International Journal of Industrial Engineering Computations 7 (2016) 245–256

Contents lists available at GrowingScience

International Journal of Industrial Engineering Computations

homepage: www.GrowingScience.com/ijiec

A population-based algorithm for the multi travelling salesman problem

Rubén Iván Bolaños[a], Eliana M. Toro O[b] and Mauricio Granada E[a*]

[a]*Department of Electrical Engineering, Universidad Tecnológica de Pereira, Colombia*
[b]*Department of Industrial Engineering, Universidad Tecnológica de Pereira, Colombia*

C H R O N I C L E

Article history:
Received July 22 2015
Received in Revised Format Septmber17 2015
Accepted October 14 2015
Available online October 15 2015

Keywords:
Combinatorial optimization
Multi-traveling salesman problem
Population-based algorithm
Local search operators

A B S T R A C T

This paper presents the implementation of an efficient modified genetic algorithm for solving the multi-traveling salesman problem (mTSP). The main characteristics of the method are the construction of an initial population of high quality and the implementation of several local search operators which are important in the efficient and effective exploration of promising regions of the solution space. Due to the combinatorial complexity of mTSP, the proposed methodology is especially applicable for real-world problems. The proposed algorithm was tested on a set of six benchmark instances, which have from 76 and 1002 cities to be visited. In all cases, the best known solution was improved. The results are also compared with other existing solutions procedure in the literature.

1. Introduction

The mTSP problem can be viewed from the perspective of two well-known problems: i) as a generalization of the Travelling Salesman Problem (TSP), where a set of routes is assigned to m salesmen who all start from and return to a home city, and ii) as a special case of the vehicle routing problem (VRP), in which customers are considered unitary demands and every travelling salesman only visits a predetermined number of cities. Thus, the mTSP can also be utilized for solving several types of VRPs and all formulations and solution approaches for the VRP are valid for the mTSP.

Although the VRP and TSP have been widely discussed in the literature, the research on the mTSP is limited. Moreover, few papers in the literature address the mTSP through efficient population-based algorithms. The main motivation for formulating a population methodology lies in the ease of integration with multi-objective strategies, which allow introducing practical aspects, such as profit, fuel consumption and environmental impact, among others. Therefore, it becomes relevant to develop and implement an effective and robust optimization methodology based on population.

* Corresponding author.
E-mail: magra@utp.edu.co (M. Granada E)

doi: 10.5267/j.ijiec.2015.10.005

The main contributions of this paper are as follows: 1) An effective population algorithm is proposed; 2) different heuristic strategies that improve the quality of the initial population are presented; and 3) six local search operators are integrated into the methodology, with which a modified and improved version of the Genetic Algorithm presented by Chu and Beasley (1997) is obtained.
The obtained results demonstrate the efficiency of the improved method with respect to those reported in the literature.

The rest of the paper will proceed as follows. In Section 2, the literature of the mTSP is reviewed. In Section 3, the problem is defined, and the mathematical model is formulated. Section 4 describes the proposed algorithm. Results are reported in Section 5, and conclusions and perspectives are discussed in Section 6.

2. Literature review

While in specialized literature the work related to TSP and VRP are abundant and numerous instances and test systems are presented, relatively few studies are found on mTSP with which comparisons can be made regarding the best known solutions. In Bektas (2004), a complete review of the state of the art solutions that includes aspects of the mathematical formulation and solution procedures is presented. The author makes an interesting analysis of applications and connections with other problems, where the mTSP is associated with many applications such as, scheduling of print press (Gorenstein, 1970; Carter & Ragsdale, 2002), bank crew (Svestka & Huckfelt, 1973), interview (Gilbert & Hofstra, 1992), photographer team (Zhang et al., 1999), security service (Calvo & Cordone, 2003) and hot rolling (Tang et al., 2000), school bus routing (Angel et al., 1972), work force planning, workload balancing (Okonjo, 1988), mission planning for mobile robots can be found in Brumitt and Stentz (1998); Yu et al. (2002); Sariel et al. (2009) and inspection task-rescue scenarios, in which possible victims have to be identified (Faigl et al., 2005).

Likewise, several methodologies have been raised to solve the mTSP, such as heuristic and metaheuristic algorithms, neural network-based methods, ant systems and exact techniques. The mTSP is very time consuming due to its NP-hard nature. Therefore, heuristic algorithms are the preferred method. In this regard, Russell (1977) proposes a technique denominated as mTOUR heuristic that consists of two stages: cluster first-route second. In that work, the extended version of the Lin Kernighan algorithm (1973) is used for routing. Sze and Tiong (2007) perform a comparison between the Nearest Neighbor Algorithm (NNA) and Genetic Algorithm (GA) for solving the mTSP. The results obtained by the NNA are superior to the GA in terms of performance and computing time. However, a conventional GA was used in that case.

In a more recent work, Sedighpour et al. (2011) present an effective GA for solving the mTSP, in which the 2-Opt local search algorithm is used for improving solutions. In the aforementioned work, 6 benchmark instances from the TSPLIB (Reinelt, 2014) are used and, in all but four instances, the best known solution was improved.

