An Investigation of DNA Mapping with Genetic Algorithms: Preliminary Results

Walter Cedeño and Venkateswararao Vemuri
Department of Applied Science and LLNL
University of California, Davis
Lawrence Livermore National Laboratory
Livermore, CA 94550 (wcedeno@llnl.gov, vemuri@icdc.llnl.gov)

Abstract

The physical mapping of DNA structure for every human chromosome is a task that has occupied many Genome labs for years now. Progress towards this goal can be accelerated with algorithms that are less susceptible to noise in the data and can present the scientist with various optimal or near optimal solutions from the experimental data. Genetic Algorithms (GAs) have been applied successfully to noisy environments and multi-modal search spaces. This paper describes a GA model to sequence a set of overlapping clones from restriction fragment data using genetic operators suited for multi-modal function optimization. The performance of this model is compared against other simpler GA approaches. Utility of this model for mapping parts of human chromosome 19 is described.

1 Introduction

Mapping the entire human genome is a task that has occupied many Genome labs for years now (An excellent survey of the state of the art can be found in a special issue of *Los Alamos Science*, number 20, 1992, devoted exclusively to The Human Genome Project). The ultimate goal is to obtain the *DNA sequence* (i.e., order of the bases thymine, cytosine, guanine, and adenine) in every human chromosome. The DNA sequence can be used to identify and locate genes more accurately in the chromosomes. Current technology limits the direct sequencing of DNA to fragments composed of approximately 0.5k base-pairs (Istvanick *et al.* 1993). In order to divide the DNA into fragments up to this resolution level, techniques using restriction enzymes are used. The restriction enzymes cut the DNA at specific locations which are uniformly distributed in the chromosome. Depending on the number of different restriction enzymes used to obtain the fragments the data is called single-digest (one enzyme), double-digest (two enzymes), or n-digest (n enzymes) data. Most mapping is done using single- and double-digest data. Scientists use different restriction enzymes to obtain DNA fragments of the appropriate size.

The Human Genome Center at LLNL is mapping human chromosome 19 which is estimated to be approximately 60 million base-pairs long. The chromosome is divided into islands (contigs ranging between 150k to 200k base-pairs) of overlapping cosmid clones, where each cosmid clone is approximately 40k base-pairs. Each cosmid clone is further divided using a single restriction enzyme (EcoRI) followed by *fingerprinting* (recording the fragment lengths) into fragments ranging from 0.5k to 15k base-pairs. A pictorial view of this process is shown in figure 1. During the process information about the relative location of the islands, cosmid clones, and fragments in the original DNA is lost. Techniques using larger clones are being used to order, orient, and connect the islands in the original DNA (Olson *et al*. 1986; Waterman and Griggs 1986; Branscomb *et al*. 1990; Stallings *et al*. 1990; Fickett 1993).

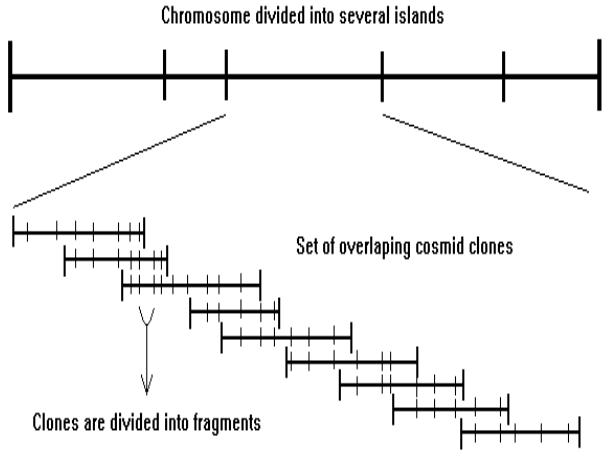

Figure 1: Pictorial view of DNA cutting.

The DNA Restriction Fragment Map Assembly problem consists in finding possible locations of the fragments in each clone, and the sequence of the clones in the island so that the fragment overlap is maximized and the total error between the overlapped fragments is minimized. The problem can be pictured in figure 1 as trying to assemble the islands from the fragments and clones. There are other considerations, like the total length of the mapping should be close to the islands original size. The problem is complicated further by the uncertainty in the data and the possibility of data loss (fragments of the same size are hard to distinguish during fingerprinting). In this paper we only consider the problem of finding the best clone sequences from the fragment sizes on a set of overlapping cosmid clones.

