A Hybrid Metaheuristic Algorithm for the Integrated Vehicle Routing and Three-Dimensional Container-Loading Problem

Christos D. Tarantilis, Emmanouil E. Zachariadis, and Chris T. Kiranoudis

***Abstract*—This paper examines a recently addressed practical variant of the capacitated vehicle routing problem (VRP) called the Capacitated Vehicle Routing Problem with 3-D Loading Constraints (3L-CVRP). This problem considers customer demand to be formed by 3-D rectangular items. Additional loading constraints often encountered in real-life applications of transportation logistics are imposed on the examined problem model. In addition to 3L-CVRP, we also introduce and solve a new practical problem version that was dictated by a transportation logistics company and covers cases in which transported items are manually unloaded from the loading spaces of the vehicles. Both problem versions are solved by a hybrid metaheuristic methodology that combines the strategies of tabu search (TS) and guided local search (GLS). The loading characteristics are tackled by employing a collection of packing heuristics. The proposed algorithm's robustness was tested for both problem versions, solving benchmark instances derived from the literature and new benchmark problems with diverse features in terms of customer set size and transported-item dimensions. It produced fine results, improving most of the best solutions that were previously reported.**

***Index Terms*—Fleet management, guided local search (GLS), tabu search (TS), vehicle routing and packing integration.**

I. Introduction

In THIS paper, we examine a practical variant of the standard version of the vehicle routing problem (VRP) [1] in which customer demand is formed by a set of 3-D, rectangular, and weighted items. Although routing problems with 3-D loading constraints are frequently encountered in real-life applications, only recently did Gendreau *et al.* [2] publish the first paper introducing the Capacitated Vehicle Routing Problem with 3-D Loading Constraints (3L-CVRP). Briefly, 3L-CVRP aims to design the least cost route set for covering customer demand and determine a feasible packing of the transported items into the available loading space.

The standard version of the capacitated VRP (CVRP) is a widely studied combinatorial optimization problem [3]. It calls for the minimization of the cost of satisfying the customer demand using a fleet of vehicles operating from a central station. In particular, the CVRP model considers a number of geographically dispersed customers with known demands and a central depot that acts as the base of a homogeneous fleet of vehicles with a fixed maximum carrying load. Moving between a pair of points is associated with a cost that may reflect the distance, travel time, or actual cost of implementing this transition. The goal of the problem is to design a set of routes (one route per vehicle) that start and terminate at the central depot such that the demand of customers is fully satisfied, each customer is visited exactly once by a single vehicle, the total demand of the customers that are assigned to a route does not exceed carrying capacity of the vehicle, and the total cost of the designed route set is minimized. Except for the classic static routing problems (CVRP variants), recent advances in information technology allow dynamic characteristics of the network to be taken into account when the optimal routing policies are determined [4], [5].

As previously mentioned, apart from the constraints of the standard CVRP model, the examined problem aims at determining a feasible—nonoverlapping—packing of the transported items into the loading spaces (containers) of the vehicles. It considers a variety of operational constraints that often arise in practical situations and involve the stability of the packing, the unloading process, and the fragility of the items. Thus, 3L-CVRP can be seen as a generalization of the CVRP; aside from the weight attribute, the physical dimensions of the transported goods are also considered. Packing the items into the containers is closely related to nondeterministic polynomial-time hard (NP-hard) 3-D bin packing (3-D-BPP) [6], which consists of packing a given set of 3-D rectangular items into the minimum number of identical bins. The problem presented in this paper is reduced to the CVRP in special cases involving items with zero volume or loading spaces with infinite dimensions for which the loading feasibility is always guaranteed.

The examined problem is of great value from both the theoretical and commercial viewpoints. Concerning theoretical importance, 3L-CVRP is a challenging problem to solve, as it is composed of two NP-hard combinatorial optimization problems, i.e., the VRP (routing aspects) and the 3-D-BPP (packing aspects). From the commercial point of view, modeling the container transportation of goods, which plays a central role in the field of distribution/collection management, is a particularly interesting problem and has become increasingly popular in recent years mainly because of the fact that it favors intermodal means of transportation involving road, rail, and sea freight networks.

Manuscript received August 30, 2007; revised February 16, 2008 and September 4, 2008. First published April 28, 2009; current version published June 2, 2009. The Associate Editor for this paper was A. Erera.

C. D. Tarantilis is with the Department of Management Science and Technology, Athens University of Economics and Business, 11362 Athens, Greece (e-mail: tarantil@ aueb.gr).

E. E. Zachariadis and C. T. Kiranoudis are with the School of Chemical Engineering, National Technical of Athens, 15780 Athens, Greece (e-mail: ezach@ ntua.gr; kyr@chemeng.ntua.gr).

Digital Object Identifier 10.1109/TITS.2009.2020187

In terms of published solution approaches for 3L-CVRP, Gendreau *et al.* [2] tackled the routing features of the problem by employing the tabu search (TS) method, which relocates customers using the GENI [7] heuristic. The loading constraints are handled by applying two packing heuristics for solving the 3-D strip-packing problem. Again, TS is applied for searching through the possible orders by which items are inserted into the container. The performance of their algorithmic methodology was evaluated on benchmark instances derived from the CVRP literature and was modified to express customer demand as a set of 3-D rectangular items.

Combinations of the CVRP with additional loading constraints have also been investigated in the case of 2-D loading surfaces and items [CVRP with 2-D Loading Constraints (2L-CVRP)]. Iori *et al.* [8] presented an exact methodology that employs a branch-and-cut algorithm for the routing aspects of the problem and a branch-and-bound algorithm for determining the loading feasibility of an item set. The solution approach of Iori *et al.* [8] is applicable to problems involving up to 30 customers and 90 items. Gendreau *et al.* [9] designed a TS methodology for solving larger scale 2L-CVRP instances. Zachariadis *et al.* [10] proposed a metaheuristic development that deals with the routing aspects of 2L-CVRP by hybridization of TS and guided local search (GLS), whereas the loading constraints are tackled by applying a series of packing heuristics. Moving from the 2L-CVRP [10] to the 3L-CVRP model raised numerous challenges in terms of the packing aspects of the algorithm. Specifically, 3L-CVRP considers the transportation of 3-D items in containers so that item stacking is allowed. Stacking of items both incorporates additional loading constraints (i.e., stability and fragility issues) and complicates some of the constraints imposed by 2L-CVRP (e.g., unloading issues and rotation of items), as will be discussed later in detail. Therefore, our primary aim is to effectively tackle these complex packing requirements while keeping the computational effort of our algorithm to manageable levels. Another study on a combined routing and loading problem was published by Doerner *et al.* [11].

The purpose of this paper is to study combined routing and 3-D-packing problems. Specifically, apart from 3L-CVRP, a new practical problem version is also introduced. This version was dictated by a major company in the transportation industry and is suited for cases in which transported items are not of excessive size or weight, so that they can manually be unloaded from the containers [Capacitated Vehicle Routing Problem with Manual 3-D Loading Constraints (M3L-CVRP)]. We propose an effective hybrid metaheuristic development for solving both problem versions (3L-CVRP and M3L-CVRP). Regarding the routing characteristics, the proposed algorithm explores the solution space using a TS methodology [12]. To help the algorithm escape from the local optima encountered and move along diverse trajectories of the solution space, the conducted search is coordinated by a guiding mechanism that periodically modifies the objective function of the problem. The objective function alteration mechanism incorporates the rationale of GLS [13] in the search process and works by penalizing low-quality features of the candidate solutions. Both TS and GLS have successfully been used for solving a routing variant, which considers vehicles to be replenished at intermediate stops [14]. However, the algorithmic framework of the work involved the execution of a pure GLS method for improving the solution obtained by a hybrid TS within a variable neighborhood search [15] block. To determine whether a candidate route is feasible in terms of the loading constraints posed by the problem, we employ a bundle of heuristics that aim at generating diverse packing structures to increase the probability of achieving feasible loadings. To reduce the required computational time, as the search progress evolves, the obtained loading feasibility information is recorded in a tree-based memory structure. To test the robustness of the proposed method, we developed new benchmark instances involving diverse customer populations and item dimensions. Our algorithmic framework was employed to solve instances derived from the literature and our new benchmark problems for both problem versions. It produced fine results, significantly improving the majority of the best solutions previously reported.

The remainder of this paper is outlined as follows: In Section II, a description of the examined problem is provided. In Section III, all of the proposed algorithmic features are discussed in detail. Section IV presents the experimental computational results and introduces our proposed new benchmark instances. Finally, Section V concludes this paper.

II. Problem

Following the notation of Gendreau *et al.* [2], the 3L-CVRP variant is defined on a graph $G = (N, A)$, where N is the vertex set containing the central depot and n geographically dispersed customers, and $A = \{(i, j) : i, j \in N, i \neq j\}$ is the arc set. A travel cost c_{ij} corresponding to the required travel cost of transiting from i to j is associated with each arc (i, j). The demand of each customer i is formed by a set IT_i of m_i 3-D items having total weight d_i and total volume s_i. The length, width, and height dimensions of an item $I_{ik} \in IT_i$ $(k = 1, 2, \ldots, m_i)$ are denoted by l_{ik}, w_{ik}, and h_{ik}, respectively. A fragility status fr_{ik} (whose value is 1 if item I_{ik} is fragile and 0 otherwise) is associated with each item I_{ik}. The demanded goods are transported by a fleet of v homogeneous vehicles that are available at the central depot. The vehicles have a maximum carrying load Q and a loading space of length L, width W, and height H, respectively. The goal of 3L-CVRP is to design the minimum cost set of routes satisfying five constraints.

1) The size of the generated route set must not exceed the number of available vehicles (at most one route per vehicle).
2) Every route must originate and terminate at the central depot.
3) The demand of all customers must fully be satisfied.
4) Every customer must be visited once by exactly one vehicle.
5) The total demand weight of the customer set assigned to any route must not exceed vehicle capacity Q.

In terms of the loading aspects of the problem, for every item set transported by a vehicle, five constraints must also hold.

1) There must be an orthogonal nonoverlapping loading of the items into the vehicle's loading space.

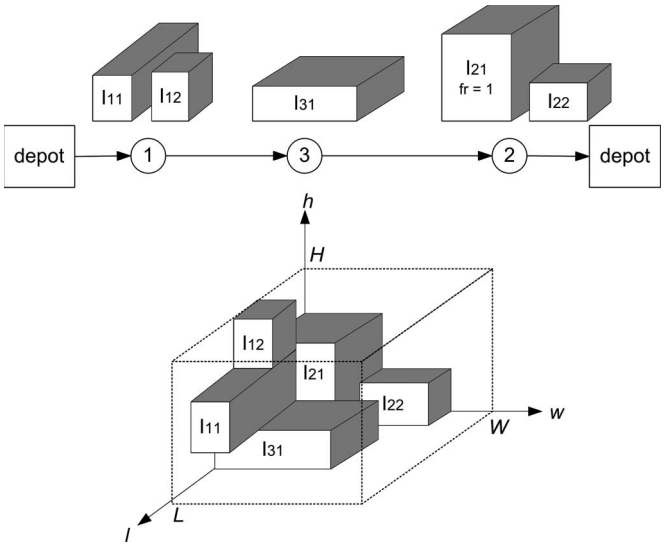

Fig. 1. Feasible 3L-CVRP route example.

