Available online at www.sciencedirect.com

ScienceDirect

Transportation Research Procedia 30 (2018) 197–206

Transportation Research Procedia

www.elsevier.com/locate/procedia

EURO Mini Conference on "Advances in Freight Transportation and Logistics" (emc-ftl-2018)

Algorithms for a Vehicle Routing Tool Supporting Express Freight Delivery in Small Trucking Companies

Luigi De Giovanni[a], Nicola Gastaldon[a,b,*], Massimilano Losego[b], Filippo Sottovia[b]

[a] *Dipartimento di Matematica "Tullio Levi-Civita", Università di Padova, via Trieste 63, 35124 Padova, Italy*
[b] *Trans-Cel Autotrasporti, via L. Da Zara 33, 35020, Albignasego, Italy*

Abstract

We present a tool to support the operations manager of small trucking companies. The tool integrates modules to manage pickup/delivery requests, determine initial assignment to trucks and service routes, react to dynamic events, support demand forecasting. Modules are interconnected by a cloud platform and take advantage from real-time data sharing. The core of the tool is a neighborhood search heuristic for a Vehicle Routing Problem with side attributes, to be solved in both static and dynamic settings. The efficiency of the algorithm is improved by enhanced search strategies, neighbor filtering and parallel implementation. A prototype implementation of the tool currently supports the operations at Trans-cel, a small freight transportation company. Tests on the field on different operational scenarios show that the algorithm fits both static and dynamic settings in terms of running times and quality of the suggested solutions.

Keywords: Express freight pickup and delivery; Vehicle routing; Decision support systems

1. Introduction

Freight trucking is an important component of the supply chain that needs a clever setup of operations. Trucking companies have made a considerable effort to provide their operational managers with decision support systems to suggest optimized vehicle routing and scheduling strategies, as well as actions to dynamically react to many

* Corresponding author. Tel.: +39-049-8626-855; fax: +39-049-7313-190.
E-mail address: nicola.gastaldon@phd.unipd.it

10.1016/j.trpro.2018.09.022

operational issues like vehicles or transportation network failures, real-time issued demands etc. For the most part, big companies have invested for such systems, whereas small enterprises usually operate with small computer support and rarely find off-the-shelf software that fits their specific business model. The scope of this paper is the development of a decision support platform tailored for small trucking companies for express freight delivery. The study has been inspired by the daily operations at Trans-Cel and by the development of *Chainment*, a cloud platform to integrate operations research, artificial intelligence and networking tools to support the actors of the supply chain, such as carriers, manufacturers, fuel companies, maintenance operators. The platform is designed to host different modules that collect, share and exploit data towards the collaborative optimization of several daily operations, including static and dynamic route planning, driver scheduling, consolidated truck refueling and maintenance, demand pricing and revenue management.

We focus on the modules that support or interact with the vehicle routing operations, based on the current vehicles' status and planned activities, as well as on information about pending and forecast transportation requests. The company operates a fleet of heterogeneous vehicles with specific capacity, loading facilities (e.g. tail lift, crane, side door) and operational costs. Each transportation request (or order) defines one or more pickup and delivery addresses: for example, in a multi-delivery order, freight has to be picked-up in one point and divided into parts to be delivered at different addresses. Each pick-up or delivery operation specifies a hard or soft time window and the size of the freight (from small items to full truckload). The orders are spread in a regional area, covering an area up to about 500 kilometers around the depot. Clients issue their requests in small advance (even a few hours before the first operation) and ask for a just-in-time service, with pickup and delivery within the same day or two consecutive days. A revenue is negotiated and associated to orders, and, in case of mandatory orders, a penalty for missed execution. The routes define the daily operations of each vehicle in terms of sequence of pickups and/or deliveries, starting from the position of the last operation of the day before (open routes). Routes are subject to hours-of-service regulations, and, sometimes, maximum duration, ending position and other side constraints. Preferences may exist on, e.g., assigning orders to specific vehicles, or order execution precedence.

Based on the information on orders, fleet-status and transportation network, available from the web platform, the operations manager has to determine, in different operational scenarios, the assignment of orders to vehicles and the related pickup/delivery routes that maximize the profit, taking operational constraints and preference matching into account.