Zhou and Li (2010) solve the mTSP problem by a modified GA. A greedy strategy is implemented to create the initial population, and the mutation operator is combined with the local search strategy 2−Opt, which allows one to quickly determine quality neighboring solutions and accelerates the convergence of the algorithm.

Chen et al. (2011) propose a genetic algorithm in which new mutation and recombination operators are suggested within a codification technique named two-part chromosome, representing the solution to the mTSP. This type of encoding takes into account the sequence and number of cities that must be visited by each traveling salesman.

The work presented by Junjie and Dingwei (2006) applies an Ant Colony Optimization (ACO) to the mTSP with capacity constraints. The ACO tests several standard problems from TSPLIB, finding competitive solutions in reasonable computation times. However, for large instances, the algorithm does not reach the best known solution. Similarly, in Seidighpour et al. (2011), the results obtained by ACO are taken for comparison with those obtained in this paper.

A hybrid two-stage algorithm is presented by Yousefikhoshbakh and Seidighpour (2012); in the first stage, the mTSP is solved by the so-called sweep algorithm and in the second stage by so-called elite ant colony optimization in conjunction with a local search strategy 3-Opt to improve the solution found in the first stage. Six instances of the TSPLIB library are resolved showing the development and competitiveness of the algorithm.

Song et al. (2003) propose an extended simulated annealing, which is an open system, and this gives rise for perturbation schemes according to the problem-augmented TSP or mTSP. The entropy constraints are combined with an energy function to equally distribute the salesmen's workload. Further, that algorithm can solve the problem without any transformation into the standard form as proposed in Bellmore and Hong (1974) and GuoXing (1995).

Exact techniques have been explored to solve the problem. Initially, Laporte and Nobert (1980) solve mTSP without sub-tour elimination constraints. These restrictions are only included when an integer solution is found, whereby a labeling process is performed in order to verify if any subtour constraint has been violated. If a subtour is found in the solution, then the restrictions for each subtour are included in the problem, and a re-optimization is performed. The integrality is obtained using Gomory's cutting planes. The methodology is able to find optimal solutions for instances of size less than or equal to 100 cities.

3. Problem definition and mathematical model

The mTSP is NP-hard in the strong sense and is commonly formulated as an integer programming model that considers a complete directed graph $G(V,E)$ where E is the set of edges and $V=\{0\;...\;n\}$ is the vertex set and corresponds to the cities; whereas vertex 0 corresponds to the depot. Graph G must be strongly connected and is generally assumed to be complete. D Is the matrix formed with the nonnegative metric cost d_{ij} associated with each *edge (i, j)*$\in E$. The metric cost can represent cost, distance or time. x_{ij} Is defined as a binary variable associated with each *edge (i, j)*$\in E$ that takes the value 1 if *edge (i, j)* belongs to the optimal solution and takes the value 0 otherwise. In the mTSP, the cost matrix satisfies the triangle inequality, meaning $d_{ik}+d_{kj}\geq d_{ij};\quad \forall i,j,k\in V$. Therefore, given a set of cities where m salesmen and a metric cost are located, the objective of the mTSP is to determine a tour for each salesman such that:

- The total tour cost is minimized.
- All of the routes must start and end at the depot.
- Each city must be visited exactly once by only one salesman.

Starting from a model based on a two-index vehicle flow formulation, the mTSP model may be defined as:

$$\min \sum_{i\in V}\sum_{j\in V} d_{ij}x_{ij} \tag{1}$$

s.t

$$\sum_{i\in V} x_{ij} = 1, \quad \forall j \in V\setminus\{0\} \tag{2}$$

$$\sum_{j\in V} x_{ij} = 1, \quad \forall i \in V \setminus \{0\} \tag{3}$$

$$\sum_{i\in V} x_{i0} = m \tag{4}$$

$$\sum_{j\in V} x_{0j} = m \tag{5}$$

$$\sum_{i\notin S}\sum_{j\in S} x_{ij} \geq r(S) \quad \forall S \subseteq V \setminus \{0\}, S \neq \varnothing \tag{6}$$

$$x_{ij} \in \{0,1\}; \quad \forall i, j \in E \tag{7}$$

The indegree and outdegree Constraints (2) and (3) impose that exactly one edge enters and leaves the vertex associated with each city, respectively. Constraints (4) and (5) ensure that exactly *m* salesmen leave from and return to the depot. The so-called capacity-cut constraints (6) impose both the connectivity of the solution and the maximum number of nodes that can be visited by any salesman. In fact, they indicate that each cut $(V \setminus S, S)$ defined by a customer set S is crossed by a number of edges not smaller than $r(S)$ (minimum number of salesmen needed to serve set S). Alternatively, a family of equivalent constraints may be obtained by considering the subtour elimination constraints proposed for the TSP by Miller et al. (1960).

$$u_i - u_j + p x_{ij} \leq p - 1 \qquad \forall i, j \in V \setminus \{0\}, i \neq j \tag{8}$$

where $u_i \in V \setminus 0$, is an additional continuous variable representing the number of cities that salesman i has not visited yet and *p* is the maximum number of nodes that can be visited by any salesman. The above mathematical model (2) - (7) can be implemented in commercial solvers to obtain exact solutions of the mTSP only for small-sized instances (approximately less than 100 cities). Given *n* is the number of cities to be visited and *m* the total number of salesmen, the total number of possible routes covering all cities is given by $m(n-1)!/2$. Due to its high computational complexity, solving even moderate-sized mTSPs takes unreasonable computational time. The approximate approach never guarantees an optimal solution but provides a near optimal solution in an acceptable computational time.