2) Items must be packed with a fixed vertical orientation.
3) Every item must be unloaded with a straight movement parallel to the horizontal plane (*last-in–first-out (LIFO) policy*).
4) Every item's bottom surface must sufficiently be supported by another item's top surface or by the container's bottom surface (*supporting-area constraint*).
5) No nonfragile item can be stacked on any fragile item (*fragility constraint*).

Fig. 1 shows a feasible route for the 3L-CVRP problem. In particular, the route covers customers 1, 3, and 2. Customer 1 demands items I_{11} and I_{12}, customer 3 demands a single item I_{31}, and customer 2 demands two items, i.e., I_{21} and I_{22}. Note that the only fragile item transported by the vehicle is I_{21}, as indicated by its fragility status fr. As shown in Fig. 1, the container is placed with its bottom–back–left corner at position (0, 0, 0) of a 3-D coordinate system. The W, L, and H edges of the loading space are parallel to the w-, l-, and h-axis, respectively. The container's loading/unloading door lies at the $W \times H$ rectangle originating from point (0, L, 0) (the container's front end or the vehicle's rear end).

Concerning the loading constraints imposed, loading constraint 2 ensures that all items are packed with a fixed vertical orientation; however, they can be rotated 90° on the horizontal plane. The LIFO policy (loading constraint 3) guarantees that all items can directly be delivered into the container without any repositioning of the other items. This constraint frequently arises in practical situations of freight distribution when the weight of items makes repositioning impossible or in the case of automated unloading. Thus, if item I_{ik} is to be delivered before another item I_{jl}, item I_{jl} must not obstruct the direct unloading of I_{ik}. To ensure the stability of the loading patterns, supporting-area loading constraint 4 is introduced. Every item's bottom surface must partially lie on either the bottom of the container or the top surfaces of other items. In particular, an item I_{ik} is considered to be sufficiently supported if its bottom surface lies on top of a supporting surface with an area of not lower than $a \cdot w_{ik} \cdot l_{ik}$, where a is the supporting-area factor. The fragility constraint ensures that no nonfragile item is stacked on any fragile item. On the contrary, fragile items can be stacked on both fragile and nonfragile items.

To better present the loading constraints 2–5 of the problem formulation, we provide Fig. 2, which shows three packing examples (for the route shown in Fig. 1) that violate the aforementioned loading constraints. The packing of Fig. 2(a) violates the fragility constraint, as the nonfragile item I_{22} is stacked on the fragile item I_{21}. In the case of Fig. 2(b), the supporting-area constraint is violated due to the fact that item I_{22} is supported by neither the top surface of any other item nor the bottom surface of the container. Finally, the loading pattern of Fig. 2(c) does not satisfy the LIFO-policy constraint. In particular, two violations occur: I_{21} is stacked on I_{31}, obstructing the direct unloading of I_{31}, and I_{22} is placed between item I_{12} and the unloading door, blocking the direct unloading of I_{12}.

In addition to 3L-CVRP, a new problem version called M3L-CVRP is introduced. The M3L-CVRP model is distinguished from that of Gendreau *et al.* [2] in terms of the LIFO policy adopted. Specifically, Gendreau *et al.* [2] considered that a loading pattern violates the LIFO constraint if an item A is placed under item B, A is to be delivered before B, and A and B have overlapping projections on the w–l plane [see Fig. 3(a)]. This policy is suited for scenarios that involve the use of forklifts to unload the transported items. In such cases, the delivered items are first elevated and then moved toward the unloading door. On the contrary, the M3L-CVRP version considers a packing to be infeasible only if item B is stacked on A (i.e., the top surface of item A touches the bottom surface of B), item A is to be delivered before B, and A and B have overlapping projections on the w–l plane [see Fig. 3(b)]. The M3L-CVRP version models cases in which transported items are manually unloaded from the vehicle loading space; therefore, they are not necessarily elevated before pulling them out of the container. Note that, for both problem versions, a packing is considered to violate the LIFO policy if item B, which is to be delivered after item A, is packed between A and the unloading door of the vehicle, and items A and B have overlapping projections on the w–h plane.

III. Proposed Solution Approach

As previously mentioned, 3L-CVRP is composed of two highly complex combinatorial optimization problems (i.e., CVRP and 3-D-BPP). Exact algorithmic methodologies are inapplicable of solving real-life problems of large customer and item sets. Therefore, our interest is focused on metaheuristic methodologies that are able to produce near-optimal solutions within reasonable computing times. The routing aspects of the examined problem were tackled by an algorithmic framework that incorporates the search strategies of two powerful metaheuristic approaches, i.e., TS and GLS. As far as the loading features of the problem are concerned, a collection of six packing heuristics is employed, each of which is designed to favor a different packing structure to increase the probability of achieving feasible loadings of the transported items into the containers.

The proposed algorithmic framework is motivated by the successful application of a scheme that is similar to 2L-CVRP [10]. The components of the aforementioned scheme are not

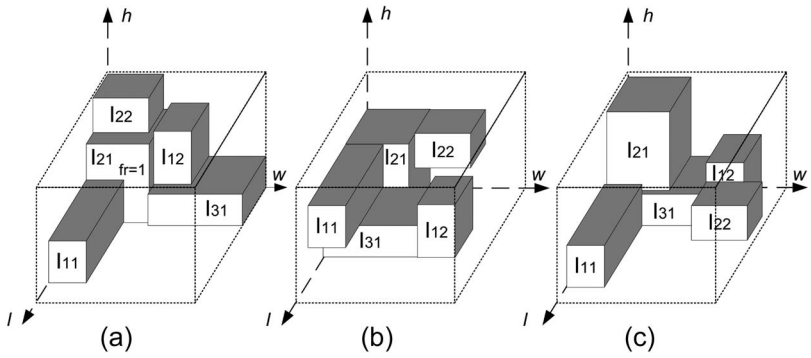

Fig. 2. Loading constraints violations for the route shown in Fig. 1. (a) Fragility violation. (b) Supporting-area violation. (c) LIFO violation.

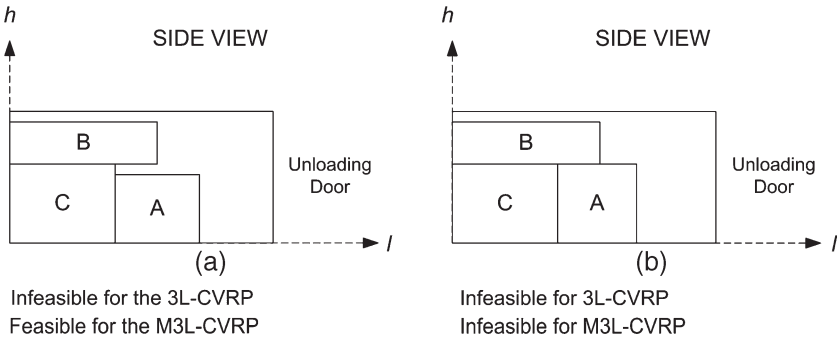

Fig. 3. LIFO policies of the examined problem versions.

problem specific; therefore, they can confidently be transferred to the more complex 3L-CVRP. Regarding the routing aspects, both TS and GLS have effectively been applied to numerous vehicle routing variants. Their hybridization aims at successfully balancing the intensification and diversification of the search process, so that a thorough exploration of the solution space is performed. In terms of the loading constraints, although the problem examined in [10] imposes fewer and simpler loading requirements, compared with those of 3L-CVRP, a similar bundle of packing heuristics is used mainly because of its simple structure and computational economy. In this section, we provide an analytic description of the proposed packing heuristics, followed by a presentation of the guided TS (GTS) methodology.

A. Packing Heuristics Bundle

To determine whether customer sequence $CS(k)$, which is assigned to vehicle k, is feasible in terms of the loading constraints of the problem, we employ the following procedure, aiming at feasibly packing the transported item set $I(k) = \bigcup_{i \in CS(k)} IT_i$ into the loading space of vehicle k: We create three orders $Ord_t, t = 1, 2,$ and 3 of the items present in set $I(k)$. For all three orders, the items are sorted in descending order of visit, breaking ties based on ascending order of fragility status, where the order of visit denotes the position of the corresponding customer within its route. The orders are different in terms of the second tie breaker used: For Ord_1, Ord_2, and Ord_3, items are sorted in descending order of volume $(w \cdot l \cdot h)$, bottom area $(w \cdot l)$, and height (h), respectively. The aforementioned sortings correspond to the order in which items are loaded into the container. In all cases, the items delivered first are loaded last, as dictated by the LIFO policy. Note, however, that, if the LIFO policy is not considered, the order of visit is not taken into account when generating the item orders. Regarding the fragility of the items, nonfragile items precede fragile items, so that the former's top surface can be used for hosting subsequent items of any fragility status. Ord_1 aims at packing larger boxes first to occupy the initially available wide space. Ord_2 and Ord_3 facilitate stacking of items. Loosely speaking, the use of Ord_2 entails loading items with a large bottom area first to adequately support subsequent items; with Ord_3, short items are placed last so that they can be loaded on top of previously inserted items. Items are successively picked from set Ord_1 to be inserted into the container using the first packing heuristic $Heur_1$. For each item and for both item orientations, all available loading positions are scanned. Note that, in the beginning, the only available loading position lies at the bottom–left–back corner of the container (0, 0, 0). The item is inserted into a position that does not lead to any loading constraint violation and satisfies the packing criterion dictated by the heuristic that is currently employed, i.e., $Heur_1$. When an item occupies the selected position, the list of available loading positions is updated, and new spaces become available for the subsequent items. If all items of Ord_1 are feasibly packed into the container, the route is considered to be feasible in terms of the loading positions of the problem. In the opposite case, if, for any item, no feasible loading position is available, the container is emptied, and the insertion procedure is again initiated using the next packing heuristic $Heur_{i+1}$. If none of the six available heuristics succeeds in producing a feasible loading, the heuristic bundle is reapplied to the next item

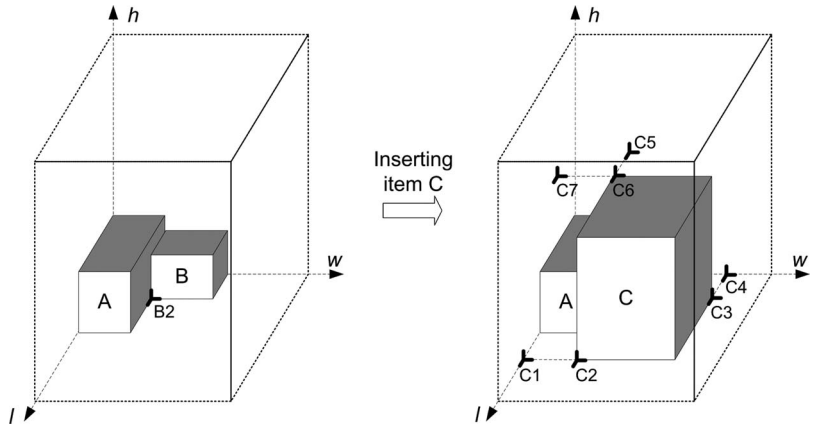

Fig. 4. Loading an item into the container.

order Ord_2. If no feasible packing is achieved for any of the three item orders, the investigated route is considered to be infeasible in terms of the loading constraints.

