Solving the Free Clustered TSP Using a Memetic Algorithm

Abdullah Alsheddy
College of Computer and Information Sciences
Al Imam Mohammad Ibn Saud Islamic University (IMSIU)
Riyadh, Saudi Arabia

***Abstract*—The free clustered travelling salesman problem (FCTSP) is an extension of the classical travelling salesman problem where the set of vertices is partitioned into clusters, and the task is to find a minimum cost Hamiltonian tour such that the vertices in any cluster are visited contiguously. This paper proposes the use of a memetic algorithm (MA) that combines the global search ability of Genetic Algorithm with local search to refine solutions to the FCTSP. The effectiveness of the proposed algorithm is examined on a set of TSPLIB instances with up to 318 vertices and clusters varying between 2 and 50 clusters. Moreover, the performance of the MA is compared with a Genetic Algorithm and a GRASP with path relinking. The computational results confirm the effectiveness of the MA in terms of both solution quality and computational time.**

Keywords*—Combinatorial optimization; clustered travelling salesman problem; memetic algorithm; guided local search; genetic algorithm

I. Introduction

The travelling salesman problem (TSP) is one of the best known and most widely studied combinatorial optimization problems. Many variants of the TSP have been proposed and solved during the last decades. This paper focuses on the clustered travelling salesman problem (CTSP), a variant of the TSP that was introduced by Chisman [1]. Similar to the TSP, the objective of the CTSP is to construct a Hamiltonian path with minimum distance, visiting all cities exactly once. Cities in the CTSP, however, are partitioned into predefined clusters and all cities belonging to the same cluster should be visited consecutively.

The CTSP has several applications in various fields. Examples of CTSP applications include automated warehouse routing [1], shops and grocery suppliers [2] and emergency vehicle dispatching [3] in the vehicle routing domain; disk fragmentation and computer operations in the IT domain [4]; machine scheduling and production planning [5] in the manufacturing domain; and microscopy (cytology) [4].

Most of the related research addressed the so-called *ordered* CTSP (OCTSP) in which the clusters has to be visited in a prespecified order. Although such a prespecified order is not necessarily defined in real-life applications, there are few algorithms developed for the CTSP without a pre-order [6]. This variant of the CTSP is referred to as the free CTSP (FCTSP).

This paper considers the FCTSP which can be formally defined as follows. Given a complete undirected graph $G = (V, E)$ with vertex set $V = \{v_1, v_2, ..., v_n\}$, and edge set $E = \{(v_i, v_j) : v_i, v_j \in V, i \neq j\}$. The vertex set V is partitioned into m predefined clusters: $V_1, V_2, ..., V_m$. Assuming that a non-negative cost or distance c_{ij} is associated with each edge $(v_i, v_j) \in E$, the FCTSP consists of determining a least cost Hamiltonian cycle on G such that the vertices of each cluster are visited contiguously and the clusters can be visited in any order. For illustration, Fig. 1 shows examples of several solutions to a FCTSP.

There have been several exact and metaheuristics algorithms developed for the OCTSP. An approximation algorithm and some other heurisics were proposed in [7]. The LBD-COMP is an exact partitioning algorithm proposed in [8]. Other approximation algorithms were developed in [9] and [10]. A hybrid of Tabu Search (TS) and Genetic Algorithm (GA) was developed for solving the OCTSP in [11]. This hybrid algorithm runs multiple TS search threads while periodically applying a phase of diversification using the Edge Recombination crossover operator to generate offspring solutions that will seed the TS search threads again. Ahmed [12] developed a hybrid genetic algorithm using sequential constructive crossover, the 2-opt algorithm and local search for the OCTSP.

For the FCTSP, a genetic algorithm was proposed in [13], which first searches for an optimal inter-cluster edges and then the intra-cluster edges. The two-level Genetic Algorithm (TLGA) is another algorithm developed for the FCTSP in [14]. The TLGA consists of two interrelated levels; the lower level focuses on finding the shortest Hamiltonian cycle for each cluster, whereas the higher level constructs the complete tour by randomly deleting an edge from each cycle and then heuristically connecting the clusters through the intra-cluster edges. As reported in [14], the TLGA performed well in comparison to other GA variants. Later, Mestria et al. [6] developed several path-relinking strategies incorporated to a Greedy Randomized Adaptive Search Procedure (GRASP) for solving the FCTSP. The best performing heuristic was the GRASP that uses path-relinking in each iteration and as a post-optimization strategy, outperforming other GRASP variants and the TLGA [14].

