European Journal of Operational Research xxx (xxxx) xxx

Contents lists available at ScienceDirect

European Journal of Operational Research

journal homepage: www.elsevier.com/locate/ejor

Invited Review

A comprehensive survey on the generalized traveling salesman problem

Petrică C. Pop[a,*], Ovidiu Cosma[a], Cosmin Sabo[a], Corina Pop Sitar[b]

[a] Technical University of Cluj-Napoca, North University Center of Baia Mare, Department of Mathematics and Computer Science, Baia Mare, Romania
[b] Technical University of Cluj-Napoca, North University Center of Baia Mare, Department of Economics, Baia Mare, Romania

ARTICLE INFO

Article history:
Received 18 April 2022
Accepted 14 July 2023
Available online xxx

Keywords:
Combinatorial optimization
Traveling salesman problem
Generalized traveling salesman problem
Mathematical formulations
Heuristic and metaheuristic algorithms

ABSTRACT

The generalized traveling salesman problem (GTSP) is an extension of the classical traveling salesman problem (TSP) and it is among the most researched combinatorial optimization problems due to its theoretical properties, complexity aspects and real-life applications in various areas: location-routing problems, material flow design problem, distribution of medical supplies, urban waste collection management, airport selection and routing the courier airplanes, image retrieval and ranking, digital garment manufacturing, etc. Even though the importance of this combinatorial optimization problem was highlighted in several publications and there were developed several methods for solving it, there is no survey dedicated to the GTSP. The scope of this paper is to close this gap by providing a comprehensive survey on mathematical formulations, solution approaches and latest advances regarding the GTSP. The paper is organized around the following issues: problem definition, variations and related problems, real-life applications of the GTSP, mathematical formulations, solution approaches designed for solving the investigated problem, datasets, computational results and comparative analysis of the performance of the existing state-of-the-art algorithms. Additionally, we discuss certain open problems and potential research directions.

1. Introduction

The traveling salesman problem (TSP) has been recorded in the history of combinatorial optimization since 1930. Merrill M. Flood first provided a mathematical formulation of the problem when he came across it while solving a school bus routing problem (Flood, 1948). Given a set of cities, the TSP looks for the least-cost tour that visits all of the cities. It can be modeled as an undirected graph whose vertices are the cities and the edges are the paths between the cities. The TSP is one of the most investigated problems in optimization, being used as a benchmark for several optimization techniques. There exist several aspects which justify the popularity and relevance of the TSP. Although the problem is computationally difficult there are various efficient exact or heuristic algorithms which make it solvable when dealing with large size graphs with up to millions of nodes (https://www.math.uwaterloo.ca/tsp/world/). Many exact solution approaches have been developed, making it possible to provide optimal solutions to instances having tens of thousands of cities (Applegate, Bixby, Chvatal, Cook, & Problem, 2006), and several heuristic algorithms have been proposed, approximating within a small fraction of 1% of optimality even problems having millions of cities (https://www.math.uwaterloo.ca/tsp/world/). The TSP has many real-life applications (Matai, Singh, & Mitall, 2010) in various areas such as planning, logistics, machine scheduling problems, the manufacture of microchips, and DNA sequencing.

Based on various research directions in this field, see for example Chisman (1975), Garg & Ravi (2000), Henry-Labordere (1969), one can draw the conclusion that there is high appeal on the TSP extensions and variants. The most important extensions of the TSP that have been investigated are the vehicle routing problem (Laporte, 1992), the traveling salesman problem with time windows (da Silva & Urrutia, 2010), the Steiner traveling salesman problem (Rodríguez-Pereira, Fernández, Laporte, Benavent, & Sykora, 2019), the selective traveling salesman problem (Laporte & Martello, 1990), the multi-objective traveling salesman problem (Psychas, Delimpasi, & Marinakis, 2015), the multiple traveling salesman problem (Cheikhrouhou & Khoufi, 2021), the bottleneck TSP (Garfinkel & Gilbert, 1978), the prize collecting traveling salesman problem (Balas, 1989), the clustered traveling salesman problem (Chisman, 1975), and the generalized traveling salesman problem (Henry-Labordere, 1969; Saskena, 1970; Srivastava, Kumar, Garg, & Sen, 1969).

* Corresponding author.
E-mail address: petrica.pop@mi.utcluj.ro (P.C. Pop).

https://doi.org/10.1016/j.ejor.2023.07.022

In this paper, we investigate an extension of the classical TSP, called the generalized traveling salesman problem (GTSP). We define the GTSP as follows. Given an undirected graph $G = (V, E)$ whose vertices are divided into a given number of clusters, denoted by $C_1, C_2, \ldots, C_k$, the GTSP searches for the shortest Hamiltonian cycle visiting a collection of vertices with the property that exactly one vertex from each cluster is visited. The GTSP was introduced independently by Henry-Labordere (1969), Saskena (1970) and Srivastava et al. (1969), in relation to real-world applications including record balancing problems encountered in computer design and the routing of clients through welfare agencies.

We observe that the TSP is a particular case of the GTSP, where each cluster is a singleton (i.e., it contains exactly one vertex).

Lien, Ma, & Wah (1993) pointed out that the GTSP provides a more accurate model than the TSP for several real-world problems having a hierarchical structure (i.e., a structure with multiple levels (clusters)). In addition, the GTSP may include the case when the vertices are partitioned into a given number of possible intersecting clusters. Furthermore, as we pointed out, the TSP is a particular case of the GTSP, and the application areas of the GTSP are more numerous than those of the TSP. Therefore, we conclude that the GTSP gives a more ideal modeling tool in comparison to the TSP for many real problems.

The GTSP belongs to the class of generalized network design problems. This class of problems generalizes in a natural way classical network design problems (NDPs), and has the subsequent primary characteristics. The vertices of the graph associated with the NDP are partitioned into a given number of sets of vertices, called clusters and, the feasibility restrictions of the initial optimization problem are expressed in relation to the clusters rather than as individual vertices. Some interesting and challenging problems that belong to this class of optimization problems are the generalized minimum spanning tree problem (Myung, Lee, & Tcha, 1995; Pop, 2020; Pop, Matei, Sabo, & Petrovan, 2018b), the generalized vehicle routing problem (Cosma, Pop, & Sitar, 2022; Ghiani & Improta, 2000; Pop, Matei, & Sitar, 2013), the partition graph coloring problem (Demange, Monnot, Pop, & Ries, 2014; Fidanova & Pop, 2016), the clustered shortest-path tree problem (Cosma, Pop, & Zelina, 2020; Cosma, Pop, & Zelina, 2021b; Petrovan, Pop, Sabo, & Zelina, 2023b), the clustered minimum routing cost problem (Thang, Long, Hoang, & Binh, 2021), etc. For further information on this class of optimization problems, we refer the reader to Dror & Haouari (2000), Feremans, Labbé, & Laporte (2003) and Pop (2012).

The rest of the paper is organized as follows. In Section 2, we provide the formal definition of the GTSP and present the variants and related problems to the investigated problem. In Section 3, we describe the most important real-world applications of the GTSP, while in Section 4, we present and analyze integer and mixed integer formulations of the GTSP. In Section 5, we provide a summary of exact and approximate algorithms, transformation methods and heuristic and metaheuristic algorithms for solving the GTSP. In Section 6, we describe the existing datasets used for testing the performance of different solution approaches for solving the GTSP and provide a comparative analysis of the performance of existing state-of-the-art algorithms. In Section 7, we present closing comments, along with extra remarks and future research areas.

2. Defining the GTSP, variants and related problems

This part of the paper is dedicated to explicitly defining the GTSP as a graph theoretic model. We consider an undirected, connected and weighted graph $G = (V, E)$ that consists of a set of n vertices $V = \{1, 2, \ldots, n\}$ and a set of edges $E = \{e_1, \ldots, e_m\}$ where:

$$E \subseteq \{\{i, j\} \mid i, j \in V \text{ and } i \neq j\},$$

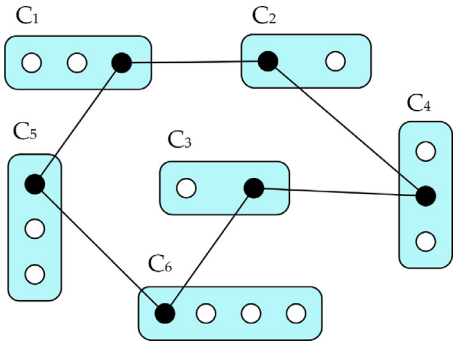

Fig. 1. A feasible solution of the GTSP.

and the cost function $c : E \to R_+$, which associates to every edge $e = \{i, j\} \in E$ of the graph, a positive number $c(e) = c_e = c_{ij} \in R_+$, called the cost of the edge e. The associated cost matrix is denoted by W. If the triangle inequality holds, then the problems are referred to as Euclidean, otherwise there are referred to as non-Euclidean problems.

We divide the set of vertices V into k mutually exclusive nonempty subsets, each containing a subset of the vertices from G, which are called clusters and which are denoted by $C_1, \ldots, C_k$, meaning that the following conditions hold:

1. $C_1 \cup C_2 \cup \ldots \cup C_k = V$
2. $C_l \cap C_p = \emptyset,\ \forall\, l, p \in \{1, \ldots, k\}$ with $l \neq p$.

The edges of the graph are divided into two classes: intra-cluster edges that link vertices from the same cluster and inter-cluster edges that link vertices from different clusters.

The aim of the GTSP is to look for a shortest Hamiltonian tour (with respect to visiting all clusters of the graph). According to the type of the Hamiltonian tour, there are two different versions of the GTSP:

- the Hamiltonian tour visiting every cluster exactly once (i.e., exactly one vertex from each cluster is visited);
- the Hamiltonian tour passing through each cluster at least once (i.e., at least one vertex from each cluster is visited), denoted by (L-GTSP).

In the case of Euclidean problems, the optimal solution of the L-GTSP contains exactly one vertex from every cluster and the GTSP and L-GTSP have the same optimal solution.

In this paper, we focus on the first version, namely the vertices of the graph are split into a given set of vertex-disjoint clusters, the cost matrix is symmetric and exactly one vertex from every cluster is selected.

An illustration of the GTSP with 17 vertices which are partitioned into six clusters $C_1, \ldots, C_6$, is displayed in Fig. 1:

Such Hamiltonian tours, visiting a subset of vertices with the property that exactly one vertex is visited from every cluster, are called generalized Hamiltonian tours.

The GTSP is an NP-hard optimization problem because it includes the classical TSP as a particular case when all clusters are singletons.

If the cost matrix C is asymmetric then the GTSP is called the asymmetric GTSP. This variant was investigated by Laporte, Mercure, & Norbert (1985, 1987). Their first paper (Laporte et al., 1985) describes a Lagrangian relaxation approach in which the subtour elimination restrictions and the node flow conservation restrictions of an integer linear program are relaxed in a Lagrangian fashion by summing them to the objective function. The resulting dual program is solved as a network problem to obtain lower bounds. The same authors (Laporte, Mercure, & Norbert, 1987) proposed an approach based on a modification of the branch-and-bound algorithm

developed by Carpaneto & Toth (1980) in the case of the asymmetric TSP. In this way, the authors were able to optimally solve Euclidean and non-Euclidean problems involving graphs with up to 100 vertices. Noon & Bean (1991) described an optimal method that was able to solve asymmetric GTSP instances with up to 104 nodes. Their approach has three procedures. The first one uses Lagrangian relaxation to compute lower bounds of the problem and uses a heuristic to compute upper bounds. The second one uses the computed lower and upper bounds to determine and eliminate non-optimal arcs and vertices. Finally, the third procedure uses implicit enumeration to discover an optimal solution.

In another variant of the GTSP (for example, see Lien et al., 1993), the vertices are partitioned into possibly intersecting clusters. In this case, proved by Lien et al. (1993), every graph G with possibly intersecting clusters may be converted into a graph G' that has no overlapping clusters such that each generalized Hamiltonian tour in G' may be converted into a generalized Hamiltonian tour in G of no greater cost. Each optimal generalized Hamiltonian tour in G' may be converted into an optimal generalized Hamiltonian tour in G.

The clustered traveling salesman problem (CTSP) is a variant of the GTSP introduced by Chisman (1975). The CTSP finds a shortest Hamiltonian tour visiting all the vertices of the graph, with the property that the vertices belonging to the same cluster are visited consecutively. Laporte & Palekar (2002) described real-world applications of the CTSP arising in vehicle routing problems, manufacturing, computer operations and computer program restructuring, and examination timetabling.

The generalized traveling salesman problem with time windows (GTSP-TW) was investigated by Yuan, Cattaruzza, Ogier, & Semet (2020b), Yuan, Cattaruzza, Ogier, Rousselot, & Semet (2020a) motivated by applications in the field of delivery services. The GTSP-TW is defined on directed graphs with the set of vertices divided into clusters with the following properties. One cluster includes only the depot and for every vertex we associate a time window, during which the visit must take place if the corresponding vertex is visited. The goal of the GTSP-TW is to find the shortest Hamiltonian tour beginning and ending at the depot so that each cluster is visited exactly once and the time constraints are fulfilled, i.e., for each cluster the selected vertex is visited within its time window.

A more general variant of the GTSP known as the selective generalized traveling salesman problem (SGTSP) was investigated by Derya, Dinler, & Kececi (2020). The objective of this variant is to find the most profitable tour within a given threshold of the tour's duration, which is made up of a subset of clusters and a subset of vertices belonging to the selected clusters, visited on the tour. This variant was motivated by a practical application connected with the distribution of mass products. Derya et al. (2020) described eight mixed integer programming models of the SGTSP and their performance was assessed on a set of instances by conducting 4608 experiments.

Tang & Hooks (2007) investigated a probabilistic GTSP characterized by the fact that each of the clusters has a set of vertices (customers), and for each cluster C_l, $1 \leq l \leq k$ there is a probability P_l that at least one vertex will be present within C_l. The authors presented an exact method based on the integer L-shaped technique which was able to solve small and medium size problems optimally, and three tour construction heuristic algorithms. The Max-Min insertion heuristic algorithm provided good solutions in very reasonable computation times.

The precedence constrained generalized traveling salesman problem (PCGTSP) was introduced by Salman, Ekstedt, & Damaschke (2020) and combines the GTSP and the Sequential Ordering Problem (SOP). The authors presented a branch-and-bound algorithm that was tested on a set consisting of 23 synthetic PCGTSP instances and 5 industrial cases.