As previously mentioned, the insertion of an item results in the generation of new loading positions for hosting the following items. The rationale of managing the available loading positions for the inserted items is similar to the corner-point approach in [6], which was modified to deal with the loading constraints posed by the problem examined. Fig. 4 shows a sample item insertion, where item C is placed at loading position $B2$. In particular, $B2$ is erased from the list of available loading positions, whereas seven new loading positions ($C1$–$C7$) are generated and become available to accommodate subsequent items: $C1$ is created at the leftmost nonoccupied point of the item's bottom–front edge projection $(0, l_B + l_C, 0)$. $C2$ lies at the bottom–left–front corner of the inserted item $(w_A, l_B + l_C, 0)$. Position $C3$ is generated at the bottom, right, back corner of C $(w_A + w_c, l_B, 0)$. $C4$ is created at the backmost nonoccupied point of the inserted item's right–bottom edge projection $(w_A + w_c, 0, 0)$. Loading position $C5$ lies at the backmost available point of the item's top–left edge projection $(w_A, 0, h_C)$, whereas $C6$ lies at the top–back–left corner of C (w_A, l_B, h_C). Finally, loading position $C7$ lies at the leftmost nonoccupied point of the inserted item's top–back edge projection $(0, l_B, h_C)$. Note that we do not generate a position at $(0, 0, h_C)$, because this loading position rarely offers an adequate supporting area for subsequent items. Two more loading positions are generated when the inserted item is not placed at $h = 0$: The first loading position lies at the lowest nonoccupied point of the item's left–front edge projection, whereas the second loading position is generated at the lowest nonoccupied point of the item's right–back edge projection. When the list of available loading positions is updated, any duplicate entries are removed. Moreover, loading positions in the space occupied by the inserted item are also deleted. This process of updating the available loading positions aims to keep track of possible empty spaces between inserted items so that they can later be filled to obtain dense packing patterns.

The criteria for assigning each item into a specific loading position are dictated by the aforementioned six packing heuristics $Heur_i$, $(i = 1, 2, \ldots, 6)$, each of which aims at generating a different packing structure. As previously stated, the selected loading position for a given item must not lead to any fragility, supporting-area, or LIFO-policy constraint violation. Since 3L-CVRP and M3L-CVRP are different in terms of the LIFO policy adopted, the packing heuristics examine the LIFO feasibility of a specific position by employing the appropriate criterion for each problem version. The underlying mechanism of all six packing heuristics is provided here.

1) *BackLeftLow* $(Heur_1)$: The position with the minimum l-axis coordinate is selected, breaking ties with the minimum w- and h-axis coordinates. Employing this criterion, the packing tends to evolve in the form of layers that are parallel to the w–h plane.
2) *LeftBackLow* $(Heur_2)$: The position with the minimum w-axis coordinate is selected, breaking ties with the minimum l- and h-axis coordinates. Using this heuristic, the packing tends to evolve in the form of layers that are parallel to the l–h plane.
3) *MaxTouchingAreaW* $(Heur_3)$: The selected position is that which maximizes the inserted item's touching area. The total touching area of an item is obtained by summing the area of its surface that is adjacent to either the surface of the items that were already placed or the container walls. To favor stacking of items, the bottom surface of the container is not taken into account. In addition, the common surface of two fragile items, with one item placed on top of the other, is doubled to take advantage of the top surface of the fragile items. In the case of equal touching areas, the position with the minimum w-axis position is selected to break the tie. Using this heuristic, items tend to be initially spread out at the container's boundaries and later fill the inner part of the loading space. Regarding the tie breaker employed, it drives the packing to evolve in the direction of the w-axis.
4) *MaxTouchingAreaNoWallsW* $(Heur_4)$: This heuristic operates with the exact same mechanism as that in the case of the *MaxTouchingAreaW* heuristic. As its name suggests, the walls of the containers are not taken into account when calculating the touching area of an inserted item. Following this rationale, the items tend to be initially packed into the inner parts of the container and later be spread out at the container's boundaries.

5) *MaxTouchingAreaL* ($Heur_5$): This packing method shares the same mechanism with the *MaxTouchingAreaW* heuristic. In this case, the position lying at the minimum l-axis coordinate is selected to break the tie. The aforementioned criterion favors the evolution of the packing in the direction of the l-axis.
6) *MaxTouchingAreaNoWallsL* ($Heur_6$): The last heuristic employed has the same behavior as that of the *MaxTouchingAreaL* method, which is apart from the fact that the walls of the container are not taken into account when calculating the touching area of an inserted item.

The heuristics of the bundle are sorted in increasing order of complexity. First, the simple $Heur_1$ and $Heur_2$ are executed, followed by $Heur_3$–$Heur_6$, which require additional computational effort for the calculation of touching perimeters. Preliminary experiments indicated that the aforementioned order reduces the total CPU time required by the heuristic bundle since, for the cases in which a feasible loading pattern can easily be obtained, it is preferable to first use the fastest heuristics. If these heuristics fail to produce a feasible packing, they are followed by the slower but more effective (see Section IV-E) $Heur_3 - Heur_6$.

B. Constructing an Initial Feasible Solution

The solution approach is initiated by employing a construction algorithm to generate an initial solution that is going to be improved by the application of the proposed metaheuristic methodology. This initial solution must respect all of the constraints imposed by the problem formulation. The feasibility of a solution heavily depends on the loading characteristics of the designed routes. Therefore, to ensure the construction of a feasible solution, our major goal is to maximize the volume utilization of the available containers. The actual quality of the produced solution, as far as the routing features are concerned, is of secondary importance for the construction algorithm. The proposed methodology managed to successfully produce feasible solutions for all test problems and for both problem versions.

The mechanism of the construction algorithm works as follows: Customers are sorted in descending order of demanded item volume s_i, and v new empty-of-customer routes originating and terminating at the central depot are generated (one route per vehicle). Customers are successively picked to be assigned to some route. The feasibility of inserting customer i into any position, of all the available routes, is investigated using the packing heuristic bundle. Let S_{feas} denote the set of routes to which customer i can feasibly be assigned. For every route $j \in S_{\text{feas}}$, the quantity $freeVolume_j$ is calculated, where $freeVolume_j$ denotes the nonoccupied volume of the container of route j. The customer is assigned to the route that yields the minimum $freeVolume_j$ value. Regarding the insertion position within the selected route, the customer is inserted into the point that minimizes the objective function increase.

As previously mentioned, the construction heuristic managed to build feasible initial solutions in terms of both the loading constraints and the fleet size for all test instances and for both problem versions. However, if the construction method fails to generate such a solution, additional vehicles become available so that all produced routes satisfy the loading constraints, as this is the only precondition for executing the improvement metaheuristic. Then, the proposed algorithm is applied, having as primary objective the elimination of these additional routes.

C. Defining Neighborhood Structures

The proposed metaheuristic methodology, as every local search procedure, works by performing moves for transiting between solutions and exploring the solution space. We consider three move types, with each defining a neighborhood structure $NS_i, i = 1, 2,$ and 3.

1) *Customer relocation (NS1)*: This move type relocates a customer from its current position to another [1]. Every possible reinsertion position is taken into consideration.
2) *Customer swap (NS2)*: This move type exchanges the positions of a given customer pair [1]. Every possible customer pair is taken into consideration.
3) *Route interchange (NS3)*: The mechanism of the route interchange move (2-opt) [1] depends on the route pair involved. If the move is performed within a single route, two arcs are removed, two new arcs are created, and the path lying between the created arcs is reversed. In the case where the move is applied between a route pair, the routes are divided into their starting and terminating sections. The initial section of the first route is connected to the terminating section of the second route, and *vice versa*, by replacing a pair of arcs. Every arc pair is considered a candidate for replacement.

D. GTS Methodology

The initial feasible solution produced by the construction algorithm is improved by the proposed metaheuristic methodology called GTS. The GTS methodology explores the solution space by embodying the GLS principles into the TS framework. A guiding mechanism, based on the GLS, coordinates the search process of a TS procedure by controlling the objective function of the problem. The purpose of this guiding mechanism is twofold: It induces diversification by driving the algorithm to unexplored search trajectories and aims to eliminate low-quality features from the final solution obtained.

The GTS metaheuristic begins from the initial solution of the construction heuristic and works as follows: At each iteration, a move type is stochastically selected (NS_i). All three move types have the same probability of being chosen. Neighborhood structure NS_i is fully explored, and the method implements the most economical move that does not violate any of the constraints posed by the problem model and is not declared tabu, unless it leads to the highest quality solution ever encountered (*aspiration criterion*). The loading feasibility of all tentative solutions is investigated by employing the packing heuristic bundle presented in Section III-A. To avoid any cycling phenomena caused by the deterministic criterion of moving to subsequent solutions, when a move is performed, its reversal is declared tabu for a number of iterations equal to *tabuTenure*.

For every *guidFreq* completed iterations of the GTS framework, the guiding mechanism is initiated. As for the GLS method, a set of undesirable solution features must be defined and penalized. In the case of the GTS methodology, as well as numerous routing problem approaches, expensive (long) arcs are not considered to be part of a near-optimal solution. Therefore, long arcs present in the candidate solution are located, and their cost is penalized to be eliminated from the final solution. The arc selected to be penalized is that which maximizes the following utility function: $U(i,j) = (c_{ij}/avg_{ij})/(1+p_{ij})$, where avg_{ij} denotes the average cost of all arcs originating from vertices i and j in arc set A and p_{ij} is equal to the number of times that arc (i,j) has been selected for penalization. The proposed utility function not only selects the longest arc of the solution but also takes into account the vertex topology by considering the positions of i and j relative to the rest of the customer population (term avg_{ij}). This behavior results in a fairer and more balanced arc selection strategy [14]. Let arcPen denote the term selected to be penalized. The cost of arcPen is doubled for a penalization horizon that is equal to $(guidFreq/2)$ number of iterations before being restored to its original value. Using this penalization policy, the methodology is driven to search trajectories that do not contain the penalized arc for a number of iterations that is at least equal to the penalization horizon. If, during the penalization period, a tentative solution improves the highest quality solution ever encountered, the penalization policy is overridden. The GTS methodology is terminated after *maxNoImp* nonimproving iterations by returning the highest quality solution ever encountered during the search.

E. Acceleration Strategies

To restrict the computational time required by the GTS methodology, we designed two accelerating strategies: The first strategy reduces the size of the neighborhoods explored, whereas the second strategy records the loading feasibility of all encountered routes.

As far as the neighborhood reduction policy is concerned, the moves defined by the selected neighborhood structure NS_i are evaluated, only if they result in the generation of arcs linking two neighboring vertices. In detail, for every vertex k, quantity avg_k is obtained as the average cost of all arcs beginning from k in arc set A. Then, the set of neighboring vertices of k, which is denoted by NV_k, is composed of every vertex m, such that $c_{km} \leq avg_k$. Note that the central depot is considered to be a neighbor of every customer. A solution's neighborhood structure NS_i is reduced to RNS_i by considering only the moves that lead to the connection of vertices k and m such that $k \in NV_m$ and $m \in NV_k$. Thus, at each GTS iteration, the search explores the limited RNS_i of the current solution.

