Hybridization of Evolutionary Techniques for the TSP with a Genotypic Termination Criterion

JESSICA A. CARBALLIDO, IGNACIO PONZONI, NÉLIDA B. BRIGNOLE
Grupo de Investigación y Desarrollo en Computación Científica (GIDeCC)
Departamento de Ciencias e Ingeniería de la Computación - Universidad Nacional del Sur
Av. Alem 1253 – 8000 - Bahía Blanca
ARGENTINA
Planta Piloto de Ingeniería Química (PLAPIQUI)
Universidad Nacional del Sur - CONICET
Complejo CRIBABB – Camino La Carrindanga km 7 – CC 717 – 8000 Bahía Blanca
ARGENTINA

Abstract: - In this paper we present new evolutionary algorithms to solve the Travelling Salesman Problem (TSP). The new hybrid algorithms combine classical representations of the TSP with two different crossover methods, namely Multiple Crossover Per Couple (MCPC) and Multiple Crossovers between Multiple Parents (MCMP). A genotypic termination condition was imposed in order to gain proper insight into some behavioural features. The performance analysis revealed that the MCPC algorithms with path representation lead to the best results.

Key-Words: - TSP, Evolutionary Algorithm, Termination Criterion, Hybridization Methods, Multiple Crossover.

1 Introduction

In this work we analyse several variants of Evolutionary Algorithms (EAs) for the TSP. Interest on this problem arose due to the great diversity of its application fields. In particular, our research line is focused on the use of graph theory for process-plant instrumentation design [1] [2].

Since various representations and genetic operators have been developed for the TSP, it became necessary to decide on the best algorithmic approach for our problem instance. In particular, a crucial aspect is the selection of the crossover method, which constitutes the main concern of this paper. In principle, the recombination technique may have significant influence on algorithmic performance for a given representation. In this article we have addressed two of the commonest chromosome structures for TSP, namely the ordinal and path representations. The former has already shown poor results [3], which could potentially be improved through adequate modifications in the crossover method. Besides, a new termination criterion was developed to detect possible premature convergence problems. In short, our end goal is to evaluate the impact of the application of different crossover methods for the TSP so as to analyse whether they outperform EAs with classic operators.

The paper is organized as follows: in Section 2 the TSP is introduced, while the EAs are explained in Section 3; the relationship between the problem and the algorithms is discussed next; in Section 5 our implementations are presented and finally, the experiments, main results and conclusions are put forward.

2 The Travelling Salesman Problem

The Travelling Salesman Problem (TSP) is a classic combinatorial optimisation problem. Given a set of cities and the cost of travelling between each pair of them, the aim is to find the minimum-cost itinerary, provided every city is visited only once. One possible way of representing the cost is by associating it to the distance between cities.

If there are n cities in the territory and any pair of cities is directly connected by a road, the size of the associated search space is $n!$, i.e. the number of permutations for those n cities. Any permutation leads to a feasible solution, and the optimum corresponds to the minimum-cost tour. In the Asymmetric TSP, the cost of going from city i to city j may differ from the cost of visiting city i after city j.

By way of illustration let us consider the graph shown in Fig.1, which corresponds to $n = 6$, where the nodes represent cities and the numbers on each edge contain the cost of traveling from one place to the other.

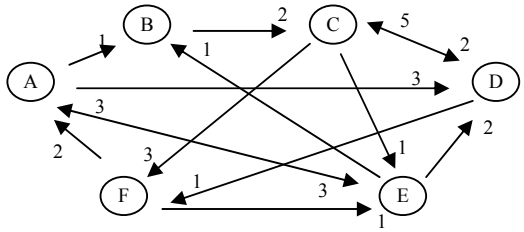

Fig.1: the TSP for n=6

Fig.2 shows the permutations that correspond to possible paths, i.e. feasible solutions.

Path 1: A B C E D F A

Path 2: A B C D F E A

Fig.2: Feasible solutions for the TSP with n = 6.

The costs of travelling along Paths 1 and 2 amount to 9 and 10 units, respectively. Therefore, Path 1 constitutes the optimum solution.

The TSP can be theoretically defined as follows: given n cities $[c_1, c_2, .., c_n]$ and an nxn distance matrix D, whose elements d_{ij} represent the distance between c_i and c_j, the objective is to find a tour, i.e. a permutation of those n cities, that minimizes the total length defined as the sum of the distances that constitute a closed tour.