The GRASP algorithm proposed in [6] has two important characteristics: 1) it deals with the whole FCTSP in a single phase without differentiation between the search for the inter-cluster and for the intra-cluster edges, unlike the two phases approach of the TLGA; and 2) the underlying local search procedure implements the well-known 2-Opt heuristic as one of the most effective local search heuristic for the classical TSP. These characteristics suggest the potential of adapting and then applying successful TSP heuristics to the FCTSP.

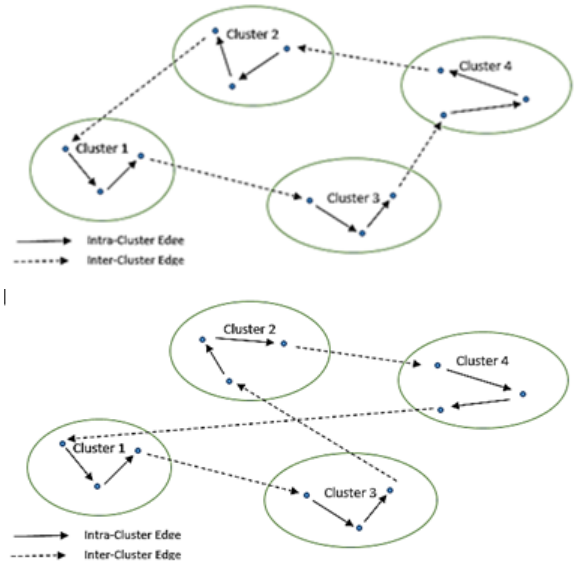

Fig. 1. Examples of two possible solutions to a FCTSP with 4 clusters and 12 vertices.

Thus, we propose in this paper a memetic algorithm that combines the global search ability of Genetic Algorithm with a local-search-based metaheuristic, namely Guided Local Search (GLS) [15], to refine solutions to the FCTSP. GLS, which is a successful algorithm for the TSP [16], incorporates a straight forward extension of the 2-Opt heuristic to handle the clustering constraint of FCTSP. The performance of the proposed memetic algorithm is evaluated through several experiments that include comparisons with the TLGA [14] and the GRASP algorithm [6].

This paper is organized as follows. The proposed approach and its application to the FCTSP is described in section II. Then, the experimental results in comparison with state-of-the-art algorithms are presented in Section III. Finally, concluded remarks and further research are given in Section IV.

II. Proposed Approach

A. Memetic Algorithms

Local search is the basis of many heuristic methods for combinatorial optimization problems. Starting from an initial solution, local search algorithms iteratively improve the current solution by exploring its neighbourhood for better movements. Although local search algorithms usually return good solutions, these would easily get stuck in local optima, which are typically overcome by an escape mechanism as in Tabu Search, Simulated Annealing and Guided Local Search (GLS) [17]. In GLS, the escaping mechanism is based on augmenting the objective function with penalties. Every time the underlying local search algorithm reaches a local optimal, GLS augments the cost function by adding penalties to selected bad features, and then restarts the local search algorithm while using the augmented function $h(s)$ in the search process instead of the main objective function, $g(s)$:

$$h(s) = g(s) + \lambda \sum_{i \in F} p_i * I_i(s) \quad (1)$$

where, s is a candidate solution and λ is a parameter of the GLS algorithm. F refers to the set of all features that were used to distinguish between solutions with different characteristics. p_i is the penalty of feature i (each p_i is initialized to 0, and incremented by 1 whenever it is selected for penalization), and $I_i(s)$ is an indicator which is equal to 1 only if s exhibits the feature i; 0 otherwise. GLS penalizes the most costly features in the current solution, weighted by the number of times the feature has been penalized so far.