The second policy of reducing the computational time demanded by the GTS methodology employs a memory structure that records the loading feasibility information of all previously encountered routes. Since the packing heuristic bundle is a major contributing factor to the overall computational effort required by the proposed methodology, any duplicate executions of these packing heuristics must be eliminated. During the search process, the loading feasibility of all tentative solutions is investigated using the packing heuristic bundle. Already examined routes may repeatedly be encountered as the search evolves. Therefore, once the loading feasibility of a given route has been checked, the corresponding information is stored in a memory structure to be available for later stages of the search. To reduce the time needed for accessing and retrieving the obtained loading information, the memory structure was designed in the form of a sorted tree.

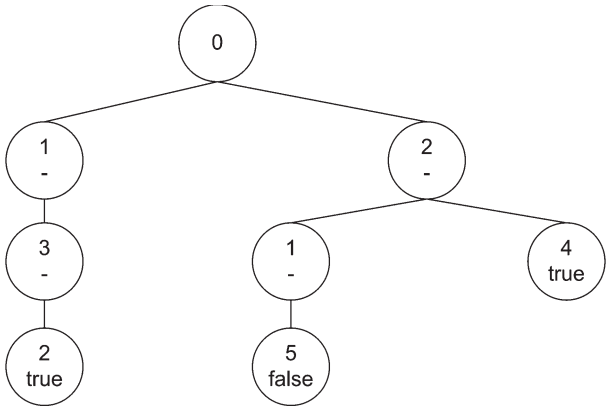

Fig. 5. Recording the loading feasibility.

Fig. 5 shows an example of the memory structure employed, particularly in the case where the packing heuristic bundle achieved a feasible loading pattern for routes 0-2-4-0 and 0-1-3-2-0, and no such pattern was generated for route 0-2-1-5-0. If these routes are to be reencountered during the search progress, their loading feasibility is going to be retrieved from the memory structure rather than determined by the application of the packing heuristic bundle. Recording and retrieving the obtained loading information resulted in a significant decrease in the total GTS computational time, which ranged from 64% for the smallest instances involving up to 15 customers to 35% for the largest instances with 125 customers. Note that, as the search process evolves, the required physical memory for storing the loading feasibility information may exceed the memory availability of the computer executing the GTS algorithm. This danger becomes greater in the case of larger scale instances when the cardinality of the potential routes becomes higher. Therefore, to avoid any memory overflow situations, the tree-based structure may need to be destroyed and rebuilt from the beginning after a fixed number of completed GTS iterations. Fixing the destruction period at 5000 iterations worked fine when executing the GTS algorithm on a computer with 1 GB of RAM, even for the largest scale instances.

IV. Computational Result

The proposed metaheuristic solution approach was coded in Visual C# and executed on a Pentium IV 2.8-GHz computer with 1 GB of RAM. To determine the standard setting for all algorithmic parameters, we conducted extensive tests, solving both the instances of Gendreau *et al.* [2] and the new test problems that we developed. Then, the performance of the GTS methodology was evaluated using the standard parameter setting. Note that, to directly compare the performance of the

TABLE I
DEMANDED ITEM SET CHARACTERISTICS OF THE NEW BENCHMARK INSTANCES

	m_{min}	m_{max}	d_{min}	d_{max}
Class 1	2	4	0.2	0.4
Class 2	1	2	0.4	0.6
Class 3	1	3	0.1	0.7

m_{min}: minimum number of demanded items per customer, m_{max}: maximum number of demanded items per customer, d_{min}: factor used to calculate the lower bound of the item dimensions, d_{max}: factor used to calculate the upper bound of the item dimensions

TABLE II
BENCHMARK INSTANCES SOLVED

Gendreau et al. instances				New instances			
Instance	***n***	***M***	***v***	**Instance**	***n***	***M***	***v***
E016-03m	15	32	5	**50-1**	50	143	9
E016-05m	15	26	5	**50-2**	50	73	14
E021-04m	20	37	5	**50-3**	50	97	10
E021-06m	20	36	6	**75-1**	75	227	14
E022-04g	21	45	7	**75-2**	75	114	20
E022-06m	21	40	6	**75-3**	75	142	15
E023-03g	22	46	6	**100-1**	100	309	17
E023-05s	22	43	8	**100-2**	100	149	29
E026-08m	25	50	8	**100-3**	100	189	20
E030-03g	29	62	10	**125-1**	125	379	21
E030-04s	29	58	9	**125-2**	125	184	35
E031-09h	30	63	9	**125-3**	125	246	24
E033-03n	32	61	9				
E033-04g	32	72	11				
E033-05s	32	68	10				
E036-11h	35	63	11				
E041-14h	40	79	14				
E045-04f	44	94	14				
E051-05e	50	99	13				
E072-04f	71	147	20				
E076-07s	75	155	18				
E076-08s	75	146	19				
E076-10e	75	150	18				
E076-14s	75	143	18				
E101-08e	100	193	24				
E101-10c	100	199	28				
E101-14s	100	198	25				

n: number of customers, ***M***: number of transported items, ***v***: size of the available fleet of vehicles

GTS algorithm to the solution approach of Gendreau *et al.* [2], we set the supporting-area factor $a = 0.75$, as in the work of Gendreau *et al.* [2]. All benchmark instances and analytic solutions obtained by the GTS methodology are available at www.msl.aueb.gr/management_science/networks.htm.

A. Benchmark Instances

Gendreau *et al.* [2] introduced 27 instances for routing problems with 3-D loading constraints. These test problems were derived from previous CVRP instances [1] by expressing the customer demand in the form of 3-D rectangular items. In particular, the length L, width W, and height H of the containers were set equal to 60, 25, and 30, respectively. For each customer i, a set of items IT_i was created. The size of IT_i is stochastically generated within the [1, 3] interval. The dimensions l_{it}, w_{it}, and h_{it} of a generated item it take random integer values from $[d_{\min} L, d_{\max} L]$, $[d_{\min} W, d_{\max} W]$, and $[d_{\min} H, d_{\max} H]$, respectively, where $d_{\min} = 0.2$ and $d_{\max} = 0.6$. The number of available vehicles v was experimentally determined to guarantee the feasibility of the developed instances.

Apart from the aforementioned instances, the GTS algorithm was applied to 12 new benchmark problems. To determine a robust parameter setting, these instances involve diverse item dimensions and customer set sizes. In particular, we developed four graphs containing 50–125 customers. The central depot lies at $(x_{\text{dep}}, y_{\text{dep}}) = (0, 0)$, and each customer i is placed at $(x_i, y_i) = (r\ \cos(\theta), r\ \sin(\theta))$, where r and θ are stochastically taken from [10, 100] and $[0, 2\pi]$, respectively. The dimensions of the containers were defined as $L = 60$, $W = 30$, and $H = 30$. Furthermore, we introduced three classes of item demand characteristics: Class 1 contains small items, Class 2 contains large items, and Class 3 involves diverse item dimensions. For each of the four graphs, we have created three test problems according to the aforementioned item classes: With each customer i, an item set IT_i is associated. The size of IT_i is randomly generated in the interval $[m_{\min}, m_{\max}]$. The dimensions l_{it}, w_{it}, and h_{it} of each item it are uniformly distributed in $[d_{\min} L, d_{\max} L]$, $[d_{\min} W, d_{\max} W]$, and $[d_{\min} H, d_{\max} H]$, respectively. The values of $m_{\min}$, $m_{\max}$, $d_{\min}$, and $d_{\max}$ for each class are presented in Table I. Every generated item has 20% probability of being fragile. The demand d_i of a customer i is randomly valued within the range [5, 35]. To ensure the feasibility of the developed test problems, the number of available vehicles v was experimentally defined by applying the construction heuristic algorithm described in Section III-B.

TABLE III
SENSITIVITY ANALYSIS SUMMARY

Parameter	Description	Range Tested	Standard Setting
tabuTenure	The number of iterations, for which the reversal of a performed move is declared tabu	[9,18]	12
guidFreq	The number of iterations, before which the guiding mechanism is employed	[5,30]	18
maxNoImp	The maximum number of non-improving iterations before the GTS procedure is terminated	[2000,15000]	6000

TABLE IV
CALIBRATING THE *tabuTenure* PARAMETER

	tabuTenure						
Instance	**9**	**12**	**15**	**18**	***min***	***max***	***%gap***
E022-06m	509.03	**508.79**	508.86	509.25	508.79	509.25	0.09
E031-09h	627.83	625.12	**624.84**	626.20	624.84	627.83	0.48
E045-04f	1301.09	**1300.76**	1302.61	1306.32	1300.76	1306.32	0.43
E076-14s	1208.42	**1203.45**	1204.82	1209.20	1203.45	1209.20	0.48
100-1	**2693.22**	2693.31	2698.83	2710.02	2693.22	2710.02	0.62
100-2	4345.80	4346.48	**4344.91**	4348.55	4344.91	4348.55	0.08
100-3	4208.42	**4203.70**	4210.73	4211.41	4203.70	4211.41	0.18
Average							0.34

min**:** minimum average objective function value over 100 runs, ***max*****:** maximum average objective function value over 100 runs, ***%gap*****:** percent gap between the best and the worst scores achieved (relative to the minima). Bold characters indicate higher quality values

TABLE V
CALIBRATING THE *guidFreq* PARAMETER

	guidFreq								
Instance	**5**	**10**	**15**	**20**	**25**	**30**	***min***	***max***	***%gap***
E022-06m	514.95	511.34	509.73	**508.95**	511.07	513.52	508.95	514.95	1.18
E031-09h	631.40	626.56	**625.10**	627.92	629.00	633.48	625.10	633.48	1.34
E045-04f	1328.34	1305.87	**1303.40**	1308.76	1320.12	1329.76	1303.40	1329.76	2.02
E076-14s	1251.61	1216.54	1205.26	**1203.67**	1227.71	1240.65	1203.67	1251.61	3.98
100-1	2703.52	2712.82	**2693.73**	2714.31	2726.20	2751.14	2693.73	2751.14	2.13
100-2	4431.28	4392.06	4363.76	**4345.92**	4374.70	4388.98	4345.92	4431.28	1.96
100-3	4243.15	**4215.61**	4256.76	4248.86	4284.10	4338.02	4215.61	4338.02	2.90
Average									2.22

min**:** minimum average objective function value over 100 runs, ***max*****:** maximum average objective function value over 100 runs, ***%gap*****:** percent gap between the best and the worst scores achieved (relative to the minima). Bold characters indicate higher quality values

The capacity of the vehicles was set equal to 120. Table II presents a description of both the 27 instances introduced by Gendreau *et al.* [2] and our new 12 benchmark problems.

B. Calibration of the Parameters

The GTS algorithmic framework contains three parameters, the values of which have to be determined before the algorithm is executed. To define the setting of these parameters, we performed a thorough calibration procedure, solving instances of various scales and item characteristics for the 3L-CVRP version. The test problems that were used for the sensitivity analysis procedure involve 21–100 customers and 40–309 transported items. The calibrated parameters were individually valued in relatively wide ranges, and the resulting GTS performance was evaluated. Table III presents a brief summary of the calibration procedure and results.

