A Representation for Permutation Optimization with a Combinatorial Genetic Algorithm

Robert E. Smith
Engineering Science and Mechanics
The University of Alabama
Box 870278
Tuscaloosa, AL 35487
Email: rob@comec4.mh.ua.edu

Dan Holtkamp
Hewlett-Packard
Fort Collins, CO
Email: dgh@hpfitst2.fc.hp.com

Abstract

This paper presents a way of representing problems of sequence in a traditional, combinatorial genetic algorithm (GA). This representation differs from so-called order-based (or permutational) GA representations. A combinatorial representation is advantageous, given that order-based GAs are complex, and have limited analytical and empirical success when compared to combinatorial GAs. The representation involves letting the genetic algorithm specify a rank for each element of the problem, and then sequencing the elements according to the rankings. Ties in rank are resolved either at random or by a problem specific (knowledge augmented) operator. Promising results are presented using this sort of representation for traveling salesperson problems. Extension of this sort of representation to other problems is also discussed.

Keywords: Genetic algorithms, traveling salesperson problems, order-based genetic algorithms, representation

Introduction

In 1985, Goldberg and Lingle were examining the problem of adjusting linkages between bits in genetic algorithm (GA) representations. Usually, GAs are *combinatorial* optimizers that are based on the mechanics of natural genetics. In their exploration, Goldberg and Lingle developed a method for re-arranging bits in a GA through a crossover-like operator. Their efforts to test this method led to a new area of application for GAs: problems of sequence or permutation problems. The resulting GAs are often called *order-based GAs*.

This development, and the popularity of certain NP-complete permutation problems (chiefly the *traveling salesperson problem* or *TSP*, which Goldberg and Lingle used as a test problem), led to a flurry of developments in order-based GAs. Many used direct representations of permutations, with special crossover operators (like Goldberg and Lingle's PMX (1985)) that preserved valid permutations (Oliver, Smith & Holland, 1987). Some used combinatorial representations, with specialized and often complex decoding and constraint satisfaction operators to ensure valid permutations (Starkweather, McDaniel, Mathias, Whitley & Whitley, 1991).

Order-based GAs have met with mixed success when compared to GAs applied to combinatorial problems. Although theory on their operation exists, it is somewhat less firm than the theory for combinatorial GAs. This paper presents a new, simple representation that allows a combinatorial GA to operate on a permutation problem in a natural way.

Combinatorial Versus Permutational GAs

In a GA, one must combine different possible solutions as a part of the search process. The typical GA recombination operator is crossover, where half of one solution is spliced to half of another. For instance, given two bit strings that represent solutions to a problem:

1 0 1 0 1 0 1
0 1 1 0 1 1 0

One could split the strings at a randomly selected crossover point (say, between the second and third bits), and create the strings

0 1 1 0 1 0 1
1 0 1 0 1 1 0

This example problem is combinatorial: GA solutions are represented by combinations of bits. Theory and empirical evidence suggest that this sort of GA is robust. That is, it is effective across a broad class of problems (Goldberg, 1989, Holland, 1975).

To see how things differ in a problem of sequence, consider the TSP. This problem is specified by a list of city locations. The goal of the search is to find a route that visits each city once and only once, and has minimum length.

A natural representation for this problem is a list of city names, or, more generally, city numbers. For instance, the string

1 2 4 3

could represent a tour (or sub-tour) that visits city 1, followed by city 2, followed by city 4, and finally visiting city 3.

Now consider two such solutions:

1 2 4 3
2 3 4 1

If one attempts to cross these strings at their midpoint, the following two strings result:

1 2 4 1
2 3 4 3

Neither of these tours is valid, since they violate the restriction of visiting each city once and only once.

To prevent this quandary, new forms of GAs were developed. They used special crossover operators and repair schemes to ensure that no invalid tours were generated. The results from these efforts have been mixed (Fox & McMahon, 1991; Goldberg & Lingle, 1985; Oliver, Smith & Holland, 1987; Starkweather, McDaniel, Mathias, Whitley & Whitley, 1991). Given their variety and complexity, these techniques are not discussed in detail here.

A New, Combinatorial Representation

Using permutational representations in GAs leads to the need for complex operators to ensure that the GA generates valid solutions. Despite their complexity, these techniques have met with mixed results. This section introduces a combinatorial representation scheme for problems of sequence in GAs.