In contrast to local search algorithms, global search algorithms try to overcome local optima in order to find more globally optimal solutions. Genetic Algorithms are very popular global search algorithms which have been successfully applied to many combinatorial optimisation problems. A GA is a population-based search technique inspired from the biological principals of natural selection and genetic recombination. At every generation, a GA maintains a population of individuals that represent candidate solutions to the problem. This population evolves throughout the optimization process to find global solutions by applying reproduction operators. Each individual is evaluated to give some measure of its "fitness". Selection of parents for reproduction is based on their fitness. The reproduction operators include crossover and mutation which are both applied with certain probabilities. The evolution of the GA continues until either an optimal solution is found, or some other stopping criteria have been met.

Memetic algorithms [18] denote a broad class of metaheuristics that extend global search methods, such as GA, by incorporating problem-specific knowledge, usually in the form of a local search strategy or through the use of specialised search operators. Thus, a memetic algorithm hybridizes global and local search, such that the individuals of a population in a global search algorithm have the opportunity for local improvement in terms of local search. The applications of MAs are enormous, including areas such as routing, assignment and planning problems [18].

B. Memetic Algorithm for the FCTSP

Our memetic algorithm (MA) is a hybrid of GA for global search and GLS for local search. In GLS, the underlying local search is the 2-Opt heuristic which is a well-known TSP heuristic. GLS is a simple local-search-based metaheuristic with only one parameter to tune. Nevertheless, GLS was shown to be a very effective method for the TSP [16] and other routing and planning problems [15]. In the proposed MA, GLS is complemented by the genetic operators to enhance the exploration of the space of FCTSP solutions.

The proposed MA is outlined in Algorithm 1. It starts by randomly generating an initial population of N solutions. Each individual in this population undergoes local improvement by applying GLS. Then, the algorithm iteratively evolves the population by applying genetic operators and GLS. Since GLS is more computationally expensive than the genetic operators, it is performed in the proposed MA periodically, every 10 generations. A description of the algorithm design for the FCTSP is given below:

C. Guided Local Search (GLS)

In the proposed method, a solution of the FCTSP is a tour that is represented by a permutation (i.e. vector) of cities.

MA(Problem instance ($problem$), population size (N), stopping criterion ($maxGene$))
$P \leftarrow RandomPopoulation(problem, N)$;
$P^* \leftarrow GuidedLocalSearch(P)$;
for $generation \leftarrow 1$ until $maxGene$ **do**
$P^{'} \leftarrow MatingSelection(P^*)$;
$P^{'} \leftarrow GeneticOperators(P^{'})$;
if $generation$ mod 10 = 0 **then**
$P^{'} \leftarrow GuidedLocalSearch(P^{'})$;
end if
$P^* \leftarrow SurvivorSelection(P \cup P^{'})$;
end for
return P^*;

Algorithm 1: Memetic Algorithm

The permutation determines the order of the cities in the tour. The 2-Opt heuristic is a well-known and very simple, yet effective local search algorithm for the classical TSP. The 2-Opt heuristic iteratively improves an initial tour by testing all neighbour solutions obtained by applying the 2-exchange neighbourhood operator (i.e. a neighbour is obtained from the current tour by deleting two edges and reconnecting the two resulting paths with the only possible way that yields a new tour). To implement the 2-Opt heuristic for the FCTSP, the feasibility of the newly generated solutions with respect to the clustering constraint is maintained by applying the 2-exchange operator to any two non-adjacent edges if and only if both edges are in the same cluster or are inter-cluster edges.

As reported in [16], GLS can sit on top of the 2-Opt heuristic and guide it to escape local minima in an efficient and effective manner. GLS converges quickly to a close to optimal solution, particularly when it is coupled with Fast Local Search [15]. The latter is a general method that divides a neighbourhood set into sub-groups. Each subgroup is associated with an activation bit, to control which sub-groups will be explored during the search process. The proposed MA incorporates GLS as the local search procedure, and the same algorithm design presented in [16] to implement GLS for the FCTSP are followed in this study.