This problem can be applied to a wide variety of practical situations, such as vehicle routing, task scheduling or connection between several kinds of devices. It also represents an important issue in Complexity Theory since it has been proved to be NP-Hard [4]. This implies that the required time for finding the exact solution increases at least exponentially with the size of the considered instance. That is why it becomes necessary to use heuristics that find nearly optimal solutions in short times.

3 Evolutionary Algorithms

Genetic Algorithms (GAs) are search algorithms based on the mechanics of the natural selection process (Darwin evolution). The most basic concept is that the strong individuals tend to adapt themselves and survive, while the weak ones tend to die out. In other words, optimization is based on evolution and the survival-of-the-fittest concept.

GAs have the ability to create an initial population of feasible solutions and then recombine them so that their search is guided towards the most promising areas of the state space. Each feasible solution is encoded as a chromosome (string) also called a genotype, and each chromosome is given a measure of fitness via a fitness (evaluation or objective) function. The fitness of a chromosome determines its ability to survive and produce offspring. A finite population of chromosomes is maintained.

GAs use probabilistic rules to make a population evolve from one generation to the next. The new generations are developed by means of the following three genetic recombination operators:

Selection: its function is to select the fittest individuals for reproduction.

Crossover: it combines parents' chromosomes to produce children's chromosomes. The operator makes use of the fittest chromosomes, thus transmitting the superior genes to the next generation.

Mutation: it alters some genes in a chromosome. This operator ensures that the entire state-space will be searched sooner or later and leads the population away from local minima.

The most important parameters in a GA are its population size, the evaluation function, the crossover method and its mutation rate. Determining the size of the population is a crucial factor. If an excessively small population size is chosen, the risk of converging prematurely to a local minimum increases significantly, because the population does not have enough genetic material to cover the whole search space adequately. In contrast, a sufficiently large population has greater chances of finding the global optimum at the expense of more CPU time.

When the canonical (binary) representation of the individual is changed, new crossover and mutation operators need to be defined. In these cases, GAs become EAs.

4 Evolutionary Algorithms for the TSP

It has been shown that the TSP problem is NP-Hard. Therefore, it is unreasonable to employ brute-force methods to solve it for a large number of cities. In the last few decades some alternative methods have been proposed in order to find solutions that tend to the optimum. In this respect, the EAs [3][5] are gaining ground as one of the most promising techniques.

There are various reasons why the TSP constitutes an important test problem for these heuristic methods [6]. First of all, its associated decision problem is considered representative in the class of NP-Hard problems. Besides, the problem has a long history and many heuristic approaches are a real challenge for the search of new methods. Big instances of the problem have been solved until optimality and are available either in the literature or through Internet.

4.1 Our proposal

In this work we present a comparison among EA variants based on two genetic representations (permutations and decoders) to solve the asymmetric

TSP. In both cases, instead of employing the traditional approach, we have applied the MCPC and MCMP crossover operators defined in Esquivel et al. [7] [8] because those operators have proved to exhibit very good performance for other applications such as the job shop scheduling problem [9].
Our hypothesis in this work was that the hybridization of simple GAs used on TSP with MCPC and MCMP would improve the behavior of the search process. In addition, we implemented a new genotypic convergence criterion so as to obtain information about possible premature convergence of the operators.

5 EA Representation of Individuals for the TSP

There are several different representations for the individuals in this problem. In this work we will focus on two of the most often used ones, namely decoders (ordinal representation) and permutations (path representation).

5.1 Decoders

The ordinal representation builds lists, known as decoders, which represent tours. For n cities, the ith element on each list is a natural number between 1 and n-i+1.
For instance, decoder d = (1 1 2 1 4 1 3 1 1) represents tour T = / 1 – 2 – 4 – 3 – 8 – 5 – 9 – 6 – 7 /.
The decoder should be interpreted as follows:

- It is assumed that R = / 1 – 2 – 3 – 4 – 5 – 6 – 7 – 8 – 9/ is the reference tour.
- The first element in d is 1. Therefore, the first city in R is taken as the first in the tour and it is eliminated from R, thus yielding the partial tour T = / 1 /.
- The next element in d is 1. So, the first city in R, which is now 2, is added to T and eliminated from R. Thus, T = / 1 – 2 /.
- The following element in d is 2. So, the second city in R (city 4) is added to T and eliminated from R. At this point, T = / 1 – 2 – 4 / and R = / 3 – 5 – 6 – 7 – 8 – 9/.
- These steps are repeated until all the elements in d have been visited and all the cities in R have been eliminated, thus yielding a complete tour T.