The *tabuTenure* parameter controls the period for which the reversal of a performed move is declared tabu. This tabu period is expressed in GTS iterations and guarantees that the search process avoids cycling phenomena and moves toward new unexplored solution trajectories. We valued the *tabuTenure* parameter within three to six iterations per move type. Since the GTS method considers three neighborhood structures, *tabuTenure* varied from 9 to 18 in increments of three. The GTS methodology was executed 100 times on each test problem, and the results obtained are presented in Table IV. The GTS methodology proved to be rather insensitive to the value of the *tabuTenure* parameter, as the average gap between the minimum and the maximum scores achieved was limited to 0.34%. For the majority of test problems, the best results were obtained by fixing *tabuTenure* at 12. Therefore, *tabuTenure* was set equal to 12.

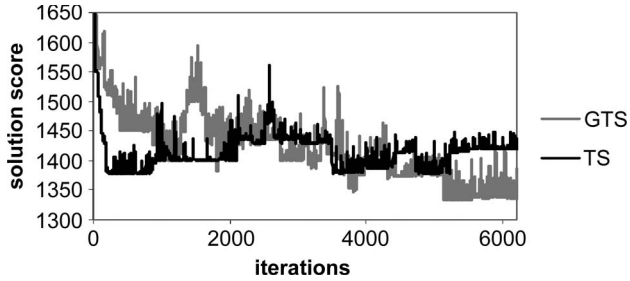

Fig. 6. Role of the guiding mechanism for instance E045-04f.

The *guidFreq* parameter controls the period of employing the guiding mechanism. The lower the value of *guidFreq*, the more

TABLE VI
COMPUTATIONAL RESULTS FOR 3L-CVRP ON GENDREAU *et al.* [2] INSTANCES

Instance	C_{GTS}	C_{TS}	*%gap*	bt_{GTS}	bt_{TS}	tt_{GTS}	tt_{TS}
E016-03m	321.47	**316.32**	-1.63	7.8	129.5	13.2	1800
E016-05m	**334.96**	350.58	4.46	7.2	5.3	11.5	1800
E021-04m	**430.95**	447.73	3.75	352.6	461.1	540.6	1800
E021-06m	458.04	**448.48**	-2.13	204.0	181.1	323.5	1800
E022-04g	465.79	**464.24**	-0.33	61.3	75.8	99.6	1800
E022-06m	507.96	**504.46**	-0.69	768.8	1167.9	1212.4	1800
E023-03g	**796.61**	831.66	4.21	241.5	181.1	364.8	1800
E023-05s	880.93	**871.77**	-1.05	140.0	156.1	230.0	1800
E026-08m	**642.22**	666.10	3.59	604.7	1468.5	982.2	3600
E030-03g	**884.74**	911.16	2.90	803.1	714.0	1308.4	3600
E030-04s	873.43	**819.36**	-6.60	308.5	396.4	522.5	3600
E031-09h	**624.24**	651.58	4.20	180.8	268.1	294.6	3600
E033-03n	**2799.74**	2928.34	4.39	1309.5	1639.1	2193.1	3600
E033-04g	**1504.44**	1559.64	3.54	2678.1	3451.6	4581.3	3600
E033-05s	**1415.42**	1452.34	2.54	1466.3	2327.4	2528.3	3600
E036-11h	**698.61**	707.85	1.31	2803.2	2550.3	4256.5	3600
E041-14h	**872.79**	920.87	5.22	1208.6	2142.5	2096.0	3600
E045-04f	**1296.59**	1400.52	7.42	1300.9	1452.9	2275.2	3600
E051-05e	**818.68**	871.29	6.04	1438.4	1822.3	2509.0	7200
E072-04f	**641.57**	732.12	12.37	1284.8	790.0	1940.9	7200
E076-07s	**1159.72**	1275.20	9.06	1704.8	2370.3	2823.4	7200
E076-08s	**1245.35**	1277.94	2.55	1663.5	1611.3	2685.6	7200
E076-10e	**1231.92**	1258.16	2.09	3048.2	6725.6	4659.1	7200
E076-14s	**1201.96**	1307.09	8.04	2876.8	6619.3	4854.1	7200
E101-08e	**1457.46**	1570.72	7.21	3432.0	5630.9	5725.8	7200
E101-10c	**1711.93**	1847.95	7.36	3974.8	4123.7	6283.1	7200
E101-14s	**1646.44**	1747.52	5.78	5864.2	7127.2	9915.7	7200
Average			3.54	1471.6	2058.9	2415.9	4266.7

***C*:** cost of the highest quality solution obtained, ***bt*:** computational time elapsed when the best solution was encountered (sec), ***tt*:** required computational times for the algorithm to run to completion (sec), ***%gap*:** percent gap between the scores achieved by GTS and TS methods (relatively to TS), ***GTS*:** Pentium IV 2.8 GHz, 1GB RAM, Visual C#, ***TS*:** Pentium IV 3 GHz, 512MB RAM, C (Gendreau et al. [2]). Bold characters indicate higher quality values

TABLE VII
COMPUTATIONAL RESULTS FOR 3L-CVRP ON THE NEW BENCHMARK INSTANCES

Instance	C_{GTS}	bt_{GTS}	tt_{GTS}	Instance	C_{GTS}	bt_{GTS}	tt_{GTS}
50-1	1457.78	1387.6	2308.7	**100-1**	2690.23	4895.5	8457.5
50-2	2257.60	975.8	1600.4	**100-2**	4342.64	3281.8	5967.9
50-3	1838.40	1065.8	1719.9	**100-3**	4190.92	4682.5	9301.9
75-1	2059.32	2286.0	3776.2	**125-1**	3298.22	6850.2	12590.0
75-2	3279.16	1460.1	2495.7	**125-2**	5788.12	5281.7	9727.3
75-3	2508.17	1768.3	3507.1	**125-3**	5177.97	5704.4	11191.9

***C*:** cost of the highest quality solution obtained, ***bt*:** computational time elapsed when the best solution was encountered (sec), ***tt*:** required computational times for the algorithm to run to completion (sec), ***GTS*:** Pentium IV 2.8 GHz, 1GB RAM, Visual C#

frequent the arcs are penalized, and the stronger the diversification induced to the search process. To define the value yielding the most satisfactory interplay between the intensification and diversification of the search, the *guidFreq* parameter was valued within [5, 30] in increments of five. For each parameter value and each test problem, the algorithm was applied 100 times. The results obtained are presented in Table V. They indicate a considerable dependence of the GTS method on the *guidFreq* parameter, as the gaps between the best and worst average scores ranged between 1.18% and 3.98%. The lowest average objective function values were obtained, when *guidFreq* was set equal to 15 and 20. Thus, the standard setting of the *guidFreq* parameter was fixed at 18.

The last parameter to be valued controls the terminating condition of the GTS methodology. GTS is terminated after the completion of *maxNoImp* iterations without improving the highest quality solution encountered. Obviously, the greater the value of *maxNoImp*, the higher the probabilities of achieving better-quality solutions, and the longer the computational time required by the GTS method. To tune the *maxNoImp* parameter, we tested GTS with *maxNoImp* values taken from {2000, 4000, 6000, 10 000, 15 000}. For small-scale test problems, 4000 proved to be high enough to guarantee the production of high-quality solutions. On the contrary, when GTS was applied for solving large-scale instances, we observed solution improvements after the completion of 6000 or even 10 000 non-improving iterations. These high values of *maxNoImp* involve the risk of consuming excessive overall computational effort. Therefore, we fixed *maxNoImp* at 6000, for which we observed a satisfactory balance between the final solution quality and the demanded computational time.

C. Comparison of GTS With Pure TS

To visualize both the effectiveness and the diversification effect of the guiding mechanism, we provide Fig. 6,

TABLE VIII
EFFECT OF THE LOADING CONSTRAINTS FOR 3L-CVRP

	No Fragility		No LIFO		No Supporting Area ($a = 0$)		3D loading only	
Instance	***%Cons***	***%TS***	***%Cons***	***%TS***	***%Cons***	***%TS***	***%Cons***	***%TS***
Gendreau et al. instances								
E016-03m	1.44	-0.16	6.05	-0.09	3.18	-3.15	7.08	-0.35
E016-05m	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
E021-04m	3.82	3.61	12.68	4.11	9.94	0.00	15.94	0.00
E021-06m	0.06	-3.88	5.93	0.00	2.14	-4.03	5.93	0.00
E022-04g	0.45	-0.24	3.72	-6.04	4.39	-2.16	9.43	-6.63
E022-06m	2.30	0.41	1.90	-0.50	1.90	-0.03	2.38	0.00
E023-03g	0.39	0.94	4.27	-4.11	2.85	-1.48	6.83	0.00
E023-05s	6.11	4.33	3.02	-5.39	7.41	-2.03	13.14	-4.09
E026-08m	0.20	5.34	1.88	0.00	1.88	1.69	1.88	0.00
E030-03g	5.10	0.44	5.04	-1.57	8.16	-1.17	18.69	-0.20
E030-04s	6.68	0.52	6.25	-6.73	10.61	-1.06	14.34	-4.17
E031-09h	1.55	8.16	1.55	3.24	1.55	0.38	2.24	0.71
E033-03n	2.04	0.41	2.94	-6.58	5.53	-2.05	12.50	-5.75
E033-04g	3.06	3.94	7.66	0.01	8.49	-2.11	13.98	-1.37
E033-05s	3.53	3.45	4.20	-0.70	6.01	1.62	15.46	0.00
E036-11h	-0.68	1.09	0.00	0.70	0.00	0.88	0.00	0.00
E041-14h	0.73	5.92	0.73	4.42	0.13	5.35	1.26	4.92
E045-04f	4.69	13.80	3.24	5.79	6.01	2.16	13.33	0.05
E051-05e	2.55	6.48	6.38	1.96	7.74	-2.80	14.80	-2.54
E072-04f	3.99	5.74	6.45	4.69	10.23	5.69	16.66	-1.08
E076-07s	2.87	5.00	4.97	8.98	5.69	7.98	14.95	1.80
E076-08s	2.76	2.59	4.82	-2.12	7.41	1.63	14.04	-0.14
E076-10e	8.12	9.33	10.16	4.06	10.01	-0.19	17.15	-0.80
E076-14s	4.59	3.44	6.15	2.29	7.94	0.66	11.97	0.52
E101-08e	2.25	14.85	4.43	1.94	5.12	-0.49	13.85	8.44
E101-10c	3.17	6.64	2.98	-3.45	7.94	0.22	13.32	4.70
E101-14s	5.28	6.17	7.11	1.73	9.07	2.56	16.74	0.56
Average	2.85	4.01	4.61	0.25	5.61	0.30	10.66	-0.20
New Instances								
50-1	0.09	-	2.74	-	0.99	-	2.74	-
50-2	0.00	-	1.83	-	0.87	-	4.24	-
50-3	5.35	-	10.09	-	7.55	-	13.80	-
75-1	0.57	-	1.86	-	1.14	-	1.71	-
75-2	1.94	-	5.17	-	4.57	-	10.97	-
75-3	-2.77	-	3.47	-	3.83	-	9.51	-
100-1	1.85	-	6.37	-	4.39	-	7.16	-
100-2	-0.47	-	4.96	-	2.68	-	5.61	-
100-3	8.42	-	15.16	-	17.11	-	23.65	-
125-1	1.46	-	2.90	-	2.40	-	3.59	-
125-2	3.09	-	6.87	-	6.67	-	9.82	-
125-3	3.88	-	17.57	-	16.71	-	25.90	-
Average	1.95	-	6.58	-	5.74	-	9.89	-

***%Cons*:** percent improvement of the GTS solution values for the examined constraint configuration (Table IX) over the fully constraint problem values (Table VI – Table VII), ***%TS*:** percent improvement of the GTS over the TS solution values for the examined constraint configuration (Table IX)

which compares the solution function value through the search process, with and without applying the guiding strategy. The problem used to conduct this comparison is benchmark instance E-045-04f (see Section IV-A). The black line represents the pure TS method, whereas the gray line corresponds to the proposed GTS methodology. In terms of the solution quality of the final solution obtained, the guiding mechanism resulted in a 3.13% solution improvement, compared with the pure TS algorithm (GTS: 1338.38, TS: 1376.52). To measure the diversification character of both search processes, we calculate the standard deviation of the candidate solution scores. The results obtained indicate a remarkable 82.79% increase in the solution score standard deviation (GTS: 63.61, TS: 34.80).