4. Methodology: Modified Chu-Beasley Genetic Algorithm (MCBGA)

In this section, a modified genetic algorithm based on the Chu and Beasley (1997) approaches is proposed. The first part of the MCBGA requires a completely diverse initial population where every chromosome is unique. A tournament selection is performed to choose two parent solutions from the initial population. The chosen two are used for a special crossover adapted to the mTSP. The resulting child solutions from the crossover operator are converted into individual routes. The objective function is evaluated for each chromosome, and the one with the best score is chosen to undergo a mutation process. The mutation process is different from the traditional mutation. This is an improvement step. This stage includes some inter and intra-route heuristics. Finally, a new individual enters into the population. The process is performed iteratively until a convergence point. Each of these steps is presented in detail in the following sub-sections:

4.1. Codification

Let n be the number of cities to be visited and m the number of traveling salesmen available to visit all cities. Let $C = \{c_1, c_2, ..., c_i, ..., c_m\}$ be a set of size m , where c_i is the number of cities contained in cluster TSP_i . Thus, TSP_i is a permuted set of cities that have to be connected through a feasible route for the traveling salesman i such that $TSP_i \subseteq V \setminus 0$. In general, the representation of a feasible solution Π to the mTSP is encoded by a chromosome that contains all cities to be visited (n-th order permutation), as shown in Expression (9).

$$\Pi(V) = \begin{cases} \underbrace{p_1, p_2, \dots, p_{c_1}}_{\text{TSP}_1}, \underbrace{p_{c_1+1}, \dots, p_{c_2}}_{\text{TSP}_2}, \dots, \\ \underbrace{p_{c_{i-1}+1}, \dots, p_k, p_{k+1}, \dots, p_{c_i}}_{TSP_i}, \dots, \\ \underbrace{p_{c_{j-1}+1}, \dots, p_l, p_{l+1}, \dots, p_{c_j}}_{TSP_j}, \dots, \\ \underbrace{p_{c_{n-1}+1}, \dots, p_{c_n}}_{\text{TSP}_m} \end{cases} \tag{9}$$

The number of cities in a cluster i, $c_i = |TSP_i|$, has a lower limit $c_{\min}$ and an upper limit $c_{\max}$, as shown in Eq. (10).

$$c_{\min} \le |TSP_i| \le c_{\max} \tag{10}$$

The limit $c_{\max}$ is an input parameter of the problem, while the limit $c_{\min}$, which represents the minimum number of cities that each salesman must visit, is calculated by the following equation:

$$c_{\min} = \left\lfloor \frac{n}{TSP_{min} + 1} \right\rfloor, \tag{11}$$

where $TSP_{\min}$ represent the minimum number of salesmen needed to cover all cities and is calculated as:

$$TSP_{\min} = \left\lceil \frac{n}{c_{\max}} \right\rceil \tag{12}$$

4.2. Initial population

The initial population is obtained from *N* approximated solutions of the traditional TSP considering the following: a) the complete set of cities except the deposit and b) a different starting city in each solution process. This strategy allows *N* sequences or permutations of cities called *Giant Tours*. Using this sequence as a starting point, the generation of high quality individuals is achieved, which can be explained by the fact of being incorporated within each solution Π several arcs for which the objective function is highly sensitive. The steps to build the individuals of the initial population are as follows:

- Each giant tour is built considering three heuristics: the nearest neighbor (Gutin et al., 2002), the heuristic methods for TSP of Christofides (1976) and Lin and Kernighan (1973). Initially, a percentage of the population (30% in this case) is formed with the sequences obtained by the nearest neighbor method considering different starting cities. Another percentage of the population (30 % in this case) is generated using Christofides heuristics involving different starting cities on the Eulerian graph. It should be noted that this process begins with the construction of a Minimum Spanning Tree (MST) on the complete graph $G(V,E)$. From an Eulerian graph, the MST is constructed by pairing the vertices of odd degree of the MST, and, as a final step, a Hamiltonian graph is constructed from the Eulerian graph. Finally, the remaining percentage of the population is obtained from a solution resulting by applying LKH. This solution is slightly modified through exchanges between genes in order to obtain the remaining individuals of the population. It should ensure a diverse population.

- The giant tour is divided into m clusters, where the number of cities of each cluster is chosen randomly, ensuring compliance with Constraint (10) resulting in $m-1$ routes. The route m is formed by the cities that have yet to be allocated according the order established on each chromosome.
- Each chromosome is evaluated according to Equation (1) in order to obtain the initial value of the objective function.

4.3. Selection and crossover

Tournament-selection is used to select individuals who will be submitted to crossover under a given probability ρ_c . Goldberg and Deb (2004) indicate that tournament selection has the best convergence properties and computational complexity when compared with other selection operators presented in the literature.