Morán-Mirabal, González-Velarde, & Resende (2014) investigated the family traveling salesman problem (FTSP), which was defined as follows. Given a graph whose vertices are divided into a predefined number of sets of vertices, called families, determine a minimum cost tour visiting an a priori known number of vertices from each family. In this problem, a family is equivalent to a cluster in the GTSP. The investigation and consideration of the FTSP was justified due to the existence of a practical application, namely the order picking problem in warehouses where items of the same type are stored in different warehouses or in different places in the same warehouse. The authors developed an integer programming formulation of the FTSP and generated solutions in CPLEX 11 for small instances. Moreover, they developed two randomized heuristic algorithms, a biased random-key genetic algorithm and a greedy randomized adaptive search procedure (GRASP) with evolutionary path-relinking (Morán-Mirabal et al., 2014). Bernardino & Paias (2018) presented several compact and non-compact models of the problem, while Pop, Matei, & Pintea (2018a) described an innovative technique to solve the FTSP. They split the problem into two smaller subproblems operating at the macro and micro levels, and solved them individually. The macro level subproblem is aimed at providing a collection of tours visiting the families while employing a classical genetic algorithm (GA) and a diploid GA (i.e., GA with individuals consisting of two coupled chromosomes Petrovan, Matei, & Pop, 2023a). The micro level subproblem is aimed at finding the minimum-cost tour, associated with each generated tour at the macro level, that visits a given number of vertices belonging to each family.

Sundar & Rathinam (2016) investigated the generalized multiple depot traveling salesmen problem (GMDTSP). They were motivated by several applications in network design, health-care logistics and scheduling. The problem was defined as follows. Given a set of customers divided into a predefined number of clusters and a set of K depots, search for a collection of at most K cycles having the following properties. Each cycle begins and ends at a certain depot, at least one customer from each cluster is visited by some cycle, and the total costs of the collection of cycles are minimized.

Hadjicharalambous, Pop, Pyrga, Tsaggouris, & Zaroliagis (2007) considered an interesting issue in railway optimization, which was named the railway traveling salesman problem (RTSP). In this problem, a salesman making use of the railway network desires to pass through a given number of cities to accomplish his/her task, starting and finishing at the same place, and minimize the total traveling time. In order to model the problem, the authors introduced the time-expanded digraph G built from the timetable information. The vertices of this digraph are associated with every time event (departure or arrival) at a station, and edges among vertices display either elementary links among the two events, such as served by a train that does not stop in-between, or waiting within a station. The cost of an edge represents the time difference between the time events associated with its endpoints. In spite of their similarity, the RTSP differs from the GTSP in at least two aspects. First, the RTSP starts from a specific vertex within a specific cluster, while the GTSP starts from any vertex within any cluster. Second, the GTSP visits all of the clusters, while the RTSP visits only a subset of them.

An extension of the GTSP considers automated storage and retrieval systems (logistics inside the warehouse) and drone-assisted parcel delivery service (logistics outside the warehouse). This extension was recently investigated by Baniasadi, Foumani, Smith-Miles, Ejov, & Logistics (2020) and it was named the clustered generalized traveling salesman problem (CGTSP). The CGTSP extends the GTSP by considering a partition of the clusters in subclusters and looking for a shortest Hamiltonian tour with the property that the tour visits exactly one vertex from each subcluster and within every cluster all the selected vertices must be visited consecutively.

Khan & Maiti (2018) investigated the GTSP in fuzzy environments, by modelling the costs as fuzzy numbers. The authors described a novel hybrid algorithm obtained by combining a swap sequence-based particle swarm optimization (PSO) method with a genetic algorithm that solved the GTSP in a crisp environment (i.e., characterized by a fixed traveling cost from one city to another) and a fuzzy environment (i.e., characterized by an imprecise traveling cost).

3. Real-life applications of the GTSP

The GTSP offers a promising way to model various real-world applications. Its hierarchical structure and versatility offers accurate models for several practical applications.

The GTSP first appeared in the scientific literature in the late 1960s and early 1970s motivated by the real-world applications in record balancing problems encountered in computer design (Henry-Labordere, 1969) and the routing of clients through welfare agencies (Saskena, 1970). For example, in the case of the problem of routing the clients through welfare agencies, a client may wish to get a number of services provided by welfare agencies. The cluster C_l is defined as the set of agencies offering the lth type of service. If each welfare agency offers only one type of service, then the clusters are disjoint. Thus, this problem can be viewed as a GTSP.

In the 1980s, researchers considered applications in the field of postal routing that determined the optimal route of vehicles used to pick up the contents of the post boxes in an urban area (see Bovet, 1983 and Rousseau, 1985). In these applications, a postal vehicle has to be routed in order to collect mail from a given number of urban mailboxes. The first application considered by Bovet (1983) aims to find the location of the post boxes within a city in order to deliver services to the customers. The post boxes may be located at street intersections, having different alternatives depending on which side of the street they are going to be positioned. Their precise location is determined by means of a GTSP, where the clusters C_l, $1 \le l \le k-1$ correspond to the set of potential locations of post box l, and cluster C_k has only one vertex, and represents the sorting office. The second application investigated by Rousseau (1985) assumes that the location of the post boxes is known a priori and aims to determine the collection routes. The difficulty of this problem comes from the fact that the driver of the post vehicle is forbidden to cross a street on foot to collect the correspondence from mailbox. For a post box located on a street corner, the collection can be done only if the postal van stops in one of the two streets adjacent to the corner. In order to solve this optimization problem, Rousseau (1985) duplicated the post boxes located at street corners and allocated a different edge to each copy of the post box. Then the problem can be stated as a GTSP in which the clusters contain two vertices for post boxes located at a street intersection and one vertex corresponding to the post box otherwise.

Noon (1988) presented different applications of the GTSP including warehouse order picking with multiple stock locations, airport selection and routing courier airplanes and certain categories of flexible manufacturing scheduling.

Laporte, Asef-Vaziri, & Sriskandarajah (1996) described four novel real-world applications that can be modeled as a GTSP.

1. Location-routing problems. These problems are encountered in several practical contexts where we need to locate facilities and to find vehicle routes visiting these facilities and a number of other sites. GTSP applications result when exactly one vehicle is involved, as in the case of the covering tour problem (CTP). Given a graph $H = (V \cup W, E)$, where V is a set of vertices, W is a set of vertices that must be covered, the set of edges $E = \{\{i, j\} \mid i < j,\ i, j \in V \cup W\}$ and the cost function $c : E \to R_+$, then the CTP looks for a minimum length Hamiltonian cycle on a subset of V such that every vertex of W is within a prespecified distance M from any vertex of the cycle. We refer to Gendreau, Laporte, & Semet (1997) for more information on this problem. Choosing the clusters $C_l = \{i \in V \mid c_{il} \le M\}$, for every $l \in W$, then the CTP may be modeled as a GTSP on the subgraph G of the graph H induced by $\cup_{l \in W} C_l$.
2. Material flow system design. This is a crucial problem in production settings. Given the topology of a production plant divided into k polygonal zones, denoted by $Z_1, \ldots, Z_k$, and containing only 90° and 270° angles, the objective is to design a minimal length loop along the edges of the zones. Two variants of this problem were considered. In the first variant, the loop must contain at least one vertex from each zone. In the second variant, the loop must contain at least one edge from each zone. To model these two variants as a L-GTSP, Laporte et al. (1996) noticed that degrees of all the zone vertices are equal to 2, 3 or 4.
   In the case of the first variant of the problem, vertices of degree 2 are ignored. Because, if a vertex of degree 2 belongs to the loop, its two neighbors obviously also belong to it. We define the clusters C_l corresponding to the zones Z_l, $1 \le l \le k$, as the set of vertices of the zone Z_l having a degree at least equal to 3. Then the problem of determining a minimal length loop is reduced to solving a L-GTSP over the clusters $C_1, \ldots, C_k$.
   In the case of the second variant of the problem, the vertices of degree 2 are ignored again. If the loop crosses a vertex of degree 3, evidently, it will contain an edge from each of the three zones to which it belongs. Therefore, if all the vertices have degree 3, then a loop will contain at least one edge from each zone. In the case of vertices of degree 4, it may be possible that a loop contains a vertex from a zone, but not necessarily an edge. In this case, Laporte et al. (1996) considered all the edges $\{i, j\}$ with the end points having degree 4. Then, they introduce an additional vertex on the edge $\{i, j\}$ and include it in the clusters C_l and C_p, where Z_l and Z_p are the zones that contain the edge $\{i, j\}$. Finally, all the vertices of degree 4 are removed from all the clusters, and again the problem of determining a minimum length loop is reduced to solving a L-GTSP over the clusters $C_1, \ldots, C_k$.
3. Stochastic vehicle routing. Given a depot v_1 and a set of customers $v_i, i = 2, \ldots, k$ with stochastic demands, we need to design routes visiting the customers to deliver (collect) goods, while meeting the capacity constraints of the vehicles. First, a route is established by visiting the customers and a vehicle visits customers on the route while not exceeding the vehicle capacity. When this happens, a route failure occurs, and a recourse action must follow. The objective of the problem is to generate a route for visiting the customers such that the cost of the resulting route and the expected recourse cost are minimized. This problem can be modeled as a GTSP by considering the clusters $C_l = \{v_l\}$ for $l = 1, \ldots, k$ and the cluster C_{k+1} containing two vertices v_{k+1} and v_{k+2} having the meaning: if v_{k+1} is visited from a vertex belonging to $V \setminus \{v_1\}$ then the solution contains a preventive break (i.e., a point (customer) along the planned route, in which the vehicle would return to the depot to unload before resuming collections, even if its capacity is not reached) and if v_{k+2} is visited from a vertex belonging to $V \setminus \{v_1\}$ then the solution contains no preventive break, but failure can occur at the last vertex. Only one of the vertices v_{k+1} and v_{k+2} must be visited because risking failure and planning a preventive break are mutually exclusive scenarios.
4. Arc routing problem. Given a graph $G = (V, A \cup E)$, where V is the set of vertices, A is the set of arcs, and E is the set of edges, an arc routing problem aims to find a least-cost traversal of

a given subsets of arcs or edges of G (for further information, Ávila, Corberán, Plana, Sanchis, & problem, 2016). In the case of completely directed arc routing problems, Beltrami & Bodin (1974) showed how to transform these problems into equivalent vertex routing problems. In the case when G is not completely directed, Laporte et al. (1996) showed how an arc routing problem may be transformed into a vertex routing problem. Their approach works as follows. First, we replace each edge $\{v_i, v_j\}$ of G by two arcs (v_i, v_j) and (v_j, v_i), then construct an auxiliary graph $H = (W, B)$, where W contains a vertex w_{ij} for every arc $(v_i, v_j) \in A$, $B = \{(w_{ij}, w_{pq}) \mid w_{ij}, w_{pq} \in W, w_{ij} \neq w_{pq}\}$, and the cost of the arc (w_{ij}, w_{pq}) is the cost of a shortest path from v_j to v_p. Second, define the clusters as follows. If (v_i, v_j) was an arc in G, then $C_{ij} = \{w_{ij}\}$ and if $\{v_i, v_j\}$ was an edge of G, then $C_{ij} = \{w_{ij}, w_{ji}\}$. In this way, the original arc routing problem was transformed into a GTSP fulfilling the constraint that exactly one vertex is visited from each cluster.

An interesting application of the GTSP in the field of image retrieval and ranking was described by Wang, Ishwar, Konrad, Gazen, & Saboo (2012). The authors modeled the problem of comparing sets of different images in order to determine representative images, one from each set of images, such that the selected images are coherent in a given sense. They proposed an efficient solution approach by decomposing the problem into two subproblems.

As we have already mentioned, several variants and problems related to the GTSP have been investigated in the literature. These problems are motivated by the potential real-world applications in various fields such as modern logistics, drone-assisted parcel delivery service, robotic automated storage and retrieval systems, railway optimization, network design, and delivery and collection services.

4. Mathematical formulations of the GTSP

Consider an undirected graph $G = (V, E)$ and a subset of vertices $S \subset V$. We define and denote by $E(S)$ the set of edges belonging to E with both vertices in S and the cutset denoted by $\delta(S)$ as the set of edges belonging to E with the property that exactly one vertex belongs to S, that is,

$$E(S) = \{e = \{i, j\} \in E \mid i \in S, j \in S\} \text{ and } \delta(S) = \{e = \{i, j\} \in E \mid i \in S, j \notin S\}.$$

With the scope of modeling the GTSP, we consider the following binary decision variables

$$x_e = x_{ij} = \begin{cases} 1 & \text{if the edge e} = \{i, j\} \in E \text{ is selected in the solution} \\ 0 & \text{otherwise} \end{cases}$$

$$z_i = \begin{cases} 1 & \text{if the vertex } i \text{ is selected in the solution} \\ 0 & \text{otherwise} \end{cases}$$

where $i, j \in V$ and $l, p \in \{1, \ldots, k\}$.

We use the following notation: $\forall\ E' \subseteq E$, $x(E') = \sum_{\{i,j\} \in E'} x_{ij}$, and $\forall\ V' \subseteq V$, $z(V') = \sum_{i \in V'} z_i$.

A feasible solution of the GTSP can be viewed as a subgraph with no subtours (cycles), with one vertex chosen from every cluster, satisfying the degree restrictions and with k edges linking all the clusters. Consequently, the GTSP may be expressed as the following 0–1 linear program.

$$\min \sum_{e \in E} c_e x_e$$

$$s.t.\ \ z(C_p) = 1 \qquad \forall\ p \in \{1, \ldots, k\} \tag{1}$$

$$x(\delta(v)) = 2z_v \qquad \forall\ v \in V \tag{2}$$

$$x(E(S)) \leq z(S - i)\ \ \forall\ S \subset V \text{ with } 2 \leq |S| \leq n - 2,\ \forall\ i \in S \tag{3}$$

$$x_e \in \{0, 1\} \qquad \forall\ e \in E \tag{4}$$

$$z_i \in \{0, 1\} \qquad \forall\ i \in V. \tag{5}$$

For simplicity, we use the notation $S - i$ instead of $S \setminus \{i\}$. As it is obvious from the above model, constraints (1) ensure that we choose exactly one vertex from every cluster, constraints (2) are called the degree equations and they guarantee that if a vertex belongs to the solution its degree has to be equal to two. We use the clique packing constraints (3) in order to exclude all the cycles. We observe an exponential number of constraints of type (3) and these are called *generalized subtour elimination constraints*. Constraints (4) and (5) are the 0–1 integer constraints describing the selection of vertices and edges within the solution.