The problem belongs to the broad class of Vehicle Routing Problems (VRP, for a general review, see Toth and Vigo 2014 among others). In order to support operations, it is solved in both the static version, to a-priori define initial daily routes, and the dynamic version, e.g., to accept and, in case, assign real-time issued orders in an online fashion. The static version is a Multi Attribute VRP (MAVRP, e.g. Layhani et al. 2015, Vidal et al. 2013) and, for similar problems, literature proposes both exact and heuristic approaches. Exact approaches are generally based on mathematical programming formulations solved by decomposition techniques (e.g. Baldacci et al. 2009 and Ceselli et al. 2009). Most of heuristics implement genetic algorithms and tabu search (e.g. Vidal et al. 2014, De Giovanni et al. 2017 or, for a survey, Vidal et al. 2013). For Dynamic VRPs with Pickup and Deliveries (D-PDP), literature mainly proposes two types of strategies to insert the new requests into the currently operated routes after the next destination in the route plan: reactive and anticipatory (e.g., Pillac et al. 2013, Psaraftis et al. 2016). At the time a new request arrives, reactive policies consider the pending requests already in the route plan and insert the new one. Sometimes, in case of low demand dynamism, reactive policies may use fast heuristics for the static version to re-optimize the routes. With the aim of better accommodating the new request, anticipatory algorithms exploit some stochastic information on further requests that are likely to show up in the future.

Off-the-shelf customizable tools are available and include systems to track and manage the fleet in real time (e.g. TomTom 2017), to optimize truck routes (e.g. Workwave 2017), to interconnect stakeholders and to synchronize their time windows (e.g. Transporeon 2017). In this paper, we introduce a new tool to support routing operations in small trucking companies: it integrates several functionalities that, to the best of our knowledge, are spread in different existing tools, or not present at all. In Section 2, we present the tool architecture and some relevant modules, focusing on their integration through a web platform for real-time data sharing. Modules rely on an optimization algorithm that solves the MAVRP introduced above and presenting interesting peculiarities in both the static and dynamic versions, as discussed in Section 3, where we propose a tabu-search approach based on De Giovanni et al. (2017) and improved by enhanced neighborhood exploration strategies, move filtering, hot start and parallel implementation. Section 4 presents significant application scenarios and related computational results, obtained by a prototype implementation of the platform modules that are currently working at Trans-cel and effectively supporting the operations management office.

2. Integrated modules of the decision support tool

The operations management office of a small trucking company has to face several decisional scenarios during a working day, as shown by the following illustrative example. Clients issue their transportation requests (in terms of e.g. freight features, pickup/delivery positions, time window preferences and dock slot availability) that are stored in an order repository. Based on this information, at the beginning of the working day, the office assigns requests to trucks and determines their routes. During the operations, further requests may be issued having pickup and/or delivery time windows falling in the current day. According to the real-time status of the fleet, the company has to decide if the order can be profitably taken in charge and, in case, modify vehicle routes accordingly. After deliveries, by exploiting the knowledge on customers' behavior, the trucking company may solicit some orders in convenient position, as to avoid driving empty trucks. By monitoring trucks' tank status, fuel requests are opportunely consolidated and negotiated with trusted fuel companies. Similarly, standard and extraordinary trucks maintenance operations are planned and interleaved with usual pickup and delivery operations. At present, several tools provide small trucking companies with some computer support for the above tasks, often obtained from customized off-the-shelf software. However, in many enterprises, business support applications are still tailor-made and often loosely coupled to each other. The tool we propose is made of several modules integrated through the web platform *Chainment*. The platform is designed as a single-page web application according to the Software-as-a-service (SAAS) model and follows a social network paradigm to interconnect users (trucking companies, customers, logistics operators, fuel companies etc.). As a part of Chainment, the modules interact by means of real-time data sharing, and take advantage from the available real-time and historical information (e.g., pending orders, vehicles, orders and road network real-time status), as illustrated in Figure 1. In the following, we describe the modules included in the operations management support tool.