The main advantage of this representation is that it works well together with the classic one-point crossover function.

5.2 Permutations

The path representation is perhaps the most straightforward way of denoting a tour. For example, the tour T = / 5 – 1 – 7 – 8 – 9 – 4 – 6 – 2 – 3 / is simply indicated as (5 1 7 8 9 4 6 2 3).
Three basic crossover methods have been defined for this representation: PMX (Partially Mapped Crossover), OX (Order Crossover) and CX (Cycle Crossover). In this work, the implementations were carried out with OX [10] because it has been experimentally shown that its performance on the TSP is 11% and 15% better than PMX's and CX's, respectively [11].
The OX technique exploits the path-representation property that states that the main attribute is the order of the cities, not their locations. In accordance with this principle, the method builds the children by choosing a subsequence from a parent's tour, while preserving the relative order of the cities in the other parent's tour.
As an example, let us start from parents p_1 = (1 2 3 / 4 5 6 7 / 8 9) and p_2 = (4 5 2 / 1 8 7 6 / 9 3), where the slashes are split points. First, we obtain children h1 = (* * * 4 5 6 7 * *) and h_2 = (* * * 1 8 7 6 * *), preserving the sub tour ranging between the two respective split points for p_1 and p_2. Then, starting from the second split point, the cities in p_2 that are not present in h_1 are taken in order to complete the cities in h_1. In this case, the list of cities in p_2 starting from the second split point is: 9 3 4 5 2 1 8 7 6, but if the cities already in h_1 (i. e., 4 5 6 7) are eliminated, the tour becomes 9 3 2 1 8. These cities are added to h_1, starting from the second split point. When the end is reached, the remaining ones are added at the beginning of h_1.
In short, h_1 = (* * * 4 5 6 7 * *) => (* * * 4 5 6 7 9 *) => (* * * 4 5 6 7 9 3) => (2 * * 4 5 6 7 9 3) => (2 1 * 4 5 6 7 9 3) => h_1 = (2 1 8 4 5 6 7 9 3). Likewise, h_2 = (3 4 5 1 8 7 6 9 2) is obtained.
All the traditional crossover methods work in a similar way. Two parents are selected; then, the method is applied once and two children are thus generated. This approach has been called Single Crossover Per Couple (SCPC). New alternatives, such as Multiple Crossover Per Couple (MCPC) and Multiple Crossover between Multiple Parents (MCMP), arose later in the search for better results. In the next section we will explain details on our MCPC and MCMP implementations and we will later describe the termination condition that was used in all cases.

6 EA Variants: Convergence Criterion

6.1 MCPC

In MCPC the basic crossover method is applied several times to give birth to a variable amount of

children, always keeping the size of the new population within pre-established limits. A variant is the MCPC with Fitness Proportional Couple Selection (MCPC-FPCS), where the parents' pool is built with the help of a proportional selection criterion. A fitness value is assigned to each couple and the parents to be crossed over are chosen via proportional selection based on the couple's fitness. This index may be calculated as either the average value of the individual fitness values for each parent (AF) or the distance between those two values (DA).

6.2 MCMP

In MCMP a couple pool (*cp)* is generated based on population (*pop*), where each element is selected with a Fitness Proportional Selection (FPS) criterion. The size of *cp* amounts to one half of *pop*'s dimension. An average fitness is calculated for every element in cp and two couples are selected using FPS. Thus, four parents are obtained and crossed to get four children, as shown in Fig.3.

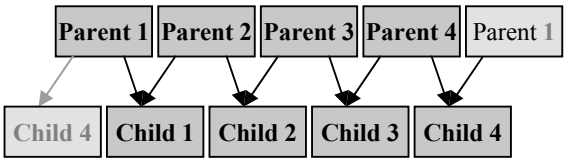

Fig.3: Four-Parent Combination

These children are not inserted in the new population in a direct way. Only two of them are selected instead, in accordance with their fitness. We will briefly explain below how the crossover is carried out, according to the type of representation.

Ordinal Representation – One Point Crossover

In this case, it is possible to graphically see how the children are built (see Fig.4). The one-point crossover builds each child by assigning to it the cities that are on one side of the cut point of one of the parents, together with the cities that are on the other side of another parent's. In the case of four parents, this idea is maintained, but the combination shown in Fig.3 is taken into account in order to choose the parents.

