Research Report AI-1993-05

Exploring Optimal Parameters for Multiple Fault Diagnosis Using the Simple Genetic Algorithm

Mark Juric

Artificial Intelligence Programs
The University of Georgia
Athens, Georgia 30602–7415 U.S.A.

Exploring Optimal Parameters for Multiple Fault Diagnosis Using the Simple Genetic Algorithm

Mark Juric
Artificial Intelligence Programs
The University of Georgia

September 7, 1993

Abstract

Multiple Fault Diagnosis (MFD) is the process of determining the correct fault or faults that are responsible for a given set of symptoms. MFD problems are generally characterized by problem-spaces containing many local minima and maxima. We show that when using Genetic Algorithms to solve these kinds of problems, best results can be achieved with higher than "normal" mutation rates. Schemata theory is then used to analyze this data and show why this genetic operator would give these results.

1 Introduction

Multiple Fault Diagnosis (MFD) is the process of identifying one or more problems or faults which are most likely responsible for a given set of symptoms. In this paper we examine the results of using a simple Genetic Algorithm (sGA) for evaluating the likelihood that a specific set of disorders is the cause of a particular set of symptoms. Optimal results were obtained using an atypically high mutation rate. We will examine the sGA parameters which produced these results, and attempt to explain these atypical configurations in terms of schema theory.

The fitness function for determining the suitability of a diagnosis is the Modified Relative Likelihood function taken from [P$^+$90]. A rigorous mathematical treatment of its theoretical basis is given in [PR87a] and [P$^+$90]. It is an efficient algorithm that requires relatively small amounts of data to operate, and is flexible enough to be used in real-world applications under many different computational methods [PR87a, PR87b, PR89, P$^+$90, P$^+$92].

The MFD problem solved consisted of ten possible symptoms and fifteen possible disorders. This creates a search space of 2^{15} possible diagnoses for any one of the possible 1023 symptom sets. The symptom set with no symptoms was not considered. An exhaustive search for the best diagnosis is impractical in a real time situation; any increase in possible disorders significantly enlarges the search space to the point of making an exhaustive search infeasible.

2 The sGA

The simple Genetic Algorithm (sGA) used was based loosely on [Gol89]. A C port of the sGA by R. E. Smith and Jeff Earickson (available by anonymous ftp from `ftp.aic.nrl.navy.mil`) was

[*]Special thanks to Dr. W. D. Potter for his analytical and editorial advice.

modified for the MFD problem. Any optimizations made were to increase execution speed and did not affect the genetic operators of the sGA.

2.1 Relative Likelihood and the Objective Function

The original "relative likelihood" function comes from [PR87a]. Given a finite set of disorders D and a finite set of symptoms or manifestations M, a tendency matrix C (a subset of $D \times M$) is created. Each member c_{ij} of C defines the probability $P(m_i : d_j \mid d_j)$; given d_j, the probability that d_j causes m_i. Note that this is not equivalent to the conditional probability $P(m_i \mid d_j)$. The probability with which d_j causes m_i remains constant irrespective of the frequency of occurrence of either d_j or m_i [PR87a, P$^+$90]. We also create the vector p_i; the probability that a common disorder causes the symptoms. This information is used to determine the likelihood that a diagnosis DI sufficiently explains observed symptoms M^+.

The values in the tendency matrix may come from historical trends, the analysis of relationships between faults and symptoms, or research data. For instance, in [P$^+$92] a modification of the relative likelihood scheme was used in the Communication Alarm Processor expert system to aid system operators in diagnosing communication network problems in a near real-time capacity. This system was developed at Oak Ridge National Laboratory for the Bonneville Power Administration, and monitored 19 components and 41 alarm groups (for a solution space of 2^{19}). Values in this tendency matrix were taken from "fault trees," structures which showed the relationships among major and minor alarms, and their associated problems.

