Breeding Perturbed City Coordinates and Fooling Travelling Salesman Heuristic Algorithms

R. Bradwell, L.P. Williams and C. L. Valenzuela
University of Teesside, Middlesbrough, TS1 3BA
Email: christine@tees.ac.uk

Abstract

Standard heuristic algorithms for the geometric travelling salesman problem (GTSP) frequently produce poor solutions in excess of 25% above the true optimum. In this paper we present some preliminary work that demonstrates the potential of genetic algorithms (GAs) to perturb city coordinates in such a way that the heuristic is 'fooled' into producing much better solutions to the GTSP. Initial results for our GA show that by using the nearest neighbour tour construction heuristic on perturbed coordinate sets it is possible to consistently obtain solutions to within a fraction of a percent of the optimum for problems of several hundred cities.

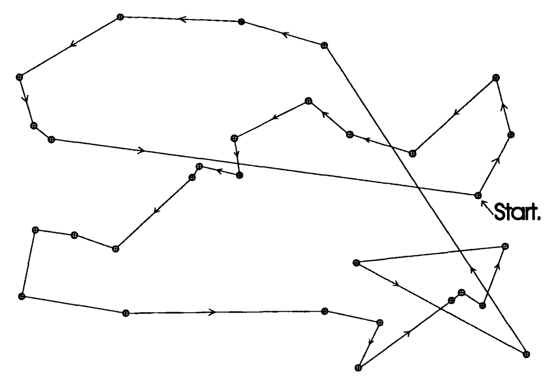

Figure 1: A nearest neighbour solution to a 20 city problem.

1 Introduction

Weaknesses in many of the standard GTSP heuristic algorithms are easy to spot by examining typical solutions produced by such techniques. Tour construction heuristic algorithms, for example, tend to start off well but are often observed to 'run out of steam' towards the end of the process.

The tour in Figure 1 was produced using a tour construction algorithm called the *nearest neighbour* heuristic algorithm (*NNHA*) [4]. This algorithm proceeds as follows: starting with an arbitrarily chosen starting city, choose for the next city the unvisited city closest to the current one. This new city is then labelled as the current one and the process repeated until all the cities have been chosen. The tour is then closed by returning to the initial city. 'Good parts' and 'bad parts' of the tour are clearly visible in Figure 1, illustrating the reduced effectiveness of the *nearest neighbour* (*NN*) heuristic when only a few cities remain to be incorporated in the tour.

The aim of this paper is to demonstrate the potential of a genetic algorithm (GA) to perturb the city coordinates in such a way that heuristics such as these are 'fooled' into producing better solutions to the GTSP than they otherwise would.

The original motivation for this work was to use perturbed city coordinates in conjunction with the cellular dissection algorithms of Richard Karp [8]. The run-time guarantees of Karp's algorithms (they run very fast and scale $O(n \log n)$) make them an attractive proposition for solving very large travelling salesman problems. The success of the *evolutionary divide and conquer* (*EDAC*) algorithm [11, 13, 14] in producing high quality solutions in $O(n \log n)$ time suggests that an approach that endeavours to combine divide-and-conquer with GAs has much potential. The effect we are hoping eventually to achieve with perturbed coordinates is to 'fool' a much simpler and faster version of Karp's algorithm than we used in *EDAC* into producing better solutions.

It would appear, however, that the potential of using a GA to breed perturbed city coordinates is not restricted to the algorithms of Richard Karp. Indeed the general approach could be applied to *any* heuristic algorithm for solving the TSP. The purpose of the present paper is to explore the po-

G. D. Smith et al., *Artificial Neural Nets and Genetic Algorithms*

tential of a GA based on perturbed coordinates on the simplest of heuristic algorithms before we extend the approach to compete with *EDAC*. We have chosen the *NNHA* as our starting point.

The basic idea is to perturb the city coordinates slightly, and use these perturbed coordinates to produce a tour using the chosen heuristic algorithm. The cities in the permutation list resulting from this tour are then moved back to their original position and a 'true tour' is produced.

The use of perturbed coordinate sets for solving the GTSP is not new. Codenotti, Manzini, Margara and Resta [2, 3] incorporated this technique into their version of Iterated Local Search, randomly perturbing the city coordinates of the TSP instance I by small amounts to give a new instance I' every time a locally optimal tour, T, is found on I. T will not normally be locally optimum with respect to I', so local optimization is then performed with respect to I' to give a new tour T'. T' then provides a new starting point for locally optimizing the GTSP with respect to I. In this way the perturbed coordinates provide a simple 'mutation' enabling the local search algorithm to escape local optima. About half of the time is spent applying the heuristic algorithm to the original coordinates and half to the perturbed coordinates. Our approach differs from that of Codenotti *et al* in three very important ways:

- In our study the TSP heuristic is applied *only* to perturbed coordinate sets.
- We use a genetic algorithm to breed perturbed coordinate sets.
- Our approach can be applied to tour construction heuristics as well as to local search heuristics.