D. Computational Result for Both Problem Versions

Using the standard parameter setting determined by the calibration experiments, we applied the GTS methodology to all 27 benchmark instances of Gendreau *et al.* [2] and our 12 new developed test problems for both problem versions (3L-CVRP and M3L-CVRP). The performance of the GTS algorithm is compared with that of the TS methodology of Gendreau *et al.* [2]. In addition, to gain insight into the effect of the loading constraints on the final solution quality and compare the performance of the GTS and TS methods for different constraint configurations, we executed our algorithm, disregarding the loading constraints imposed by the problem. In particular, we ran the GTS methodology with four different loading constraint configurations: The first three do not consider the fragility,

TABLE IX
3L-CVRP SOLUTION VALUES FOR VARIOUS LOADING CONSTRAINT CONFIGURATIONS

	No Fragility		No LIFO		No Supporting Area ($a = 0$)		3D loading only	
Instance	C_{GTS}	C_{TS}	C_{GTS}	C_{TS}	C_{GTS}	C_{TS}	C_{GTS}	C_{TS}
Gendreau et al. instances								
E016-03m	316.83	316.32	302.02	301.74	311.25	301.74	298.70	297.65
E016-05m	334.96	334.96	334.96	334.96	334.96	334.96	334.96	334.96
E021-04m	414.49	430.02	376.30	392.44	388.10	388.10	362.27	362.27
E021-06m	457.78	440.68	430.88	430.88	448.24	430.88	430.88	430.88
E022-04g	463.68	462.59	448.45	422.90	445.35	435.93	421.86	395.64
E022-06m	496.29	498.32	498.32	495.85	498.32	498.16	495.85	495.85
E023-03g	793.50	801.03	762.63	732.51	773.88	762.63	742.23	742.23
E023-05s	827.09	864.54	854.36	810.65	815.62	799.38	765.18	735.14
E026-08m	640.92	677.06	630.13	630.13	630.13	640.94	630.13	630.13
E030-03g	839.65	843.33	840.12	827.11	812.58	803.18	719.34	717.90
E030-04s	815.12	819.36	818.85	767.22	780.75	772.55	748.20	718.24
E031-09h	614.59	669.16	614.59	635.15	614.59	616.95	610.23	614.60
E033-03n	2742.74	2753.91	2717.44	2549.68	2645.04	2591.84	2449.81	2316.56
E033-04g	1458.42	1518.26	1389.22	1389.31	1376.67	1348.19	1294.10	1276.60
E033-05s	1365.45	1414.19	1355.92	1346.44	1330.32	1352.20	1196.54	1196.55
E036-11h	703.38	711.11	698.61	703.57	698.61	704.80	698.61	698.61
E041-14h	866.40	920.87	866.40	906.42	871.63	920.87	861.79	906.42
E045-04f	1235.73	1433.51	1254.63	1331.71	1218.63	1245.57	1123.77	1124.33
E051-05e	797.78	853.05	766.41	781.77	755.33	734.77	697.54	680.29
E072-04f	615.96	653.47	600.21	629.77	575.91	610.63	534.69	529.00
E076-07s	1126.42	1185.67	1102.10	1210.78	1093.74	1188.60	986.31	1004.40
E076-08s	1211.04	1243.22	1185.27	1160.67	1153.06	1172.20	1070.47	1068.96
E076-10e	1131.83	1248.25	1106.72	1153.60	1108.55	1106.43	1020.62	1012.51
E076-14s	1146.79	1187.68	1128.08	1154.51	1106.48	1113.80	1058.03	1063.61
E101-08e	1424.62	1673.08	1392.89	1420.51	1382.79	1375.99	1255.54	1371.32
E101-10c	1657.64	1775.52	1660.99	1605.54	1576.07	1579.47	1483.92	1557.12
E101-14s	1559.58	1662.10	1529.35	1556.33	1497.19	1536.49	1370.86	1378.52
New Instances								
50-1	1456.48	-	1417.88	-	1443.38	-	1417.88	-
50-2	2257.60	-	2216.36	-	2237.91	-	2161.99	-
50-3	1740.06	-	1652.82	-	1699.64	-	1584.76	-
75-1	2047.63	-	2020.92	-	2035.81	-	2024.10	-
75-2	3215.64	-	3109.49	-	3129.31	-	2919.31	-
75-3	2577.58	-	2421.18	-	2412.01	-	2269.53	-
100-1	2640.54	-	2518.82	-	2572.04	-	2497.69	-
100-2	4363.26	-	4127.35	-	4226.45	-	4098.88	-
100-3	3838.06	-	3555.68	-	3473.76	-	3199.65	-
125-1	3250.21	-	3202.72	-	3219.03	-	3179.65	-
125-2	5609.28	-	5390.74	-	5401.81	-	5219.66	-
125-3	4977.16	-	4268.25	-	4312.57	-	3836.86	-

***C*: cost of the highest quality solution obtained, *GTS*: the proposed method, *TS*: the Tabu Search approach of Gendreau et al. [2]**

LIFO, and supporting-area constraints, respectively. For the fourth configuration, all three aforementioned constraints are ignored.

1) Computational Result for 3L-CVRP: Table VI compares the results of the GTS algorithm with those of the TS method for the 27 problems of Gendreau *et al.* [2]. GTS improves 21 out of the 27 best solutions obtained by TS. The average solution improvement is equal to 3.54%. As shown in Table VI, the gap between the GTS and TS scores increases with the problem scale. In particular, when up to 30 customers are considered, the TS method presents better results for six out of the 11 test problems. However, for the rest of the sixteen test problems (31–101 customers), the GTS method consistently produces higher quality solutions. This indicates that the relative advantage of the proposed method comes from the interplay between the computational effort dedicated to the routing and packing aspects; instead of spending computational time searching through item orders to overcome loading infeasibilities [2], GTS uses a simpler packing component to determine if loading constraints are satisfied. In this way, the proposed method is mainly oriented toward the minimization of the routing cost, which constitutes the actual objective of the problem. Therefore, for small-scale instances, where the routing aspects can easily be dealt with and the solution quality primarily depends on the loading constraints, the TS and GTS methods present comparable results. However, for the large-scale instances, where the routing aspects are more challenging and become the decisive factor for the solution quality, GTS exhibits better performance than the TS method. Regarding the computational times required, GTS appears to be faster on average; however, no direct comparisons can be made, as the computational times heavily depend on a variety of factors, including the programming languages, compilers, computer systems, and the coding skills of the developers. Table VII presents the 3L-CVRP solution scores and the CPU times for the 12 new benchmark instances.

Table VIII presents the effect of the loading constraints on the solution quality. It summarizes the gaps between the solution values for the fully constrained problem and for each of the aforementioned constraint configurations (no fragility, no LIFO, no supporting area, and 3-D packing). Table VIII also presents the gaps between the GTS and TS methods for each constraint configuration. The solution values obtained by the

TABLE X
COMPUTATIONAL RESULTS FOR M3L-CVRP

Gendreau et al. instances					New instances				
Instance	***C**$_{GTS}$*	***%Gap3L***	***bt**$_{GTS}$*	***tt**$_{GTS}$*	**Instance**	***C**$_{GTS}$*	***%Gap3L***	***bt**$_{GTS}$*	***tt**$_{GTS}$*
E016-03m	322.47	-0.31	6.7	12.1	**50-1**	1457.78	0.00	1270.6	2314.7
E016-05m	334.96	0.00	9.9	18.2	**50-2**	2242.18	0.68	903.8	1404.7
E021-04m	403.51	6.37	286.8	517.0	**50-3**	1853.43	-0.82	990.1	1746.7
E021-06m	458.04	0.00	207.6	325.1	**75-1**	2061.56	-0.11	2134.0	3410.1
E022-04g	465.79	0.00	53.0	86.0	**75-2**	3226.62	1.60	1526.5	2690.4
E022-06m	507.96	0.00	683.7	1254.0	**75-3**	2477.61	1.22	1672.3	2686.6
E023-03g	796.61	0.00	256.1	449.5	**100-1**	2653.91	1.35	4562.5	8958.4
E023-05s	843.76	4.22	124.2	245.3	**100-2**	4325.19	0.40	3319.0	5674.8
E026-08m	642.22	0.00	563.8	957.1	**100-3**	4069.59	2.90	4798.1	7923.9
E030-03g	879.79	0.56	738.5	1209.1	**125-1**	3273.95	0.74	6472.2	10784.9
E030-04s	873.43	0.00	308.1	511.9	**125-2**	5759.24	0.50	5384.7	9171.3
E031-09h	624.24	0.00	164.7	263.2	**125-3**	5110.85	1.30	5524.9	8559.0
E033-03n	2782.89	0.60	1272.3	2051.0	**Average**		0.81		
E033-04g	1508.83	-0.29	2777.9	5303.4					
E033-05s	1456.97	-2.94	1542.8	2753.3					
E036-11h	698.61	0.00	2640.2	4311.9					
E041-14h	872.79	0.00	1054.6	1987.2					
E045-04f	1299.91	-0.26	1106.0	1893.0					
E051-05e	817.19	0.18	1503.1	2519.1					
E072-04f	635.90	0.88	1126.4	2197.1					
E076-07s	1163.96	-0.37	1704.8	3206.6					
E076-08s	1224.25	1.69	1450.3	2416.8					
E076-10e	1199.37	2.64	2975.2	4888.5					
E076-14s	1183.11	1.57	2932.7	5527.6					
E101-08e	1482.85	-1.74	3387.6	6748.4					
E101-10c	1695.39	0.97	4106.0	6282.0					
E101-14s	1613.32	2.01	5594.2	9591.4					
Average		0.58							

C*:** cost of the highest quality solution obtained, ***%Gap3L*:** percent gap between the M3L-CVRP and the 3L-CVRP solution values, ***bt*:** computational time elapsed when the best solution was encountered (sec), ***tt: required computational times for the algorithm to run to completion (sec), ***GTS*:** Pentium IV 2.8 GHz, 1GB RAM, Visual C#

GTS and TS methods for each constraint configuration of the 3L-CVRP model are presented in Table IX.