The key element of GLS is the definition of a set of solution features. A feature should contribute to part of the overall solution cost. For the FCTSP, a tour includes a number of edges, and each edge is associated with a cost (edge length). Therefore, the set of all edges defines the set of features for the FCTSP. Each tour either includes (i.e. exhibits) an edge (i.e. feature) or not.

D. Fitness Function and Mating Selection

For the FCTSP, the fitness of each individual chromosome in the population is the length of the entire tour specified by the chromosome. Mating selection determines the procedure to choose individuals from the current population to undergo the genetic operators. In the proposed MA, the idea is to use a selection strategy that favours exploration. Thus, all individuals in the current population are subjected to undergo the genetic operators. This is attained by randomly ordering parents, and then the genetic operator will use the first and second parents to generate the first offspring, the second and third parents to generate the second offspring, and so on.

E. Genetic Operator

The genetic operators, both crossover and mutation, for the FCTSP can be used at the inter-cluster level by changing the visiting sequence of clusters, or at the intra-cluster level by changing the gene segment for each cluster. In this study, the genetic operator at the inter-cluster level is implemented in order to intensify exploration. The following describe the details of the crossover and mutation operators.

Among the several effective crossover operators that have been proposed for the TSP and its variants is the sequential constructive crossover (SCX). The SCX operator has been modified and applied to the OCTSP in [12]. We follow the same implementation to apply the SCX to the FCTSP, however, with slight modifications. The following procedure describes how the offspring is constructed from $Parent_1$ and $Parent_2$ using the modified SCX:

- Step 1: The first vertex of $Parent_1$ is chosen to be the first gene of the offspring chromosome.
- Step 2: Given the current vertex v of the offspring chromosome, and the two candidate vertices v_1 and v_2 that represent the first legitimate vertices appeared after v in the chromosome of $Parent_1$ and $Parent_2$ respectively, the next gene in the offspring chromosome will be v_1 if it is nearer to v, and v_2 otherwise.
- Step 3: Once all vertices in the current cluster are added to the offspring chromosome, move to the next cluster according to the order of clusters in $Parent_1$ and repeat Step 2 until the offspring chromosome is completed.

Mutation plays an important role to help GA avoids establishing a uniform population unable to evolve. It usually modifies the genes of a chromosome selected with a mutation probability. For the FCTSP, the reciprocal exchange mutation operator is implemented, which selects two positions within a chromosome at random and then swaps their contents to produce new chromosomes. The swap is applied to every cluster in the chromosome. This mutation was used in a GA proposed for the OCTSP in [12].

F. Survivor Selection

The survivor selection procedure selects the next generation from parents in the current population and the offspring that are generated by the genetic operators. The proposed MA implements a fitness-biased survivor selection method where all candidate individuals are ranked, and the fitter N individuals are chosen to form the population of the next generation.

III. Experiments and Results

This section presents the conducted experiments and their results during the evaluation of the performance of the proposed MA for the FCTSP. The MA was implemented in Java programming language and executed on a PC with 3.40 GHz Intel(R) Core(TM) i7-2600 CPU and 4.00 GB RAM under MS Windows 7 operating system. The algorithm is evaluated on a set of TSPLIB instances [1] as used in [6].

