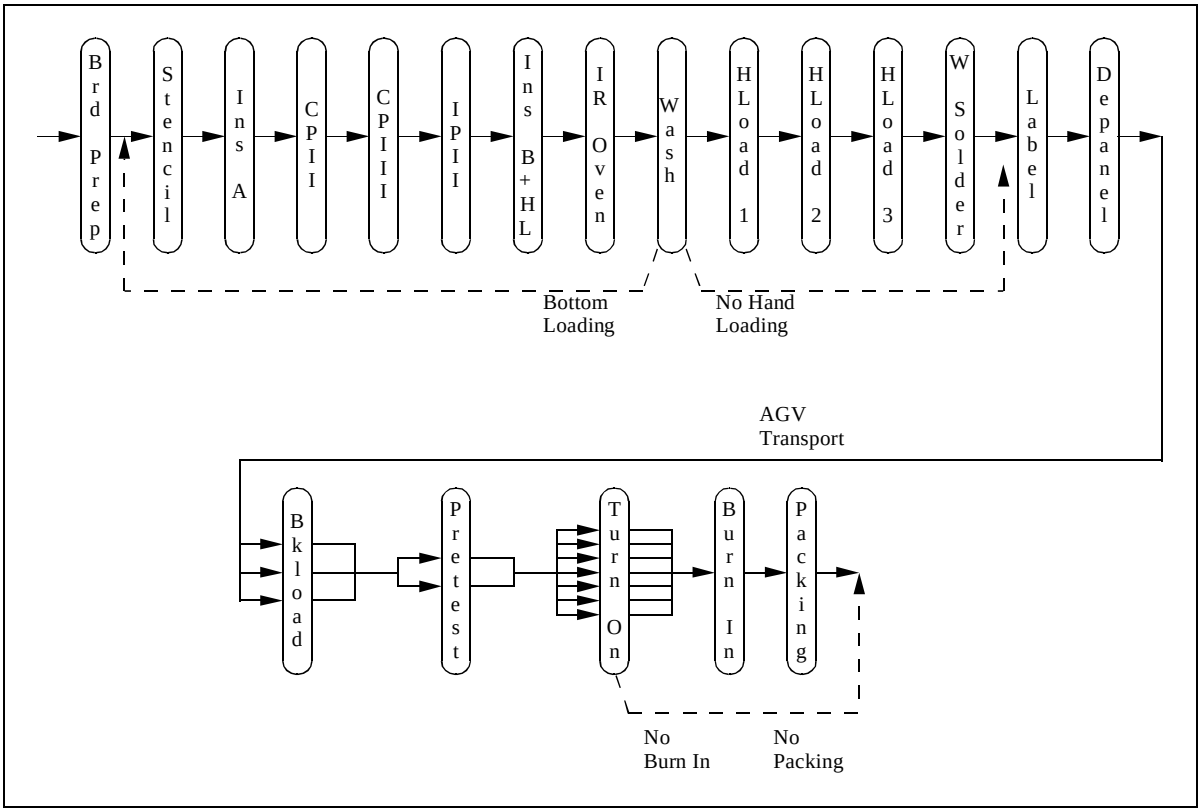

Figure 4: Example Production Line

skilled technicians. We were able to control the parameters of our optimization by creating software that translates the values of our decision variables into changes in the ASCII input files of TestSim/X, executes replications, and collects statistics. Practically every detail of the production is at our disposal for consideration by the Genetic Algorithm—everything from the order of production starts to the length and time of the lunch breaks.

2.3 The Objective and Decision Variables

To demonstrate the usefulness of using a Genetic Algorithm for decision support, we have chosen to manipulate the service times at three of the workstations, because we can readily plot the results in three dimensions (3-D). These stations (Inspection/Hand Loading, Burn-in Oven, and Packing) were chosen because they presently represent bottlenecks to production throughput, which we will attempt to maximize. As an aside, bottlenecks toward the end of the production line seem to be more responsive than others. We vary the service times from their nominal values to one fifth of the nominal value on a discrete scale with 32 points ranging from 1 to 5. Reducing a service time by a factor of two is a gross approximation to adding a new machine in parallel.