In the suggested combinatorial representation, a binary string is used to describe a path of all cities needing to be visited. This string is divided into n fields of length $log_2(n)$ bits each. Each field represents an integer rating for the associated city. The GA constructs a tour by visiting the cities in an order from lowest rating to highest rating. Ties are resolved at random.

As an example of this representation, consider the following bit string for 4 cities in a 16 city TSP:

0101|0111|1010|1000 ...

This gives city 1's rating = 5, city 2's rating = 7, city 3's rating = 10, and city 4's rating = 8. Given these ratings, the sub-sequence ordering city 1, city 2, city 4, and city3.

As another example, consider the bit string

0111|0111|1010|1000 ...

which give city 1's rating = 5, city 2's rating = 5, city 3's rating = 10, and city 4's rating = 8. In this case, two sub-sequences are possible, depending on a random decision between city 1 and city 2 for first place. In other words, the sub-sequence city 1, city 2, city 4, and city 3, and the sub-sequence city 2, city 1, city 4, and city 3 are equally likely.

Note that this representation does not give a one-to-one mapping between encoded bit strings and actual TSP tours. For most search techniques, this would hopelessly complicate the search process. However, GAs search by selecting and recombining building blocks based on emergent average fitness values. When the underlying mapping between encodings and solutions is not one-to-one, the GA will simply average the effects on building blocks, and select accordingly. It is well known that the combinatorial GA can handle a great deal of noise in its evaluation function, and still perform well. The GA sees noise in the decoding process (i.e., the random breaking of rating ties) simply as evaluation noise. Therefore, if the GA can tie the space of ratings in this representation to meaningful building blocks, it should be capable of constructing good solutions.

One need not use a purely random strategy for resolving ratings ties in this representation. For instance, in the TSP, one can resolve ties by simply picking the option with the shortest local path. In problems other than the TSP, it may be necessary to use more sophisticated methods to resolve ties and ensure a valid permutation solution.

The suggested combinatorial representation can be easily extended to other problems of sequence. For instance, consider a simple job shop scheduling problem. N jobs must be assigned to M machines. Any given job must spend a certain amount of time on each machine, with possible constraints on the order in which the machines are visited. One could construct a combinatorial representation by assigning each job a ranking for each machine. Then, a separate schedule builder could construct the complete job shop schedule by ordering the jobs by rank, breaking ties either by random or by some knowledge-augmented method, and ensuring that no job is assigned to a machine before it is ready. In other words, each machine monitors for a set of jobs that are ready for its processing, and selects from this set based on rank.

First Experiments

In a preliminary investigation, TSP experiments have been conducted with combinatorial representation presented above. Both random and knowledge augmented tie-breaking methods have been explored. Results from two separate problems are presented here. The first problem uses the coordinates of the 48 capitals of the continental US. This problem was selected because a known optimum result exists. The x, y coordinates of the 48 capitals were provided by Gerhard Reinelt at the University of Augsburg and Bob Bixby at Rice University. The second problem uses 32 cities with randomly generated coordinates.

In the 48 capitals problem, the knowledge augmented technique performs best. This is an intuitive result, since one would think a "greedy" interpretation of the chromosome would yield most accurate fitness values. Results from the 48 capitals run are shown in Figure 1.

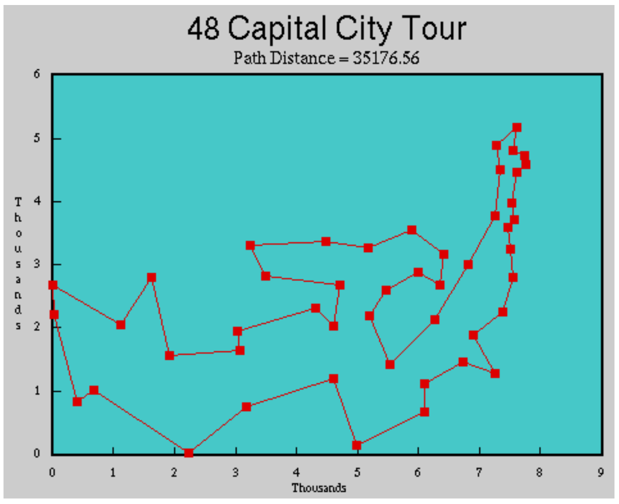

Figure 1: Best capital city tour found by the GA.

The GA run that generated these results used a population size of 1200, single point crossover with probability of 0.8, and single point mutation with a probability of 0.001667. The results are within five percent of the known optima. This solution is very similar to the known optima for this problem, which is given in Figure 2.