This 0–1 linear programming formulation of the GTSP was presented by Pop (2007) and it was called the *generalized subtour elimination formulation*.

The subtour elimination constraints (3) may be replaced by the connectivity constraints (6). This results in the 0–1 linear programming formulation described by Fischetti, Gonzales, & Toth (1997) known as the *generalized cutset formulation*

$$\begin{aligned} &\min \sum_{e \in E} c_e x_e \\ &s.t.\ \ (1), (2), (4), (5) \text{ and} \\ &\quad x(\delta(S)) \geq 2(z_i + z_j - 1)\ \forall\ i \in S \subset V, j \notin S,\ 1 \leq |S| \leq n - 2 \end{aligned} \tag{6}$$

We observe that this 0–1 linear programming formulation of the GTSP also includes an exponential number of constraints.

These two formulations of the GTSP use only variables associated with the vertices and edges of the graph G. Their main disadvantage is that they have an exponential size.

By using a polynomial number of supplementary constraints and variables, it is possible to ensure that the solution does not contain subtours. These have been called compact representations of the subtour elimination constraints.

By considering a polynomial number of auxiliary variables and constraints, Kara, Guden, & Koc (2012) provided two compact mixed 0–1 linear programming formulations of the GTSP. In order to provide compact formulations of the GTSP, Pop (2007) introduced supplementary flow variables in addition to the natural binary decision variables associated with the edges and vertices. The idea is to send a flow between the vertices of the network. An edge variable is defined to show if the edge is capable of transporting any flow or not. Even though the edges are undirected, the flow variables are going to be directed. This means that for every edge $\{i, j\} \in E$, we have flow in both directions, i.e., from i to j and from j to i. Pop (2007) described a single commodity formulation, a multicommodity formulation and a bidirectional flow formulation.

We present here the single commodity formulation, in which the source cluster C_1 sends one unit of flow to every other cluster. Let f_{ij} be the flow on the edge from i to j. This leads to the following formulation of the GTSP

$$\begin{aligned} &\min \sum_{e \in E} c_e x_e \\ &s.t.\ \ (1), (2), (4), (5) \text{ and} \\ &\quad \sum_{e \in \delta^+(i)} f_e - \sum_{e \in \delta^-(i)} f_e = \begin{cases} (k-1)z_i & \text{for } i \in C_1 \\ -z_i & \text{for } i \in V \setminus C_1 \end{cases} \end{aligned} \tag{7}$$

$$f_{ij} \leq (k-1)x_e \qquad \forall\, e=\{i,j\} \in E \tag{8}$$

$$f_{ji} \leq (k-1)x_e \qquad \forall\, e=\{i,j\} \in E \tag{9}$$

$$f_{ij}, f_{ji} \geq 0 \qquad \forall\, e=\{i,j\} \in E. \tag{10}$$

In this compact mixed 0–1 linear programming formulation of the GTSP, constraints (7) guarantee that $k-1$ units of a single commodity flow into cluster C_1 and 1 unit flows out of each of the other clusters. Constraints (7) are known as mass balance equations. These equations imply that the network defined by any feasible solution is connected. Together with the other constraints of the formulation, it results that every feasible solution is a generalized Hamiltonian tour.

For further information concerning compact formulations of the GTSP, we refer the reader to Kara et al. (2012) and Pop (2007).

The last model that we present was described by Pop (2007). It was obtained by distinguishing between variables associated with the inter-cluster (global) connections, called global variables, and variables describing if an edge is chosen between two clusters connected in the global graph, called local variables. This approach known as the *local-global approach*, was first described by Pop (2002) in the case of the generalized minimum spanning tree problem. Based on this approach, the problem is broken down into a macro-level subproblem (i.e., global) and a micro-level subproblem (i.e., local). These are two hierarchically structured subproblems. Combining the solutions obtained for both subproblems, one may obtain a feasible solution of the investigated problem. This decomposition approach provided mathematical models and different solution methods for solving various generalized combinatorial optimization problems (see Hintsch, 2021 and Pop, 2020).

Substituting all the vertices of a cluster C_p with a supervertex representing that cluster, we obtain a novel graph named the *global graph*, denoted $G_{global} = (V_{global}, A_{global})$ and defined as follows.

1. Every cluster C_p of G represents a vertex of V_{global}, with $|V_{global}| = k$.
2. The arcs of the global graph G_{global} are determined by the connections between the supervertices $\{C_1, \ldots, C_k\}$.

We define the binary variables y_{pq}, $p, q \in \{1, \ldots, k\}$ that define the global connections between the clusters by

$$y_{pq} = \begin{cases} 1 & \text{if the clusters } C_p \text{and } C_q \text{are connected in the solution} \\ 0 & \text{otherwise.} \end{cases}$$

It is assumed that the vector y depicts a Hamiltonian tour. The convex hull of all these y-vectors is called the *Hamiltonian tour polytope* associated with the global graph with vertex set $\{C_1, \ldots, C_m\}$ which is assumed to be complete.

In accordance with Miller, Tucker, & Zemlin (1960), the polytope, denoted by P_{TSP}, is described by the following polynomial number of constraints.

$$\sum_{p=1, p\neq q}^{k} y_{pq} = 1 \qquad \forall\, q \in \{1, \ldots, k\} \tag{11}$$

$$\sum_{q=1, q\neq p}^{k} y_{pq} = 1 \qquad \forall\, p \in \{1, \ldots, k\} \tag{12}$$

$$(k-1)y_{pq} - t_p - t_q \leq k-2 \;\forall\; p, q \in \{2, \ldots, k\} \text{ and } p \neq q. \tag{13}$$

where t_p, $p = 2, \ldots, k$ are unrestricted real numbers. Constraints (13) guarantee that the solution contains no subtours.

If the vector y describes a Hamiltonian tour on the global graph, the corresponding best (with respect to cost minimization) generalized Hamiltonian tour $(x, z)^{|E|\times|V|}$ may be obtained by solving the following 0–1 linear programming problem.

$$\begin{aligned} \min\ & \sum_{e\in E} c_e x_e \\ s.t.\ & z(C_p) = 1 \qquad \forall\, p \in \{1, \ldots, k\} \\ & x(C_p, C_q) = y_{pq}\ \forall\, p, q \in \{1, \ldots, k\},\ p \neq q \end{aligned} \tag{14}$$

$$\begin{aligned} & x(i, C_p) \leq z_i \qquad \forall\, p \in \{1, \ldots, k\}, \forall\, i \in V \setminus C_p \\ & x_e, z_i \in \{0, 1\} \qquad \forall\, e=\{i,j\} \in E,\ \forall\, i \in V, \end{aligned} \tag{15}$$

where $x(C_p, C_q) = \sum_{i\in C_p, j\in C_q} x_{ij}$ and $x(i, C_p) = \sum_{j\in C_p} x_{ij}$. Constraints (14) guarantee that if the clusters C_p and C_q are connected, then there is an arc $\{i, j\} \in E$ with $i \in C_p$ and $j \in C_q$ in the solution. Constraints (15) ensure that a vertex $i \in V \setminus C_p$ can be adjacent to at most one edge connecting the cluster C_p if the vertex i belongs to the solution.

We denote the feasible set of the linear programming relaxation of this integer linear program by $P_{local}(y)$, for a given Hamiltonian tour y. Then by including the constraints characterizing P_{TSP}, with $y \in \{0, 1\}$, into $P_{local}(y)$, we obtain the following 0–1 linear programming formulation of the GTSP.

$$\begin{aligned} \min\ & \sum_{e\in E} c_e x_e \\ s.t.\ & (x, z) \in P_{local}(y) \\ & y \in P_{TSP} \\ & y_{pq} \in \{0, 1\} \qquad \forall\, p, q \in \{1, \ldots, k\}, p \neq q. \\ & x_e, z_i \in \{0, 1\} \qquad \forall\, e=\{i,j\} \in E,\ \forall\, i \in V. \end{aligned}$$

As far as we know, there exists no unifying framework that compares the strengths of the linear programming relaxations of the GTSP formulations (i.e., there exist only a few relationships between the polytopes of the associated LP relaxations of the GTSP formulations). Therefore, an analysis of the strengths of the associated linear programming relaxations of the GTSP formulations is worth conducting.

5. GTSP solution approaches

In this section, we examine different approaches to solve the GTSP. We have classified these solution approaches according to the optimization methods that have been used, as shown in Fig. 2.

5.1. Exact algorithms

The exact methods are able to produce the optimal solution for a given optimization problem. In general, for NP-hard problems, these methods are very time-consuming due to the complexity of the calculations, thereby making them applicable in very limited cases. Therefore, exact approaches are adopted in only a few papers and they perform well only on small problem instances.

Henry-Labordere (1969), Srivastava et al. (1969) and Saskena (1970) proposed dynamic programming (DP) approaches for solving the GTSP derived from DP methods for the classical TSP, in which a state is defined by the clusters that have been already visited. A major drawback of these DP approaches is that the number of states grows exponentially as the number of clusters increases.

Dimitrijević et al. (1996) described a branch-and-bound algorithm for solving the GTSP based on the minimal rooted tree as a relaxation. The authors were able to solve instances with up to 13 clusters and 52 vertices optimally.

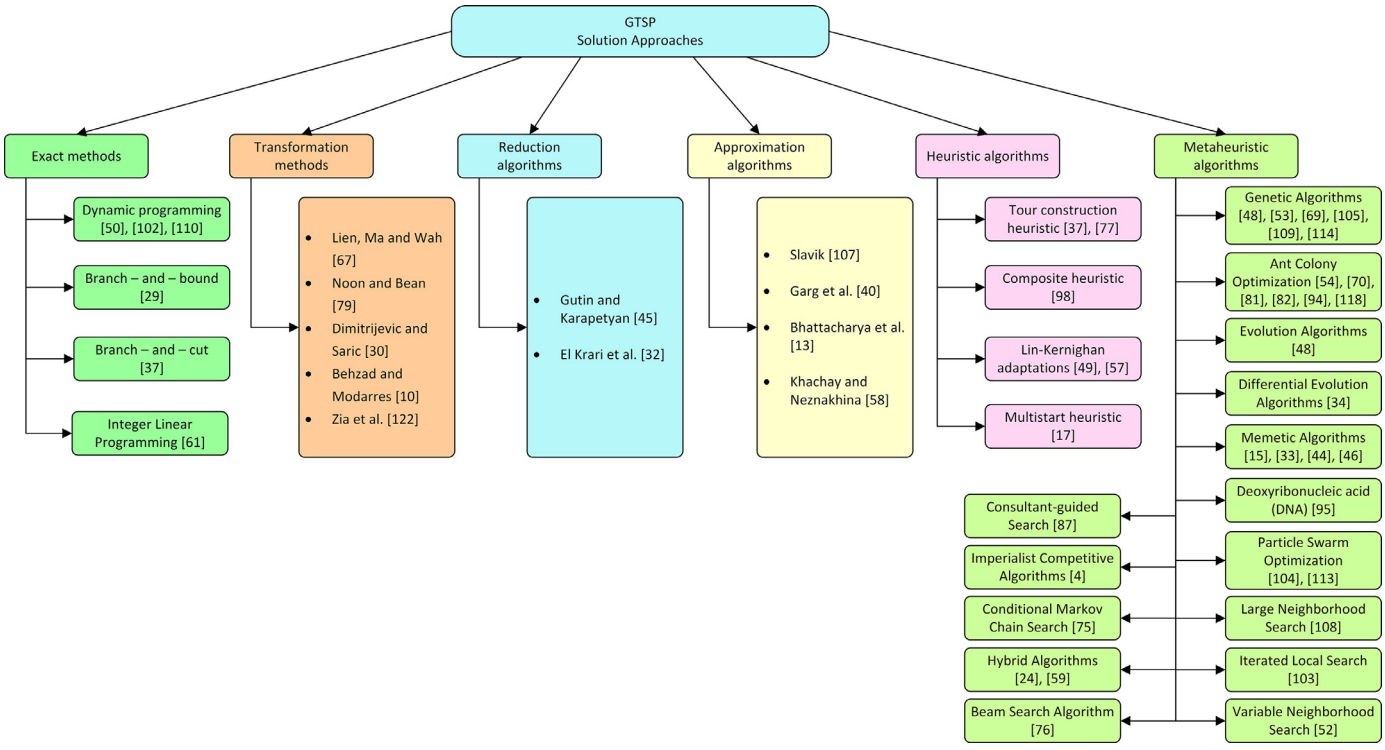

Fig. 2. A taxonomy of the GTSP solution approaches.

Laporte & Norbert (1983) provided a method based on integer linear programming that can be applied to both Euclidean and non-Euclidean problems, with the only restriction that the distance matrix is symmetric. Their proposed exact method was rather effective. It was able to solve much larger problems, surpassing the methods based on complete enumeration and dynamic programming. The largest instance solved had 50 vertices and 10 clusters.

Fischetti et al. (1997) described a branch-and-cut method that generated optimal solutions for problems with up to 89 clusters and 442 vertices (size that is still insufficient for real-world applications). Branch-and-cut is a well-known combinatorial optimization method for solving integer linear programs, which is a hybridization of branch-and-bound and cutting plane techniques. While branch-and-cut methods drastically decrease the size of the solution space and perform well on small and medium problem instances, these methods are impractical for large instances, because they are exponential time algorithms. This type of algorithm requires very large computation times to solve large instances.

5.2. Transformation methods

Due to the complexity of generalized combinatorial optimization problems, transforming them into classical combinatorial optimization problems is a convenient solution, even though such transformations are usually implying a growth in the size of the problems' instance.

Transformation methods of the GTSP into the TSP have been investigated by several researchers. The GTSP is a generalization of the classical TSP and, therefore, we may exploit the similarities between them. There exist many efficient solution approaches for solving the TSP, and most of the existing approaches for solving the GTSP are extensions of TSP solution approaches.