2.4 Results

Our experiments were conducted in two phases. First, we identified the global optimum by scanning the entire three-dimensional parameter space. This method is guaranteed to produce the global optimum, but it is very time consuming for general practice. The scan serves as a basis for comparing the results of the second phase, where we employ the Genetic Algorithm. Not only can we say that the algorithm finds the global optimum, but it does so very quickly compared to the scanning method.

2.4.1 Scan

Given three decision variables, each taking 32 discrete values, and 5 replications of each experiment to obtain a sample mean, our scan of the parameter space required 163,840 simulations. This took about 11 days elapsed time, and cannot be recommended in light of the available alternatives.

In Figure 5, the origin is in the back left corner with *x* being Packing, *y* being Burn-in, and *z* being Inspection and Hand Loading. Each ball is a candidate solution with throughput greater than 20600 parts per month.

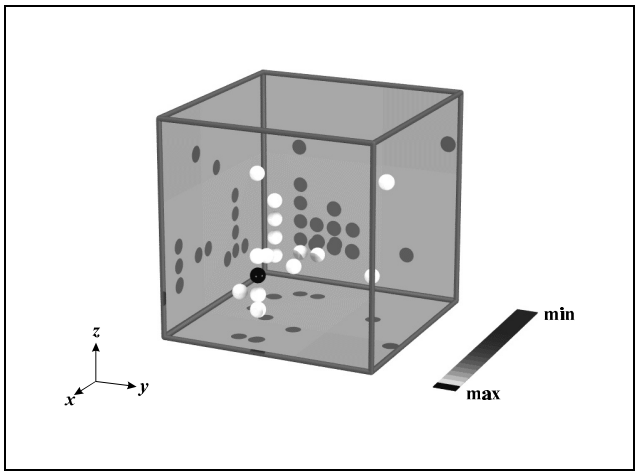

Figure 5: 3-D Plot of the Most-Fit Solutions in the Parameter Space

The shades of gray indicate the fitness, where the lighter shades are more fit. The black ball, however, is the global optimum.

Each of the back panels of the cube has a projection of the balls so that you can better place them in the cube. Notice that from these projections, it is easy to visualize clusters of good solutions. When we experiment with the Genetic Algorithm, we should expect to see that it produces similar cluster patterns.

The global optimum is located at (0.8222, 0.3778, 0.2889) with a throughput of 27090 parts per month. This is much improved over the original configuration, represented by the origin, where the throughput is 18700 parts per month.

Since these are the top 13 points in the parameter space, we are viewing only the tops of three-dimensional mountains. Many of the peaks are isolated. Seen in two dimensions as projections on the back panels, there are ridges, plateaus, and lone peaks. This random function is indeed multi-modal, and it should present quite a challenge to any algorithm.

2.4.2 Genetic Algorithm

In the second phase of our experiments, we applied a Genetic Algorithm package (Vössner and Braunstingl 1996b) to the same problem, and it performed very well. With only 30 individuals in the population and again 5 replications to compute the sample mean, the algorithm converged to the global optimum within 15 generations. That translates into only 2,250 simulation runs in about 12 minutes elapsed time. We used standard algorithm settings (crossover probability 0.7 and mutation probability 0.003).

The global optimum was actually identified at a very early stage, but it was not until about generation 10 that individuals began to cluster there in force. Let us take a look at the progress of the algorithm using the 3-D plots in Figure 6.

In Figure 6, there are four snapshots of the progress of the algorithm taken of generations 1, 5, 10, and 15. Each ball again represents an individual with fitness greater than 20600 parts per month. The colors range again according to fitness, with lighter being better. All plots use the same fitness scale.

The two main clusters are located around the global optimum, in much the same shape as the cluster formed in the scan. They are elongated in the vertical direction and somewhat symmetric in the projection onto the x-y (bottom) panel. For the production line, this means that in the area of the optimum, the production line is relatively less sensitive to variations in the performance of the Inspection and Hand Loading workstation and equally sensitive to variations at the Burn-in Oven and Packing workstations, which are the last two in the line.