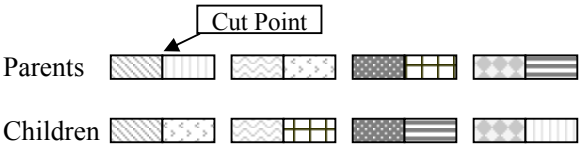

Fig.4: Four parent combination for one point crossover.

6.2.1 Path Representation - OX

OX builds every child by selecting a sub tour of one of the parents, while preserving the relative order of the other parent's cities. A sub tour is a succession of cities that fall into two cut points. In the four-parent case this idea remains, but the parents for each child are selected as shown in Fig.3.

6.3 Convergence Criterion

The termination condition proposed in this paper is based on the schema concept. A schema is a template that helps to establish similarities between chromosomes. In the binary-representation case, a schema is represented by a string of symbols in the set (0, 1, #), where # is a wildcard. For example, string 011001 is an instance of schema 01##0#. When the possible values for the alleles vary between 1 and N, the set of symbols is (1, 2, .. , N, #).

As stated in Radcliffe [12], when two parents are instances of the same schema, the child will also be an instance of that schema. In particular, when a schema carries good fitness to its instances, the whole population will tend to converge over the bits defined by that schema. As convergence is reached, all the offspring will be instances of the schema. Then, if the method that solves the problem is effectively implemented, the solution will also be an instance of that schema. Therefore, our convergence criterion analyses the individuals genotypes until a high percentage of them become instances of the same schema. This method can be explained through the following algorithm:

Input: population p of m individuals with n genes; α % of individuals that coincide in a β % of the genes.
Auxiliary information: ratio vector r of length n
Output: 1- convergence was reached,
0- convergence was not reached
1- Initialise *i* with 1.
2- Analyse the *i*-th allele of the whole population, i.e. from p[1, *i*] to p[m, *i*], getting the number r*i* of times that appears the value of allele that is repeated more times in p.
3- Increment *i*.
4- If *i* > n {all the genes were seen}
then Go to step 5.
else Go to step 2.
5- Count q*s*, the quantity of elements in r that are greater than α.
6- If *qs* represents a % greater than β
then Return 1(convergence criterion was satisfied).
else Return 0.

This control can be made either on each generation, or after a fixed number of generations. The algorithm can be applied to the canonical (binary) representation in a direct way but when the representation changes, some special considerations have to be taken into account. For the ordinal

representation the individual alleles are not used directly. A transformation needs to be carried out in order to convert the decoder into the corresponding path, and a reordering is required to keep city 1 as the initial point of the tour. The alleles of this "new genotype" are the ones analysed by the algorithm. As to the path representation, the procedure is equivalent to the second part of the method explained above. All these modifications are introduced so as to regard those individuals that represent the same tour as equal, while starting from different cities.

7 Experiments and Results

An auxiliary algorithm of Exhaustive Search (ES) was implemented to obtain reference optima. This algorithm uses the same fitness function as the EA, applying it to the whole search space.

The study cases were divided in two groups: sc_1, sc_2 and sc_3 constitute Group 1, while sc_4, sc_5 and sc_6 make up Group 2 (representing 8 and 10 city tours, respectively). The tours were represented by matrixes with random costs for the edges. The following parameters were employed:

Mutation probability: 0.2 %
Crossover probability: 0.8 %
Selection method: fitness proportional selection.
Popsize size: 15% of the total size of the search space.

The following performance variables were analysed:

- EBest: the difference between the optimum and the best value that could be obtained, divided by the optimum.
- GBest: the generation where the best value was obtained.
- GOpt: the generation where the optimum was obtained.
- GConv: the generation where convergence was achieved.
- Hit Ratio: the percentage of times the optimum was reached.

The reported values correspond to the average of 100 algorithmic runs for each group. The main results are shown in Tables 1-4, where:

- CEA refers to the results of the classic evolutionary algorithm (with SCPC).
- MCPC refers to the results of the MCPC variant.
- MCMP refers to the results of the MCMP variant.

The Hit-Ratio and EBest values show that in all cases the MCPC outperforms the other two crossover operators. MCMP exhibits poor performance and it is clear from the Gconv values that this operator presents premature convergence problems. By comparing Tables 1 and 2 with Tables 3 and 4, it can be deduced that the path representation still works better that the ordinal one, even with the performance improvement of the last one with MCPC.

