

MCPC and MCMP Evolutionary Algorithms for the TSP

Jessica A. Carballido, Ignacio Ponzoni, Nélida B. Brignole

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
jcarballido@plapiqui.edu.ar ; ip@cs.uns.edu.ar ; dybrigno@criba.edu.ar

Abstract. The Travelling Salesman Problem (TSP) is an NP-complete problem that has to be solved with optimisation tools, such as the evolutionary algorithms (EA). There are various ways of representing the TSP tours through EAs. Two of the most often employed genetic representations are the ordinal and path representations. In this work we present several EAs based on those representations combined with different crossover operators, namely Single Crossover Per Couple (SCPC), Multiple Crossover Per Couple (MCPC) and Multiple Crossover with Multiple Parents (MCMP). The implementations were tested for several academic case studies generated at random. The EAs with MCPC exhibited the best performance, while the MCMP variants yielded premature convergence problems.

1 Introduction

In the Travelling Salesman Problem (TSP), a salesman has to visit all the cities in his territory only once and come back to the starting point. Given the costs associated to each stage of the journey, the problem consists in planning the best itinerary so that the total expenditure is minimised.

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 the number of permutations for the n cities, which is calculated as $n!$. Each permutation represents a feasible tour, and the optimum corresponds to the minimum-cost tour. In particular, the asymmetric TSP is a special case where the cost of going from a city i to a city j may be different from the cost of visiting a city i starting at city j .

By way of illustration let us consider the graph shown in Figure 1, which corresponds to $n=6$, where the nodes represent cities and the numbers on each edge contain the cost of travelling from one place to another. This example contains only two possible tours, which are shown in Figure 2. The costs of travelling along Paths 1

and 2 amount to 9 and 10 units, respectively. Therefore, Path 1 constitutes the optimum solution.

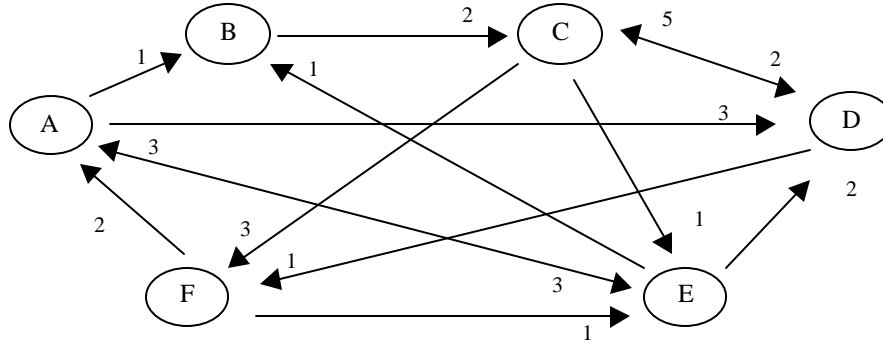


Fig. 1. The graph for an n -city tour

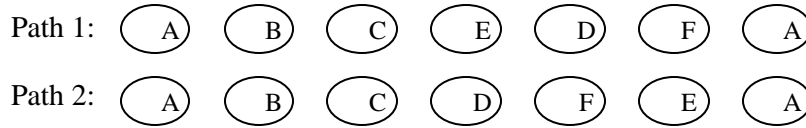


Fig. 2. The feasible solutions.

It has been shown that the TSP problem is NP-Complete [1]. Therefore, it is not reasonable 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 to find solutions that tend to the optimum. Evolutionary Algorithms (EAs) [2, 3] are among the most promising ones. EAs try to evolve towards the optimum by keeping a population of potential solutions that is subjected to unary and binary transformations –called mutations and crossovers, respectively- under a selection scheme based on individual fitness.

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 apply the MCPC and MCMP crossover operators defined in Esquivel *et al.* [4, 5] because those operators have proved to exhibit good performance for problems with other genetic representations. Our end goal is to evaluate the impact of their application for the TSP so as to analyse whether they outperform EAs with the classic operators.

2 Genetic Representations

2.1 Decoders

The ordinal representation builds lists called decoders that represent tours. For n cities, the i^{th} element on each list is a natural number between 1 and $n-i+1$.

For instance, decoder $\mathbf{d} = (1 \ 1 \ 2 \ 1 \ 4 \ 1 \ 3 \ 1 \ 1)$ represents tour $\mathbf{T} = / 1 - 2 - 4 - 3 - 8 - 5 - 9 - 6 - 7 /$. The decoder should be interpreted as follows:

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

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

2.2 Permutations

The path representation is perhaps the most straightforward way of denoting a tour. For example, the tour $\mathbf{T} = / 5 - 1 - 7 - 8 - 9 - 4 - 6 - 2 - 3 /$ 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 [6] because it has been experimentally shown [7] that its performance on the TSP is 11% and 15% better than PMX's and CX's, respectively.