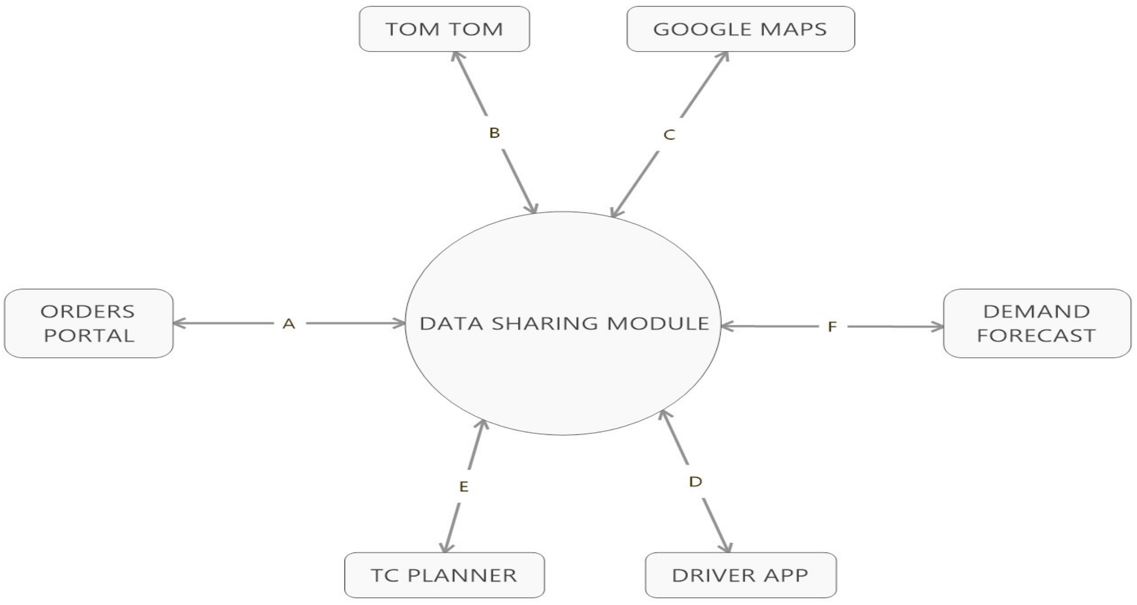

Fig. 1. The modules integrated in the support tool.

The *Orders Portal* module supports inserting and managing orders. It collects data on orders and provides different views to both carriers and their customers. Customers can make their own transportation requests by filling up an online form connected to the Data-Sharing module. Any change to the order data is tracked and notified to both parties in real-time. In particular, an interactive negotiation system that establishes the price of the incumbent order is a main component of such module. The carrier's operations manager has access to a profit estimator tool based on the evaluation of possible prices and marginal costs for the incumbent order, as suggested by historical and real-time data. A machine learning algorithm analyses historical data to fix a suitable price interval for the order to take in charge, based on past decisions concerning similar orders, as defined by some selected order features. Moreover, the Order Portal interacts with the Planning module to estimates the minimum and the maximum marginal cost of the order. This is done by dynamically inserting the incumbent order into the current route planning, taking into account the real-time status of the fleet and of the remaining currently assigned orders. The price and the cost intervals give an estimation of the possible profit related to a new order, guiding the operations manager in the negotiation process. After confirmation from both parties, orders are not changeable and appear in the data repository as orders in charge for the carrier, ready to be inserted into the daily planning (manually or by running the routing algorithm).

The *Fleet* module (*Driver App*) handles the tracking of vehicles' operations and movements: a mobile application has been developed in order to connect the drivers to the data-sharing module. By means of this module, drivers obtain their daily routes, available either as a sequence on a timeline or in navigation mode. As they progress with the trip, the drivers send feedbacks to the data-sharing module. This allows updating the order and vehicle status, as well as collecting useful information, such as departure and arrival time at each position, starting and ending time of each pickup or delivery operation or other activities (e.g. breaks). Moreover, the drivers can keep updated the status of the main vehicle components: they can regularly insert the pressure value of tires, the fuel status and overall kilometers cumulated by the vehicle, for the sake of maintenance. Drivers can also make use of a photo sharing system, in order to report miscellaneous events, such as damages to the vehicle or to the packaging, or documents received by customers after completing an operation. Moreover, the Tom Tom Telematics API provides further data that are automatically sent to the data-sharing module, thanks to the Tom Tom device integrated in each vehicle (e.g. driving or idle status of the vehicle, the actually travelled routes, etc.).