2.4.3 Cluster Analysis

A cluster analysis (Vössner and Braunstingl 1996c) was performed on each of the same generations shown in Figure 6, and not surprisingly they agree well with the conclusions above. The potential of this approach can only be appreciated when dealing with even more parameters than three. In which case, it is no longer possible to make 3-D plots, and our only guide will be such cluster diagrams, which are dimension invariant.

Figure 7 shows each of the cluster diagrams for generations 1, 5, 10, and 15. The progression of the algorithm can be traced through the generations in three ways.

1. The number of clusters reduces, showing that fewer areas of fitness are being considered.
2. The color of the clusters becomes lighter, showing that their average fitness is increasing.
3. The compactness of the clusters

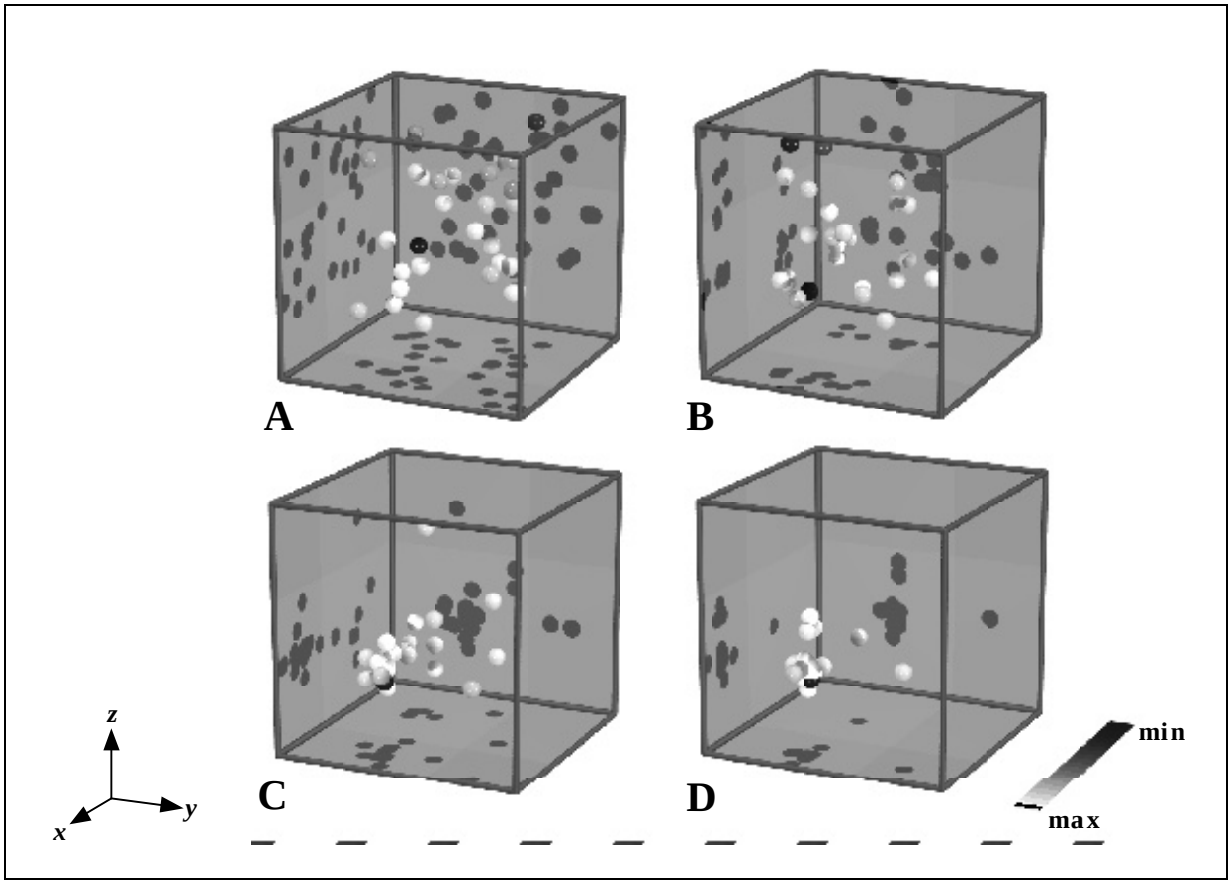

Figure 6: 3-D Plots of GA Generations 1, 5, 10, and 15

increases, showing that individuals are pressing for better performance.