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 $\mathbf{p1} = (1 \ 2 \ 3 / 4 \ 5 \ 6 \ 7 / 8 \ 9)$ and $\mathbf{p2} = (4 \ 5 \ 2 / 1 \ 8 \ 7 \ 6 / 9 \ 3)$, where the slashes are split points. First, we obtain children $\mathbf{h1} = (* \ * \ * \ 4 \ 5 \ 6 \ 7 \ * \ *)$ and $\mathbf{h2} = (* \ * \ * \ 1 \ 8 \ 7 \ 6 \ * \ *)$, preserving the subtour ranging between the two respective split points for $\mathbf{p1}$ and $\mathbf{p2}$. Then, starting from the second split point, the cities in $\mathbf{p2}$ that are not present in $\mathbf{h1}$ are taken in order to complete the cities in $\mathbf{h1}$. In this case, the list of cities in $\mathbf{p2}$ starting from the second split point is: 9 3 4 5 2 1 8 7 6, but if the cities already in $\mathbf{h1}$ (i. e., 4 5 6 7) are eliminated, the tour becomes 9 3 2 1 8. These cities are added to $\mathbf{h1}$, starting from the second split point. When the end is reached, the remaining ones are added at the beginning of $\mathbf{h1}$.

In short, $\mathbf{h1} = (* * * 4 5 6 7 * *) \Rightarrow (* * * 4 5 6 7 9 *) \Rightarrow (* * * 4 5 6 7 9 3) \Rightarrow (2 * * 4 5 6 7 9 3) \Rightarrow (2 1 * 4 5 6 7 9 3) \Rightarrow \mathbf{h1} = (2 1 8 4 5 6 7 9 3)$. Likewise, $\mathbf{h2} = (3 4 5 1 8 7 6 9 2)$ is obtained.

3 Multiple Crossover Per Couple and Multiple Crossover between Multiple Parents

In general terms, the simplest crossover operation consists in choosing two parents and then applying the crossover method once to yield two children. This basic approach is known as Single Crossover Per Couple (SCPC) [4]. Other alternatives, such as Multiple Crossover Per Couple (MCPC) and Multiple Crossover between Multiple Parents (MCMP) [4, 5], were also explored in order to improve the results.

For 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).

In MCMP more than two parents are chosen and the crossover operation is carried out by generating a variable number of children, and the population size is controlled in the same way as for MCPC.

4 Computational Experiments

Evolutionary algorithms with the combined strategies reported in Table 1 were implemented in this work in order to solve the TSP. In all cases, the same population (104 individuals), crossover (0.7) and mutation (0.25) probabilities were employed.

REPRESENTATION	CROSSOVER OPERATOR	CROSSOVER METHOD
Decoders	1-SCPC	Classic one-point
	2-MCPC-FPCS with AF	Classic one-point
	3-MCMP with AF	Classic one-point
Permutations	1-SCPC	Order Crossover
	2-MCPC-FPCS with AF	Order Crossover
	3-MCMP with AF	Order Crossover

Table 1. Evolutionary Algorithm Variants

A brute-force algorithm (BF) was also implemented so as to calculate the optima for small problem instances. The exact solutions thus obtained served as reference values to assess algorithmic quality and performance.

The MCPC-FPCS operator was implemented as follows. From population **p1**, a pool of couples **pp**, whose members were chosen via FPS, was generated. This population has half the amount of individuals in **p1**. Each couple had either 2 or 4 children. The following procedure was employed to obtain 4 children:


Decoders Two children are generated with one split point and another split point is chosen at random in order to yield the other two children.

Permutations The split points, which amount to 2 in OX, also vary.

The procedures for MCMP and MCPC are similar, but when reaching the second selection stage (FPCS), two elements from **pp** are chosen for MCMP, thus yielding four parents to be crossed over. Then, the number of children is either 4 or 8.

The four parents are crossed over as follows:

Decoders

p1 = /aaaabbbb/		h1 = /aaaadddd/
p2 = /ccccdddd/		h2 = /ccccffff/
p3 = /eeeeffff/		h3 = /eeeehhhh/
p4 = /ggggghhhh/		h4 = /ggggbbbb/

Permutations In this case, **p1**'s subtour, i.e. the cities between two split points, is taken for **h1**, as well as the cities in **p2** that do not belong to the subtour. Likewise, **p2**'s subtour is included in **h2**, together with the cities in **p3** that do not belong to the subtour. Similarly, for **h3** and **h4**, **p3**'s and **p4**'s subtours are taken together with the cities in **p4** and **p1** that do not belong to the subtours in **p3** and **p4**, respectively.

In both representations, the way of generating more than four children is by choosing new split points.

The case studies can be classified in two groups of three graphs each. The first one (*g1*, *g2* and *g3*) denote tours along 8 cities, while the other (*g4*, *g5* and *g6*) includes elements that indicate tours along 10 cities. The graphs are represented by matrices and random values were assigned for the travelling expenses associated to each edge. A hundred generations were considered for the experiments and population size was always varied between 50 and 100 individuals.

In all cases, the optima could be obtained by means of the BF algorithm. The following variables were measured:

EBEST: the difference between the known or estimated optimum value and the best calculated value, divided by the optimum value.

GBEST: the generation that yielded the best calculated value, disregarding whether it was the optimum value.

HIT RATIO: the number of times the optimum was reached.

The average values obtained for these quantities after 50 runs of each algorithm are reported in Tables 2-7.