This problem is known to be NP-hard. For example, the case with 10 cosmid clones, there are 10! / 2 possible clone sequences. The problem is complicated further when the location of the fragments within each clone must be determined. Because each clone contains an average of nine fragments, the number of solutions grows rapidly. Exhaustive methods for solving this problem are therefore time consuming.

Restriction-site mapping deals with the problem of locating the fragments within clones using restriction digest data. Most techniques being applied to restriction-site mapping make use of single- and double-digest data to determine the location of the fragments within a clone. Refer to the following works for more information {Stefik (1978), Pearson (1982), Krawczak (1988), Platt and Dix (1993)}.

Other techniques can also be used to sequence larger DNA regions. Branscomb *et al*. (1990) developed a greedy ordering algorithm to get the most probable clone sequence using overlap probabilities between the clones. The algorithm works well when a large amount of overlap between the clones exists and the fragment data has small errors. This approach is prone to getting stuck in local minima and does not use all the available data gathered at great expense.

Cuticchia *et al*. (1992) constructed maps using simulated annealing techniques. In their work clones are ordered according to a measure of similarity between them given by the presence or absence of specific sequences. A signature is assigned to each clone and the algorithm uses it to minimize the error between the actual length of the contig and the given length by the hypothetical clone ordering. Matching signatures are also used to order the clones. In their work they only considered the relationship between consecutive clones.

This paper describes a GA that uses single-digest restriction data on a set of overlapping clones to find multiple solutions for the clone sequences. The application uses genetic operators suited for multi-modal function optimization (Cedeño & Vemuri 1992) to determine solutions to the problem from the fragment data sizes. A mating operator based on Genetic Edge Recombination (Whitley *et al*. 1989) is used to preserve adjacency information and improve convergence towards multiple solutions at the same time. Results are given for two data sets of overlapping clones from human chromosome 19.

Section 2 of the paper describes the GA model used for this problem. Section 3 defines the genetic operators for this problem. Section 4 shows the results obtained on different data sets. Section 5 gives comments and conclusions about the approach and section 6 describes future work.

2 GAs for Multi-modal Search

Genetic algorithms (Holland 1975) are a search technique based on the principles of natural selection and genetic recombination. GAs have been shown to work well in noisy environments and complex search spaces and have been applied to a wide variety of problems ranging from scheduling to aircraft design (Schaffer 1989; Belew & Booker 1991). In this section a GA suitable for multi-modal function optimization is described.

The GA model used has the ability to converge to multiple solutions at the same time by encouraging competition between individuals within the same local optima. The following pseudo-code summarizes the salient features of the method:

```
1.Generate initial population of N individuals
2.For gen = 1 to MAX_GEN
3. For i = 1 to N
4.  Select mates using crowding selection
5.  Mate and mutate
6.  Insert offspring in the population using
    worst among most similar replacement.
```

If we use fitness-proportionate reproduction in step 4 shown above and replace the lowest fitted individuals in the population (step 6) with the

newly generated offspring, this model corresponds to a steady-state GA (Whitley 1988; Syswerda 1989). In contrast with the most common generational GA, offspring are available for mating as soon as they are generated, and good individuals can survive for many generations (a generation is every N mating operations in this model).

What makes this model useful in multi-modal search spaces is the use of *crowding selection* and *worst among most similar* replacement. In crowding selection individual i is selected from the population. It's mate is selected as the most similar individual from a group of *CSS* (crowding selection size) individuals chosen at random from the population. Crowding was introduced by De Jong (1975) where offspring replaced similar individuals from the population. In this model crowding selection is used to promote mating between individuals within the same local minima. In worst among most similar replacement (see figure 2) each offspring is likely to replace a low fit individual from the same niche. Form *CF* (crowding factor) groups with *CS* (crowding size) individuals taken at random from the population. Identify the individual most similar to the offspring in each group. Select from those the individual with the lowest fitness to die. A similar technique called *enhanced crowding* (Goldberg 1989) has been used before, but there the most similar individual out of a group of worst candidates is replaced.