The first transformation of the GTSP into the TSP was proposed by Lien et al. (1993). The transformation was done in two steps. In the first step, the graph G, which is supposed to have intersecting clusters, is transformed into a graph G' in which all the clusters are nonintersecting. In the second step, G' is transformed into an instance of G'' of the TSP with the following properties.

1. There is a one-to-one correspondence between Hamiltonian tours in G'' and the non-redundant generalized tours in G'.
2. The costs of the corresponding Hamiltonian tours in G'' and non-redundant generalized tours in G' pairs are the same.

In this way, any Hamiltonian tour in G'' may be converted into a generalized tour in G without increasing the cost and any optimal Hamiltonian tour in G'' may be transformed into an optimal generalized tour in G. If all clusters are non-intersecting the first step is omitted. A major drawback of this transformation is that the number of vertices of the transformed TSP is more than three times larger than the number of vertices in the corresponding GTSP.

Noon & Bean (1993) described an efficient transformation of the asymmetric GTSP into the classical asymmetric TSP. Their transformation was done in two steps and is characterized by the fact that it does not increase the number of vertices but slightly increases the number of arcs. The first step converts an instance of the asymmetric GTSP into an instance of the clustered traveling salesman problem (CTSP) by redefining the arcs and the arc costs. In the second step, the instance of the CTSP is transformed into a classical asymmetric TSP by simply adding a large cost to all inter-cluster arcs, using a similar approach to that of Chisman (1975). Using this transformation, given an optimal solution to an instance of an asymmetric TSP which costs less than $(k+1)M$, an optimal solution of the asymmetric GTSP can be constructed, where M is a large positive number.

Dimitrijevic & Săric (1997) proposed a different transformation that dropped off the size of the corresponding TSP. In their transformation, a GTSP instance is transformed into an instance of the asymmetric TSP having twice the number of vertices as the original GTSP. They proved that the optimal solution of the TSP instance corresponds to a unique optimal solution of the GTSP instance with no greater cost.

Modifying the transformation presented by Dimitrijevic & Săric (1997), Behzad & Moddares (2002) described a more efficient transformation in which the number of vertices in the transformed TSP does not surpass the number of vertices in the original GTSP. In their transformation, the directed graph G' corresponding to G has the same set of vertices and the cost of every generalized tour in G is equal to the cost of the Hamiltonian tour in G' less kM, where $M > \sum_{(i,j)\in E} c_{ij}$.

Zia, Cakir, & Seker (2018) described five different search algorithms: Euclidean search (E-search), Dijkstra search (D-search), Radial search (R-search), RE-search that combines R-search with E-search and RD-search that combines R-search with D-search, for transforming GTSP into classical TSP that operate spatially. The performance of the proposed approaches was assessed on a set of 75 instances with 15 cities, taken from the OpenStreetMap (OSM) dataset and classified into different road networks.

5.3. Reduction algorithms

Distinctive preprocessing procedures have been used for complex problems in order to shorten the computation time due to their important role within the algorithms. Such approaches have been encountered in integer and linear programming, as well as for different combinatorial optimization problems.

An important characteristic of GTSP is that it is not necessary to visit all the vertices of the graph. The GTSP can contain vertices that do not belong to the optimal solution and thus they may be eliminated. An analogous situation is encountered regarding the edges. Thus, reduction algorithms that eliminate either vertices or edges of the underlying graph might be of interest.

Gutin & Karapetyan (2009) described three reduction algorithms that eliminate redundant vertices and edges, while maintaining the value of the optimal solution. The first reduction algorithm eliminates the redundant vertices, the second one eliminates the redundant edges, and the third one is a combined algorithm which applies a vertex reduction algorithm and then an edge reduction algorithm. All proposed reduction methods keep the vertices and edges that are in the optimal solution. The experimental results showed that the combined reduction algorithm is the most efficient among the three reduction algorithms. The proposed combined reduction algorithm has a running time of $O(n^3)$ in the worst case scenario. It reduced the size of the instance by 15 – 20% and lowered the computational time by approximately 45%.

Another reduction method was proposed by El Krari & Benani (2019). Their pre-processing technique selects from every cluster the closest vertices to the other clusters and removes the vertices that have never been chosen to reduce the solution search space size. The proposed method had very small running times, while the rate of the reduction is up to 98%, therefore being very competitive against the reduction algorithms proposed by Gutin & Karapetyan (2009). Different GTSP solvers were applied to the reduced instances in order to assess their performance. The results showed that reduced instances helped the solvers find good feasible solutions in very short computational times, but are not guaranteed to find optimal solutions.

5.4. Approximation algorithms

Approximation algorithms are polynomial time algorithms that produce approximate solutions to NP-hard optimization problems, with demonstrable guarantees on the quality of the solution. The design and analysis of approximation algorithms include a mathematical proof confirming the quality of the generated solutions in the worst case, distinguishing them from heuristic approaches, which find reasonably good solutions, but do not provide any clear indication about the quality of the solutions.

Slavik (1997) defined a problem called the errand scheduling problem, which actually corresponds to the GTSP, and proposed an approximation algorithm for this combinatorial optimization problem whose performance ratio is $\frac{3}{2}\rho$, where ρ is the maximum size of the clusters, $\rho = \max\{|C_i| : i = 1, \ldots, k\}$, assuming that the cost function satisfies the triangle inequality.

Garg & Ravi (2000) provided a poly-logarithmic approximation algorithm for the group Steiner problem, that aims at determining the minimum cost subtree with at least one vertex from each group. Every group Steiner tree can be transformed (shortcut) into a generalized traveling tour of at most $\frac{3}{2}$ times its cost, by defining the groups to be exactly the clusters and using the classical Christofides approximation algorithm (Christofides, 1976). A ρ-approximation algorithm in the case of the group Steiner problem gives a $\frac{3}{2}\rho$-approximation algorithm for the GTSP. The algorithm developed by Garg & Ravi (2000) in the case of the group Steiner problem, gives an $O(\log^2 n \log\log n \log k)$ poly-logarithmic approximation algorithm for GTSP.

Bhattacharya, Custic, Rafiey, Rafiey, & Sokol (2015) described a simple $(1 + 4\sqrt{2} + \epsilon)$ – approximation algorithm for the geometric generalized minimum spanning tree problem defined on grid clusters, where the given set of points belong to an integer grid in the Euclidean plane with every non-empty 1 x 1 cell of the grid forming a cluster. The authors used this result to provide an $(1.5 + 8\sqrt{2} + \epsilon)$ approximation algorithm for the geometric GTSP defined on grid clusters with the complexity bound depending polynomially on the total number of vertices n and the number of clusters k.

Khachay & Neznakhina (2016) described two novel approximation algorithms for the geometric GTSP defined on grid clusters and proved that for any fixed k, both algorithms are polynomial-time approximation scheme (PTAS). They provide an $(1 + \epsilon)$ approximation solution within polynomial time. The first approximation algorithm combines exhaustive search for sub-optimal representatives for the clusters with dynamic programming for estimating the obtained variants and it is PTAS when $k = O(\log n)$. The second one is based on the Arora (1998)'s PTAS in the case of classical Euclidean TSP and is PTAS when $k = n - O(\log n)$.

5.5. Heuristic and metaheuristic algorithms

When it comes to solving problems faster and traditional methods are too slow, or when an exact solution cannot be found by means of a traditional method and a suboptimal solution needs to be identified, heuristic algorithms can be used. The main difference between these classes of techniques is that exact methods guarantee that a solution is optimal, while heuristic methods may provide good quality solutions but without any optimality guarantee.

As opposed to individual heuristic algorithms that provide a solution to a particular problem, metaheuristic algorithms are strategic problem solving frameworks that can be customized to provide solutions to a broad range of optimization problems. More specifically, a metaheuristic framework has a set of principles for solving the problems, which sometimes depend on physical or natural processes, as well as a set of control parameters. When a metaheuristic strategy is chosen to solve a particular problem, these parameters need to be chosen and fine-tuned. We do not expect metaheuristic algorithms to surpass the performance of specialized heuristics which are designed for certain problems. The principal goal of metaheuristics is to solve complex optimization problems for which no specialized algorithms exist.

As opposed to exact methods, heuristic and metaheuristic algorithms provide no assurance that an optimal solution will be found, but they are used for several reasons. Sometimes they are

Table 1
Heuristic and metaheuristic algorithms for solving the GTSP.

Heuristic algorithms	References
Tour Construction Heuristic	Noon (1988), Fischetti et al. (1997)
Composite Heuristic	Renaud & Boctor (1998)
Lin-Kernighan Adaptation	Karapetyan & Gutin (2011), Helsgaun (2015)
Multistart Heuristic	Cacchiani, Muritiba, Negreiros, & Toth (2011)
Metaheuristic algorithms	**References**
Genetic Algorithm	Snyder & Daskin (2006), Huang et al. (2005), Matei & Pop (2010) Hao, Huang, & Cai (2008), Silberholz et al. (2007), Tasgetiren, Suganthan, & Pan (2007b) Cosma, Pop, & Cosma (2021a), Wu, Liang, Lee, & Lu (2004), Zhao, Lin, & Lu (2006), Pop, Matei, & Sabo (2010)
Ant Colony Optimization	Yang, Shi, Marchese, & Liang (2008), Reihaneh & Karapetyan (2012), Pintea, Pop, & Chira (2017) Pintea (2015), Meng, Lin, Qing, & Wenjing (2019), Kan & Zhang (2012)
Evolution Algorithm	Hao et al. (2008)
Differential Evolution Algorithm	Tasgetiren, Suganthan, & Pan (2010)
Memetic Algorithms	Gutin & Karapetyan (2008), Bontoux et al. (2010), Gutin & Karapetyan (2010) El Krari & Benani (2020)
Deoxyribonucleic Acid (DNA)	Ren, Wang, Wang, & Wu (2020)
Particle Swarm Optimization	Shi, Lianga, Leeb, Lub, & Wang (2007), Tasgetiren, Suganthan, & Pan (2007a)
Large Neighborhood Search	Smith & Imeson (2017)
Iterated Local Search	Schmidt & Irnich (2022)
Variable Neighborhood Search	Hu & Raidl (2008)
Beam Search Algorithm	Nejma & M'Hallah (2021)
Consultant-guided Search	Pop & Iordache (2011)
Imperialist Competitive Algorithm	Ardalan, Karimi, Poursabzi, & Naderi (2015)
Conditional Markov Chain Search	Nalivajevs & Karapetyan (2019)
Hybrid Algorithm	Khan & Maiti (2018), Cosma et al. (2021a)

the only solution methods of choice for solving complex optimization problems for which no exact solution methods or even exact formulations are known. Even if exact solution methods are known, they may be computationally expensive, making them unusable for real-world applications.

Several heuristic and metaheuristic algorithms have been described to solve medium-size and large-size GTSP problem instances. In Table 1, we present a list of heuristic and metaheuristic algorithms for solving the GTSP.

From Table 1 we observe that the genetic algorithm (GA) approach is most used for the GTSP, while the second most used approach is the ant colony optimization (ACO) metaheuristic. It is worth mentioning that the GA approach was combined with other techniques in several papers, such as a GA combined with local search procedures resulting in efficient memetic algorithms (see Bontoux, Artigues, & Feillet, 2010; Gutin & Karapetyan, 2008; Gutin & Karapetyan, 2010) or with a swap sequence-based particle swarm optimization technique (see Khan & Maiti, 2018).

Next, we present some of the most efficient heuristic and metaheuristic solution approaches for solving the GTSP.

Renaud & Boctor (1998) proposed a composite heuristic for solving the GTSP, called Generalized Initialization, Insertion and Improvement, denoted by GI3, which is an extension of the heuristic algorithm presented by Renaud, Boctor, & Laporte (1996) for the classical TSP. Their heuristic has an initialization step in which a partial tour is built, an insertion step which completes the tour, by adding from clusters that have not been visited, vertices with the cheapest costs, and an improvement step based on three adaptations of the Lin's 2-opt, 3-opt and 4-opt heuristics to the GTSP. In addition to the improvements of the generalized tours based on changing the order of the vertices, the set of visited vertices is allowed to change.

Snyder & Daskin (2006) provided a random-key genetic algorithm for solving the GTSP. The advantage of their approach was that solutions produced by the crossover or the mutation operators are feasible solutions of the GTSP. The proposed GA was combined with two local search improvements (a swap procedure and a 2-opt neighborhood search) to produce a memetic algorithm.

Silberholz et al. (2007), using a general structure of GAs, evaluated different genetic structures and crossover operators. The main characteristics of their solution approach was the use of isolated initial populations and a novel reproduction mechanism that rely on the TSP ordered crossover operator. In addition, they used two local improvements heuristics to improve the performance of their described GA. The first was the classical 2-opt heuristic developed for the TSP and the second one was a swap improvement algorithm.

Bontoux et al. (2010) provided a memetic algorithm in which a genetic algorithm is combined with local search techniques. The main contribution of the authors was the originality of the crossover operator which relies on large neighborhood search. Gutin & Karapetyan (2010) described a novel memetic algorithm to solve the GTSP that couples a GA with a powerful local search procedure that runs sequentially for six local search heuristics. Their algorithms may be applied for symmetric and asymmetric instances of the GTSP.

Helsgaun (2015) combined the efficient transformation of the asymmetric GTSP into the classical asymmetric TSP developed by Noon & Bean (1993) with the powerful Lin-Kernighan-Helsgaun (LKH) TSP solver to solve the transformed GTSP instances. In this way, Helsgaun was able to improve the quality of the solutions for the GTSP_LIB (Fischetti et al., 1997) instances over the best existing algorithms at that time. The improvement in solution quality came at the expense of computation time with some instances needing large computation times.

Smith & Imeson (2017) presented an efficient solution approach for solving the GTSP that relies on adaptive large neighborhood search. Their algorithm removes and inserts vertices over and over again into the generalized tour. Its main characteristic is the introduction of a general insertion mechanism that includes as specific cases the well-known nearest, farthest, and random insertion mechanisms.