	DECODERS			PERMUTATIONS		
	1	2	3	1	2	3
EBEST	0.001083	0.001083	0.384615	0.000333	0.0	0.001583
GBEST	27.6	20.5	18.5	33	33.9	12.7
HIT RATIO	40%	40%	40%	80%	100%	30%

Table 2. *g1* runs

	DECODERS			PERMUTATIONS		
	1	2	3	1	2	3
EBEST	0.000368	0.000211	0.007895	0.000211	0	0.000474
GBEST	34	22.9	18	25.79	25.79	10.6
HIT RATIO	60%	90%	40%	60%	100%	10%

Table 3. *g2* runs

	DECODERS			PERMUTATIONS		
	1	2	3	1	2	3
EBEST	0.001091	0.000455	0.01	0.001455	0.000182	0.01
GBEST	23.875	29	14.2	33	29.3	22.2
HIT RATIO	60%	80%	50%	20%	90%	40%

Table 4. *g3* runs

	DECODERS			PERMUTATIONS		
	1	2	3	1	2	3
EBEST	0.000292	0.000292	0.004583	0.0005	0.000292	0.000375
GBEST	52.4	17	9.4	49.9	25.5	15.2
HIT RATIO	70%	70%	50%	40%	50%	50%

Table 5. *g4* runs

	DECODERS			PERMUTATIONS		
	1	2	3	1	2	3
EBEST	0.001071	0.000429	0.01	0.000786	0.000429	0.001143
GBEST	30.4	21.5	44.8	37.2	21.5	11.6
HIT RATIO	30%	60%	20%	40%	50%	20%

Table 6. *g5* runs

	DECODERS			PERMUTATIONS		
	1	2	3	1	2	3
EBEST	0.000267	0.000333	0.003333	0.000133	0	0.000333
GBEST	35	26	14.2	38.4	41.6	9.45
HIT RATIO	80%	80%	70%	90%	100%	70%

Table 7. *g6* runs

The **EBEST** and **HIT RATIO** values show that in most cases the best result was obtained with the MCPC-FPCS permutations. The second best combination as regards

performance was achieved with the MCPC-FPCS decoders. As to convergence speed, which can be quantified through **GBEST**, it is interesting to note that the implementations with MCMP yielded the best individual in the lowest amount of generations. Nevertheless, this must be a premature convergence problem because in most cases the **HIT RATIOS** reveal that these algorithms failed to converge to the optimum solution. The problem exposed in the MCMP variant is due to the strong selective pressure imposed by this operator.

5 Conclusions and future work

The aim of this research was to evaluate the performance of the MCPC and MCMP operators in the context of different representations for the TSP. Six implementations of EAs designed to solve the classic asymmetric TSP were discussed in this paper. The variants differ in the representation chosen for the individuals as well as in the crossover operator. Two alternative representations, namely permutations and decoders, were considered. Within each of them, three ways of carrying out the crossover operations were tried out: SCPC, MCMP and MCPC with AF. For the EAs based on decoders, the classic one-point crossover method was employed, while for those based on permutations, OX was used.

The algorithmic performance was assessed by analysing graphs for 8 and 10 cities. The best results were achieved with the EA with permutations and MCPC, followed by the implementation with MCPC decoders. In contrast, the MCMP variants exhibited very low hit ratios. Nevertheless, this approach could also be applicable with more lenient selection criteria, provided the low hit ratios were due to premature convergence problems. Therefore, our future work will be focused on finding an adequate termination criterion in order to have a sound basis to confirm or reject the premature convergence hypothesis.

6 Acknowledgements

The authors gratefully acknowledge Dr. Raúl Gallard for his helpful comments and suggestions.

References

1. Garey, M. and Jonson, D., *Computers and Interactability*, W. H. Freeman, San Francisco, 1979.
2. Greffenstette, J. J., Gopal, R., Rosmaita, B. and Van Gucht, D., *Genetic Algorithm for the TSP*, Laurence Erlbaum Associates, Hillsdale, NJ, 1985.
3. Jog, P., Suh J. Y. and Gucht, D. V., *The Effects of Population Size, Heuristic Crossover and Local Improvement on a Genetic Algorithm for the Traveling Salesman Problem*, Proceedings of the Third International Conference on Genetic Algorithms, Morgan Kaufmann Publishers, San Mateo, CA, 1989.

4. Esquivel, S. C., Leiva A. and Gallard R. H., *Multiple Crossovers Per Couple in Genetic Algorithms*, Research Interest Group in Computer Systems, UNSL, Argentina.
5. Esquivel, S. C., Leiva A. and Gallard R. H., *Multiple Crossovers between Multiple Parents to improve Search in Evolutionary Algorithms*, Project UNSL-338403, UNSL, Argentina.
6. Davis L., *Applying Adaptive Algorithms to Epistatic Domains*, Proceedings of the International Joint Conference on Artificial Intelligence, 162-164, 1985.
7. Oliver, I. M., Smith, D. J. and Holland J. R. C., *A Study of Permutation Crossover Operators on the Travelling Salesman Problem*, Laurence Erlbaum Associates, Hillsdale, NJ, 1987.