The diagram for generation 15 shows that there are two main clusters, with one being more stable, as indicated by its containing more individuals.

3 CONCLUSIONS

The explanation of the Genetic Algorithm is simple and intuitive, because it relates to every-day life in so many ways, and results are readily interpreted because the algorithm deals only with solutions that are implementable. These two points make Genetic Algorithms very attractive. Further, Genetic Algorithms are robust, do not require gradient calculations, and remarkably efficient on very difficult problems like the production line we presented in this paper. These points make this approach very attractive for the field of stochastic optimization.

Our experience has shown that the Genetic Algorithm was surprisingly quick to apply and readily presented useful and interesting results. It has also the promise of handling greater levels of complexity and more decision parameters than the three we presented, which is a direction for future research. We are currently experimenting with refined algorithms for cluster identification that allow more precise stopping criteria and for speeding up the simulation by adjusting the sample size dynamically.

ACKNOWLEDGMENTS

The authors are indebted to Tyecin Systems Inc. for the use of their TestSim/X simulation software, and the EES&OR Department of Stanford University for the computer resources needed to do these experiments. Furthermore we would like to thank Professor George B. Dantzig and Dr. Gerd Infanger for their support throughout this project and Professor Donald L. Iglehart and Professor Peter W. Glynn for their comments and valuable discussions.