The similarity between two individuals is determined by measuring their proximity in the decoded parameter space (phenotype). The smaller the value, the closer are the two solutions for the problem. An example of such a metric is the Euclidean distance for function optimization. Measuring distance between decoded genes instead of the genes themselves has been shown (Goldberg & Richardson 1987; Deb 1989) to give better results for multi-modal function optimization. An example of the use of this function during replacement is shown figure 2.

		CF	Sim	Most Similar Individual	
POP	FIT	Groups	val	of each group	Replaced
10101010	89	11111000	10	11111000	10111001
00010010	65	00000111	20		
11111000	53				
00100111	45	11100011	30	00011011	
00000111	90	00011011	5		
10111001	23				
11100011	12	00000111	20	10111001	
00011011	11	10111001	24		

Figure 2: Overview of worst among most similar replacement with $CF = 3$ and $CS = 2$.

Other approaches for multi-modal function optimization do exist. De Jong (1975) used crowding to improve convergence of the GA in multi-modal functions. In crowding offspring are inserted in the population by replacing the most similar individual from a group taken at random from the population.

Goldberg & Richardson (1987) used the sharing concept (Holland 1975) to divide the population into niches according to a similarity measure between them. In sharing, the individuals fitness is degraded according to the presence of other individuals in the same niche. Deb (1989) applied mating restriction to sharing methods. Mating restriction only allows individuals within the same niche to mate. Yin & Germay (1993) introduced cluster analysis to sharing in order to reduce it's complexity.

Mahfoud (1992) introduced *deterministic crowding*. In this approach all individuals are allowed to mate in every generation and offspring competed with their parents for slots in the population. The two parents are paired with the most similar offspring and the individual with the lowest fitness dies.

Beasley *et al.* (1993) applied traditional GAs to multi-modal function optimization by using a fitness derating function to prevent convergence to a known local optima. In their approach the GA is applied iteratively to the problem and every solution found in previous iterations is used to derate the fitness of individuals near them.

The approach used here has been successful in locating and maintaining multiple solutions in many generations for function optimization and combinatorial problems (Cedeño 1992). The complexity of the algorithm is not increased substantially. No prior knowledge of the search space is needed and niches get formed naturally. The idea is to maintain diversity by encouraging competition between members from the same niche. To improve convergence lowest fitted individuals are likely to be replaced by offspring belonging to the same niche. No restriction is imposed during mating to allow the exploration of other regions of the space. In the same manner an offspring may replace an individual from another niche, thus allowing competition within niches as well. Survival of the fittest is applied during the re-

placement step allowing the GA to converge to better solutions within each niche.

3 Problem Representation

In this section the problem's genetic operators are defined. First, we will examine the encoding for the chromosome to describe how clone sequences are represented. Second, we describe how the fragment sizes are used to define the fitness function for the problem. Third, the edge recombination mating operator is described. And last, the function that measures similarity between two clone sequences is described.

Before going into the details about the operators, it is important to show how the data for the problem is presented to the GA. Figure 3 shows fragment sizes obtained from fingerprinting for a set of overlapping cosmid clones. Some of the fragments from different clones have similar sizes since some of them come from the same original DNA and are obtained using a single restriction enzyme. The data for this problem consist of the M cosmid clones with their fragments and the tolerance measure E which is use to determine if two fragments are of the same size. That is, two fragments $F1$ and $F2$ are considered to be of the same size if $| F1 - F2 | < E$.

ALLELE NUMBER	CLONE ID	FRAGMENT SIZES (k base-pairs)
0	5154	16.55 4.4 1.68 1.07 4.81 8.5
1	7442	.79 .79 2.6 4.35 8.24 2.7 6.9 5.16
2	21230	.96 1.68 1.08 4.77 8.47 1.44 2.37 6.29 0.62
3	8131	.92 3.73 19.8 4.43 1.69 1.25 4.68 5.63
4	18993	.96 6.31 5.48 8.61 7.29 .81 .81 2.6 4.36 1.92
5	5435	2.89 8.24 2.7 6.9 5.14 5.14 2.89 1.54
6	7255	1.04 8.21 2.69 6.89 5.12 5.12 2.88 1.94 2.42 1.37 3.33
7	12282	4.52 5.13 5.13 2.87 1.94 2.42 1.39 3.35 5.41
8	27714	6.69 5.07 5.41 2.88 1.92 2.32 1.4 3.35 5.46 17.65 1 10.49 .58 1.74
9	10406	2.03 1.43 2.34 6.28 5.46 8.58 7.27

Figure 3: Sample fingerprinting data set.