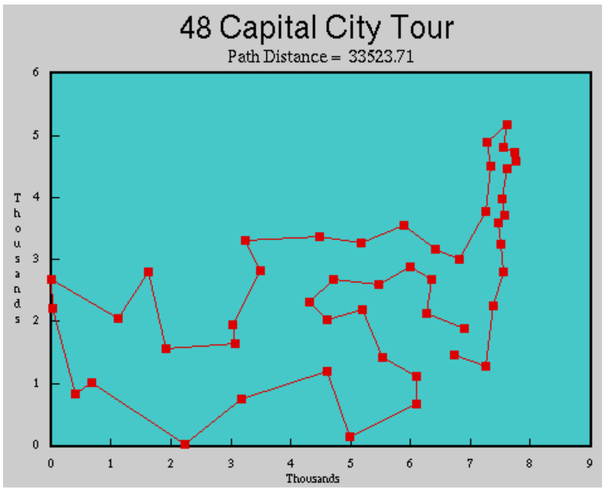

Figure 2: Optimal capital city tour.

Results on the 32 random cities showed better performance when random tie breaking was used. This counter intuitive result points out that being greedy may in fact bias the GA search, rather than aiding it. Best results on the 32 random cities run are shown in Figure 3.

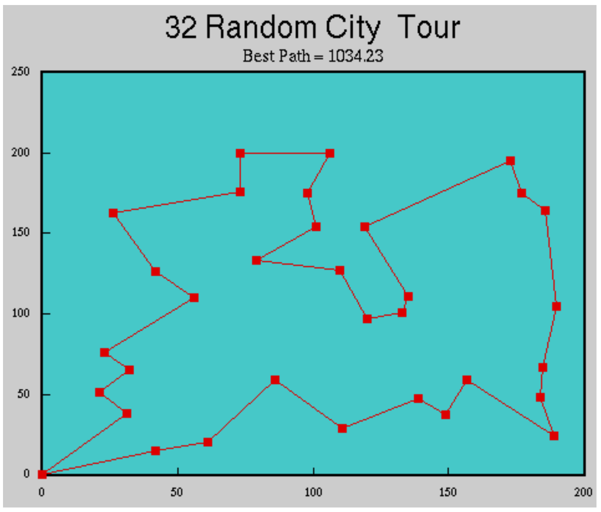

Figure 3: Best tour through 32 random cities found by the GA.

The GA that generated these results used a population size of 800, a crossover rate of 0.9, and a mutation rate of 0.00125.

Final Comments

The results presented here are certainly preliminary. They employed the simplest variety of combinatorial GA, and the simplest form of fitness function (tour length).

However, they do emphasize an important point. The GA approach can handle ambiguous representations, and generate near-optimal results. Sometimes an ambiguous (that is, not one-to-one) representation will allow for a more natural, and more easily encoded attempt at a GA solution. This allows standard, well-researched combinatorial GAs to be used in a broader class of problems, including permutation problems like the TSP. Trials comparing combinatorial and permutational GAs on more complex problems of sequence are an important area for future investigation.

References

Fox, B. R. and McMahon, M. B. (1991). Genetic Operators for Sequencing Problems. In G. Rawlins (Ed.), *Foundations of Genetic Algorithms* (pp. 284-300). San Mateo, CA: Morgan-Kaufmann.

Goldberg, D. E. and Lingle, R. (1985) Alleles, loci, and the traveling salesman problem. In Grefenstette, J. (Ed.) *Proceedings of an International Conference on Genetic Algorithms and Their Applications* (pp. 154-159).

Goldberg, D. E. (1989). *Genetic algorithms in search, optimization, and machine learning*. Reading, MA.: Addison-Wesley.

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

Michalewicz, Zbigniew (1992). *Genetic Algorithms + Data Structures = Evolution Programs*, Springer-Verlag.

Oliver, I. M. and Smith, D. J. and Holland, J. R. C. (1987). A study of permutation crossover operators on the traveling salesman. In Grefenstette, J. (Ed.) *Proceedings of the Second International Conference on Genetic Algorithms* (pp. 224-230). Hillsdale, NJ: Lawrence Erlbaum.

Starkweather, T., McDaniel, S., Mathias, K. Whitley, D. and Whitley, C. (1991). A Comparison of Genetic Sequencing Operators. In Belew, R. K. and Booker, L. B. (Eds.) *Proceedings of the Fourth International Conference on Genetic Algorithms* (pp. 69-76). San Mateo, CA: Morgan Kaufmann.