Once the tendency matrix is established, the relative likelihood, $L(DI, M^+)$, is then the product of three terms: the likelihood that a diagnosis covers less than the given symptom set (L1), the likelihood that a given diagnosis covers more symptoms than the set given (L2), and the likelihood that a common disorder is the cause of the symptoms (L3). The product of these three likelihoods is the fitness of a given diagnosis. These terms are given by:

$$L_1 = \prod_{m_i \in M^+} \left(1 - \prod_{d_j \in DI} (1 - c_{ij}) \right),$$

$$L_2 = \prod_{d_j \in DI} \prod_{m_i \in effects(d_j) - M^+} (1 - c_{ij}), \text{ and}$$

$$L_3 = \prod_{d_j \in DI} \frac{p_j}{(1 - p_j)}.$$

It is clear that L_1 and L_2 will be forced to zero in the cases of incomplete coverage and super-coverage ($c_{ij} = 0$ and $c_{ij} = 1$ respectively). This is unsatisfactory for a fitness function which must be able to categorize DI based on its relative "goodness".

In [P$^+$90] a change to the relative likelihood function, the *modified relative likelihood*, (MRL), is made that resolves this problem. When calculating L_1, all $c_{ij} = 0$ are set to a number close to, but not equal to 0. Likewise, with L_2, all $c_{ij} = 1$ are set to a value near, but not equal to 1. This allows the sGA to rank and compare the relative fitness of diagnoses whose values might otherwise be forced to 0 using the unmodified relative likelihood function.

2.2 Genetic Operators

Genetic Algorithms use the Darwinian principles of natural selection to guide a heuristic search across a problem space. A randomly generated "population" of individuals, each of whom represent a possible solution to the problem at hand, are manipulated using genetic operators according to problem specific probabilities. After these operations, individuals in the population are evaluated by a fitness function and assigned a value that determines their "goodness" or proximity to a solution. In this manner, *population* number of distinct points in the problem space can be searched at once.

The three basic genetic operators: selection, crossover, and mutation, as described in [Gol89] were used in this experiment. Since our purpose was to explore optimal sGA parameter settings and not optimize the sGA itself, we did not incorporate many of the common genetic operator improvements such as elitism or hybrid schemes [P$^+$92, Gol89].

Selection is the process by which an individual in the population is allowed to "mate" or exchange genetic data with another member of the population. Selection probabilities are generated according to the fitness of the current population. Those individuals with higher fitness will generate more offspring in the next generation because their fitness value apportions them a higher probability of being selected for reproduction.

The original sGA used an inefficient summation selection procedure. First, a selection probability was calculated, then a linear search consisting of a summation of individual fitnesses was performed until the selection probability was reached. To expedite execution speed, this linear summation was changed to a modified binary search. Each individual carried with it the sum of the fitness of the population up to that point. A binary search could then be implemented based on that sum. The search was modified so that each pivot in the search had to check a range of values to decide which direction to search next. If the search goal was between the current sum of fitness and the sum of fitness minus the current individual's fitness, then that individual was chosen.

Crossover is the actual process by which mated individuals exchange genetic material. One or more crossover points are selected on each parent, and the children become the result of swapping the genetic materials between these crossover points. In this experiment, crossover was changed to a two point crossover from the simple crossover in the original sGA. This represents the only modification that could be considered an optimization of the actual genetic operators. Each diagnosis and symptom set was represented as an `unsigned` bit string. Each individual bit position set to one represented the presence of that manifestation or disorder in the respective symptom or diagnosis set. For instance, symptom set $M^+_{879} = \{m_1, m_2, m_4, m_5, m_7, m_8, m_9, m_{10}\}$ would be represented by the binary string *1101101111*. During crossover, two loci on the chromosome were chosen at random. Since the diagnoses were represented as bit strings, a function was implemented that would exchange bits between two strings from a start to a finish position. If the first crossover point occurred before the second, the bits in between these points were swapped. Otherwise, the bits from those points to their respective ends were swapped.

The final genetic operator, mutation, ensures a more thorough coverage of the problem space. Mutation stochastically changes the value of a particular locus on an individual. This "error" can aid in preventing the population from stagnating or converging on local maxima or minima by forcing an individual into a previously unexplored area of the search space. Mutation is not generally considered as important as crossover and selection in the genetic algorithm heuristic [Gol89, DeJ75], and for many problems a low mutation rate ($\simeq 1/\textit{population size}$) produces best results. However there seems to be a growing body of work that would indicate that for certain classes of problems, mutation plays a significantly more important role [BG92, FA90, S$^+$89]. Our results bear out these indications.