Being a permutation-based encoding, the application of the crossover operator requires some care. A constraint of the problem is that all cities are visited only once. This implies that two genes on the chromosome cannot have the same value. Different strategies can be applied to prevent infeasible sequences, such as Partially Mapped Crossover (PMX) (Goldberg & Lingle, 1985), Order Crossover (OX) (Oliver et al., 1987), Order Crossover #2 (OX2) (Syswerda, 1991), Position Based Crossover (PBX) (Syswerda, 1991) and Cycle Crossover (CX) (Oliver et al.,1987). In this paper, the PMX method is used.

4.4. Local-search

The offspring obtained so far is subjected to mutation under a given probability _m. The mutation process is based on the application of neighborhood structures seeking to improve each of the routes obtained after application of the crossover operator. The upgrading steps applied are neighborhood structures known as inter-routes and intra-routes.

4.4.1. Inter-route structures

Five neighborhood structures involving inter-routes movements are applied (Subramanian et al. 2012), each of which is described below:

- **Shift (1; 0)**, city p_k is transferred from route TSP*i* to route TSP_j.
- **Shift (2; 0)**, two adjacent cities p_k and p_{k+1}, are transferred from route TSP_i to route TSP_j .
- **Swap (1; 1)**, a permutation is performed between city p_k from route TSP_i and city p_l from route TSP_j.
- **Swap (2; 1)**, two adjacent cities p_k and p_{k+1} from route TSP_i are permuted with city p_l from route TSP_j.
- **Swap (2; 2)**, two adjacent cities p_k and p_{k+1} from route TSP_i are permuted with two adjacent cities p_l and p_{l+1} from route TSP_j.

4.4.2. Intra-route structure

The local search algorithm **2-Opt** is used as the intra-route neighborhood structure, which consists of removing two nonadjacent edges *E* (p_k; p_{k+1}) and *E* (p_l; p_{l+1}) belonging to the same route TSP_i, and then two arcs *E'* (p_k; p_{k+1}) and *E'* (p_l; p_{l+1}) are added so that a new route TSP'_i is created.

Inter and intra-route neighborhood structures are applied in the order in which they were described and applied exhaustively, in order to choose the feasible movement to improve to a greater extent the value of the objective function and thus accelerate convergence to quality solutions.

4.5. Population substitution

When a new individual is obtained by the genetic operators of selection, recombination and mutation, an evaluation is necessary to determine whether that individual can be part of the population. The steps for population substitution are as follows:

- Verify that the objective function value of the new individual is less than the value of the objective function of the worst individual in the current population.
- Verify that the new individual does not exist in the current population.

If the two conditions described above are met, then the new individual enters the population replacing the individual of lower quality. The general methodology structure is presented in Algorithm 1.

Data: Instance size n, population size N, probability ρ_m and ρ_c, maximum cities per route $c_{\max}$, P $\leftarrow \{\ \}$.
Result: Feasible optimal solution Π to the the mTSP
while $|P| \neq n$ **do**
    P $\leftarrow$ P $\cup$ chromosomes from nearest neighbor, Christo_des algorithm and Lin-Kernighan heuristic;
end
Calculate the minimum of routes TSPmin according to (12);
Calculate the minimum of cities per route cmin according to (11);
for $i \leftarrow 1..N$ **do**
    Randomly split each chromosome with route size between $c_{\min}$ and $c_{\max}$;
end
while ⌐ *stopping criteria* (the fitness of the best solution has converged or a maximum number of generations has been reached) **do**
    $p_1, p_2 \leftarrow$ tournament-selection (P);
    if ($p_1 \neq p_2$) and ($rand() \leq \rho_c$) **then**
        $h \leftarrow$ Crossover(p_1, p_2);
    else
        $h \leftarrow p_1$;
    end
    if ($rand() \leq \rho_m$) **then**
        $h \leftarrow$ local-search (h);
        Place h into the population;
        generation ++
    else
        Place h into the population;
        generation ++
    end
end
return The best solution (P)

Algorithm 1. MCBGA Pseudo code

5. Computational results

In order to demonstrate the effectiveness of the proposed methodology tests on six standard instances of the library TSPLIB (Reinelt, 2014) are carried out, and the results obtained are compared to the results obtained by the algorithms ACO (Junjie & Dingwei, 2006) and MGA (Seidighpour et al. 2011). The instances used in the tests are referred to as pr76, pr152, pr226, pr299, pr436 and pr1002.

The proposed algorithm is implemented in the programming environment Matlab 2010a on a PC Core 2 Duo with 3 GHz, 4 GB RAM Memory and Windows 7.

The MCBGA is characterized by an intensive neighborhood search. For this reason, the tests were conducted with a mutation rate of $\rho_m = 0.9$. The crossover rate used was $\rho_c = 1.0$.

Table 1 shows a comparison between the results obtained by algorithm MCBGA and ACO and MGA algorithms. The best solution found for each instance and the average value of the objective function after 10 runs of the algorithm are reported. The amount of travel agents used (m); the maximum number of cities (cmax) and the total number of cities to visit (n) are also reported.