The perturbed coordinate sets are the 'chromosomes' in our experiments, and the *NNHA* produces a different tour for each of the virtual coordinate sets in the population at any one time. These tours, which are easily represented by permutation lists, can then be evaluated with respect to the original city coordinates and actual intercity distances. The 'true' tour lengths form the basis for the fitness function of the GA.

All the TSP problems used in our study are uniform random points in a square region of the Euclidean plane.

In order to assess the quality of the solutions obtained by our genetic algorithm, we use a problem specific lower bound known as the Held-Karp lower bound [5, 6, 12]. This technique is known to produce very good estimates of optimal solutions for uniform random points, optimal tour lengths averaging less than 0.8% over the Held-Karp bound [9].

2 The Genetic Algorithm

A simple GA which appeared in [13] is used here. It is derived from the model of [7] and is an example of a 'steady state' GA (based on the classification of [10]). It uses the 'weaker parent replacement strategy' first described by [1]. The GA applies the genetic operators to the perturbed coordinate sets. The fitness values are based on tour lengths, as already explained.

The process for generating an initial population of perturbed coordinates required some thought. The most obvious way to generate a 'random' population at the start of a genetic algorithm is to randomise each x and y coordinate in such a way that all the original cities are effectively free to move anywhere within a region containing the GTSP problem. Intuitively, however, we favoured a scheme which perturbed the city coordinates within some small preset rectangular region surrounding each city, letting a suitable size for this rectangular region be determined by some early experimentation.

Figure 2 shows a snapshot of our GA running on a 20 city problem with the *NNHA*. The rectangular regions surrounding each city delimit the perturbation zones. The tour has been drawn through the cities located at their original positions, but the 'virtual positions' (which are randomly generated within each rectangle) are also visible in the diagram.

3 Perturbation Zone Scaling

How large should the rectangular zones in Figure 2 be? For random uniform points the size of the ideal perturbation zone is likely to be inversely proportional to the density of the cities. More formally if R represents the area of the (square) region, and n the size of the problem then,

$$l = k\sqrt{\frac{R}{n}} \tag{1}$$

relates the required dimension of the side of the perturbation zone, l, to city density through a factor, k, which can be estimated experimentally.

We carried out a set of experiments to establish a suitable value for the parameter k, the perturbation

Table 1: Results of GA and Random Search on 100 City Problem

Heuristic	Mean	Standard Deviation	95% Confidence Intervals
One Point GA	98.21	0.74	98.07, 98.34
Two Point GA	99.08	1.16	98.87, 99.29
Uniform GA	97.77	0.33	97.70, 97.83
Random Search	104.39	1.06	104.01, 104.77
NNHA	119.11	6.91	118.42, 119.80

factor of Equation (1). We ran our GA 30 times for each of a range of values for k on a single 100 city problem (random uniform points), and recorded the mean and standard deviation for each run. For all of our experiments we set the population size to 100, the mutation rate to 0.1% and used one-point crossover (see results section). Each run terminated after 300 generations. $k = 0.6$ gave the best result and was used for the remaining experiments.

4 Results

Uniform crossover was the obvious choice for our main genetic operator [10]. Since the predetermined sequence of the coordinate pairs in the 'chromosomes' is effectively generated at random (i.e. the sequence in which the city coordinates appear on the problem file) the use of either one-point or two-point crossover is unlikely to be any more effective at preserving subtours than uniform crossover. However, if one rearranges the lists of coordinates in an attempt to improve the propagation rate of subtours between parents and their offspring, it is feasible that the more traditional one-point and two-point crossovers will become more effective.

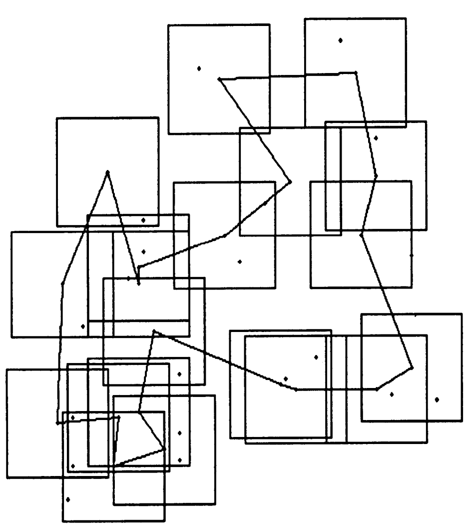

Figure 2: A good tour obtained on a 20 city problem using the GA on perturbed coordinates for the NN heuristic algorithm.