3 Run Data

The focus of this experiment was to determine the parameter settings which gave the highest number of best fit diagnoses (as determined by an exhaustive search) over the range of 1023 symptom sets. The best fit genotype for each symptom set was written to a file at the end of every run. Each run was stopped after a best solution was reached or until the maximum number of generations was complete.

A population size of 200 was used on all runs. To eliminate the possibility of a particular run getting a randomly better initial population, all populations were started with the same initial population. Each symptom set was run for 25 generations. 2 suites of runs were performed in which only one parameter, either crossover or mutation probability, was varied for each set of 1023 symptoms. The tendency matrix was half-dense (ratio of zero entries to non-zero entries) and randomly generated.

In the first suite of 200 runs, crossover probabilities ranged from 0.05 to 0.50, in increments of 0.05. Mutation probabilities ranged from 0.003 to 0.3 in increments of 0.01563. From the results of previous experiments, this was the range where the best results were expected. In the second suite, crossover probabilities were in two ranges. The first set of 50 runs used a finer grained mutation increment of 0.00058 with crossover varying from 0.05 to 0.5 and mutation ranging across 0.0001 to 0.003. The purpose of this range was to test the effectiveness of extremely low mutation parameters on the MFD problem. In the second set of 135 runs, the crossover range was from 0.55 to 0.75 with mutation ranging from 0.003 to 0.75 in increments of 0.01563. This range was included to test the "generally accepted" parameters, and to test the effect of both high mutation and crossover rates.

All runs were performed on five Sun SPARC workstations. Average run time was between 20 to 50 minutes per trial of 1023 symptom sets. For comparison, the exhaustive search took 27 hours and 36 minutes to complete a full trial.

4 Results

Results for suite one are shown in Figure 1. As expected, a large portion of this run, 45 out of 200 (approximately 23%) produced perfect or near perfect results (1020 to 1023 correct out of 1023). The best result, 1023 out of 1023, used a crossover rate of 0.25 and a mutation rate of 0.175. The worst result, using a crossover of 0.05 and mutation of 0.003, got 711 correct.

Results for set one of suite two (Figure 2) were considerably lower. None of the 50 runs reached more than marginally above 76% accuracy, or 779 out of 1023 correct. This is not surprising since this range covered mutation rates even lower than the recommended settings that would be obtained using a rate inversely proportional to the population[Gol89]. Mutation rates were also significantly lower than those that had previously shown good results. The second set of suite two (Figure 3) showed good results, however not in the ranges that might be expected. For instance [Gol89] recommends a crossover rate of 0.6 with a mutation rate of 0.005 for this type of problem. Though these parameters did not fall on the incremental boundaries used, a special run was made using these settings. This run achieved a success rate of only 77.3%, or 791 out of 1023. Best results in the range for the second set were 1022, achieved using crossover of 0.55 with mutation at 0.175.