In detail, the $\%Cons$ column of Table VIII presents the percent gaps between the GTS solution values achieved for the fully constrained problem denoted by the $FC\_C_{\rm GTS}$ (column $C_{\rm GTS}$ of Tables VI and VII) and GTS values achieved for each examined constraint configuration $CC\_C_{\rm GTS}$ (column $C_{\rm GTS}$ of Table IX). These gaps have been evaluated as $(FC\_C_{\rm GTS} - CC\_C_{\rm GTS})/FC\_C_{\rm GTS}$. The $\%TS$ columns of Table VIII correspond to the percent gaps between the values of GTS (column $C_{\rm GTS}$ of Table IX) and TS (column $C_{\rm TS}$ of Table IX) for each constraint configuration, which are denoted by $CC\_C_{\rm GTS}$ and $CC\_C_{\rm TS}$, respectively. They have been computed as $(CC\_C_{\rm TS} - CC\_C_{\rm GTS})/CC\_C_{\rm TS}$. In terms of the CPU times, ignoring the loading constraints of the original model significantly accelerated the performance of the algorithm, as less feasibility checks were required by the packing heuristics. We observe that removing each of the loading constraints results in a significant objective function decrease. Disregarding the fragility constraint, the results indicate a 2.58% (2.85% for the instances of Gendreau *et al.* [2] and 1.95% for the new instances) average decrease in the solution costs. The LIFO policy plays a more important role, as its removal yields an average solution quality improvement of 5.22% (4.61% and 6.58% for the two data sets, respectively). The supporting-area constraint has a stronger impact on the final solution quality. In particular, setting the supporting-area factor $a = 0$ results in an average solution cost decrease of 5.65% (5.61% and 5.74% for the two data sets, respectively). Finally, removing all three aforementioned constraints leads to a 10.43% average solution quality improvement (10.66% and 9.89% for the two data sets, respectively). Comparing the GTS and TS method performances for the various constraint configurations, we observe that GTS consistently improves the results of TS when ignoring the fragility, LIFO, and supporting-area constraints. In particular, disregarding the fragility constraint, GTS improves the average solution values obtained by the TS approach by 4.01%. The average improvement is lower when removing the LIFO and supporting-area constraints (0.25% and 0.30%). Surprisingly, when all of the aforementioned loading constraints are ignored, GTS increases the average solution cost achieved by the TS method by 0.20%. This behavior may be due to the fact that, since the packing has no special requirements (e.g., the supporting-area, LIFO-policy, and fragility constraints), the order by which items are loaded into the container is not strictly dictated by the loading constraints; therefore, by searching through possible item orders, the TS of Gendreau *et al.* [2] more effectively tackles the unrestricted problem version.

2) Computational Results for M3L-CVRP: Table X presents the best solution scores obtained by the GTS algorithm, solving all 39 benchmark instances for the M3L-CVRP version of the problem. It also presents the percent differences between the solution values achieved for M3L-CVRP and 3L-CVRP presented in Tables VI and VII. These differences have been evaluated as $(C_{\rm 3L} - C_{\rm M3L})/C_{\rm 3L}$, where $C_{\rm 3L}$ and $C_{\rm M3L}$ denote the solution values for the 3L-CVRP and M3L-CVRP versions,

TABLE XI
EFFECT OF THE LOADING CONSTRAINTS FOR M3L-CVRP

	No Fragility	No LIFO	No Supporting Area ($a = 0$)	3D loading only
Instance	***%Cons***	***%Cons***	***%Cons***	***%Cons***
Gendreau et al. instances				
E016-03m	0.87	6.34	3.61	7.37
E016-05m	0.00	0.00	0.00	0.00
E021-04m	-2.88	6.74	4.78	10.22
E021-06m	0.06	5.93	2.20	5.93
E022-04g	0.75	3.72	3.45	9.43
E022-06m	0.25	1.90	1.83	2.38
E023-03g	-1.93	4.27	3.21	6.83
E023-05s	-0.16	-1.26	3.15	9.31
E026-08m	0.20	1.88	1.40	1.88
E030-03g	3.96	4.51	8.88	18.24
E030-04s	6.49	6.25	12.92	14.34
E031-09h	1.55	1.55	1.55	2.24
E033-03n	1.27	2.35	4.85	11.97
E033-04g	1.55	7.93	10.58	14.23
E033-05s	7.28	6.94	8.62	17.87
E036-11h	-0.59	0.00	0.00	0.00
E041-14h	0.00	0.73	0.73	1.26
E045-04f	4.72	3.48	7.22	13.55
E051-05e	2.69	6.21	8.71	14.64
E072-04f	3.68	5.61	10.82	15.92
E076-07s	1.58	5.31	8.26	15.26
E076-08s	0.69	3.18	4.54	12.56
E076-10e	4.02	7.72	8.49	14.90
E076-14s	1.91	4.65	6.98	10.57
E101-08e	2.77	6.07	7.49	15.33
E101-10c	1.07	2.03	7.43	12.47
E101-14s	1.21	5.20	10.11	15.03
New Instances				
50-1	0.18	2.74	2.54	2.74
50-2	1.28	1.15	1.93	3.58
50-3	5.74	10.82	10.32	14.50
75-1	0.52	1.97	2.20	1.82
75-2	1.53	3.63	4.88	9.52
75-3	-3.31	2.28	5.17	8.40
100-1	2.63	5.09	5.68	5.89
100-2	0.76	4.57	2.45	5.23
100-3	12.08	12.63	17.94	21.38
125-1	0.94	2.18	2.91	2.88
125-2	4.77	6.40	5.57	9.37
125-3	11.01	16.49	20.56	24.93
Average	2.08	4.60	6.00	9.85

%Cons**:** percent improvement of the GTS solution values for the examined constraint configuration (Table XII) over the fully constraint problem values (Table IX)

respectively. The 3L-CVRP scores are slightly higher than the M3L-CVRP scores as the 3L-CVRP problem version follows a more restrictive LIFO policy.

To investigate the effect of the loading constraints on the solution quality, we solved the M3L-CVRP problem version for all four constraint configurations previously described. Table XI presents the results obtained. Note that the GTS solution values for each of the four examined constraint configurations of the M3L-CVRP version are presented in Table XII. As shown in Table XI, ignoring the supporting area leads to a significant average decrease of 6.00% in the solution values obtained. The average decrease in the solution scores for ignoring the fragility and LIFO constraints is limited to 2.08% and 4.60%, respectively. Removing all of the aforementioned constraints yields a 9.85% average solution improvement. The greatest decrease in the solution scores is observed for the new benchmarks of Class 3 (involving diverse item dimensions). In particular, ignoring the fragility, LIFO, and supporting-area constraints results in a 17.30% average solution improvement for the Class-3 instances (50-3: 14.50%, 75-3: 8.40%, 100-3: 21.38%, and 125-3: 24.93%). The total computational times were reduced, compared with those demanded to solve the fully constrained problem, as the packing heuristics performed fewer operations for the loading feasibility checks.

E. Relative Effectiveness of the Packing Heuristics

To give insight into the relative effectiveness of the six packing heuristics included in the bundle of Section III-A, we have performed the following experimental procedure: For both problem versions, we applied the GTS method to each benchmark instance. Instead of terminating the execution of the bundle when the first feasible packing was obtained, all six packing heuristics were always applied to the investigation of

TABLE XII
M3L-CVRP SOLUTION VALUES FOR VARIOUS LOADING CONSTRAINT CONFIGURATIONS

	No Fragility	No LIFO	No Supporting Area ($a = 0$)	3D loading only
Instance	C_{GTS}	C_{GTS}	C_{GTS}	C_{GTS}
Gendreau et al. instances				
E016-03m	319.66	302.02	310.84	298.70
E016-05m	334.96	334.96	334.96	334.96
E021-04m	415.13	376.30	384.24	362.27
E021-06m	457.78	430.88	447.98	430.88
E022-04g	462.28	448.45	449.72	421.86
E022-06m	506.68	498.32	498.65	495.85
E023-03g	812.00	762.63	771.00	742.23
E023-05s	845.07	854.36	817.18	765.18
E026-08m	640.92	630.13	633.21	630.13
E030-03g	844.93	840.12	801.65	719.34
E030-04s	816.77	818.85	760.56	748.20
E031-09h	614.59	614.59	614.59	610.23
E033-03n	2747.58	2717.44	2648.01	2449.81
E033-04g	1485.39	1389.22	1349.24	1294.10
E033-05s	1350.88	1355.92	1331.45	1196.54
E036-11h	702.70	698.61	698.61	698.61
E041-14h	872.79	866.40	866.40	861.79
E045-04f	1238.55	1254.63	1206.07	1123.77
E051-05e	795.22	766.41	746.01	697.54
E072-04f	612.49	600.21	567.07	534.69
E076-07s	1145.57	1102.10	1067.76	986.31
E076-08s	1215.79	1185.27	1168.70	1070.47
E076-10e	1151.17	1106.72	1097.51	1020.62
E076-14s	1160.56	1128.08	1100.54	1058.03
E101-08e	1441.75	1392.89	1371.72	1255.54
E101-10c	1677.25	1660.99	1569.44	1483.92
E101-14s	1593.87	1529.35	1450.20	1370.86
New Instances				
50-1	1455.14	1417.88	1420.74	1417.88
50-2	2213.51	2216.36	2198.97	2161.99
50-3	1746.97	1652.82	1662.14	1584.76
75-1	2050.81	2020.92	2016.30	2024.10
75-2	3177.40	3109.49	3069.26	2919.31
75-3	2559.72	2421.18	2349.49	2269.53
100-1	2584.24	2518.82	2503.07	2497.69
100-2	4292.47	4127.35	4219.39	4098.88
100-3	3578.02	3555.68	3339.60	3199.65
125-1	3243.26	3202.72	3178.71	3179.65
125-2	5484.67	5390.74	5438.26	5219.66
125-3	4548.02	4268.25	4060.27	3836.86

***C*:** cost of the highest quality solution obtained, ***GTS*:** the proposed method

the feasibility of the tentative routes. In this way, we can empirically compare the relative performance of the proposed packing heuristics. The results found are presented in Table XIII. In detail, for each instance, the heuristic that obtained the most feasible loadings is assigned a score of 100%. The score for each of the remaining heuristics is proportional to the number of feasible loading patterns that it generated. As seen from the results, the four heuristics that involve the calculation of the touching perimeters ($Heur_3$–$Heur_6$) exhibited better performance, compared with the simpler and faster $Heur_1$ and $Heur_2$. For both problem versions, $Heur_4$, which ignores the container walls when calculating the touching perimeters and favors the evolution of packing in the form of layers parallel to the l–h plane, yielded the best average scores or, in other words, managed to generate the most feasible loading patterns.

V. CONCLUSION

In this paper, we have investigated combined routing and 3-D-packing problems. Specifically, we have studied a recently addressed practical variant of the CVRP called 3L-CVRP. This problem arises in the context of transportation logistics when vehicles must be routed to satisfy customer demand, which consists of 3-D, rectangular, and weighted items.