The results of our experiments are presented in Table 1. In order to assess the potential of one-point and two-point crossover, we incorporated a sorting algorithm into our program which, at regular intervals of time, rearranged the virtual coordinate sets in the current population together with the coordinates for the original problem into a sequence matching the current 'best tour so far'. This rearrangement was not expected to fulfil any useful purpose when the GA was using uniform crossover, so the sorting routine was left out in this case.

For each operator we ran 30 trials using the same 100 city problem. Each trial ran for 300 generations on a population of 100 members (a total of 30,000 evaluations plus the initial population). The perturbation parameter used was 0.6, with the mutation rate set to 0.1% for the GA. The random search was based on 30 trials each consisting of 30,100 randomly generated individuals with $k = 0.6$. The *NNHA* was executed 100 times, using each of the 100 cities in turn as the starting city for the algorithm.

Given that the *NNHA* produced a mean of 119.11 on the original coordinate data for the problem, all the other results presented represent a considerable improvement on the basic algorithm. In addition, the GA clearly outperforms the random search. From Table 1 it is possible to see that in our trials the uniform crossover was the most effective, producing on average a result that measures 1.01% above the Held-Karp lower bound of 96.37. The superior performance of uniform crossover seems counter intuitive since one would expect that operators such as two point crossover would preserve subtours more readily.

Initial experiments indicate that significant improvements over *NNHA* are maintained by using our GA for problems up to 500 cities, although runtime overheads for *NNHA* are becoming significant at this level.

Preliminary trials using a version of Karp's algorithm with our GA are encouraging. For a 500 city problem the GA consistently lifts the solution by about 20%.

5 Conclusion

In this paper we have clearly demonstrated that our technique for breeding perturbed coordinates for the TSP heuristics has potential. The method has succeeded in improving the quality of solution produced by the nearest neighbour heuristic algorithm by about 20% on a 100 city problem, to within a fraction of a percent of the optimum. Work is underway to incorporate the technique for breeding perturbed coordinates into Karp's cellular dissection algorithms.

References

[1] D. J. Cavicchio. *Adaptive search using simulated evolution.* Unpublished doctorial dissertation, University of Michigan, Ann Arbor

[2] B. Codenotti L. Margara, G. Manzini and G Resta. *Global strategies for augmenting the efficiency of TSP heuristics*, volume 709. Springer-Verlag, Berlin, 1993.

[3] B. Codenotti L. Margara, G. Manzini and G Resta. Perturbation: an efficient technique for the solution of very large instances of the euclidean TSP. *IFORMS Journal on Computing*, 8(2):125, Spring 1996.

[4] D. S.Johnson. Local optimization and the traveling salesman problem. In *Automata Languages and Programming: 17th International Colloquium Proceedings*, 1990.

[5] M. Held and R. M. Karp. The travelling salesman problem and minimum spanning trees. *Oper. Res*, 18:1138–1162.

[6] M. Held and R. M. Karp. The travelling salesman problem and minimum spanning trees: part ii. *Maths. Programming*, 1:6–25.

[7] J. H Holland. *Adaptation in natural and artificial systems.* The University of Michigan Press, Ann Arbor

[8] R. M. Karp. Probabilistic analysis of partitioning algorithm for the travelling- salesman problem in the plane. *Mathematics of Operations Research*, 2(3):209–224, August 1977.

[9] L. A. McGeoch D. S. Johnson and E. E. Rothberg. Asymptotic experimental analysis for the Held-Karp traveling salesman bound. In *Proceeding 1996 ACM-SIAM symp. on Discrete Algorithms*, 1996.

[10] G. Syswerda. Uniform crossover in genetic algorithms. In *Proceedings of the Third International Conference on Genetic Algorithms*, Hillsdale, NJ, 1996. Lawrence Erlbaum Associates.

[11] C. L. Valenzuela and A. J.Jones. A parallel implementation of evolutionary divide and conquer for the TSP. In *Proceedings of the First IEE/IEEE conference on Genetic ALgorithms in Engineering Systems: Innovations and Applications (GALESIA)*, pages 499–504, Sheffield, U.K., September 1995.

[12] C. L. Valenzuela and A. J. Jones. Estimating the Held-Karp lower bound for the geometric TSP. *European Journal of Operational Research*, to appear.

[13] C. L. Valenzuela and A. J.Jones. Evolutionary divide and conquer (I): a novel genetic approach to the TSP. *Evolutionary Computation*, 1(4):313–333, 1995.

[14] C. L. Valenzuela. *Evolutionary Divide and Conquer: a Novel Genetic approach to the TSP.* PhD thesis, University of London, 1995.