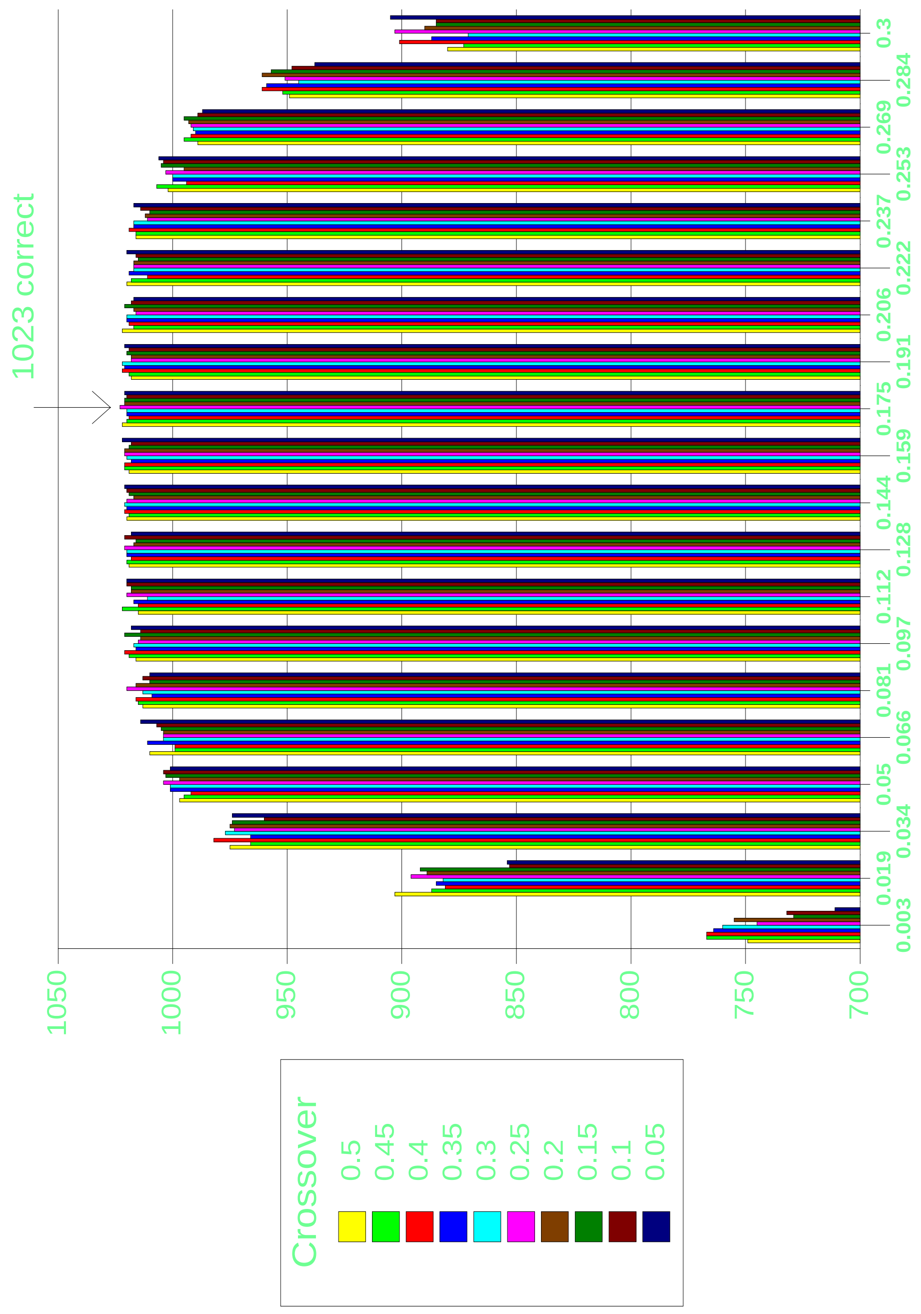

Figure 1: Crossover vs. Mutation. Number correct for Suite 1

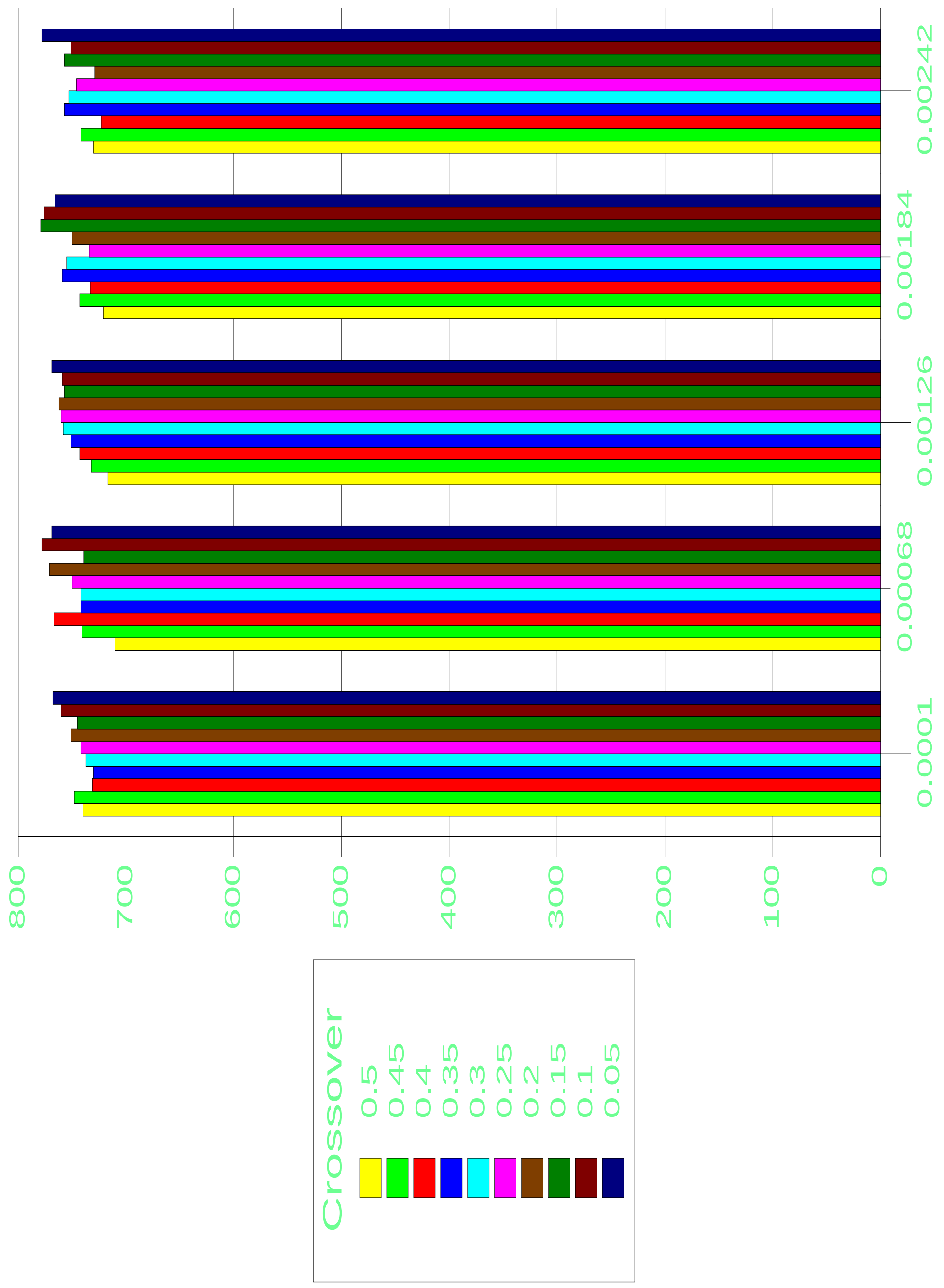

Figure 2: Crossover vs. Mutation. Number correct for Suite 2

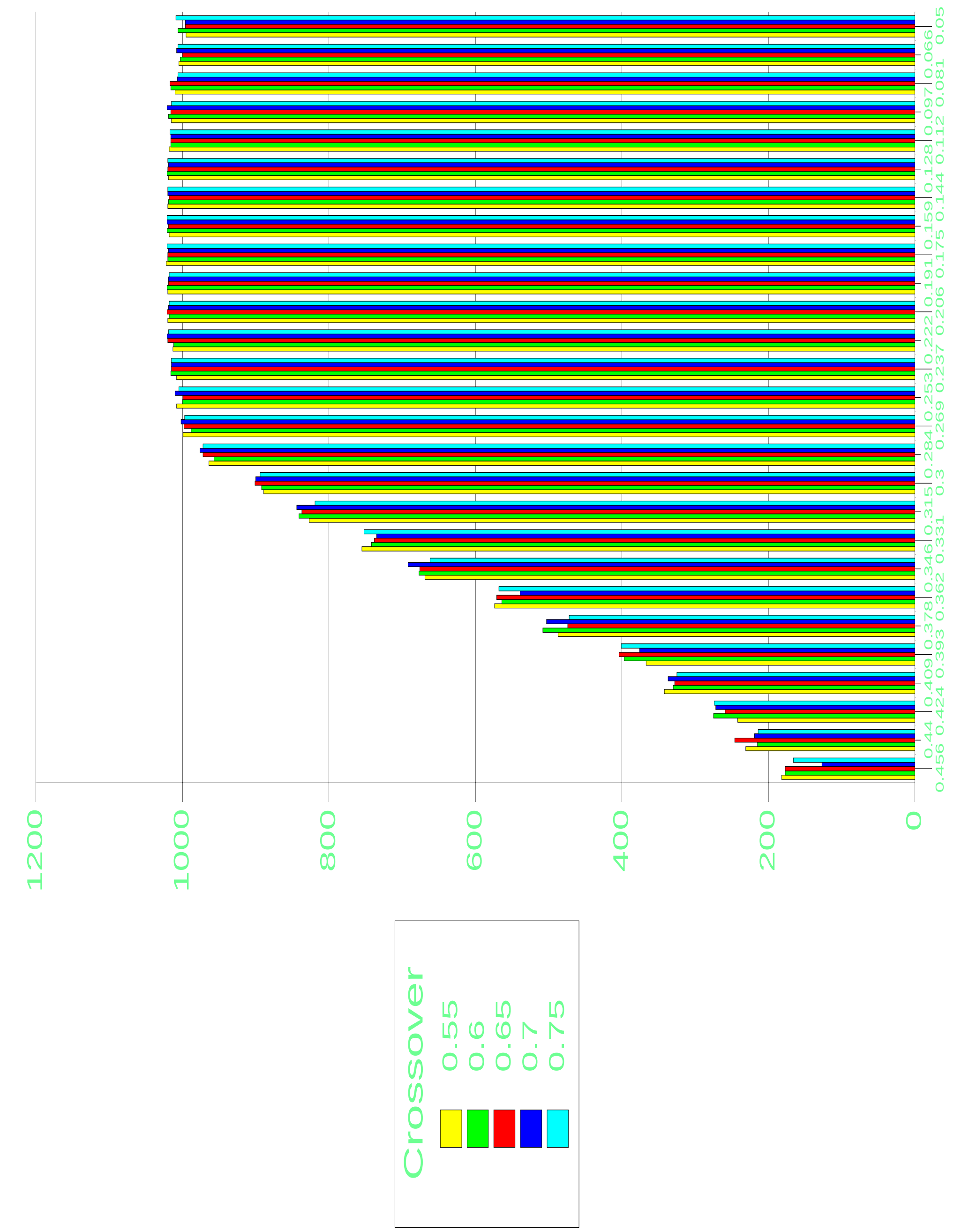

Figure 3: Crossover vs. Mutation. Number correct for Suite 3

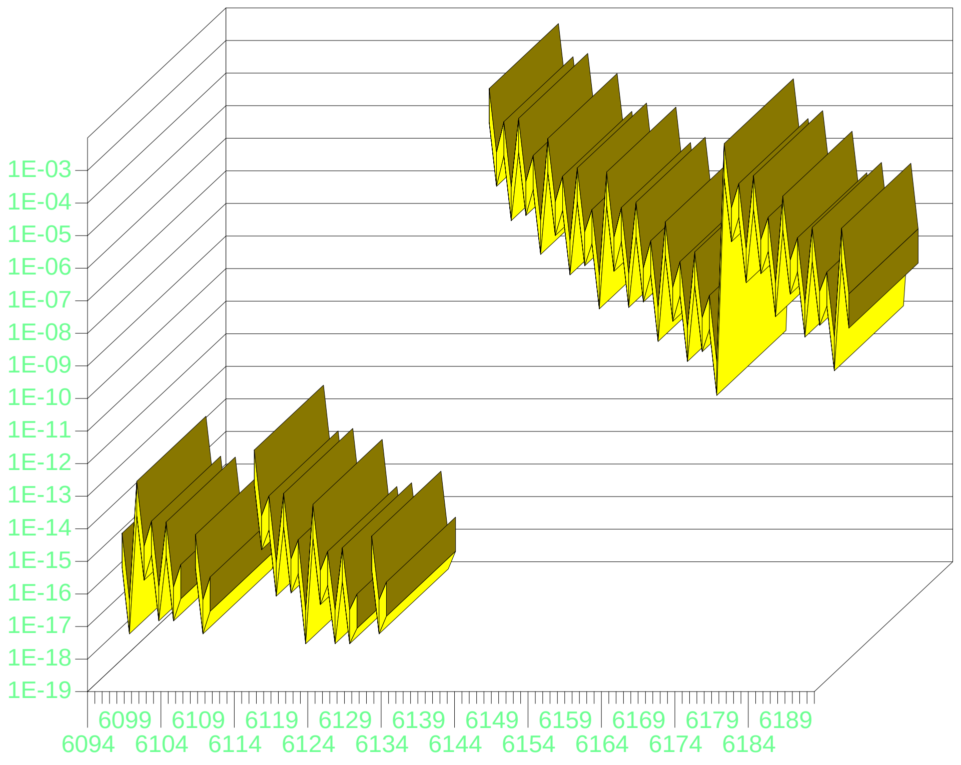

Figure 4: Logarithmically scaled MFD search space

5 Discussion

To understand these results it is necessary to look at the schemas which represent the best fit diagnoses. By comparing the effect of genetic operators on these schema with their effect on best fit schema from functions that *do* benefit from the "rule of thumb" parameters, we can draw conclusions about how mutation and crossover settings should be modified for a given search.

5.1 MFD Search Space

By examining the search space for the MFD problem, we can see why higher mutation rates are necessary to solve this problem quickly. Figure 4 shows the values for the individuals immediately surrounding the best fit diagnosis for symptom set M^+_{6144}, logarithmically scaled to show the whole problem space.