Schmidt & Irnich (2022) described three novel GTSP neighborhoods, all with polynomial worst-case computation and inspired by classical TSP neighborhoods. The authors allow, at the same time, permutation of the visiting order of the clusters and the way of choosing the vertices from every cluster. Their proposed neighborhoods, along with neighborhoods from the existing literature, are integrated into an efficient iterated local search (ILS) algorithm. They used cluster optimization due to Fischetti et al. (1997), Ballas-

Simonetti neighborhood (Balas, Simonetti, & study, 2001), Gutin's neighborhood (Gutin, Yeo, Zverovitch, Gutin, & Punnen, 2007), and the string relocation neighborhood (Schmidt & Irnich, 2022). The ingenuousness of the ILS algorithm resides in its distinct random way of choosing the neighborhoods within the local search and an ordinary record-to-record ILS acceptance criterion.

Recently, Nejma & M'Hallah (2021) proposed an approximate tree-based technique that used a beam search algorithm (i.e., a truncated branch-and-bound) with two levels of hybridization

- a low-level hybridization that employs a local search based on a swap procedure for every partial solution of a vertex of a tree, and
- a high-level hybridization that applies adaptations of the 2-opt and 3-opt local search algorithms or Lin-Kernighan heuristic to the current best solution obtained during the algorithmic search procedure.

6. Datasets, computational results and comparative analysis of the performance of existing algorithms

6.1. Datasets

The existing datasets used for testing the performance of different solution approaches for the GTSP belong to four libraries.

1. GTSP_LIB was proposed by Fischetti et al. (1997) and groups the vertices based on their closeness to each other, selecting iteratively $m = \lceil n/5 \rceil$ centers of the clusters so that every center maximizes its distance from the nearest already chosen center. Then, all remaining vertices are assigned to the cluster whose center is the nearest. This way of clustering the vertices simulates the geographical regions. GTSP_LIB contains 88 symmetric and asymmetric instances with up to 1084 vertices and 217 clusters. The instances were acquired from the Reinelt's TSP_LIB (Reinelt, 1991).
2. BAF_LIB was provided by Bontoux (2008) in his PhD thesis. The instances from this library were derived as well from Reinelt's TSP_LIB (Reinelt, 1991). A main feature of generating the clusters is that there are no geographical regions and the clusters are generated pseudo-randomly. BAF_LIB contains 56 instances which are symmetric with up to 1084 vertices and 217 clusters.
3. MOM_LIB was introduced by Mestria, Ochi, & de Lima Martins (2013) initially in the case of the clustered TSP, then for the GTSP and the clustered shortest-path tree problem (see Cosma et al., 2021b). MOM_LIB includes six kinds of Euclidean instances which were generated using distinctive algorithms (for more details, see Mestria et al., 2013). There were small instances with vertices ranging from 30 to 120, and clusters ranging from 2 to 42. There were large instances with vertices ranging from 108 to 3000, and partitioned into clusters ranging from 4 to 200.
4. LARGE_LIB was proposed by Helsgaun (2015) and has 44 very large, symmetric instances with the number of clusters ranging from 10 to 17,180 and the number of vertices ranging from 1000 to 85900. These instances were derived from TSP_LIB and the National TSP benchmark library (Applegate, Bixby, Chvatal, Cook, & problems, 2015). The clusters were obtained using Fischetti's clustering procedure (Fischetti et al., 1997).

In Table 2, we present the characteristics of the main datasets used in the literature for testing the performance of the existing solution approaches for solving the GTSP.

6.2. Computational results and comparative analysis of the performance of existing algorithms

The best-performing state-of-the-art algorithms for solving the GTSP that have been published in the literature are

- the memetic algorithm (MA) described by Gutin & Karapetyan (2010);
- the Lin-Kernighan-Helsgaun (LKH) solution approach provided by Helsgaun (2015);
- the large neighborhood search (LNS) algorithm described by Smith & Imeson (2017);
- a basic iterated local search (Basic ILS) and a refined version of the algorithm (Refined ILS) provided by Schmidt & Irnich (2022).

Next, we present a comparative analysis of these algorithms for all instances belonging to the existing GTSP libraries.

Gutin & Karapetyan (2010) tested and validated their memetic algorithm only on instances from GTSP_LIB. Helsgaun (2015) tested his solution approach on instances from GTSP_LIB and LARGE_LIB. Smith & Imeson (2017) benchmarked the performance of their LNS solver on all four GTSP libraries. In order to compare the performance of their basic iterated local search and the refined version of the algorithm against MA, LKH, and LNS, Schmidt & Irnich (2022) performed tests with the previously three mentioned algorithms on all described libraries. These reported results were taken into consideration in our analysis of performance.

To highlight the differences among the algorithms, the reported results were processed in the following way.

1. The best-known solution (BKS) has been updated for each instance to reflect the situation at the time of writing this survey paper.
2. All instances for which there are no reported results by the corresponding authors have been removed.
3. In order to emphasize the differences, the instances for which all the algorithms found the BKS at each run were removed.

In Table 3, we display the number of runs in which the five algorithms (LNS, LKH, MA, Basic ILS and Refined ILS) discovered the BKS. We considered only the instances for which there are results reported in the literature by all the five algorithms.

From Table 3, we see that the LNS achieved the largest number of BKS, followed by MA, Refined ILS, Basic ILS, and LKH.

The box plots of the success rates and the average percentage errors obtained on different GTSP libraries based on the processed results are shown in Figs. 3–6. The success rate of an algorithm, when solving an instance, represents the percentage of runs at which it found the BKS. The percentage error e that occurs when an instance is solved by an algorithm is given by the following formula

$$e = 100 \cdot \frac{foundSolution - BKS}{BKS}.$$

Analyzing the success rates, we conclude the following.

- The LKH solution approach has the best success rates in the case of GTSP_LIB, and the Refined ILS algorithm has the second best success rate.
- In the case of BAF_LIB, the LNS algorithm obtained the best results. The Basic ILS and Refined ILS algorithms produced the second best results.
- In the case of MOM_LIB, the LNS and MA success rates have the same median of 100%, which indicates that they found each time the optimal solution for at least half of the instances. The interquartile range is smaller in the case of the LNS algorithm, which indicates that more results are closer to the median. For

Table 2
Datasets used in computational experiments for the GTSP.

Datasets	No. of instances	No. of vertices	No. of clusters	Reference
GTSP_LIB	88	152–1084	31–217	Fischetti et al. (1997)
BAF_LIB	56	99–1084	20–217	Bontoux (2008)
MOM_LIB	45	30–3000	2–200	Mestria et al. (2013)
LARGE_LIB	44	1000–85,900	10–17,180	Helsgaun (2015)

Table 3
Number of runs in which five algorithms found the BKS.

Solution approaches	GTSP_LIB	BAF_LIB	MOM_LIB	LARGE_LIB	Total
MA	364	365	380	72	1181
LKH	368	227	243	79	917
LNS	349	401	413	86	1249
Basic ILS	348	385	343	64	1140
Refined ILS	366	385	343	77	1171

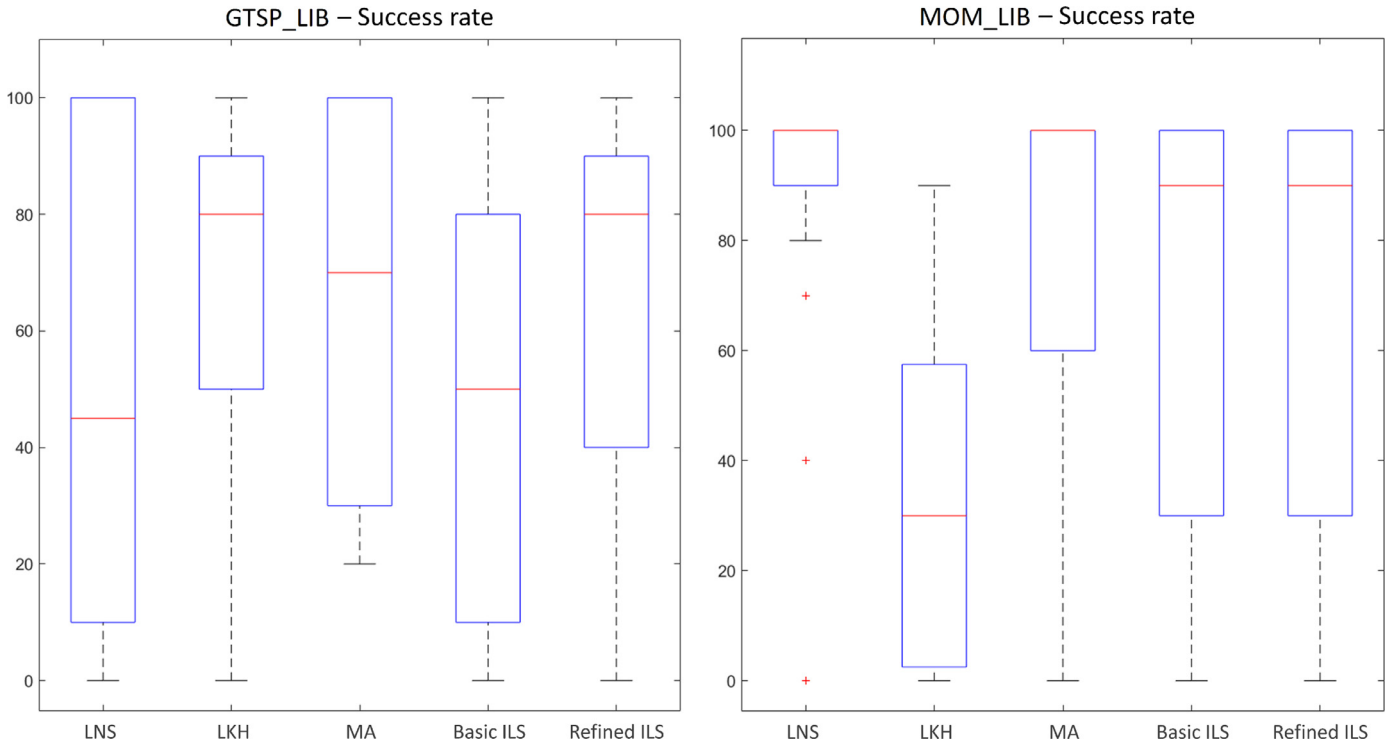

Fig. 3. Box plots of the success rate for five algorithms on the GTSP_LIB and MOM_LIB instances.

these reasons, we conclude that LNS is the best algorithm, followed by MA.

- In the case of the LARGE_LIB, the LNS algorithm took first place even though it has a lower median, since it has also very good success rates, including 100% for a few instances. The second place was taken by the LKH algorithm, which has the highest median.

Regarding the average percentage error, we conclude that the LKH algorithm produced the best results for GTSP_LIB and LARGE_LIB, while the LNS algorithm produced the best results for BAF_LIB and MOM_LIB.

A general view of the performances of LNS, LKH, MA, Basic ILS, and Refined ILS algorithms, for the GTSP_LIB, BAF_LIB, MOM_LIB, and LARGE_LIB datasets, is shown in Table 4 and Fig. 7. The columns in Table 4 contain the average percentage errors, which are calculated for each dataset.

In conclusion, we observe that the LNS algorithm developed by Smith & Imeson (2017) showed the best overall performance. It performed remarkably well in comparison to the other algorithms for those instances belonging to MOM_LIB and BAF_LIB. Regarding the success rates, it performed best for MOM_LIB, BAF_LIB and LARGE_LIB. Regarding the average percentage error, it performed best for MOM_LIB and BAF_LIB, and second best in the case of LARGE_LIB. Surprisingly, this algorithm did not perform so well compared to the other algorithms in the case of GTSP_LIB. The blue columns corresponding to the percentage errors in the case

Table 4
Average percentage errors of LNS, LKH, MA, Basic ILS, and Refined ILS algorithms.

Dataset	LNS	LKH	MA	Basic ILS	Refined ILS
GTSP_LIB	0.0105	0.0065	0.0108	0.0253	0.0148
BAF_LIB	0.07	6.51	0.29	0.17	0.17
MOM_LIB	0.02	0.82	0.03	0.09	0.09
LARGE_LIB	0.50	0.52	0.76	1.40	0.97

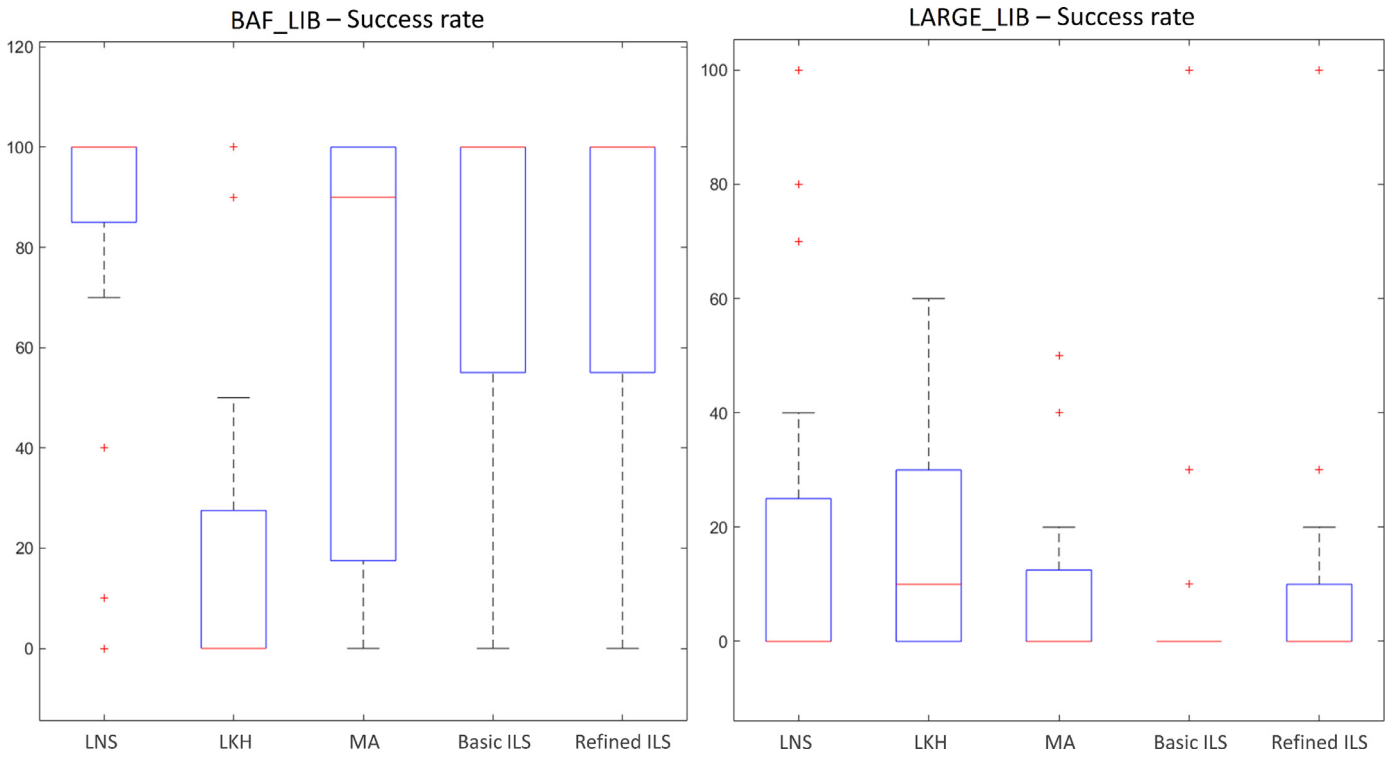

Fig. 4. Box plots of the success rate for five algorithms on the BAF_LIB and LARGE_LIB instances.