The *Planning* module (*TC Planner*) determines the assignment of daily operations to vehicles and the related routes, after collecting the information on orders, fleet and road network status from the other modules, including the ones that interconnect with Google Maps and Tom Tom Telematics to obtain real-time traffic information. To this end, the module includes a routing algorithm to solve the MAVRP stated above and provide a set of feasible routes. The module offers both a map view of the suggested routes and a drag-and-drop interface, which allows the operations manager to interact with the routing algorithm. The former view, thanks to the interaction with Google Maps and Tom Tom Telematics, shows both the actual route rode by each vehicle and the routes as arranged in the current planning. The latter view allows the user creating an initial (even partial) solution and launching the routing algorithm to complete and optimize it. The same interface gives the opportunity of overruling the routes suggested by the algorithm, by, e.g., fixing the final position of a vehicle, or the assignment of some orders to specific trucks, or a sub-route. All planning information are made available to the operations manager, such as the overall profit, the total costs of the planning, the total spatial and time distance. Moreover, any change in the planning registered through the interface can be evaluated by activating the routing algorithm to evaluate and show the impact of the modification.

At the beginning of the working day, the Planning module suggests an initial plan for daily operations after solving the static version of MAVRP. This can also be done whenever, during the day, the fleet, order or road network status diverge from the initially planned ones (e.g. road network failures or congestion, pickup or delivery operations taking longer than planned etc.), so that routes can be conveniently (re)optimized starting from the actual status gathered from the real-time data sharing module.

The Planning module is also able to suggest possible strategies to modify the current vehicle routes in response to several dynamic events. One main example are real-time issued orders. In this case, the module runs a dynamic version of the routing algorithm in order to include the new operations into the most convenient route. To this end, the interaction with the Fleet module through real-time data sharing allows collecting the information necessary to deploy a solution that is consistent with the actual status. Another example is represented by vehicle failures. In this case, the broken vehicle is excluded from the fleet, and related orders status is changed in order to let them be carried out by other vehicles. If load or unload operations are possible in the location where the vehicle broke down, a new pickup operation is associated to all orders on board, and the routing algorithm can assign the updated order to the remaining vehicles.

The *Demand Forecast* module is designed to predict possible future orders. The module exploits the historical information on orders to detect orders that are likely to be issued in the near future, so that they can be taken into account by the planning module. Moreover, profitable forecast orders may be detected by interaction with the Order module and proactively solicited by telemarketing, in order to increase the probability to show up. In order to identify fruitful regions, the module plots a heat map that shows the most visited areas: this, together with the Fleet and the Order modules, suggests convenient positions and order features that may be compatible with some running trucks. Notice that heat maps may depend on the time of the day or the day of the week, according to historical information on order features (position, date, operations time windows etc.).

3. The optimization engine

In order to achieve their goals, most of the modules integrated in the support platform need to solve a MAVRP both in the static and dynamic versions. From an operations research perspective, the problem presents interesting peculiarities since it mixes different features. Orders are typical of intercity less-than-truckload couriers, however, the presence of urgent just-in-time requests reduces the impact of freight consolidation, typical of this setting, and introduces features similar to express couriers' operations in urban contexts. In addition, literature for the dynamic version usually makes the assumption that the next destination of a vehicle cannot be changed (Pillac et al. 2013), which may not fit long haul trucking, since a driver may diverge and serve a new request if time windows allow. Moreover, in order to achieve their goals, modules take as much advantage as possible from real-time data, so that, as a consequence, they need to solve the MAVRP within limited running time. To this end, a heuristic approach is suitable: we start from the tabu-search algorithm proposed by De Giovanni et al. (2017) for a similar MAVRP, and we improve it by move filtering, hot start, parallel implementation and new search strategies, for a consistent integration with the modules of the support platform.