The proposed algorithm finds better solutions for six instances used in this work with respect to the results reported in the literature for the same six instances (Seidighpour et al., 2011; Junjie & Dingwei, 2006), reflecting the effectiveness of the proposed methodology.

Table 1
Comparison of results of the algorithms MCBGA, ACO and MGA

Instance	*n*	C_{max}	MCBGA				ACO				MGA			
			Best	Avg	T [s]	*m*	Best	Avg	T [s]	*m*	Best	Avg	T [s]	*m*
pr76	76	20	153774	157666,6	1,8	4	178597	180690	19	5	157444	160574	43	5
pr152	152	40	119938	128768,8	11,07	4	130953	136341	41	5	127839	133337	91	5
pr226	226	50	157239	160836,4	17,63	5	167646	170877	62	5	166827	178501	195	5
pr299	299	70	71081	73192,8	35,65	5	82106	83845	65	5	82176	85796	363	5
pr436	436	100	136809	140436,6	80	5	161955	165035	95	5	173839	183698	623	5
pr1002	1002	220	313561	318778,8	465,7	5	382198	387205	186	5	427269	459179	2892	5

The new best solutions found are presented below:

Instance pr76: 4 routes, cost 153774.

23 22 24 46 45 44 48 47 69 68 70 67 50 49 53 54 42 43 27 26;
21 25 55 56 57 58 59 60 41 61 62 63 64 73 72 71 65 66 51 52;
28 33 34 40 39 38 18 37 36 35 32 29 30 31 19 20 5 4;
2 3 6 7 8 9 10 12 11 17 16 15 13 14 74 75 76.

Instance pr152: 4 routes, cost 119938.

35 49 50 76 74 75 73 51 78 46 52 53 72 71 70 69 54 45 44 55 56 68 67 66 65 57 43 42 58 59 64 63 62 61 60 41 15 2 16;
3 14 4 13 5 12 6 11 7 10 8 9 17 18 40 19 21 20 22 23 39 24 25 26 27 28 38 29 30 31 32 33 37 34;
79 114 117 140 139 141 142 143 138 137 118 119 136 135 144 145 146 134 133 120 121 132 131 147 148 149 130 129 122 123 128 127 151 150 152 126 125 124 77 48;
47 92 93 94 95 116 115 96 91 97 98 99 90 80 113 100 89 81 101 102 103 88 82 112 111 104 87 105 106 107 110 109 108 85 84 86 83 36.

Instance pr226: 5 routes, cost 157239.

74 75 76 77 78 79 80 81 82 83 84 85 86 111 112 113 114 115 116 117 118 119 120 121 122 123 124 125 132 130 131 129 127 128 126 110 109 108 107 106 104 105 103 70 69 68 71 72;
212 211 210 209 208 207 206 205 204 203 202 201 200 199 198 197 196 195 194 193 192 191 190 189 188 187 186 185 184 183 182 181 180 179 178 177 176 175 174 173 172 171 143 142 148 149 147 64 67 13;
87 95 96 89 88 97 98 90 91 99 100 92 93 101 102 94 137 144 146 145 138 139 140 141 150 170 169 168 167 166 165 164 163 162 161 160 159 158 157 156 155 154 153 152 151 41 43 47 46 49;
9 10 14 15 18 19 20 21 22 23 24 25 26 27 29 30 34 35 37 38 39 40 42 44 45 36 48 33 32 31 28 50 51 52 53 54 55 56 57 58 59 60 61 62 63 65 17 16 12 8;
2 3 4 5 6 7 11 66 136 135 134 133 213 214 215 216 217 218 219 220 221 222 223 224 225 226 73.

Instance pr299: 5 routes, cost 71081.

3 7 88 90 89 91 86 94 97 96 95 98 83 84 99 100 102 82 81 79 80 76 78 77 75 72 73 74 71 69 70 67 68 105 64 65 107 62 63 66 33 34 29 28 27 26 25 24 23 21 18 17 16 15 13 12 11 10 9 8 5 6 2;
19 20 22 30 31 32 35 36 38 37 42 40 41 43 45 46 44 47 48 49 50 51 52 53 54 56 55 116 114 115 117 119 118 121 122 120 180 181 182 183 184 185 256 186 253 252 255 261 260 254 257 258 259 262 264 263 266 265 267 268 248 249 244 247 243 237 236 134 136 92;
14 85 87 101 103 168 167 197 240 242 196 193 192 191 194 195 170 169 174 172 128 109 108 106 110 61 59 60 39 57 58 112 111 113 123 127 126 124 125 175 177 176 178 187 189 179 171 173 190 188 250 251 245 246 270 269 271 239 241 238 272 273 274 235 234 198 164 163 162 139;
4 142 141 140 161 138 137 135 133 132 129 104 130 131 165 166 201 199 200 202 228 203 229 231 230 233 232 277 276 275 279 278 281 282 283 280 284 285 286 219 221 222 209 220 224 223 226 225 227 299 206 205 204 207 208 210 211 160 159 158 157 144 145 146;
92 142 212 211 217 215 216 286 287 288 290 289 291 292 294 293 296 295 297 214 213 155 154 153 152 151 150 149 148 146 147.