3.1 Chromosome Encoding

The encoding for this problem is very simple. Each allele in the chromosome has a value between 0 and M-1 corresponding to one of the cosmid clones. No two alleles have the same value and mating and mutation will preserve this constraint. Using the data in figure 3, an allele with value 0 corresponds to clone with ID 5154 and an allele with value 9 to clone ID 10406. The clone sequence (5154, 21230, 10406, 7255, 12282, 27714, 8131, 18993, 7442, 5435) is represented by the chromosome (0 2 9 6 7 8 3 4 1 5). The initial population is generated by picking at random values between 0 and M-1 without replacement for each individual.

3.2 Fitness Function

To calculate the fitness of an individual the number of fragment matches and the error between the fragments is considered between all consecutive clones. The fragment sizes are represented using integer numbers by multiplying them by 100. All values are computed using integers to accelerate computation of the fitness function. Prior to the execution of the GA two matrices are calculated; one containing the number of fragments that match within tolerance E between any two clones, the other containing the total error between those fragments that match. Figure 4 shows both matrices for the data in figure 3. The error between two clones is given by the sum of the errors between all fragments that matched. For example, between clone 8131 and 5154 there are two pairs of fragments, 169 with 168 and 443 with 440 that match within tolerance. The total error between both pairs of fragments is 1 + 3 = 4 which is the value shown in the matrix (row C3, col C0).

Number of matches between clones

CLONE		C0	C1	C2	C3	C4	C5	C6	C7	C8	C9
C0:	5154:	6	1	4	2	1	0	1	0	2	1
C1:	7442:	1	8	0	1	4	4	4	1	1	0
C2:	21230:	4	0	9	3	2	1	3	2	5	3
C3:	8131:	2	1	3	8	2	0	0	1	2	0
C4:	18993:	1	4	2	2	10	1	3	2	3	4
C5:	5435:	0	4	1	0	1	8	6	3	2	0
C6:	7255:	1	4	3	0	3	6	11	7	7	3
C7:	12282:	0	1	2	1	2	3	7	9	7	4
C8:	27714:	2	1	5	2	3	2	7	7	14	3
C9:	10406:	1	0	3	0	4	0	3	4	3	7

Total error in the matches

CLONE		C0	C1	C2	C3	C4	C5	C6	C7	C8	C9
C0:	5154:	0	5	8	4	4	0	3	0	13	8
C1:	7442:	5	0	0	8	5	12	17	3	9	0
C2:	21230:	8	0	0	14	2	10	20	10	23	5
C3:	8131:	4	8	14	0	11	0	0	9	13	0
C4:	18993:	4	5	2	11	0	10	19	9	6	10
C5:	5435:	0	2	10	0	10	0	10	4	8	0
C6:	7255:	3	9	16	0	19	10	0	7	26	23
C7:	12282:	0	3	10	9	9	4	7	0	20	26
C8:	27714:	13	9	23	13	11	8	26	20	0	10
C9:	10406:	8	0	5	0	10	0	23	26	5	0

Figure 4: *Match* and *Error* matrices (E=10).

From the matrices we can observe that by looking only at the number of fragment matches between the clones it is not sufficient to determine which two clones go together in the sequence. For

example, clone 5154 (*Match* matrix in figure 4) has the most number of matches with clone 21230 (i.e., 4 matches, as shown by the entry in row C0 and column C2). The next closest match for clone C0 is with either clone 8131 or 27714 (see the entry 2 in columns C3 and C8). Using only the number of matches between consecutive clones was not sufficient to discriminate between clones with the same count. The problem is due to false matches (by chance) between fragments of similar sizes. In this case clone 8131 is a better candidate because it contains less error than clone 27714 (see entries in *Error* matrix, row C0, columns C3 and C8). By incorporating the total error in the matches the GA is able to discriminate further between clones. Detailed information of the use of these matrices to calculate fitness is given by the following expression:

$$\sum_{i=0}^{M-1} \left(\frac{Match[C_i, C_l]}{Count[C_i]} \times \left(1 - \frac{Error[C_i, C_l]}{Match[C_i, C_l] \times E}\right) + \frac{Match[C_i, C_r]}{Count[C_i]} \times \left(1 - \frac{Error[C_i, C_r]}{Match[C_i, C_r] \times E}\right)\right) * 100$$

where C_l and C_r are the clones to the left and right (if any, otherwise $Match = 0$) respectively of clone C_i.