The basic algorithm considers a two-level optimization to deal with, respectively, the order-to-route assignment and the intra-route sequencing. The first level implements a tabu search with four neighborhoods: (i) move one order to a new route, (ii) swap two orders in different routes, (iii) move two orders from one route to a new one and (iv) move one order to a new route and replace it with an order taken from a third route. They are sequentially explored by a Variable Neighbourhood Descent (VND, Hansen et al. 2008). In particular, we apply neighborhood (i) as long as it is able to provide an improving solution. Then we look for an improvement by applying, in the order, one neighborhood (ii), (iii) and (iv) move (one neighborhood is explored only if the previous one does not provide a better solution). We thus come back to explore neighborhood (i) from the improving solution, if any, or from the best solution generated by (i), (ii), (iii) or (iv). In the last case, the selection of the next solution to visit is either deterministic (the best neighbor) or stochastic (one random neighbor among the best five). The algorithm stops after a maximum number of iterations without improvement, with a further limit on the overall number of iterations. For the sake of efficiency, (iv) considers only orders located within a given distance threshold. The second level optimization evaluates a score function associated to each neighbor, i.e., to fixed order-to-route assignments: for each route modified by the move, it runs an insertion heuristic and a local search to determine the sequence of the pickups and deliveries to take place in the day of operations, and a fast nearest neighbor heuristic to sequence next-day tasks. The score function is a weighted sum of the revenue associated to assigned orders, the operational costs, the penalties associated to missed priority orders or soft time windows, and further calibrated penalties to take violated constraints as well as preferences into account. A best insertion heuristic provides a fast initial solution. The algorithm's details are available from De Giovanni et al. 2017.

The algorithm is run by the route planning module to solve MAVRP in static settings. As described in Section 2, in addition to providing the initial routes according to orders and road network information, the same module supports the interaction between the algorithm and the operations manager, who can suggest an initial plan, modify or overrule the solution proposed by the algorithm. For the sake of this interaction, we have modified the algorithm to start from a given solution provided through the graphical interface. The operations manager is allowed to provide even partial or unfeasible initial solutions, which can be suitably processed thanks to violated constraints penalties.

Other modules solve a dynamic version of the MAVRP, e.g. to react to dynamic events during operations (Planning module) or to provide an estimate of the marginal cost of an order, which is useful to determine its price (Order Portal module). We observe that the expected inter-arrival times between different orders determines a low demand dynamism and we apply a reactive strategy based on re-optimization. In particular, we run the same algorithm to re-optimize the routes, using the input from different modules to collect the current vehicle, order and network status, and determine a synchronized initial solution.

As to reduce the algorithm response times, in particular for dynamic settings, we introduce the following methodological and implementation improvements. From a methodological point of view, we consider alternative exploration strategies for the tabu search. In De Giovanni et al. (2017), as soon as a neighborhood (i) local optimum is reached, the VND strategy tries to escape from it before allowing non-improving moves. This implies iterating the search of computationally expensive neighborhoods (ii), (iii) and (iv) whenever a local optimum is reached, until an improving solution is found. In the new exploration schema, we always allow a maximum number I_n of non improving neighborhood (i) moves before switching to one of the neighborhoods (ii), (iii) or (iv). The exploration switches back to neighborhood (i) as soon as an improving solution is found or, in any case, after a maximum number I_m of iterations. We compare two alternative switching methods: in the first (denoted by S1), at each switch we choose, in the order, either neighborhood (ii), (iii) or (iv); in the second (S2), we choose the switching neighborhood at random, with probability proportional to a measure of its *strength*. The strength of each neighborhood is preliminarily evaluated by running a simple tabu search using that neighborhood only on a set of instances, and measuring the average improvement with respect to the initial solution.

In order to speed up the search, we extend filtering to neighborhood (iii), i.e, we apply the simultaneous two-orders move only to a subset of pairs selected on proximity criterion. We recall that orders may have multiple pickups or deliveries involving different positions: we say that an order i is *close* to order j if at least one half of all the positions involved by i are within a distance threshold from any of the positions involved by j, and vice versa. Hence, filtered neighborhood (iii) includes only solution obtained by moving pairs of close orders. In our experiments, we have set this threshold to 20 km.