Instance pr436: 5 routes, cost 136809.

4 6 7 8 9 10 11 12 13 14 15 16 17 32 31 30 29 28 27 26 64 25 23 21 22 24 65 66 67 20 19 18 68 69 70 71 72 88 89 90 91 104 105 106 107 144 146 112 111 75 61 60 59 58 79 80 81 82 56 55 48 47 44 43 45 46;
49 42 86 224 223 233 234 235 236 237 238 239 240 241 242 243 244 245 246 247 248 249 250 210 211 212 213 214 215 216 217 218 219 220 221 191 184 192 183 193 194 195 196 197 198 199 200 201 202 203 205 206 172 173 174 204 175 176 177 178 179 180 181 182 154 155 156 157 158 159 160 161 162 163 134 135 136 137 138 139 140 141 142 152 153 151 150 143 108 74 73 63 62 33 34 35 36 37 40 5;
39 77 76 109 113 114 116 118 119 121 122 101 120 117 115 103 102 92 93 94 100 95 96 97 98 99 125 123 124 126 127 129 128 130 131 132 133 167 165 164 166 169 168 170 171 207 208 209 251 253 252 255 254 316 315 317 318 319 320 321 322 323 324 325 326 327 328 329 331 332 330 303 302 304 305 306 307 308 309 310 311 312 313 314 256 257 258 259 260 261 262 263 264 265 266 267 268 269 270 271;
41 38 57 78 87 110 147 148 145 185 190 222 272 301 273 274 275 276 277 278 279 294 295 296 297 298 299 300 333 334 335 336 337 338 339 360 370 361 362 363 364 365 366 367 368 369 378 385 386 387 388 401 400 402 404 405 417 418 428 429 431 416 415 419 420 421 414 406 407 413 412 423 422 432 433 430 427 424 426 425 410 411 409 408 403 434 438 437 436 435 439 284 344 343 342 281 230 187 54 50;
51 53 189 188 225 231 232 280 285 286 287 340 288 289 290 291 292 293 359 357 358 356 355 354 352 353 351 350 349 347 348 346 345 376 377 379 389 390 380 374 375 381 391 392 399 398 396 397 393 395 394 384 382 383 372 371 373 341 282 229 283 228 227 226 186 149 85 84 83 52 2 3.

Instance pr1002: 5 routes, cost 313561.

75 77 93 90 92 91 98 97 96 95 136 135 134 133 100 101 102 128 129 130 131 132 137 138 139 140 141 142 152 151 150 149 148 144 143 145 146 147 126 127 125 124 123 122 121 257 258 259 256 255 254 253 249 250 247 248 260 261 262 246 245 244 243 251 252 995 153 154 155 156 164 165 166 169 168 167 163 162 158 157 185 159 160 161 172 171 170 173 174 175 176 177 178 179 180 181 184 183 182 996 203 204 206 205 207 210 209 208 200 199 198 201 202 196 197 194 195 187 186 241 242 240 238 239 188 189 193 192 190 191 237 236 235 233 234 221 219 218 217 212 211 213 216 215 214 220 222 223 224 225 226 227 228 229 230 232 231 263 264 265 266 267 268 269 271 270 272 273 274 275 276 277 278 279 280 281 282 301 302 303 304 113 112 114 115 116 118 106 105 994 107 108 109 307 308 309 310 311 312 45 44 43 42 41 50 51 53 52 54 73 71;
74 80 83 84 48 47 46 306 305 991 297 298 286 287 288 403 402 408 409 411 412 413 414 415 417 416 410 407 406 405 421 422 420 418 419 426 425 424 423 464 463 462 461 460 458 457 456 432 433 446 444 443 442 440 438 439 441 632 631 630 629 611 610 609 608 607 606 605 604 603 602 601 599 597 596 595 594 593 592 591 559 560 558 557 556 555 554 553 552 551 550 548 542 543 544 541 539 540 545 546 547 549 562 561 563 564 565 519 518 520 521 517 516 567 566 568 570 569 514 515 523 522 525 524 530 529 526 527 528 536 537 538 535 533 534 532 531 999 510 511 512 509 508 507 506 505 504 503 502 501 498 499 500 489 488 487 513 485 486 490 491 495 496 497 494 493 492 484 483 482 481 998 580 579 571 572 573 574 590 589 588 576 575 578 577 581 582 583 584 585 449 448 450 451 452 453 455 454 475 474 473 471 472 476 477 478 479 480 470 469 468 467 120 117 119 104 103 99 94 76;
79 78 89 87 110 111 283 284 428 429 430 431 997 434 447 445 795 796 797 798 799 800 804 805 806 809 810 811 812 820 821 822 825 824 823 819 818 814 813 841 815 816 817 828 827 826 829 830 831 832 833 834 835 836 837 840 839 838 1002 859 860 862 861 863 866 865 864 856 855 854 857 858 852 853 850 851 843 842 897 898 896 894 895 844 845 849 848 846 847 893 892 891 889 890 877 875 874 873 868 867 869 872 871 870 876 878 879 880 881 882 883 884 885 886 888 887 919 900 901 902 918 917 916 904 903 906 905 909 910 911 912 915 914 913 777 776 778 779 780 781 783 782 803 802 801 787 794 793 788 786 785 784 758 757 756 789 790 791 792 751 752 753 754 755 747 746 748 750 749 733 734 735 738 744 745 743 742 741 740 739 737 736 730 731 732 616 615 614 613 612 627 628 626 625 992 633 634 635 636 637 638 639 640 641 373 372 371 370 369 377 376 375 374 437 436 435 427 404 86 88;
82 85 300 299 285 466 465 459 586 587 598 600 808 807 1001 908 907 899 920 921 922 923 924 925 927 926 928 929 930 931 932 933 934 935 936 937 938 939 957 958 959 960 769 767 766 768 770 771 772 773 775 774 762 761 760 759 1000 763 764 765 702 703 704 705 697 698 699 700 701 969 968 967 966 965 964 963 962 961 993 953 954 956 955 940 941 942 943 944 945 946 947 948 952 951 949 950 977 978 979 976 973 974 975 980 982 981 972 971 970 987 986 985 983 984 990 989 988 684 683 685 686 687 682 676 675 677 674 671 670 669 672 673 678 679 681 680 688 689 690 691 692 693 694 695 696 706 707 709 708 710 713 711 712 723 722 721 715 714 716 717 718 720 719 668 667 666 665 664 663 726 725 724 728 729 727 620 619 618 617 624 623 621 622 649 650 651 648 645 646 647 652 654 653 644 643 642 659 658 657 655 656 662 661 660 356 355 357 358 359 354 348 347 349 346 343 342 327 313 49 81;
341 344 345 350 351 353 352 360 361 362 363 364 365 366 367 368 378 379 381 380 382 385 383 384 395 394 393 387 386 388 389 390 392 391 340 339 338 337 336 335 398 397 396 400 401 399 292 291 290 289 296 295 293 294 321 322 323 320 317 318 319 324 326 325 316 315 314 331 330 329 328 334 333 332 28 27 29 30 31 26 20 19 21 18 15 14 13 16 17 22 23 25 24 32 33 34 35 36 37 38 39 40 58 57 55 56 72 70 69 68 67 66 65 59 60 61 62 64 63 12 11 10 9 8 7 6 4 3 5 2.