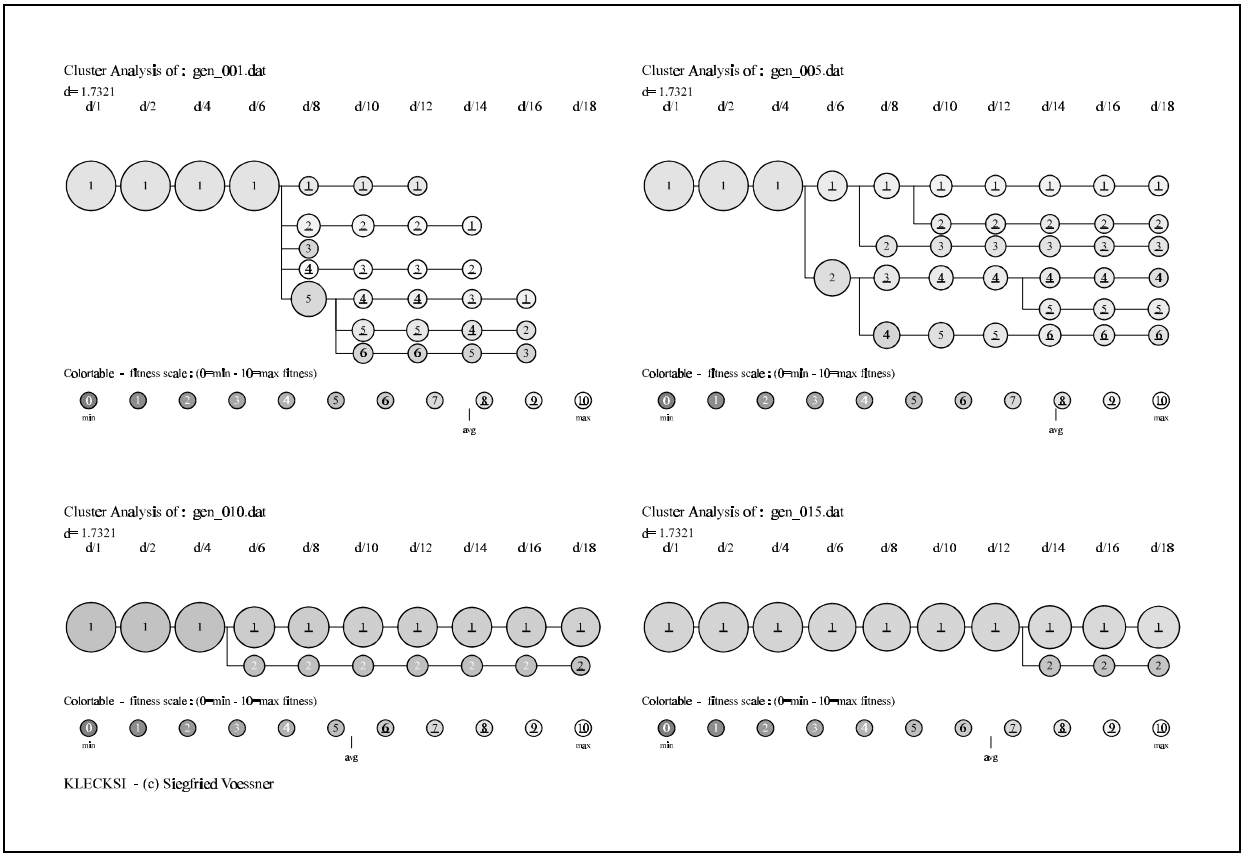

Figure 7: Cluster Diagrams for Generations 1, 5, 10, and 15

REFERENCES

Darwin, C. 1859. *On the Origin of Species by Means of Natural Selection*, J. Murray.

Davis, T. E., and Principe, J. C. A Simulated Annealing Like Convergence Theory for the Simple Genetic Algorithm. In *5th International Conference on Genetic Algorithms*, 174-181, San Diego, USA.

Goldberg, D. E. 1989. *Genetic Algorithms in Search, Optimization, and Machine Learning*, Reading, Massachusetts:Addison-Wesley.

Goldberg, D. E., Deb, K., and Clark, J. H. 1992. Genetic Algorithms, Noise, and the Sizing of Populations. *Complex Systems*, 6, 333-362.

Holland, J. 1975. *Adaptation in Natural and Artificial Systems*, Ann Arbor, Michigan:University of Michigan Press.

Miller, B. L., and Goldberg, D. E. 1995. Genetic Algorithms, Selection Schemes and the varying Effects of Noise. *95009*, IlliGAL.

Rojas, J. C. 1993. Simulation of a Just in Time Assembly Process for Computer Memory Cards at Hewlett-Packard, Puerto Rico, Engineer's Thesis, Stanford University, Stanford, CA.

Tyecin Systems, I. 1994. TestSim/X User's Manual, Tyecin Systems, Inc., Los Altos, California.

Vössner, S., and Braunstingl, R. 1996a. Analysis of Results from Genetic Algorithms. *submitted to Evolutionary Optimization*.

Vössner, S., and Braunstingl, R. 1996b. G.O.A.L. (Genetic Optimization ALgorithm), Genetic Optimization Lab, Graz, Austria.

Vössner, S., and Braunstingl, R. 1996c. KLECKS (Cluster Analysis Package for Genetic Algorithms), Genetic Optimization Lab, Graz, Austria.

AUTHOR BIOGRAPHIES

ROBERT ENTRIKEN has a Ph.D. in Operations Research from Stanford University, a BSEE from Carnegie Mellon University and has over eight years Business Analysis Experience in various industries including: computer software and hardware production, finance, oil refining, and electric power industries. Dr. Entriken is co-author of AIMMS: The Modeling System from Paragon Decision Technology, and has developed economic and financial software for Bank America, Goldman Sachs & Co., and most recently Pacific Gas & Electric.

SIEGFRIED VOESSNER has a MS in Mechanical Engineering and a Ph.D. degree in Industrial Engineering from Technical University of Graz, Austria. He is currently a visiting scholar at the Engineering Economic Systems and Operations Research Department of Stanford University. His research interests focus on Logistics, Stochastic Modelling, Materials Handling, Production Planning and Scheduling, Evolutionary Computation Methods for Nonlinear-, Stochastic Optimization - especially Genetic Algorithms and Genetic Programming.