Table 1: Average results for 8 city tours with ordinal representation

	CEA	MCPC	MCMP
Hit Ratio	50%	65%	45%
GBest	71.28	8.94	9.02
GOpt	88.71	8.89	9.22
GConv	171.1	33.1	30.2
EBest	0.0008473	0.000583	0.13417

Table 2: Average results for 10 city tours with ordinal representation

	CEA	MCPC	MCMP
Hit Ratio	65%	73%	46%
GBest	91.85	11.72	14.3
GOpt	94	11.9	8.49
GConv	222.95	36.25	12.595
EBest	0.000543	0.000351	0.00598

Table 3: Average results for 8 city tours with path representation

	CEA	MCPC	MCMP
Hit Ratio	97%	100%	37%
GBest	147.71	31.17	6.06
GOpt	150.29	31.17	6.39
GConv	550.95	-	21.9
EBest	0.0000225	0	0.0008105

Table 4: Average results for 10 city tours with path representation

	CEA	MCPC	MCMP
Hit Ratio	56%	99%	29%
GBest	151.76	144.7	13.54
GOpt	153	147.7	17.96
GConv	726.07	-	51.1
EBest	0.00031	0.00001	0.00089

A null (-) value in GConv means that the algorithm did not stop due to convergence control, but because the preset limit of generations (1000) had been reached. This indicates excellent levels of population diversity in this method.

8 Conclusions

In this work we modified the traditional TSP GAs , with either ordinal or path representations, by

changing the crossover methods and applying a new convergence criterion. Six variants were implemented and analysed. In our experiments, path representation proved to be better because its crossover operator always preserves the relative order of the cities, instead of keeping a random portion of the parent. The best performance was achieved with the MCPC path representation version, which showed excellent Hit Ratio figures together with high convergence-generation values. This means that, in our study cases, the approach regularly finds the solution while maintaining population diversity. Since MCPC exhibited the best results with both representations, this approach is eligible for its future employment in process-plant instrumentation-design applications.

References:

[1] Ponzoni I., Sánchez M.C., Brignole N.B, A New Structural Algorithm for Observability Classification, *Industrial & Engineering Chemistry Research*, Vol. 38, No. 8, 1999, pp. 3027-3035.

[2] Ponzoni I., Sánchez M.C., Brignole N.B., CDHG: a New Partitioning Algorithm based on the Detection of Cycles in Hypergraphs, *Latin American Applied Research*, Vol. 28, Nº1/2, 1998, pp.31-36.

[3] Greffenstette J.J., Gopal R., Rosmaita B. and Van Gucht D., *Genetic Algorithm for the TSP*, Laurence Erlbaum Associates, Hillsdale, NJ, 1985.

[4] Garey M. and Jonson D., *Computers and Interactability*, W. H. Freeman, San Francisco, 1979.

[5] Jog P., Suh J.Y. y Gucht, D.V., The Effects of Population Size, Heuristic Crossover, and Local Improvement on a Genetic Algorithm for the Travelling Salesman Problem, *Proceedings of the Third International Conference on Genetic Algorithms*, San Mateo, CA, 1989.

[6] Moscato P. and Tinetti F., *Blending Heuristics with a Population Based Approach: A Memetic Algorithm for the Travelling Salesman Problem*, 1992.

[7] Esquivel S., Leiva A., Gallard R., Multiple Crossover per Couple in genetic algorithms, *Proceedings of the Fourth IEEE International Conference on Evolutionary Computation (ICEC' 97)*, Indianapolis, USA, April 1997, pp. 103-106.

[8] Esquivel S., Leiva A., Gallard R., Multiple Crossovers between Multiple Parents to improve search in evolutionary algorithms, Presentation in the *1999 Congress on Evolutionary Computation IEEE*, Washington DC, 2001 (in press).

[9] Esquivel S., Ferrero S., Gallard R., Salto C., Alfonso H., Schutz M., Enhanced Evolutionary ALgorithms for single and multiobjective optimization in the job shop scheduling problem, *Knowledge-Based Systems*, Vol. 15, 2002, pp. 13-25.

[10] Davis L., Applying Adaptive Algorithms to Epistatic Domains, *Proceedings of the International Joint Conference on Artificial Intelligence*, 1985, pp. 162-164.

[11] Oliver, I. M., Smith, D. J., and Holland, J.R.C., A Study of Permutation Crossover Operators on the Traveling Salesman Problem, *Proceedings of the Second International Conference on Genetic Algorithms*, 1987, pp.224-230.

[12] Radcliffe N. J., Equivalence Class Analysis of Genetic Algorithms, *Complex Systems*, Vol. 5, 1991, pp. 183- 205.