There is no gradient leading to the best diagnosis for crossover to exploit. The binary representation of the best diagnosis is *001100000000000*, whereas the second and third best fit are *001100001000000*, and *001100000000100* respectively.

If we consider the schema as the minimum number of bits defining these strings as best fit, then our defining length becomes 10^1. These long schemata will either be quickly lost through crossover,

[1]There are 11 bit positions that characterize the schema in this example. We strictly follow Goldberg's definition of defining length: $\delta(H) = b_j - b_i - 1$ where $j > i$. See [Gol89] for details.

or, if they are even reasonably fit, cause rapid, premature population convergence. That is usually the case with the MFD problem: populations generally converge to a low fitness within 5 to 10 generations and stay there unless mutation improves their fitness.

In comparison, the search space of an example function optimization, where the function is x^2 or even $x^2+y^2+z^2$ provides a smooth grade for crossover in which most if not all improvements will move the best fit individuals towards the optimum solution. Looking at their binary representations, we can see that only a very small defining length is necessary for the representation of optimum and near-optimum solutions. For instance, the function x^2, optimized on the arbitrarily chosen value 60 has a best fit 14 bit binary representation of 3600 = *00111000010000*. Second and third best fit are 3599 = *00111000001111*, and 3601 = *00111000010001* respectively. We can see that the defining length for these strings is only 4 bits. This compact schema is just the type of building block that crossover needs to function effectively. The small defining length also makes the x^2 function more impervious to mutation during optimization.

5.2 Hamming Distance

There appears to be another relationship among the optimum solutions in the MFD and the x^2 and $x^2 + y^2 + z^2$ functions: the number of bit changes necessary to transform a lesser fit chromosome into a better or optimum fit one.

For our purposes we will define the Euclidean distance between vectors x and y as $d = \sqrt{(x_1-y_1)^2+(x_2-y_2)^2+\cdots+(x_n-y_n)^2}$, with the restriction that $x_i, y_1 \in \{0,1\}$. x and y are also vectors in $n-$dimensional Hamming space [FS91], and we can therefore define the Hamming distance as

$$h = \text{number of mismatched bits of } \ x \text{ and } \ y \ .$$

Looking at the x^2 function, optimized on the value 60, we can see that despite the small defining length of the schema, the second and third best fit have a Hamming distance of 5 and 1 respectively from the optimum value. Though there is a good chance that a third best fit chromosome might improve its fitness through mutation, it is highly improbable that the second best fit would benefit from a higher mutation rate. The average Hamming distance for the first through tenth next best fit is 2.7 for this problem; nearly three correct mutations would be necessary to change a population converged to any one of these values to optimum fitness.