From the fitness function note that not only the number of fragment matches are used, but also the percentage relative to the number of fragments in the clone. We did this because we wanted to give a higher fitness to those clones that match a higher percentage of their regions. For example, clone 5154 has two matches with clones 8131 and 27714 that have 8 and 14 fragments respectively. A higher value is given when the clone 5154 is near 8131 since 2/8 is a higher percentage than 2/14. Dividing by the number of fragments in the clone pushes clones with a higher percentage of matches closer together. The error is used to reduce this count by the percentage of error in the matched fragments. Higher error values will reduce fitness.

Before settling on this fitness function, others were considered: a) adding the number of matches alone, b) subtracting the error, c) combination of both. In those cases, the GA was not able to find the correct solution, although parts of the solutions were correct. In future work the number of matches and error between every three clones can be added to improve the fitness measure.

3.3 Mating and Mutation Operators

The mating operator is based on the Genetic Edge Recombination (Whitley *et al*. 1989) with minor modifications. This operator was applied successfully to the Traveling Salesperson Problem (TSP). Just as in the TSP, the important information in this problem is adjacency of the alleles, not the order of the alleles in the chromosome. The idea is to recombine the links (pairs of clones) between two parents such that common links are inherited by the offspring. Figure 5 presents an example of this operator. First, those links that are common between both parents are passed to the offspring. Those alleles that are not passed to the offspring are randomly assigned to the available positions.

The differences between this operator and the original work by Whitley *et al.* are in the number of offspring generated and in the assignment of alleles not transferred from the parent. Here we generated two offspring instead of one because the location of the links in the clone sequence is important to our problem. In TSP the chromosome is circular thus the location did not matter. We allow both parents to pass the location of the links to their offspring. To assign the other alleles we select them at random from those clones not passed by their parents. In the original operator the links are assigned from those present in any of the two parents. Alleles with fewer links are assigned first to prevent from running out of links for a given allele.

```
     Parent 1      Similar links    Offspring 1
(6781239450) (-781----50) (3781462950)

     Parent 2                       Offspring 2
(1879502634) (187-50----) (1873506249)
```

Figure 5: Mating operator for clone sequencing.

The mating operator used here is doing much more exploration of the search space than the edge recombination operator. This extra work is reduced somewhat by the fact that mates are selected using crowding selection and therefore they have common features between them. Exploration is limited to a smaller region within the entire search space. Setting alleles at random from unassigned clones allows links to reappear which might not have happened if mutation alone is used.

Mutation is applied on an individual basis. After an offspring is generated it is mutated if a biased coin flip is true. When this happens a link

from the offspring is selected at random and all alleles from that link to the last position of the chromosome are reversed. For example, the offspring (1 8 7 3 5 0 6 2 4 9) after mutation can result in (1 8 7 9 4 2 6 0 5 3) if the link between allele 7 and 3 is selected to mutate.

The mating and mutation operator are compatible with each other in the sense that they both operate on links. The building blocks of this problems are based on the links between clones in the sequence. The GA operates on these links so that the most useful ones are passed from generation to generation.

3.4 Similarity Function

The similarity function is very simple also. It counts the number of dissimilar links between two individuals. Using the parents from figure 5 they have 6 different links corresponding to the five alleles not assigned to the offspring. This metric measures proximity between two clone sequences by the different links they have and not the position of the alleles. For example, the sequence (0 1 2 3 4 5 6 7 8 9) and (9 8 7 6 5 4 3 2 1 0) have a distance of zero since all the links are the same. This metric captures the essential aspect of the problem since both solutions are equivalent in our problem.