[1] http://comopt.ifi.uni-heidelberg.de/software/TSPLIB95/

TABLE I. PERFORMANCE OF THE MA VS. PERFORMANCE OF THE GRASP AND THE TLGA ON FCTSP INSTANCES

	TLGA		GRASP		MA	
Instance	%	t_{sec}	%	t_{sec}	%	t_{sec}
5-eil51	8.47	0.4	0	1	**0**	**0.138**
10-eil51	2.73	0.4	0	1	**0**	**0.009**
15-eil51	7.78	0.4	0	1	**0**	**0.012**
5-berlin52	1.44	0.6	0	1	**0**	**0.008**
10-berlin52	10.18	0.4	0	1	**0**	**0.007**
15-berlin52	14.69	0.4	0	1	**0**	**0.008**
5-st70	0.86	1.6	0	2.2	**0**	**0.033**
10-st70	1.88	1	0	1.8	**0**	**0.021**
15-st70	7.23	0.8	0	1.8	**0**	**0.016**
5-eil76	3.94	1.8	0.54	2.4	**0**	**0.298**
10-eil76	9.45	1.2	0.71	2.4	**0**	**0.043**
15-eil76	2.48	1.2	0.35	2.4	**0**	**0.093**
5-pr76	1.78	2	0.92	2.6	**0**	**0.087**
10-pr76	1.02	1.2	0.01	2.4	**0**	**0.029**
15-pr76	5.95	1.2	0.15	2.4	**0**	**0.127**
10-rat99	6.7	3.4	0.24	4.6	**0**	**0.042**
25-rat99	23.48	2.4	1.02	4.6	**0**	**0.042**
25-kroA100	5.69	2.2	0	4.8	**0**	**0.271**
50-kroA100	22.23	2.8	1.02	5	**0**	**0.054**
10-kroB100	2.49	3.8	0.07	4.8	**0**	**0.095**
50-kroB100	25.36	2.2	0.16	5	**0**	**0.393**
25-eil101	6.33	2.2	1.51	4.8	**0**	**2.731**
50-eil101	20.34	2.2	2.95	5	**0**	**0.402**
25-lin105	19.39	2	0.15	5.2	**0**	**0.199**
50-lin105	26.29	3.2	0.54	5.8	**0**	**0.216**

A. Parameter Settings

The proposed MA is controlled by a number of parameters that need to be set. The genetic operators include three parameters, namely crossover probability, mutation probability and population size, which are empirically set to 1.0, 0.2 and 20 respectively. The only control parameter of GLS is λ which is calculated as follows: $0.3 \times g^*(s)/|F^*|$, where $g^*(s)$ is the cost of the first local optimal and $|F^*|$ is the average number of features exhibited in a solution. The GLS stops when the best solution is not updated for a $maxIter$ number of consecutive penalizations (i.e. local search calls). The $maxIter$ is set as a function of the problem size, i.e. $maxIter$ is set to the number of the cities in the considered FCTSP instance. The stopping criterion for the MA is controlled by the maximum number of generations ($maxGene$) which is set to 5000.

B. Comparing the MA with State-of-the-Art Techniques

In [6], an algorithm based on GRASP with path-relinking was proposed and compared to the Two-level Genetic Algorithm (TLGA) [14], on a set of small size TSPLIB instances. These algorithms were encoded in the C programming language, and executed on a 2.83 GHz Intel Core 2 Quad with 4 cores and 8 GBs of RAM running the Ubuntu Linux OS (version 4.3.2-1). In order to evaluate the performance of the proposed MA, it is applied to the same set of instances, and make direct comparisons to the results of the GRASP and the TLGA algorithms as presented in [6].

Table I shows this comparative study between MA and both the TLGA and the GRASP methods. It gives, for each combination of algorithm and problem instance, the following performance measures: the average solution quality (%) which represents the mean excess above the best known solution in 20 runs, and the average computational time (t). The reported results for TLGA and GRASP are obtained from [6]. The number that precedes the TSPLIB instance name gives the number of clusters. The results suggest the following remarks:

- In terms of solution quality, the MA solves all the 25 instances to optimality, whereas the GRASP method constantly obtains the optimal solution only on 10 instances, and none of the instances was solved to optimality by the TLGA.
- In terms of the computational time, the comparison cannot be made directly as they were executed in different machines. However, the results show a significant gap between the MA and the other two methods, and the MA is capable to obtain better performance in much less time. On average, the amount of time that the MA requires to solve the considered FCTSP instances to optimality is less than 10% of that of the GRASP algorithm on all of the instances solved by the GRASP algorithm. Even on the unsolved instances by the GRASP method, the MA solves them to optimality in a very short time compared to the GRASP and TLGA methods.

These remarks reveal the outstanding performance of the proposed MA on these FCTSP instances in terms of both solution quality and computational time.