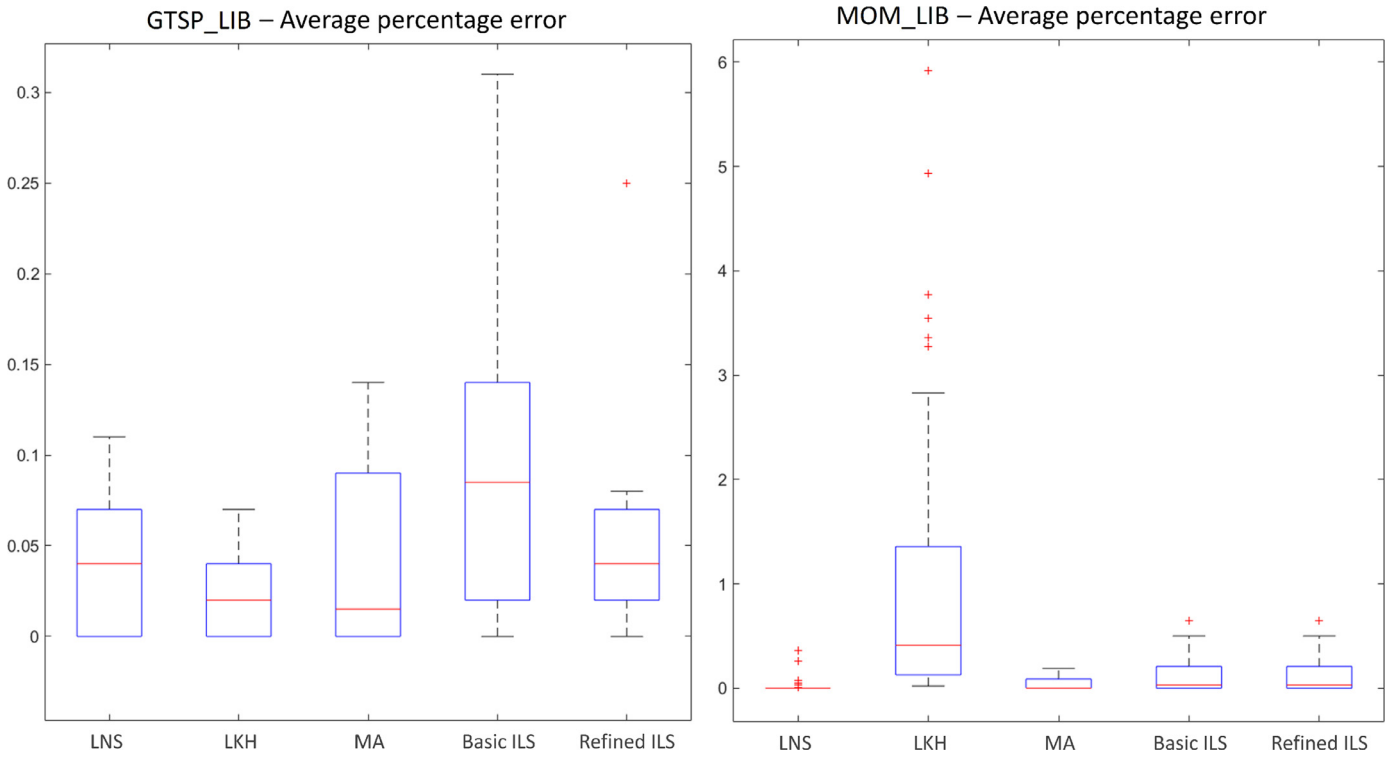

Fig. 5. Box plots of the average percentage error for five algorithms on the GTSP_LIB and MOM_LIB instances.

of GTSP_LIB do not appear in Fig. 7, because their values are close to 0.

Comparing the algorithms' efficiency is a difficult task. For accurate results, only runs which delivered similar solutions should be compared. If solutions differ, then comparing execution times is irrelevant, because better results involve more complex operations, which require longer running times. If we compare the algorithms and not their implementation, then the programming language in which they were developed should also be considered. Finally, we must take into account the characteristics of the equipment on which the experiments were performed (CPU, memory, operating system).

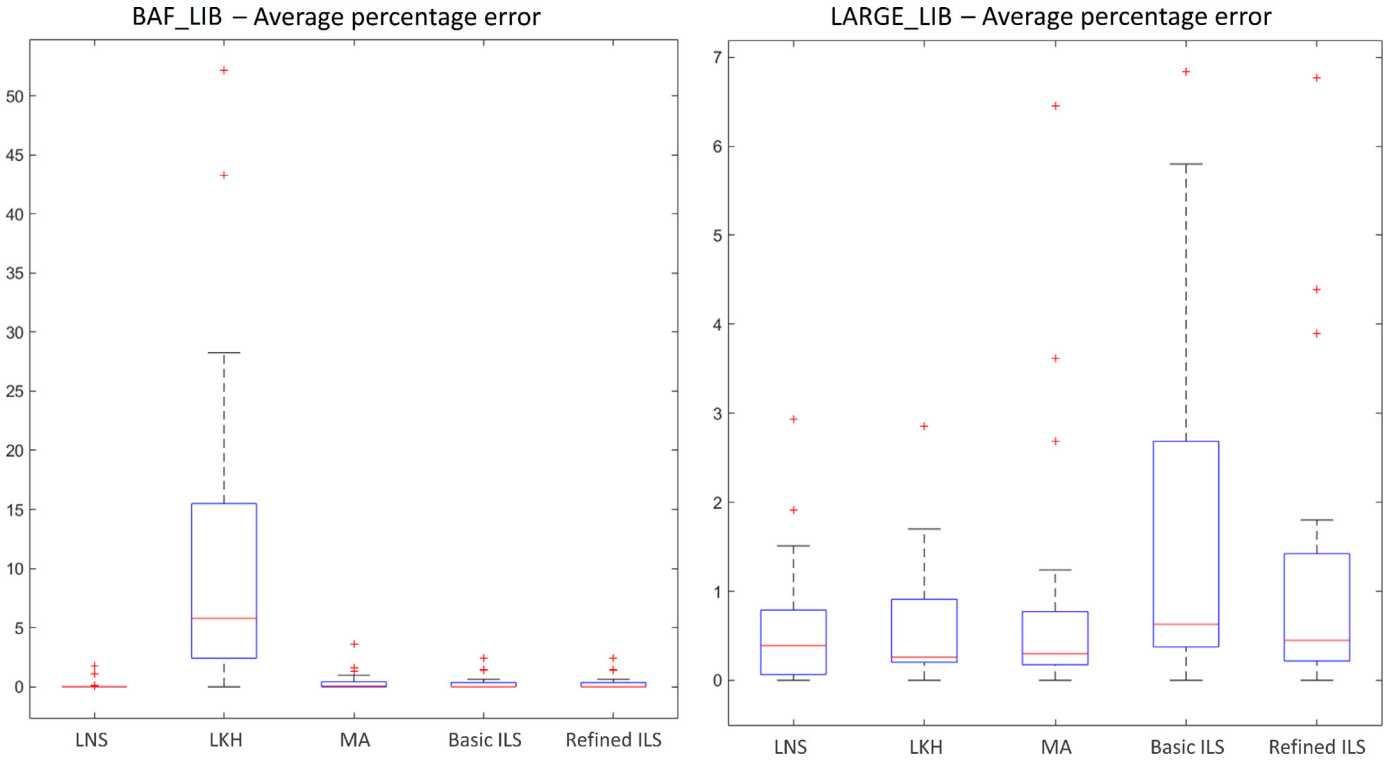

Fig. 6. Box plots of the average percentage error for five algorithms on the BAF_LIB and LARGE_LIB instances.

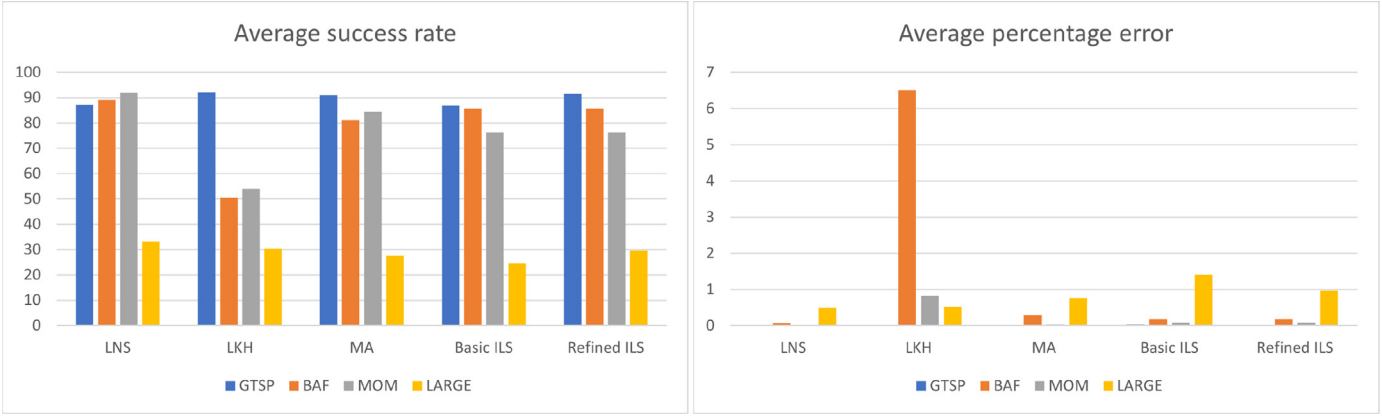

Fig. 7. Comparison of success rates and percentage errors.

Smith & Imeson (2017) provided the LNS average execution times for eight sample instances in each of the GTSP, BAF and LARGE datasets. They ran the LNS algorithm in three settings (Slow, Medium and Fast) on an Intel Core i7-6700 CPU, 3.4 GHz, with 16 GB of RAM. The LNS algorithm was implemented in the Julia language. Regarding LKH and MA, there is execution time information only in the original papers Helsgaun (2015) and Gutin & Karapetyan (2010) respectively, but only the GTSP_LIB is investigated in both papers. LKH was developed in C and executed on an Intel Core i7 CPU, 3.4 GHz, with 32 GB of RAM. MA was developed in C# and executed on an AMD Athlon 64 X2 CPU, 3.0 GHz. LNS and LKH algorithms were run on the same type of CPU, while MA was run on a different CPU, which according to the single thread ratings in PassMark Software is 2.23 times slower. Schmidt & Irnich (2022) only provided computational time bounds, and not the actual execution time for each instance.

The running times of the three algorithms (LKH, LNS and MA) on available instances are presented in Table 5, where $\bar{e}$ represents the average percentage error calculated for all runs on the same instance, and *Time* is the average running time in seconds. The MA running times were corrected to compensate for the CPU difference. The bottom line contains the average of $\bar{e}$, calculated for all instances in the table.

First, we compare LKH with LNS on the slow setting, because the average percentage errors of the other two settings are much higher. The running times comparison makes sense only in the case of similar average percentage errors. There is only one such case, for instance number 5. For this instance, the LKH algorithm is more than five times faster. For the other four instances, LKH produced smaller average percentage errors in shorter running times than LNS. In conclusion, for these instances, LKH is more efficient than LNS. Second, we compare LNS in the medium setting with MA, because the average percentage errors are close. All instances in Table 5 are solved by MA in shorter running times than LNS. For three instances, MA also produced better average percentage errors.

Table 5
Efficiency comparison for LKH, LNS and MA.

No.	Instance name	LKH		LNS						MA	
				Slow		Medium		Fast			
		$\bar{e}$ %	Time(s)	$\bar{e}$ %	Time(s)	$\bar{e}$ %	Time(s)	$\bar{e}$ %	Time(s)	$\bar{e}$ %	Time(s)
1	134gr666	0	162.3	0.09	182	0.38	22	1.17	1	0.11	6.5
2	145u724	0	145.4	0.02	321	0.06	34	0.96	1	0.14	5.2
3	157rat783	0.03	764.4	0.08	471	0.17	43	1.6	2	0.11	6.9
4	200dsj1000	0	794.4	0.01	1091	0.1	102	2.2	5	0.12	22.5
5	201pr1002	0	164.8	0	846	0.04	78	1.73	5	0.14	15.6
6	207si1032	0	1202.5	0.06	963	0.11	98	0.94	5	0.03	17.2
7	212u1060	0.02	2054.9	0.07	1048	0.19	120	2.24	5	0.27	20.1
8	217vm1084	0	209	0.04	1167	0.09	126	1.72	5	0.19	26.8
	Averages	0.006		0.046		0.142		1.57		0.139	

A proper comparison of the five algorithms (LNS, LKH, MA, Basic ILS and Refined ILS) in terms of efficiency is impossible due to lack of reported information. However, by analyzing the limited data that is available, we conclude that, in the case of the instances in Table 5, LKH is the most efficient, followed by MA and LNS.

7. Summary and future research directions

This paper has been devoted to reviewing contributions with respect to the generalized traveling salesman problem. The GTSP belongs to the class of generalized network design problems, a very dynamic area of research that has been studied intensively over the past three decades. The hierarchical structure of the GTSP offers more precise models for many real-world applications.

We have presented an overview of the existing mathematical formulations of the GTSP and different solution approaches, as well as describing existing variants and related problems, and highlighting application areas.