From an implementation point of view, further acceleration comes from a parallel implementation of the tabu search. In particular, we divide each neighborhood into subsets of equal size and we explore them by means of independent parallel threads. Notice that the algorithm adopts a steepest descent strategy to explore neighborhoods, so that the parallel implementation does not change the search trajectory and, hence, the provided solution, with respect to the sequential implementation, at least in deterministic settings.

For the sake of new order pricing, the order management module runs the dynamic MAVRP algorithm to estimate the marginal cost. In addition, the module runs a machine-learning algorithm exploiting historical data to analyze past decisions and predict a suitable interval price, to be compared to the marginal cost interval estimation. The algorithm uses a decision tree to implement an ordinal classifier (Frank and Hall, 2001), based on orders features like average distance between pickup and delivery positions, average distance from the depot, load size (weight and volume), urgency level, requested truck facilities. Given the features, the classifier applies a parametrized function that selects the interval to associate to a given order among a set of predefined intervals, sorted by average price. As usual, the parameters are obtained by training the classifier on a suitable dataset of orders. For this task, we have used a k-fold cross validation scheme, with a 4:1 ratio between training and validation sets (Picard and Cook, 1984).

4. Sample applications and computational results

A prototype of the routing support tool has been implemented and it is currently in use at Trans-Cel. The different modules store and share data from TomTom Telematics and GoogleMaps APIs (e.g., geo-referenced real-time and historical routes, information on dynamic events, order and road network status), and integrate the routing algorithm to solve the MAVRP in static and dynamic settings, as well as the price interval classifier. In the following, we analyze two typical operational scenarios: the setting up of the initial daily plan, and the insertion of new orders during the working day, with the related price negotiation.

The first scenario normally takes place at the beginning of a working day, when the operations manager assigns orders to vehicles and determines their initial routes. This task is supported by the Planning module, that runs the optimization algorithm for MAVRP in static settings, based on order, fleet and road network data collected from the others tool's modules. In order to show how the planning module is able to support the initial daily planning, we provide some sample results on the time needed to compute the vehicles routes and the quality of the proposed solutions. The results have been obtained by running the tabu search algorithm on 43 real instances collected from March to December 2017. The test benchmark is summarized in Table 1 and contains instances with up to 55 orders (27.8 on average), 116 pickup/delivery operations (72.2 on average) and 3 to 14 trucks (11.2 on average). Instance are divided into five groups, depending on the total number of operations, as reported in the first column. The number of instances per group appears in the second column. Following columns report, for each group, the average, minimum and maximum number of operations, orders, fleet vehicles.

Table 1. Instances.

Group	Count	Operations	Orders	Fleet
0-40	9	23.1 (12.0 ; 38.0)	7.7 (3.0 ; 12.0)	7.1 (3.0 ; 14.0)
41-80	9	61.9 (51.0 ; 79.0)	20.7 (14.0 ; 29.0)	12.3 (10.0 ; 14.0)
81-90	10	84.5 (81.0 ; 90.0)	33.3 (30.0 ; 40.0)	12.0 (10.0 ; 13.0)
91-100	9	93.9 (91.0 ; 97.0)	38.3 (34.0 ; 44.0)	12.3 (11.0 ; 14.0)
101-116	6	108.3 (103.0 ; 116.0)	43.7 (36.0 ; 55.0)	12.7 (12.0 ; 13.0)
All	*43*	*72.2 (12.0 ; 116.0)*	*27.8 (3.0 ; 55.0)*	*11.2 (3.0 ; 14.0)*

Results are shown in Tables 2, 3 and 4. The first two tables show the effect of neighborhoods (iii) and (iv) filtering and parallel implementation on the VND exploration strategy. Columns refers to different versions of the algorithm with Best Insertion heuristic only (BI), tabu search with VND and deterministic neighborhood exploration (VND-DET), tabu search with additional three runs of the randomized exploration (VND-RND), filtered (iv) neighborhood (+F), and parallel implementation with four threads (+P). Each row of Table 2 reports the per-cent gaps (average, minimum and maximum within the instance group) with respect to VND-RND, taken as baseline (it obtains the best solutions of all instances but four and has the best average performance). The rows of Table 3 show, for each instance group, the running times in seconds (average, minimum and maximum) on an Intel Core i5-5200 2.20 GHz CPU with 8 GB RAM. Each exploration stops after 60 not improving iterations or, in any case, after 100 iterations. To assess results, we solve the continuous relaxation of a set covering formulation by column generation (in a preliminary implementation) and obtain bounds for 38 out of the 43 instances. Bounds certifies that VND-RND finds nine optimal solutions (mainly in the group of the smallest instances) and, in the remaining cases, the solution is on average at most 1.2% far from the optimum (5.8% in the worst case).