C. Evaluating the MA on Various Instances with Different Clusters

This experiment aims to evaluate the impact of the number and size of the clusters of the FCTSP on the performance of the MA. The experiment is designed as follows:

- A set of 20 TSPLIB instances are selected. They are of various names and sizes up to 318 cities.
- Since the best-known solution for each of these instances is already known, an optimal tour for each instance is obtained, and then the clusters are defined accordingly. Consequently, the best-known solution for the TSPLIB instance will be the same for its FCTSP counterpart.
- Nine FCTSP instances are derived from each TSPLIB instance by using different number of clusters (C) of almost equal sizes, where $C \in \{2, 4, 6, 8, 10, 20, 30, 40, 50\}$.
- On each FCTSP instance, the MA is applied while using similar experimental settings to the previous experiment.

The computational results reveal that the MA always solves to optimality all the FCTSP instances without any sensitivity to the number of clusters. Therefore, only the computational time required by the MA to solve each instance is reported in Table II. The results for selected instances are plotted in Fig. III-C. The results show that the hardness of the FCTSP instance for the MA to solve an instance to optimality grows as the number of clusters shrinks. For example, the complete time required by the MA to find optimal solutions on $kroA200$ with 2, 4, 10 and 50 clusters are 16.5, 3.2, 1.9 and 0.4 seconds, respectively.

IV. CONCLUSION

In this study, a new memetic algorithm based on the GA, GLS and 2-Opt algorithms is proposed for solving the free

TABLE II. AVERAGE COMPUTATIONAL TIME REQUIRED BY THE MA TO SOLVE SOME TSPLIB INSTANCES WITH VARIOUS CLUSTERS TO OPTIMALITY

Instance	Clusters								
	2	4	6	8	10	20	30	40	50
kroA100	0.095	0.053	0.037	0.03	0.024	0.028	0.032	0.03	0.028
kroB100	0.063	0.035	0.027	0.022	0.039	0.021	0.024	0.025	0.026
rd100	0.061	0.04	0.037	0.025	0.022	0.02	0.019	0.025	0.027
eil101	0.137	0.033	0.041	0.034	0.023	0.023	0.027	0.034	0.034
lin105	0.186	0.096	0.052	0.042	0.036	0.028	0.031	0.037	0.03
pr107	0.184	0.272	0.112	0.123	0.093	0.105	0.037	0.191	0.153
pr124	0.101	0.051	0.036	0.03	0.028	0.023	0.023	0.026	0.031
bier127	0.234	0.201	0.344	0.301	0.22	0.125	0.18	0.139	0.23
pr144	0.286	0.126	0.24	0.068	0.056	0.038	0.034	0.047	0.063
kroA150	0.854	0.283	0.164	0.105	0.084	0.055	0.064	0.094	0.064
kroB150	1.911	0.718	0.213	0.351	0.273	0.084	0.081	0.122	0.085
ch150	0.756	0.154	0.118	0.085	0.093	0.059	0.07	0.068	0.055
pr152	0.375	0.304	0.273	0.069	0.101	0.074	0.091	0.095	0.085
rat195	1.163	0.853	0.127	0.094	0.096	0.108	0.095	0.086	0.131
kroA200	16.564	3.286	3.991	2.759	1.913	0.917	0.276	0.317	0.416
kroB200	93.54	17.229	3.688	3.603	0.989	0.84	0.592	0.098	0.355
ts225	5.853	1.963	0.65	0.845	0.376	0.141	0.22	0.16	0.182
a280	1.763	0.997	0.533	0.472	0.895	0.264	0.135	0.192	0.252
pr299	69.791	24.579	17.487	3.017	7.634	2.076	0.985	1.715	1.87
lin318	34.536	12.599	7.977	7.249	5.385	1.049	0.701	1.734	1.03

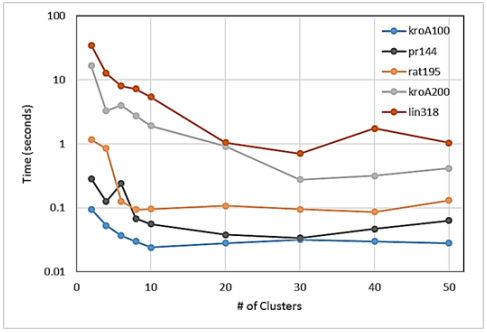

Fig. 2. Average computational time required by the MA as a function of the number of clusters, on selected TSPLIB instances.