6. Conclusion

In this paper, a genetic algorithm modified to solve the mTSP has been presented. In order to examine the efficiency of the algorithm, tests on six instances of literature ranging from 76 to 1002 cities were made. The algorithm found better solutions for all instances used with respect to the values reported by Seidighpour et al. (2011) and Junjie and Dingwei (2006). Similarly, the average solutions after 10 run of the algorithm, for each instance, are less than the average reported solutions. The results show that the efficiency of the proposed algorithm, which combines technique initialization and neighborhood structures, provide good quality solutions at each iteration of the algorithm.

References

Angel, R. D., Caudle, W. L., Noonan, R., & Whinston, A. N. D. A. (1972). Computer-assisted school bus scheduling. *Management Science*, *18*(6), B-279.

Bektas, T. (2006). The multiple traveling salesman problem: an overview of formulations and solution procedures. *Omega*, *34*(3), 209-219.

Bellmore, M., & Hong, S. (1974). Transformation of multisalesman problem to the standard traveling salesman problem. *Journal of the ACM (JACM)*, *21*(3), 500-504.

Brumitt, B. L., & Stentz, A. (1998, May). GRAMMPS: A generalized mission planner for multiple mobile robots in unstructured environments. In *Robotics and Automation, 1998. Proceedings. 1998 IEEE International Conference on*(Vol. 2, pp. 1564-1571). IEEE.

Calvo, R. W., & Cordone, R. (2003). A heuristic approach to the overnight security service problem. *Computers & Operations Research*, *30*(9), 1269-1287.

Carter, A. E., & Ragsdale, C. T. (2002). Scheduling pre-printed newspaper advertising inserts using genetic algorithms. *Omega*, *30*(6), 415-421.

Chen, S. H., & Chen, M. C. (2011, November). Operators of the two-part encoding genetic algorithm in solving the multiple traveling salesmen problem. In *Technologies and Applications of Artificial Intelligence (TAAI), 2011 International Conference on* (pp. 331-336). IEEE.

Christofides, N. (1976). *Worst-case analysis of a new heuristic for the travelling salesman problem* (No. RR-388). Carnegie-Mellon Univ Pittsburgh Pa Management Sciences Research Group.

Chu, P. C., & Beasley, J. E. (1997). A genetic algorithm for the generalised assignment problem. *Computers & Operations Research*, *24*(1), 17-23.

Faigl, J., Kulich, M., & Preucil, L. (2005, September). Multiple traveling salesmen problem with hierarchy of cities in inspection task with limited visibility. In *proceedings of the 5th Workshop on Self-Organizing Maps. Universit Paris-Sud* (pp. 91-98).

Gilbert, K. C., & Hofstra, R. B. (1992). A new multiperiod multiple traveling salesman problem with heuristic and application to a scheduling problem.*Decision Sciences*, *23*(1), 250-259.