Several solution approaches have been proposed in the literature, either to solve the GTSP directly or after its transformation into the TSP. The first solution approaches were based on exact algorithms. Even though these techniques provide the optimal solution, they are useful only for small instances because the GTSP is *NP*-hard. Metaheuristic approaches, including genetic algorithms, ant colony optimization, particle swarm optimization, iterated local search, and beam search algorithm have been extensively explored to solve the GTSP. We point out that genetic algorithms and ant colony optimization have been the most widely used metaheuristic approaches. However, in recent years, there has been a tendency to develop hybrid algorithms and to apply neighborhood search techniques including iterated local search, large neighborhood search, and beam search algorithms.

There are four benchmark libraries that have been used to test solution approaches for the GTSP. In the majority of the cases, the problem instances selected by researchers to asses the performance of their proposed algorithms belong to GTSP_LIB (Fischetti et al., 1997).

Based on our analysis, we showed that the large neighborhood search algorithm (Smith & Imeson, 2017) is the state-of-the-art solver. LNS shows quite similar performance compared to the memetic algorithm (Gutin & Karapetyan, 2010), the solution approach described by Helsgaun (2015), and the iterated local search algorithm and its refined version developed by Schmidt & Irnich (2022) for GTSP_LIB and LARGE_LIB. LNS shows significant performance improvements over the analyzed algorithms by improving several best-known solutions for MOM_LIB and BAF_LIB.

Based on our research and findings, we have identified the following future research directions:

1. A theoretical and experimental analysis of the existing GTSP formulations;
2. Advances on decomposition methods and hybrid algorithms for solving the GTSP efficiently;
3. Investigation of dynamic variants of the GTSP;
4. Appropriate modeling of uncertainty in various GTSP contexts, such as stochastic estimation, fuzzy estimation or rough estimation of the travel costs;
5. Appropriate modeling based on the multi-graph structure of the GTSP;
6. Improving and enlarging the existing benchmark instances.

Decomposition methods are suitable techniques to tackle the GTSP, by decomposing the problem into smaller, more manageable and easier subproblems, and solving them separately. We have reviewed the existing local-global approach which is a decomposition method that decomposes the GTSP into two natural, smaller and hierarchically structured subproblems which depend on each other: a macro-level problem that has cluster vertices representing the k clusters and a micro-level problem that determines in each of the clusters which vertex is visited. Further attention should be given to decomposition methods. They are a crucial source for the development of solution approaches for large-scale GTSPs, by taking advantages of the hierarchical structure of the problem, offering computational advantages by developing effective approaches for solving the subproblems and then combining the obtained results to produce feasible solutions. Another important direction involves the development of hybrid approaches by combining exact methods with metaheuristic algorithms, for example the use of metaheuristic algorithms for providing high-quality solutions and bounds to a branch-and-bound based exact algorithm, or the development of search large neighborhoods by means of mathematical programming techniques to optimize local parts within candidate solutions.

The majority of the researchers focused on the static GTSP and static variants, but real-world optimization applications are actually dynamic. They are time-dependent, as a consequence, we consider that an important direction that should be investigated is the design of solution approaches in the case of changing environments. Some dynamic generalized traveling salesman problem (DGTSP) variations of the GTSP that might be investigated are the following.

- Traffic jams on certain roads implying that the associated travel times are increased.
- Deleting or adding cities (vertices) within the clusters based on varied conditions.

Obviously, the DGTSP is more complex than the GTSP because the number of vertices and the cost matrix are time-varying.

Even though uncertainty is a feature of the investigated problem, because the traveling costs between the cities are uncertain in real-life applications depending on the means of transportation, condition of the roads, and traffic jams, most of the authors used

deterministic models for the GTSP. Researchers should focus on various estimation approaches of the travel costs such as stochastic estimation, fuzzy estimation, and rough estimation. Therefore, future studies on the GTSP should emphasize solving the problem when environments are uncertain. These stochastic optimization problems are more complex than the static variants.

An important aspect ignored by the researchers is the presence of multiple tours in the case of the GTSP. This may lead to associate with the given graph $G = (V, E)$ a corresponding multigraph $G_m = (V_m, E_m)$ that permits multiple edges between the same vertices. The multigraph G_m is defined as follows. Each cluster of the graph G is a vertex of V_m with $|V_m| = k$, and for each edge $e \in E$, it corresponds exactly with one edge $e \in E_m$ with the same cost c_e and with $|E| = |E_m|$. This should be given more attention in future research studies to the multigraph structure of the GTSP, in terms of providing different model formulations and solution approaches of the problem.

The existing benchmark instances, used to test the performance of different solution approaches designed to solve the GTSP, have not been revised since their introduction. The majority of them were derived from TSP_LIB and the clusters were generated using Fischetti's clustering procedure. In our opinion, it is important to update the benchmark instances in accordance to the scale and complexity of recent real-world applications.

Finally, it is worth mentioning that the GTSP remains a promising and challenging research field both theoretically and practically for the new optimizing problems that are emerging.

8. Conclusions

The generalized traveling salesman problem is one of the most interesting and challenging combinatorial optimization problems due to its ability to describe and model many real-life problems that are inherently hierarchical. In this survey paper, we showed that the GTSP may be used to model optimization problems occurring in several fields including location-routing problems, material flow system design, stochastic vehicle routing, image retrieval and ranking, digital garment manufacturing, distribution problems, and scheduling. Our paper provides a comprehensive review of the existing models and solution approaches for the GTSP. We have divided the existing solution approaches into five broad classes: exact algorithms, transformation methods, reduction methods, approximation algorithms, and heuristic and metaheuristic approaches.

As a closing remark, the general image resulting from the review carried out for this paper highlights an area of research with numerous contributions over the past decades, yet diverse topics worth further research are also included. We express our ambition and our hope that the current study would galvanize scientists to conduct further research in this area of combinatorial optimization which has proven to be both promising and challenging.

Acknowledgments

The authors are grateful to the anonymous referees for reading the manuscript very carefully and providing constructive comments which helped to improve the paper substantially. This work was supported by the project “Collaborative Framework for Smart Agriculture” – COSA that received funding from Romania’s National Recovery and Resilience Plan PNRR-III-C9-2022-I8, under grant agreement 760070.

References

Applegate, D., Bixby, R., Chvatal, V.,& Cook, W. (2006) The traveling salesman problem.
Applegate, D., Bixby, R., Chvatal, V., & Cook, W. (2015) National traveling salesman problems. http://www.math.uwaterloo.ca/tsp/world/countries.html.
Ardalan, Z., Karimi, S., Poursabzi, O., & Naderi, B. (2015). A novel imperialist competitive algorithm for generalized traveling salesman problems. *Applied Soft Computing, 26*, 546–555.
Arora, S. (1998). Polynomial time approximation schemes for euclidean TSP and other geometric problems. Journal of ACM, 45, 5753–782
Ávila, T., Corberán, A., Plana, I.,& Sanchis, J. M. (2016) A branch-and-cut algorithm for the profitable windy rural postman problem. European Journal of Operational Research. 249, 3, 1092–1101,
Balas, E. (1989). The prize collecting traveling salesman problem. *Networks, 19*(6), 621–636.
Balas, E., & Simonetti, N., (2001) Linear time dynamic-programming algorithms for new classes of restricted TSPs: A computational studyINFORMS Journal on Computing. 13, 1, 56–75,
Baniasadi, P., Foumani, M., Smith-Miles, K., & Ejov, V., (2020) A transformation technique for the clustered generalized traveling salesman problem with applications to logistics. European Journal of Operational Research. 285, 2, 444–457,
Behzad, A., & Moddares, M. (2002). A new efficient transformation of generalized traveling salesman problem into traveling salesman problem. In *Proceedings of the 15th international conference of systems engineering* (pp. 6–8).
Beltrami, E. L., & Bodin, L. D. (1974). Networks and vehicle routing for municipal waste collection. *Networks, 4*, 65–94.
Bernardino, R., & Paias, A. (2018). Solving the family traveling salesman problem. *European Journal of Operational Research, 267*(2), 453–466.
Bhattacharya, B., Custic, A., Rafiey, A., Rafiey, A., & Sokol, V. (2015). Approximation algorithms for generalized MST and TSP in grid clusters. *Lecture Notes in Computer Science, 9486*, 110–125.
Bontoux, B. (2008). *Techniques hybrides de recherche exacte et approchée: Application á des problémes de transport*. Université d'Avigno. France Ph.D. thesis.
Bontoux, B., Artigues, C., & Feillet, D. (2010). A memetic algorithm with a large neighborhood crossover operator for the generalized traveling salesman problem. *Computers & Operations Research, 37*(11), 1844–1852.
Bovet, J. (1983). The selective travelling salesman problem. In *Paper presented at the EURO VI conference*. Vienna
Cacchiani, V., Muritiba, A., Negreiros, M., & Toth, P. (2011). A multistart heuristic for the equality generalized traveling salesman problem. *Networks, 57*, 231–239.
Carpaneto, G., & Toth, P. (1980). Some new branching and bounding criteria for the asymmetric travelling salesman problem. *Management Science, 26*(7), 736–743.
Cheikhrouhou, O., & Khoufi, I. (2021). A comprehensive survey on the multiple traveling salesman problem: Applications, approaches and taxonomy. *Computer Science Review, 40*, 100369.
Chisman, J. A. (1975). The clustered traveling salesman problem. *Computers & Operations Research, 2*(2), 115–119.
Christofides, N. (1976). Worst-case analysis of a new heuristic for the travelling salesman problem. *Report 388*. Graduate School of Industrial Administration. Carnegie Mellon University
Cosma, O., Pop, P. C., & Cosma, L. (2021a). An effective hybrid genetic algorithm for solving the generalized traveling salesman problem. In *Proceedings of HAIS 2021, lecture notes in computer science: vol. 12886* (pp. 113–123).
Cosma, O., Pop, P. C., & Pop Sitar, C. (2022). A two-level based genetic algorithm for solving the soft-clustered vehicle routing problem. *Carpathian Journal of Mathematics, 38*(1), 117–128.
Cosma, O., Pop, P. C., & Zelina, I. (2020). A novel genetic algorithm for solving the clustered shortest-path tree problem. *Carpathian Journal of Mathematics, 36*(3), 401–414.
Cosma, O., Pop, P. C., & Zelina, I. (2021b). An effective genetic algorithm for solving the clustered shortest-path tree problem. *IEEE Access, 9*, 15570–15591.
da Silva, R. F., & Urrutia, S. (2010). A general VNS heuristic for the traveling salesman problem with time windows. *Discrete Optimization, 7*(4), 203–211.
Demange, M., Monnot, J., Pop, P. C., & Ries, B. (2014). On the complexity of the selective graph coloring problem in some special classes of graphs. *Theoretical Computer Science, 540–541*, 82–102.
Derya, T., Dinler, E., & Kececi, B. (2020). Selective generalized travelling salesman problem. *Mathematical and Computer Modelling of Dynamical Systems, 26*(1), 80–118.
Dimitrijević, V., Milosavljević, M., & Marković, M. (1996). A branch and bound algorithm for solving a generalized traveling salesman problem. *Publikacije Elektrotehnickog fakulteta, Serija Matematika, 7*, 31–35.
Dimitrijević, V., & Săric, Z. (1997). An efficient transformation of the generalized traveling salesman problem into the traveling salesman problem on digraphs. *Information Sciences, 102*(1–4), 105–110.
Dror, M., & Haouari, M. (2000). Generalized Steiner problems and other variants. *Journal of Combinatorial Optimization, 4*, 415–436.
El Krari, M., Ahiod, B., & El Benani, Y. B. (2019). A pre-processing reduction method for the generalized travelling salesman problem. *Operational Research, 21*, 2543–2591.
El Krari, M., Ahiod, B., & El Benani, Y. B. (2020). A memetic algorithm based on breakout local search for the generalized traveling salesman problem. *Applied Artificial Intelligence, 34*(7), 537–549.
Fatih Tasgetiren, M., Suganthan, P. N., & Pan, Q.-K. (2010). An ensemble of discrete differential evolution algorithms for solving the generalized traveling salesman problem. *Applied Mathematics and Computation, 215*, 3356–3368.
Feremans, C., Labbé, M., & Laporte, G. (2003). Generalized network design problems. *European Journal of Operations Research, 148*(1), 1–13.
Fidanova, S., & Pop, P. C. (2016). An improved hybrid ant-local search for the parti-