4 Results

The results presented in this section were obtained on a SGI IRIS 4D computer under IRIX operating system. running the GA application written in C. The parameters for the GA are the following:

Population size:	200
Mutation probability:	0.06
Crossover probability:	0.95
Crowding selection group size (*CSS*):	10
Crowding sub-population size:	10
Crowding factor (*CF*):	3
Maximum number of generations:	150
Tolerance *E*:	10

These parameters were picked after various trials. The population size of 200 was sufficient for the data submitted to the application. Other sizes (in multiples of 50) were tried, but higher sizes did not provide new information about the problem and lower sizes in some cases did not converge to the best solutions seen before in the allowed number of generations. Mutation was set at 0.06, therefore an average of 12 individuals were mutated every generation. This low value of mutation was selected to avoid eliminating the best of population frequently and allow faster convergence within each niche. On the other hand the mating probability was set to a high value of 0.95 because in our GA all individuals have a higher chance of mating with a similar individual, therefore their similar traits will be passed to the next generation. The group size for crowding selection and crowding sub-population was set at 5% of the population size with a crowding factor of 3. For each individual at most 5% of the population were examined for selection and at most 15% of the population were examined for replacement. These values allowed a diverse population to co-exist during the number of generations allowed and did not restricted competition between individuals from different niches. The tolerance value *E* was set to 10 to minimize false matches due to chance. Higher values of *E* increased the false matches more than true matches and therefore more possible clone sequences were found.

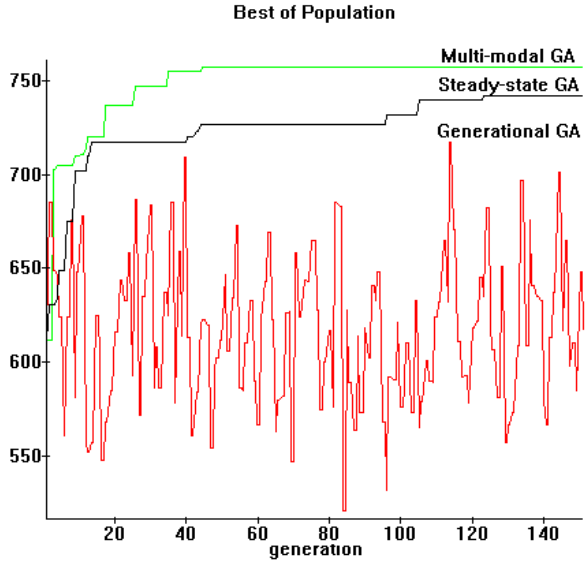

Figure 6: Performance of three GA models.

We compared the performance of the multi-modal paradigm with a generational GA and a steady-state GA. Crossover probability was set at 0.6 for the generational GA and the generation gap was set to 10 in the steady-state GA. Roulette wheel selection was used to select both mates. Figure 6 shows the best in each generation for all three models for data set 2. Similar results were obtained in data set 1 with the exception that the steady-state GA was able to find the global solution. In both cases the multi-modal GA was

able to find a superior solution faster and maintain different solutions throughout the search.

Following are some of the best sequences obtained for two different sets of overlapping clones. Data set number 1 is shown in figure 3 and the data set number 2 is not shown here. The GA took an average of 50 seconds for each run. Figure 7 shows the actual sequence for the data sets and the clone sequences (with their fitness) obtained by the GA. For data set 1 the GA was able to find the actual clone sequence. From the other sequences found, the fitness values of the next best is 15 less than the actual sequence. Similar gaps exist between all the sequences show in figure 7 for data set 1. From the solutions we can see that the last four sequences are the result of a single mutation from the actual sequence.

```
Set 1 actual sequence:
(8131 5154 21230 10406 18993 7442 5435 7255 12282 27714)  752

Best clone sequences found with their fitness:
(8131 5154 21230 10406 18993 7442 5435 7255 12282 27714)  752
(8131 5154 21230 27714 12282 7255 5435 7442 18993 10406)  737
(8131 5154 21230 10406 27714 12282 7255 5435 7442 18993)  729
(8131 27714 12282 7255 5435 7442 18993 10406 21230 5154)  718
(8131 5154 21230 10406 18993 27714 12282 7255 5435 7442)  710

Set 2 actual sequence:
(12595 6722 26999 29626 29064 18301 19811 29035 17755 28828 20235) 749

Best clone sequences found with their fitness:
(12595 26999 6722 29626 29064 18301 19811 29035 17755 28828 20235) 757
(12595 26999 6722 29626 29064 18301 19811 29035 17755 20235 28828) 755
(12595 26999 6722 29626 29064 18301 28828 20235 17755 29035 19811) 751
(12595 26999 6722 29064 29626 18301 19811 29035 17755 28828 20235) 750
(26999 6722 12595 26626 29064 18301 19811 29035 17755 28828 20235) 750
(26999 6722 12595 29626 29064 18301 19811 29035 17755 20235 28828) 748
(12595 26999 6722 29626 29064 18301 28828 20235 17755 19811 29035) 748
(12595 26999 6722 29064 29626 18301 19811 29035 17755 20235 28828) 748
(26999 12595 6722 29626 29064 18301 19811 29035 17755 28828 20235) 747
(12595 26999 6722 29626 29064 18301 19811 17755 29035 20235 28828) 744
```