Goldberg, D. E., & Deb, K. A comparison of selection scheme used in genetic algorithms. *Foundations of Genatic Algorithms*, 69-93.

Goldberg, D. E., & Lingle, R. (1985, July). Alleles, loci, and the traveling salesman problem. In *Proceedings of the first international conference on genetic algorithms and their applications* (pp. 154-159). Lawrence Erlbaum Associates, Publishers.

Gorenstein, S. (1970). Printing press scheduling for multi-edition periodicals.*Management Science*, *16*(6), B-373.

GuoXing, Y. (1995). Transformation of multidepot multisalesmen problem to the standard travelling salesman problem. *European Journal of Operational Research*, *81*(3), 557-560.

Gutin, G., Yeo, A., & Zverovich, A. (2002). Traveling salesman should not be greedy: domination analysis of greedy-type heuristics for the TSP. *Discrete Applied Mathematics*, *117*(1), 81-86.

Junjie, P., & Dingwei, W. (2006, August). An ant colony optimization algorithm for multiple travelling salesman problem. In *Innovative Computing, Information and Control, 2006. ICICIC'06. First International Conference on* (Vol. 1, pp. 210-213). IEEE.

Laporte, G., & Nobert, Y. (1980). A cutting planes algorithm for the m-salesmen problem. *Journal of the Operational Research Society*, 1017-1023.

Lin, S., & Kernighan, B. W. (1973). An effective heuristic algorithm for the traveling-salesman problem. *Operations research*, *21*(2), 498-516.

Miller, C. E., Tucker, A. W., & Zemlin, R. A. (1960). Integer programming formulation of traveling salesman problems. *Journal of the ACM (JACM)*, *7*(4), 326-329.

Okonjo-Adigwe, C. (1988). An effective method of balancing the workload amongst salesmen. *Omega*, *16*(2), 159-163.

Oliver, I. M., Smith, D., & Holland, J. R. (1987). Study of permutation crossover operators on the traveling salesman problem. In *Genetic algorithms and their applications: proceedings of the second International Conference on Genetic Algorithms: July 28-31, 1987 at the Massachusetts Institute of Technology, Cambridge, MA*. Hillsdale, NJ: L. Erlhaum Associates, 1987..

Reinhelt, G. (2014). {TSPLIB}: a library of sample instances for the TSP (and related problems) from various sources and of various types. *URL: http://comopt. ifi. uniheidelberg. de/software/TSPLIB95*.

Russell, R. A. (1977). Technical Note—An Effective Heuristic for the M-Tour Traveling Salesman Problem with Some Side Conditions. *Operations Research*, *25*(3), 517-524.

Sariel-Talay, S., Balch, T. R., & Erdogan, N. (2009). Multiple traveling robot problem: A solution based on dynamic task selection and robust execution.*Mechatronics, IEEE/ASME Transactions on*, *14*(2), 198-206.

Sedighpour, M., Yousefikhoshbakht, M., & Mahmoodi Darani, N. (2012). An Effective Genetic Algorithm for Solving the Multiple Traveling Salesman Problem. *Journal of Optimization in Industrial Engineering*, (8), 73-79.

Song, C. H., Lee, K., & Lee, W. D. (2003, July). Extended simulated annealing for augmented TSP and multi-salesmen TSP. In *Neural Networks, 2003. Proceedings of the International Joint Conference on* (Vol. 3, pp. 2340-2343). IEEE.

Subramanian, A., Penna, P. H. V., Uchoa, E., & Ochi, L. S. (2012). A hybrid algorithm for the heterogeneous fleet vehicle routing problem. *European Journal of Operational Research*, *221*(2), 285-295.

Svestka, J. A., & Huckfeldt, V. E. (1973). Computational experience with an m-salesman traveling salesman algorithm. *Management Science, 19*(7), 790-799.

Syswerda, G. (1991). Schedule optimization using genetic algorithms.*Handbook of genetic algorithms*.

Sze, S., & Tiong, W. (2007). A comparison between heuristic and meta-heuristic methods for solving the multiple traveling salesman problem. *World Academy of Science, Engineering and Technology*, *1*.

Tang, L., Liu, J., Rong, A., & Yang, Z. (2000). A multiple traveling salesman problem model for hot rolling scheduling in Shanghai Baoshan Iron & Steel Complex. *European Journal of Operational Research*, *124*(2), 267-282.

Yousefikhoshbakht, M., & Sedighpour, M. (2012). A combination of sweep algorithm and elite ant colony optimization for solving the multiple traveling salesman problem. *Proceedings of the Romanian Academy A*, *13*(4), 295-302.

Yu, Z., Jinhai, L., Guochang, G., Rubo, Z., & Haiyan, Y. (2002). An implementation of evolutionary computation for path planning of cooperative mobile robots. In *Intelligent Control and Automation, 2002. Proceedings of the 4th World Congress on* (Vol. 3, pp. 1798-1802). IEEE.

Zhang, T., Gruver, W., & Smith, M. H. (1999). Team scheduling by genetic search. In *Intelligent Processing and Manufacturing of Materials, 1999. IPMM'99. Proceedings of the Second International Conference on* (Vol. 2, pp. 839-844). IEEE.