Table 2. Computational results for the planning scenario, VND search strategy: gap towards VND-RND (baseline).

Group	BI	VND-DET	VND-DET+F(+P)	VND-RND	VND-RND+F	VND-RND+F+P
0-40	2.8 (0.0;13.3)	0.1 (0.0;0.8)	0.5 (0.0;2.4)	–	0.0 (0.0;0.0)	0.0 (0.0 ; 0.4)
41-80	5.6 (0.0;14.6)	0.1 (0.0;0.6)	0.1 (0.0;0.6)	–	0.0 (0.0;0.4)	0.0 (-0.7 ; 0.4)
81-90	6.5 (0.2;10.6)	0.6 (0.0;2.1)	0.7 (0.0;2.1)	–	0.1 (-1.9;0.8)	0.2 (0.0 ; 0.8)
91-100	8.4 (2.1;14.3)	0.3 (0.0;1.0)	1.2 (-0.9;4.1)	–	0.6 (-0.9;3.1)	0.8 (-0.9 ; 3.9)
101-116	8.2 (3.5;14.9)	0.3 (0.0;1.0)	0.4 (-0.2;1.0)	–	0.0 (-0.2;0.2)	0.3 (-0.2 ; 1.0)
All	*6.2 (0.0;14.9)*	*0.3 (0.0;2.1)*	*0.6 (-0.9;4.1)*	–	*0.1 (-1.9;3.1)*	*0.3 (-0.9 ; 3.9)*

Table 3. Computational results for the planning scenario, VND search strategy: running times.

Group	BI	VND-DET	VND-DET+F	VND-DET+F+P	VND-RND	VND-RND+F	VND-RND+F+P
0-40	0.0 (0.0;0.1)	0.6 (0.0;4.3)	0.4 (0.0;3.0)	0.3 (0.0;1.8)	2.3 (0.0;16.4)	1.5 (0.0;9.8)	1.0 (0.0;6.1)
41-80	0.2 (0.0;0.4)	5.6 (0.0;11.8)	3.2 (0.0;6.4)	1.9 (0.0;3.6)	27.4 (0.0;72.3)	14.9 (0.0;43.5)	8.6 (0.0;19.6)
81-90	0.3 (0.2;0.5)	10.5 (4.7;19.4)	6.0 (2.3;15.0)	3.4 (1.4;8.5)	44.8 (24.5;83.1)	24.2 (12.4;44.5)	15.4 (6.5;29.7)
91-100	0.6 (0.3;1.3)	15.3 (6.8;33.8)	7.8 (3.6;17.5)	4.4 (2.1;9.9)	71.5 (28.5;132.2)	34.5 (16.7;67.9)	18.5 (8.1;33.3)
101-116	1.6 (0.4;6.2)	33.3 (7.9;88.6)	14.1 (4.7;28.6)	7.6 (2.7;14.8)	145.1 (38.8;369.9)	66.3 (23.3;96.0)	33.9 (11.8;48.6)
All	*0.5 (0.0;6.2)*	*11.6 (0.0;88.6)*	*5.7 (0.0;28.6)*	*3.2 (0.0;14.8)*	*51.8 (0.0;369.9)*	*25.5 (0.0;96.0)*	*14.2 (0.0;48.6)*

VND-RND always provides the best average results (and, in fact, the best available solutions but in four cases, where VND-RND+F wins) in less than one minute, on average; for larger instances running times are about six minutes at most. Filtering neighborhoods halves running times, both for the deterministic and for the randomized versions of VND, while preserving solution quality: e.g., VND-RND+F is only 0.1% worse than VND-RND, on average. Running times are further reduced by about 40% on average thanks to the parallel implementation (that maintains solution quality unaffected, except for random fluctuations). This allows obtaining solutions in less than one minute in the worst case, even for the multistart randomized version of the algorithm, if neighborhood filtering and parallelization are switched on.