Figure 7: Sequences for data set 1.

The data for set 2 presented a more challenging problem for the GA. In this case the best sequence had clones 6722 and 26999 transposed from the actual sequence. The fitness for the actual sequence is 749, which is the fifth best score when compared with all solutions found. The actual sequence was obtained in some of the runs, but did not survive until the last generation. Another observation is that there is a difference of 13 or less in the fitness between all the sequences found. Some of the sequences are mutations of others, but there is more diversity when compared with the solutions for data set 1.

5 Comments and Conclusions

Among the things that were observed, two items deserve further discussion. Data containing clones with fewer than five fragments were normally sequenced in a wrong location by the GA. This is due to the lack of fragment matches. In a clone the corner fragments will not match with high probability their counterparts in the preceding and succeeding clones. For clones with less than five fragments, it means that in the average, half or more of their region is not useful for fitness and in some cases leads to more false matches. Clone with less than five fragments were usually placed first or last in the clone sequence by the GA.

When large amounts of overlaps existed between 3 or 4 clones, the GA had difficulty deciding the sequence between them. An example of this behavior was observed with data set 2. We believe that this is happening because the fitness function is only looking for matches between the clones to the left and to the right separately without accounting for the fragments which are common between all three clones. An improved fitness measure is needed to account for fragment matches between three or more clones.

Overall the GA worked well with the data presented to it. Using the correct set of genetic operators was very important to find a GA model that will find good solutions to the problem. Using a multi-modal approach was very useful for this problem also since it prevented premature convergence and at the same time explored the search space in a more efficient manner. Defining the operators for mating, mutation, fitness, and similarity measure to work with adjacency information between the clones rather than clone positions gave the GA the correct set of tools to converge towards the most probable solutions. More information must be incorporated into the fitness evaluation to distinguish even further between the best clone sequences and other similar ones.

6 Future Work

There are other things that we are planing to try out in the near future. Improve the fitness function to account for matches between more than two clones. Account for the error between the original island length and the one obtained from the sequence found by the GA and include clone signature information when available.

Compare results with other methods available for larger regions. Extend the application to present a partial fragment map using the fragment data. Measure the effect on the GA of different levels of error in the data .

Acknowledgments

Special thanks to Mr. Tom Slezak and Dr. Elbert Branscomb for describing the problem and providing the test data. This work was supported (in part) by the Applied Mathematics Program of the Office of Energy Research (US. department of energy) under contract number W-7405-Eng-48 to Lawrence Livermore National Laboratory.

References

D. Beasley, D. R. Bull, and R. R. Martin, A Sequential Technique for Multi-modal Function Optimization, To be published in *Evolutionary Computation*, February 1993.

R. K. Belew and L. B. Booker, eds., *Proceedings of the Fourth International Conference on Genetic Algorithms*, Morgan Kaufmann Publishers San Mateo, CA, July 1991.

E. Branscomb, T. Slezak, R. Pae, D. Galas, A. V. Carrano, and M. Waterman, Optimizing Restriction Fragment Fingerprinting Methods for Ordering Large Genomic Libraries, *Genomics* 8, 351-366, 1990.

A. J. Cuticchia, J. Arnold, and W. E. Timberlake, The use of simulated annealing in chromosome reconstruction experiments based on binary scoring, *Genetics* 132, 591-601, 1992.