Regarding our solution approach, we have proposed a hybrid metaheuristic method called GTS, which combines the strengths of the TS and GLS strategies. The conducted TS is periodically controlled by a guiding mechanism, which locates and penalizes low-quality features present in the candidate solution. The application of this guiding mechanism drastically diversifies the search process and increases the effectiveness of the proposed methodology. Concerning the packing aspects of the problem, we have proposed a collection of packing heuristics. To increase the probability of obtaining feasible loadings, each heuristic is designed to favor a different packing structure. The overall computational effort is reduced by two acceleration strategies: We have used a policy for restricting the size of the solution neighborhoods explored, together with a memory structure for storing the loading feasibility information obtained through the search progress. To calibrate the algorithmic

TABLE XIII
RELATIVE EFFECTIVENESS OF THE PACKING HEURISTICS

	3L-CVRP						M3L-CVRP					
Instance	*Heur1*	*Heur2*	*Heur3*	*Heur4*	*Heur5*	*Heur6*	*Heur1*	*Heur2*	*Heur3*	*Heur4*	*Heur5*	*Heur6*
Gendreau et al. instances												
E016-03m	61	68	100	94	74	76	60	58	95	100	85	88
E016-05m	57	72	100	91	75	73	69	70	100	96	86	82
E021-04m	66	63	93	100	83	88	58	53	88	100	79	94
E021-06m	65	64	87	100	72	76	60	66	88	90	100	82
E022-04g	62	68	89	96	95	100	62	56	95	97	89	100
E022-06m	64	64	82	100	72	80	55	70	89	86	90	100
E023-03g	60	66	87	92	100	85	68	55	92	97	95	100
E023-05s	67	73	100	92	81	76	61	53	100	95	79	73
E026-08m	54	65	100	96	75	72	69	64	94	97	100	91
E030-03g	60	57	83	100	76	87	69	75	100	84	76	70
E030-04s	59	65	93	100	88	97	56	79	88	100	81	90
E031-09h	62	59	100	92	80	76	71	78	94	89	93	100
E033-03n	68	67	100	88	90	88	70	78	97	82	100	98
E033-04g	66	74	85	100	82	78	65	81	91	100	83	83
E033-05s	63	67	100	94	80	85	56	66	96	100	92	95
E036-11h	64	70	92	100	73	86	61	61	100	99	76	78
E041-14h	59	69	86	100	78	85	68	57	93	100	96	93
E045-04f	60	73	94	100	86	86	70	77	91	100	75	74
E051-05e	62	64	100	91	82	85	62	62	98	100	73	80
E072-04f	67	62	85	100	87	100	64	57	89	100	81	100
E076-07s	66	66	95	100	88	85	58	76	99	92	100	86
E076-08s	62	73	87	95	100	84	60	78	85	98	92	100
E076-10e	58	56	100	83	76	83	60	70	97	100	82	89
E076-14s	62	66	100	90	81	90	57	66	100	100	77	83
E101-08e	67	65	94	100	90	88	70	61	90	100	75	79
E101-10c	69	68	100	100	84	78	73	60	93	92	100	93
E101-14s	65	65	100	98	90	94	62	77	94	100	84	93
New Instances												
50-1	64	68	95	96	98	100	68	76	100	82	97	98
50-2	55	56	100	90	88	85	56	67	93	100	93	80
50-3	66	66	94	95	92	100	62	54	84	91	100	93
75-1	62	65	89	91	100	93	56	57	90	100	92	95
75-2	56	73	97	100	78	86	62	70	100	100	77	85
75-3	50	64	88	85	100	93	53	72	100	89	100	94
100-1	67	61	100	95	100	100	69	68	99	100	94	80
100-2	64	69	100	93	78	83	60	74	95	100	82	88
100-3	67	63	84	90	100	94	69	81	89	93	100	95
125-1	67	62	92	92	99	100	62	65	100	96	92	100
125-2	59	62	100	93	88	86	72	66	94	100	75	78
125-3	60	65	92	100	88	85	60	69	100	95	75	83
Average	62	66	94	95	86	87	63	67	94	96	88	89

parameters of the GTS, we have performed extensive computational experiments, solving both instances derived from the literature and a set of new benchmark problems. These new benchmark instances have been developed to cover different problem characteristics, as far as the size of the customer population and the dimensions of the transported items are concerned. After determining the standard parameter setting, GTS has successfully been applied to the benchmark problems. In particular, it managed to improve the average quality of the previously reported best solutions by 3.54%. Finally, to evaluate the sensitivity of the proposed algorithmic methodology to the various packing constraints, we have executed GTS, considering various loading constraint configurations.

Despite their great practical importance, only recently have the first routing problems with loading constraints been addressed. Therefore, in terms of future research directions, several hybrid metaheuristic frameworks that have proven to be effective for routing and packing problems can be developed to deal with practical large-scale problems. In addition, for medium- and small-scale problem instances, it is challenging to design metaheuristic strategies that include mathematical-programming-related components, which have recently been defined as matheuristic methods. Practical variants of 3L-CVRP covering more operational constraints can also be investigated. In terms of the packing features of the problem, these variants may consider various container sizes and irregular shapes of transported items. Regarding the routing aspects, problem models considering time windows and split deliveries can also be examined.

ACKNOWLEDGMENT

The authors would like to thank the anonymous referees for extensively reviewing this paper and offering constructive remarks and directions for the completion of this work.

REFERENCES

[1] P. Toth and D. Vigo, *The Vehicle Routing Problem, SIAM Monographs on Discrete Mathematics and Applications*. Philadelphia, PA: SIAM, 2002.
[2] M. Gendreau, M. Iori, G. Laporte, and S. Martello, "A Tabu search algorithm for a routing and container loading problem," *Transp. Sci.*, vol. 40, no. 3, pp. 342–350, Aug. 2006.
[3] D. Mester and O. Bräysy, "Active-guided evolution strategies for large-scale capacitated vehicle routing problems," *Comput. Oper. Res.*, vol. 34, no. 10, pp. 2964–2975, Oct. 2007.
[4] S. Kim, M. E. Lewis, and C. C. White, III, "Optimal vehicle routing with real-time traffic information," *IEEE Trans. Intell. Transp. Syst.*, vol. 6, no. 2, pp. 178–188, Jun. 2005.
[5] H. Jula, M. Dessouky, and P. A. Ioannou, "Truck route planning in non-stationary stochastic networks with time windows at customer locations," *IEEE Trans. Intell. Transp. Syst.*, vol. 7, no. 1, pp. 51–62, Mar. 2007.
[6] S. Martello, D. Pisinger, and D. Vigo, "The three-dimensional bin packing problem," *Oper. Res.*, vol. 48, no. 2, pp. 256–267, Mar. 2000.
[7] M. Gendreau, A. Hertz, and G. Laporte, "New insertion and post-optimization procedures for the traveling salesman problem," *Oper. Res.*, vol. 40, no. 6, pp. 1086–1094, Nov./Dec. 1992.
[8] M. Iori, J. J. Salazar Gonzalez, and D. Vigo, "An exact approach for the symmetric capacitated vehicle routing problem with two dimensional loading constraints," *Transp. Sci.*, vol. 41, no. 2, pp. 253–264, May 2007.

[9] M. Gendreau, M. Iori, G. Laporte, and S. Martello, "A Tabu search heuristic for the vehicle routing problem with two-dimensional loading constraints," *Networks*, vol. 51, no. 1, pp. 4–18, Jan. 2008.
[10] E. E. Zachariadis, C. D. Tarantilis, and C. T. Kiranoudis, "A guided Tabu search for the vehicle routing problem with two-dimensional loading constraints," *Eur. J. Oper. Res.*, vol. 195, no. 3, pp. 729–743, Jun. 2009.
[11] K. F. Doerner, G. Fuellerer, R. F. Hartl, M. Gronalt, and M. Iori, "Metaheuristics for the vehicle routing problem with loading constraints," *Networks*, vol. 49, no. 4, pp. 294–307, Jul. 2007.
[12] M. Gendreau and J. Y. Potvin, "Tabu search, in search methodologies," in *Introductory Tutorials in Optimization and Decision Support Techniques*, E. Burke and G. Kendall, Eds. New York: Springer-Verlag, 2005, pp. 165–186.
[13] C. Voudouris and E. Tsang, "Partial constraint satisfaction problems and guided local search," in *Proc. 2nd Int. Conf. PACT*, 1996, pp. 337–356.
[14] C. D. Tarantilis, E. E. Zachariadis, and C. T. Kiranoudis, "A hybrid guided local search for the vehicle routing with intermediate replenishment facilities," *INFORMS J. Comput.*, vol. 20, no. 1, pp. 154–168, Feb. 2008.
[15] N. Mladenoviæ and P. Hansen, "Variable neighborhood search," *Comput. Oper. Res.*, vol. 24, no. 11, pp. 1097–1100, Nov. 1997.

Christos D. Tarantilis received the Diploma in mathematics from the University of Patras, Patras, Greece, in 1997, the M.S. degree in operational research from the London School of Economics, London, U.K., in 1998, and the Ph.D. degree in operations research from the National Technical University of Athens, Athens, Greece, in 2002.

He is currently an Associate Professor of operations research with the Department of Management Science and Technology, Athens University of Economics and Business, and the Director of the Operations Research and Decision Systems Centre, Athens. He has been a member of the Board of Directors of several companies, a principal investigator for several European Union and national projects, and a consultant for public organizations and private industries. He has authored more than 100 scientific papers in international journals, books, and conference proceedings. His research interests include the application of operations research techniques for complex decision systems modeled across many time and scale lengths, with specialization in the analysis, design, and development of integrated models, methodologies, and computational tools in the areas of distribution logistics services, supply-chain management, and production scheduling.

Dr. Tarantilis was the recipient (more than ten times) of the "Best Teaching Faculty Award" from the Department of Management Science and Technology, Athens University of Economics and Business.

Emmanouil E. Zachariadis received the Diploma in chemical engineering from the National Technical University of Athens, Athens, Greece, in 2004 and the M.S. degree in computing science from the Imperial College of London, London, U.K., in 2005. He is currently working toward the Ph.D. degree in logistics management with the School of Chemical Engineering, National Technical University of Athens.

His current research interests include the analysis and design of intelligent computational methodologies for tackling large-scale complex combinatorial optimization problems, which arise in logistics, transportation, and production operations.

Chris T. Kiranoudis received the Diploma in chemical engineering and the Ph.D. degree in nonlinear mathematical programming from the National Technical University of Athens (NTUA), Athens, Greece, in 1988 and 1992, respectively.

He is currently an Associate Professor of computational methods in analysis and design of industrial processes and management operations with the School of Chemical Engineering, NTUA. He has authored more than 120 international scientific journal publications and presented papers at 50 national and international conferences. In addition, he has actively participated in 20 European Union projects and more than 60 national scientific projects in the fields of computational system analysis, optimization, and control for industrial processes and operations. His current research interests include the design and control of chemical processes and plants, numerical analysis, mathematical programming, risk analysis, and the development of information systems for logistics operations.