However, looking at the MFD problem for symptom set 6144, we can see that the second and third best fit both have a Hamming distance of 1 from the best value. The average distance for the first through tenth next best fit is in fact only 1.3; rarely would more than one correct mutation be necessary to change a population converged to any one of these values to the optimum value. Because the defining length of this schema is 12 out of a string length of 15, there is a high likelihood that a random bit flip will occur in this range.

To test the effect that the density of the tendency matrix has on the Hamming distance, tests were run optimizing the MFD on third dense and three-quarters dense matrices. In these tests, 10 out of the possible 1023 symptom sets were chosen at random and all 2^{15} possible diagnoses were generated and ranked according to their ability to explain the symptom set. The Hamming distance from the best fit (determined by exhaustive search) was calculated for the first ten best fitnesses.

For the third dense matrix, the average Hamming distance was 1.2, with an average schema length of 11.6. For the three-quarter dense, the average distance was 2.1, and the average schema length dropped down to 5.3. This would seem to indicate that as the problem space gains more data to work with, gradients leading to best fit solutions become more numerous, and schema size falls into a range that can be more readily exploited by crossover.

6 Conclusions

From empirical results, it would seem that mutation should play a much larger role when dealing with a search space with little helpful gradients. Since in problems like the MFD the optimal solutions tend to have larger defining schema lengths, spikier search spaces, and smaller Hamming distances between optimal solutions, a large rate of mutation becomes necessary to efficiently explore the problem space. Problems with smooth gradients tending towards optimum results tend to have smaller schema because of the shorter jumps between fitness values involved. This would make them easier to exploit through crossover and less benefited by mutation.

Though this was not a rigorous mathematical analysis, empirical evidence seems to bear out the underestimated importance of mutation in certain types of problems. Future work on this topic will include a formal mathematical study of the relationship between Hamming distance, schema length, and mutation, and a survey of the types of problems which might benefit from higher than average mutation rates.

References

[BG92] Randall D. Beer and John C. Gallagher. Evolving dynamical neural networks for adaptive behavior. *Adaptive Behavior*, 1(1):91–122, 1992.

[DeJ75] K. A. DeJong. *An Analysis of the Behavior of a Class of Genetic Adaptive Systems*. PhD thesis, University of Michigan, 1975. University Microfilms No. 76-9381.

[FA90] D. B. Fogel and J. W. Atmar. Comparing genetic operators with gaussian mutation in simulated evolutionary processes using linear systems. *Biological Cybernetics*, 63:111–114, 1990.

[FS91] James A. Freeman and David M. Skapura. *Neural Networks Algorithms, Applications, and Programming Techniques*. Addison-Wesley, Reading MA, 1991.

[Gol89] D. E. Goldberg. *Genetic Algorithms in Search, Optimization, and Machine Learning*. Addison-Wesley, Reading MA, 1989.

[P$^+$90] W. D. Potter et al. Diagnosis, parsimony, and genetic algorithms. In *Proceedings of the Third International Conference on Industrial and Engineering Applications of Artificial Intelligence and Expert Systems*, New York, NY, 1990. Springer-Verlag.

[P$^+$92] W. D. Potter et al. Improving the reliability of heuristic multiple fault diagnosis via the EC-based genetic algorithm. *Journal of Applied Intelligence*, pages 5–23, 1992.

[PR87a] Yun Peng and James A. Reggia. Probabalistic causal model for diagnostic problem solving–part 1: Integrating symbolic causal inference with numeric probabilistic inference. *IEEE Transactions on Systems, Man, and Cybernetics*, SMC-17(2):146–162, 1987.

[PR87b] Yun Peng and James A. Reggia. Probabalistic causal model for diagnostic problem solving–part 2: Integrating symbolic causal inference with numeric probabilistic inference. *IEEE Transactions on Systems, Man, and Cybernetics*, SMC-17(3):395–406, 1987.

[PR89] Yun Peng and James A. Reggia. A connectionist model for diagnostic problem solving. *IEEE Transactions on Systems, Man, and Cybernetics*, 19(2):285–298, 1989.

[S$^+$89] J. David Schaffer et al. A study of control parameters affecting online performace of genetic algorithms for function optimization. In *Proceedings of the Third International Conference on Genetic Algorithms*, San Mateo CA, 1989. Morgan Kaufman.