W. Cedeño and V. Vemuri, Dynamic multi-modal function optimization using genetic algorithms, *Proc. of the XVIII Latin-American Informatics Conference*, Las Palmas de Gran Canaria, Spain, August 1992.

W. Cedeño, Genetic algorithms in SISAL to solve the file design problem, *Proc. of the Second SISAL User's Conference*, San Diego CA, December 1992.

K. A. De Jong, An analysis of the behavior of a class of genetic adaptive systems, Doctoral dissertation, University of Michigan, *Dissertation Abstracts International* 36(10), 5140B, 1975.

K. Deb and D. E. Goldberg, An investigation of niche and species formation in genetic function optimization, *Proceedings of the Third International Conference on Genetic Algorithms*, J. D. Schaffer, ed., 42-50, Morgan Kaufmann Publishers San Mateo, CA, June 1989.

J. W. Fickett and M. J. Cinkosky, A genetic algorithm for assembling chromosome physical maps, Unpublished, 1993.

D. E. Goldberg and J. Richardson, Genetic algorithms with sharing for multimodal function optimization, *Proceedings of the Second International Conference on Genetic Algorithms*, J. J. Grefenstette, ed., 41-49, Lawrence Erlbaum Associates, Hillsdale, NJ, June 1987.

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

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

W. Istvanick, A. Kryder, G. Lewandoeski, J. Meidânis, A. Rang, S. Wyman, and D. Joseph, Dynamic methods for fragment assembly in large scale genome sequencing projects, *Proceedings of the Twenty Sixth Annual Hawaii International Conference on System Sciences*: Architecture and Biotechnology Computing, T. N. Mudge, V. Milutinovic, and L. Hunter eds. IEEE Computer Society Press, 534-543, Wailea, Hawaii, 1993.

M. Krawczak, Algorithms for the restriction-site mapping of DNA molecules. *Proc. Natl. Acad. Sci.* USA 85, 7298-7301, 1988.

S. W. Mahfoud, Crowding and preselection revisited, *Proceedings of Parallel Problem Solving from Nature 2*, R. Männer and B. Manderick, eds., 27-36, Elsevier Science Publishers B. V., 1992.

M. V. Olson, J. W. Dutchik, M. Y. Graham, G. M. Brodeur, C. Helms, M. Frank, M. MacCollin, R. Scheinman, & T. Frank, Random-clone strategy for genomic restriction mapping yeast. *Proc. Natl Acad Sci.* USA 83, 7826-7830, 1986.

W. Pearson, Automatic construction of restriction site maps, *Nucleic Acids Research* 10, 217-228, 1982.

M. D. Platt and T. I. Dix, Construction of restriction maps using a genetic algorithm, *Proceedings of the Twenty Sixth Annual Hawaii International Conference on System Sciences*: Architecture and Biotechnology Computing, T. N. Mudge, V. Milutinovic, and L. Hunter eds. IEEE Computer Society Press, 756-762, Wailea, Hawaii, 1993.

J. D. Schaffer, ed., *Proceedings of the Third International Conference on Genetic Algorithms*, Morgan Kaufmann Publishers San Mateo, CA, June 1989.

R. L. Stallings, D. C. Torney, C. E. Hildebrand, J. L. Longmire, L. L. Deaven, J. H. Jett, N. A. Doggett, and R. K. Moyzis, Physical mapping of human chromosomes by repetitive sequence fingerprinting, *Proc. Natl. Acad. Sci.* USA 87, 6218-6222, 1990.

M. Stefik, Inferring DNA structures from segmentation data, *Artificial Intelligence* 11, 85-114, 1978.

G. Syswerda, Uniform crossover in genetic algorithms, *Proceedings of the Third International Conference on Genetic Algorithms*, Morgan Kaufmann Publishers San Mateo, CA, June 1989.

D. Whitley, GENITOR: a different genetic algorithm, *Proceedings of the Rocky Mountain Conference on Artificial Intelligence*, Denver Colorado, 1988.

X. Yin and N. Germay, A fast genetic Algorithm with sharing scheme using cluster analysis methods in multimodal function optimization, *Proceedings Artificial Neural Nets and Genetic Algorithms,* Austria, Springer-Verlag, 450-457, 1993.