tion graph coloring problem. *Journal of Computational and Applied Mathematics, 293*, 55–61.
Fischetti, M., Salazar-Gonzales, J. J., & Toth, P. (1997). A branch-and-cut algorithm for the symmetric generalized traveling salesman problem. *Operations Research, 45*(3), 378–394.
Flood, M. M. (1948). A game theoretic study of the tactics of area defense. *RAND Research Memorandum*.
Garfinkel, R. S., & Gilbert, K. C. (1978). The bottleneck traveling salesman problem: Algorithms and probabilistic analysis. *Journal of ACM, 25*, 435–448.
Garg, N., Konjevod, G., & Ravi, R. (2000). A polylogarithmic approximation algorithm for the group Steiner tree problem. *Journal of Algorithms, 37*(1), 66–84.
Gendreau, M., Laporte, G., & Semet, F. (1997). The covering tour problem. *Operations Research, 45*(4), 568–576.
Ghiani, G., & Improta, G. (2000). An efficient transformation of the generalized vehicle routing problem. *European Journal of Operational Research, 122*(1), 11–17.
Gutin, G., Karapetyan, D., & Krasnogor, N. (2008). Memetic algorithm for the generalized asymmetric traveling salesman problem. *Studies in Computational Intelligence, 129*, 199–210. Proceedings of Nature Inspired Cooperative Strategies for Optimization
Gutin, G., & Karapetyan, D. (2009). Generalized traveling salesman problem reduction algorithms. *Algorithmic Operations Research, 4*(2), 144–154.
Gutin, G., & Karapetyan, D. (2010). A memetic algorithm for the generalized traveling salesman problem. *Natural Computing, 9*(1), 47–60.
Gutin, G., Yeo, A., & Zverovitch, A. (2007). Exponential neighborhoods and domination analysis for the TSP. Gutin, G., & Punnen, A.P. The traveling salesman problem and its variations, (pp. 223–256).
Hadjicharalambous, G., Pop, P. C., Pyrga, E., Tsaggouris, G., & Zaroliagis, C. D. (2007). The railway travelling salesman problem. *Lecture Notes in Computer Science, 4359*, 264–275.
Hao, Z., Huang, H., & Cai, R. (2008). *Bio-inspired algorithms for TSP and generalized TSP* (pp. 35–62). Traveling Salesman Problem. IntechOpen Book Series
Helsgaun, K. (2015). Solving the equality generalized traveling salesman problem using the Lin-Kernighan-Helsgaun algorithm. *Mathematical Programming Computation, 7*(3), 269–287.
Henry-Labordere, A. L. (1969). The record balancing problem: A dynamic programming solution of a generalized salesman problem. *RIRO, B-2*, 43–49.
Hintsch, T. (2021). Large multiple neighborhood search for the soft-clustered vehicle-routing problem. *Computers & Operations Research, 129*, 105132.
Hu, B., & Raidl, G. R. (2008). Effective neighborhood structures for the generalized traveling salesman problem. In *Proceedings of evolutionary computation in combinatorial optimization, lecture notes in computer science: 4972* (pp. 36–47).
Huang, H., Yang, X., Hao, Z., Wu, C., Liang, Y., & Zhao, X. (2005). Hybrid chromosome genetic algorithm for generalized traveling salesman problems. *Lecture Notes in Computer Science, 3612*, 137–140. Proceedings of Advances in Natural Computation
Kan, J.-M., & Zhang, Y. (2012). Application of an improved ant colony optimization on generalized traveling salesman problem. *Energy Procedia, 17*, 319–325.
Kara, I., Guden, H., & Koc, O. N. (2012). New formulations for the generalized traveling salesman problem. In *Proceedings of the 6th international conference on applied mathematics, simulation, modelling* (pp. 60–65).
Karapetyan, D., & Gutin, G. (2011). Lin-Kernighan heuristic adaptations for the generalized traveling salesman problem. *European Journal of Operational Research, 208*(3), 221–232.
Khachay, M., & Neznakhina, K. (2016). Towards a PTAS for the generalized TSP in grid clusters. In *AIP conference proceedings: vol. 1776* (p. 050003).
Khan, I., & Maiti, M. K. (2018). A novel hybrid algorithm for generalized traveling salesman problems in different environments. *Vietnam Journal of Computer Science, 5*, 27–43.
Laporte, G. (1992). The vehicle routing problem: An overview of exact and approximate algorithms. *European Journal of Operational Research, 59*, 345–358.
Laporte, G., Asef-Vaziri, A., & Sriskandarajah, C. (1996). Some applications of the generalized travelling salesman problem. *Journal of the Operational Research Society, 47*(12), 1461–1467.
Laporte, G., & Martello, S. (1990). The selective travelling salesman problem. *Discrete Applied Mathematics, 26*(2–3), 193–207.
Laporte, G., Mercure, H., & Norbert, Y. (1985). Finding the shortest hamiltonian circuit through n clusters: A lagrangean relaxation approach. *Congressus Numerantium, 48*, 277–290.
Laporte, G., Mercure, H., & Norbert, Y. (1987). Generalized traveling salesman problem through n sets of nodes: The asymmetrical case. *Discrete Applied Mathematics, 18*, 185–197.
Laporte, G., & Norbert, Y. (1983). Generalized travelling salesman problem through n sets of nodes: An integer programming approach. *INFOR: Information Systems and Operational Research, 21*(1), 61–75.
Laporte, G., & Palekar, U. (2002). Some applications of the clustered travelling salesman problem. Journal of the Operational Research Society, 53, 972–976,
Lien, Y.-N., Ma, E., & Wah, B. W. S. (1993). Transformation of the generalized traveling-salesman problem into the standard traveling-salesman problem. *Information Sciences, 74*, 177–189.
Matai, R., Singh, S. P., & Mitall, M. L. (2010). Traveling salesman problem: An overview of applications, formulations, and solution approaches. In D. Davendra (Ed.), *Traveling salesman problem, theory and applications* (pp. 1–26). London: InTech Press.
Matei, O., & Pop, P. C. (2010). An efficient genetic algorithm for solving the generalized traveling salesman problem. In *Proceedings of 6th IEEE international conference on intelligent computer communication and processing* (pp. 87–92).
Meng, L., Lin, Y., Qing, S., & Wenjing, F. (2019). Research on generalized traveling salesman problem based on modified ant colony optimization. In *Proceedings of Chinese control and decision conference* (pp. 4570–4574).
Mestria, M., Ochi, L. S., & de Lima Martins, S. (2013). GRASP with path relinking for the symmetric euclidean clustered traveling salesman problem. *Computers & Operations Research, 40*(12), 3218–3229.
Miller, C. E., Tucker, A. W., & Zemlin, R. A. (1960). Integer programming formulation of travelling salesman problems. *Journal of the ACM, 7*(4), 326–329.
Morán-Mirabal, L. F., González-Velarde, J. L., & Resende, M. G. C. (2014). Randomized heuristics for the family traveling salesperson problem. *International Transactions in Operational Research, 21*(1), 41–57.
Myung, Y. S., Lee, C. H., & Tcha, D. W. (1995). On the generalized minimum spanning tree problem. *Networks, 26*, 231–241.
Nalivajevs, O., & Karapetyan, D. (2019). Conditional Markov chain search for the generalised travelling salesman problem for warehouse order picking. In *Proceedings of 11th computer science and electronic engineering* (pp. 75–78).
Nejma, I. B., & M'Hallah, R. (2021). A beam search for the equality generalized symmetric traveling salesman problem. *RAIRO - Operations Research, 55*(5), 3021–3039.
Noon, C. E. (1988). The generalized traveling salesman problem. In *Unpublished dissertation*. University of Michigan, USA.
Noon, C. E., & Bean, J. C. (1991). A lagrangian based approach for the asymmetric generalized traveling salesman problem. *Operations Research, 39*(4), 623–632.
Noon, C. E., & Bean, J. C. (1993). An efficient transformation of the generalized traveling salesman problem. *INFOR: Information Systems and Operational Research, 31*(1), 39–44.
PassMark Software, CPU BenchmarksAvailable at https://www.cpubenchmark.net/.
Petrovan, A., Matei, O., & Pop, P. C. (2023a). A comparative study between haploid genetic algorithms and diploid genetic algorithms. *Carpathian Journal of Mathematics, 39*(2).
Petrovan, A., Pop, P. C., Sabo, C., & Zelina, I. (2023b). Novel two-level hybrid genetic algorithms based on different Cayley-type encodings for solving the clustered shortest-path tree problem. *Expert Systems with Applications, 215*, 119372.
Pintea, C. M. (2015). A unifying survey of agent-based approaches for equality-generalized traveling salesman problem. *Informatica, 26*(3), 509–522.
Pintea, C. M., Pop, P. C., & Chira, C. (2017). The generalized traveling salesman problem solved with ant algorithms. *Complex Adaptive Modelling Systems, 5*.
Pop, P. C. (2002). *The generalized minimum spanning tree problem*. University of Twente, the Netherlands PhD thesis.
Pop, P. C. (2007). New integer programming formulations of the generalized traveling salesman problem. *American Journal of Applied Sciences, 4*(11), 932–937.
Pop, P. C. (2012). Generalized network design problems. In *Modelling and optimization*. Germany: De Gruyter.
Pop, P. C. (2020). The generalized minimum spanning tree problem: An overview of formulations, solution procedures and latest advances. *European Journal of Operational Research, 283*(1), 1–15.
Pop, P. C., & Iordache, S. (2011). A hybrid heuristic approach for solving the generalized traveling saleasman problem. In *Proceedings of GECCO 2011, association for computing machinery* (pp. 481–488).
Pop, P. C., Matei, O., & Pintea, C. M. (2018a). A two-level diploid genetic based algorithm for solving the family traveling salesman problem. In *Proceedings of GECCO: vol. 2018* (pp. 340–346).
Pop, P. C., Matei, O., & Sabo, C. (2010). A new approach for solving the generalized traveling salesman problem. *Lecture Notes in Computer Science, 6373*, 62–72. Proceedings of HM 2010
Pop, P. C., Matei, O., Sabo, C., & Petrovan, A. (2018b). A two-level solution approach for solving the generalized minimum spanning tree problem. *European Journal of Operational Research, 265*(2), 478–487.
Pop, P. C., Matei, O., & Sitar, C. P. (2013). An improved hybrid algorithm for solving the generalized vehicle routing problem. *Neurocomputing, 109*, 76–83.
Psychas, I. D., Delimpasi, E., & Marinakis, Y. (2015). Hybrid evolutionary algorithms for the multiobjective traveling salesman problem. *Expert Systems with Applications, 42*(22), 8956–8970.
Reihaneh, M., & Karapetyan, D. (2012). An efficient hybrid ant colony system for the generalized traveling salesman problem. *Algorithmic Operations Research, 7*, 22–29.
Reinelt, G. (1991). TSPLIB - a traveling salesman problem library. *ORSA Journal on Computing, 3*(4), 376–384.
Ren, X., Wang, X., Wang, Z., & Wu, T. (2020). Parallel DNA algorithms of generalized traveling salesman problem-based bioinspired computing model. *International Journal of Computational Intelligence Systems, 14*(1), 228–237.
Renaud, J., & Boctor, F. F. (1998). An efficient composite heuristic for the symmetric generalized traveling salesman problem. *European Journal of Operational Research, 108*(3), 571–584.
Renaud, J., Boctor, F. F., & Laporte, G. (1996). A fast composite heuristic for the symmetric traveling salesman problem. *INFORMS Journal on Computing, 8*, 134–143.
Rodríguez-Pereira, J., Fernández, E., Laporte, G., Benavent, E., & Martínez-Sykora, A. (2019). The Steiner traveling salesman problem and its extensions. *European Journal of Operational Research, 278*(2), 615–628.
Rousseau, J.-M. (1985). Private Communication. Centre de recherche sur les transports, University of Montreal,
Salman, R., Ekstedt, F., & Damaschke, P. (2020). Branch-and-bound for the precedence constrained generalized traveling salesman problem. *Operations Research Letters, 48*, 163–166.
Saskena, J. P. (1970). Mathematical model for scheduling clients through welfare agencies. *CORS Journal, 8*, 185–200.

Schmidt, J., & Irnich, S. (2022). New neighborhoods and an iterated local search algorithm for the generalized traveling salesman problem. *EURO Journal on Computational Optimization, 10*, 100029.
Shi, X. H., Lianga, Y. C., Leeb, H. P., Lub, C., & Wang, Q. X. (2007). Particle swarm optimization-based algorithms for TSP and generalized TSP. *Information Processing Letters, 103*, 169–176.
Silberholz, J., & Golden, B. (2007). The generalized traveling salesman problem: A new genetic algorithm approach. In E. K. Baker, A. Joseph, A. Mehrotra, & M. A. Trick (Eds.), *Extending the horizons: Advances in computing, optimization, and decision technologies*. In *Operations Research/Computer Science Interfaces Series: 37* (pp. 165–181).
Slavik, P. (1997). On the approximation of the generalized traveling salesman problem. *Working paper*. University of Buffalo, USA.
Smith, S. L., & Imeson, F. (2017). GLNS: An effective large neighborhood search heuristic for the generalized traveling salesman problem. *Computers & Operations Research, 87*, 1–19.
Snyder, L. V., & Daskin, M. S. (2006). A random-key genetic algorithm for the generalized traveling salesman problem. *European Journal of Operational Research, 174*, 38–53.
Srivastava, S., Kumar, S., Garg, R., & Sen, P. (1969). Generalized traveling salesman problem through n sets of nodes. *CORS Journal, 7*, 97–101.
Sundar, K., & Rathinam, S. (2016). Generalized multiple depot traveling salesmen problem – polyhedral study and exact algorithm. *Computers & Operations Research, 70*, 39–55.
Tang, H., & Miller-Hooks, E. (2007). Solving a generalized traveling salesperson problem with stochastic customers. *Computers & Operations Research, 34*, 1963–1987.
Tasgetiren, M. F., Suganthan, P. N., & Pan, Q. K. (2007a). A discrete particle swarm optimization algorithm for the generalized traveling salesman problem. In *Proceedings of GECCO 2007* (pp. 158–167).
Tasgetiren, M. F., Suganthan, P. N., & Pan, Q. K. (2007b). A genetic algorithm for the generalized traveling salesman problem. In *Proceedings of CEC 2007* (pp. 2382–2389).
Thang, T. B., Long, N. B., Hoang, N. V., & Binh, H. T. T. (2021). Adaptive knowledge transfer in multifactorial evolutionary algorithm for the clustered minimum routing cost problem. *Applied Soft Computing, 105*, 107253.
Wang, M., Ishwar, P., Konrad, J., Gazen, C., & Saboo, R. (2012). Coherent image selection using a fast approximation to the generalized traveling salesman problem. In *Proceedings of the 20th ACM international conference on multimedia* (pp. 981–984).
Wu, C. G., Liang, Y. C., Lee, H. P., & Lu, C. (2004). Generalized chromosome genetic algorithm for generalized traveling salesman problems and its applications for machining. *Physical Review. E, Statistical, Nonlinear, and Soft Matter Physics, 70*, 1–13.
Yang, J., Shi, X., Marchese, M., & Liang, Y. (2008). An ant colony optimization method for generalized TSP problem. *Progress in Natural Science, 18*, 1417–1422.
Yuan, Y., Cattaruzza, D., Ogier, M., Rousselot, C., & Semet, F. (2020a). Mixed integer programming formulations for the generalized traveling salesman problem with time windows. *4OR, 19*, 571–592.
Yuan, Y., Cattaruzza, D., Ogier, M., & Semet, F. (2020b). A branch-and-cut algorithm for the generalized traveling salesman problem with time windows. *European Journal of Operational Research, 286*, 849–866.
Zhao, X., Lin, J. L., & Lu, X. Q. (2006). Void vertex genetic algorithm for the second kind of generalized traveling salesman problems. *Journal of Information and Computing Science, 1*(5), 259–265.
Zia, M., Cakir, Z., & Seker, D. Z. (2018). Spatial transformation of equality - generalized travelling salesman problem to travelling salesman problem. *International Journal of Geo-Information, 7*, 115.