clustered travelling salesman problems. In this method, the GA, which implements the sequential constructive crossover and the reciprocal exchange mutation operator, is used for global search; while the GLS algorithm that sits on top of the 2-Opt heuristic is used for local search. The performance of this proposed method is evaluated in terms of solution quality and speed on a set of TSPLIB, with comparison to a GA and GRASP methods. The impacts of the different number of clusters on the performance of the proposed method are also analyzed. The obtained experimental results reveal the outstanding performance of the proposed memetic algorithm which solves to optimality all the instances used in this study in a reasonable amount of time.

REFERENCES

[1] James A. Chisman. The clustered traveling salesman problem. *Computers and Operations Research*, 2(2):115 – 119, 1975.

[2] Hassan Ghaziri and Ibrahim H Osman. A neural network algorithm for the traveling salesman problem with backhauls. *Computers and Industrial Engineering*, 44(2):267 – 281, 2003.

[3] A. Weintraub, J. Aboud, C. Fernandez, G. Laporte, and E. Ramirez. An emergency vehicle dispatching system for an electric utility in chile. *Journal of the Operational Research Society*, 50(7):690–696, 1999.

[4] G. Laporte and U. Palekar. Some applications of the clustered travelling salesman problem. *Journal of the Operational Research Society*, 53(9):972–976, 2002.

[5] F.C.J. Lokin. Procedures for travelling salesman problems with additional constraints. *European Journal of Operational Research*, 3(2):135 – 141, 1979.

[6] Mário Mestria, Luiz Satoru Ochi, and Simone de Lima Martins. GRASP with path relinking for the symmetric euclidean clustered traveling salesman problem. *Computers and Operations Research*, 40(12):3218 – 3229, 2013.

[7] Michel Gendreau, Gilbert Laporte, and Jean-Yves Potvin. Heuristics for the clustered traveling salesman problem. Technical Report CRT-94-54, Centre de Recherche sur les Transports, Université de Montréal, Montreal, Canada, 1994.

[8] T Aramgiatisiris. An exact decomposition algorithm for the traveling salesman problem with backhauls. *Journal of Research in Engineering and Technology*, 1:151–164, 2004.

[9] N. Guttmann-Beck, R. Hassin, S. Khuller, and B. Raghavachari. Approximation algorithms with bounded performance guarantees for the clustered traveling salesman problem. *Algorithmica*, 28(4):422–437, 2000.

[10] Shoshana Anily, Julien Bramel, and Alain Hertz. A 53-approximation algorithm for the clustered traveling salesman tour and path problems. *Operations Research Letters*, 24(12):29 – 35, 1999.

[11] Gilbert Laporte, Jean-Yves Potvin, and Florence Quilleret. A tabu search heuristic using genetic diversification for the clustered traveling salesman problem. *Journal of Heuristics*, 2(3):187–200, 1997.

[12] Zakir Hussain Ahmed. The ordered clustered travelling salesman problem: A hybrid genetic algorithm. *The Scientific World Journal*, 2014:1–13, 2014.

[13] Jean-Yves Potvin and François Guertin. *The Clustered Traveling Salesman Problem: A Genetic Approach*, pages 619–631. Springer US, Boston, MA, 1996.

[14] Chao Ding, Ye Cheng, and Miao He. Two-level genetic algorithm for clustered traveling salesman problem with application in large-scale tsps. *Tsinghua Science and Technology*, 12(4):459 – 465, 2007.

[15] Christos Voudouris, Edward Tsang, and Abdullah Alsheddy. Guided local search. *Handbook of metaheuristics*, pages 321–361, 2010.

[16] Christos Voudouris and Edward Tsang. Guided local search and its application to the traveling salesman problem. *European journal of operational research*, 113(2):469–499, 1999.

[17] Michel Gendreau and Jean-Yves Potvin. *Handbook of metaheuristics*, volume 146. Springer, 2010.

[18] Pablo Moscato and Carlos Cotta. *A Modern Introduction to Memetic Algorithms*, pages 141–183. Springer US, Boston, MA, 2010.