The impact of the new exploration schema is shown in Table 4. Tests takes randomization (RND), neighborhood filtering (F) and parallel implementation (P) into account, and we set $I_n = 15$ and $I_m = 5$. For each of the proposed switching methods S1 and S2, the table reports the per-cent gap with respect to VND-RND and the running time in seconds, showing, for each instance group, the average, minimum and maximum values. Concerning S2, the probability of choosing switching neighborhood (ii), (iii) or (iv) is set to, respectively 55.8%, 31.1% and 13.1%.

Table 4. Computational results for the planning scenario, S1 and S2 search strategies.

Group	S1-RND+F+P gap (%)	S1-RND+F+P time (s)	S2-RND+F+P gap (%)	S2-RND+F+P time (s)
0-40	0.0 (-0.4 ; 0.0)	0.7 (0.3 ; 3.8)	0.0 (-0.4 ; 0.0)	0.7 (0.2 ; 3.1)
41-80	0.2 (-0.2 ; 1.0)	4.9 (0.3 ; 13.1)	0.4 (-0.2 ; 1.3)	4.3 (0.2 ; 11.1)
81-90	-0.3 (-4.9 ; 0.8)	8.8 (5.8 ; 15.0)	0.1 (-5.1 ; 2.5)	8.3 (4.7 ; 12.7)
91-100	-0.6 (-3.9 ; 0.9)	11.8 (7.4 ; 16.6)	-0.1 (-3.5 ; 1.8)	11.2 (6.7 ; 15.6)
101-116	0.3 (-0.3 ; 1.8)	22.3 (8.6 ; 43.6)	0.6 (-0.1 ; 3.2)	18.5 (6.8 ; 32.4)
All	*-0.1 (-4.9 ; 1.8)*	*8.5 (0.3 ; 43.6)*	*0.2 (-5.1 ; 3.2)*	*7.6 (0.2 ; 32.4)*

The new search strategies have a small impact on the quality of the solutions: the average values for the different instance groups go from a 0.6% improvement to a 0.6% worsening (1.8% and 3.2% in the worst case for S1 and S2 respectively). Both S1 and S2 greatly improve efficiency, providing, respectively, a 40% and 47% reduction of the running times (34% and 45% respectively for larger instances). Notice that S2 is, on average, slightly faster than S1: in fact, S2 more often switches to neighborhood (ii), whose complexity is lower than (iii) and (iv). On the other hand, S1 better preserves solution quality, attesting to the role of all the proposed neighborhoods in generating improving solutions during switching.

The results confirm that the proposed optimization algorithm for MAVRP can be suitably integrated in the Route Planning module as to provide the operations manager with good quality initial daily routes within short running times.

In the second application scenario, new orders show up during the workday operations. For each new order, a price is negotiated between the customer and the carrier. As we have seen, the Order Portal module supports the carrier in this task by estimating two intervals related to a suggested order price and to the marginal cost of inserting the new order in the current routes. We have conducted some experiments to evaluate the capacity of the module to support this operational scenario.

We have estimated the ordinal classifier parameters on a dataset of 1604 orders including 13 features per order. The classifier takes negligible time to suggest a price interval and we estimate a score of 75.3%. The score is computed as follows: we consider an evaluation dataset and, for each entry, we measure the classifier error as the distance (in number of intervals) between the selected interval and the correct one; we compute the maximum error, i.e., the distance of the correct interval from the farthest one; the score is obtained by comparing the norms of the vectors associated to the classifier and to the maximum errors, such that 100% corresponds to no errors and 0% to all maximum errors.

Concerning the marginal cost interval, the lower bound is obtained solving MAVRP in dynamic setting to insert the new order: this is done by applying the tabu search algorithm (we considered version VND-RND+F+P) to re-optimize the routes starting from the current fleet and order status. We have conducted some experiments by simulating, for each instance in the benchmark, five cases, each including a new order randomly selected from all the orders in different instances. For the same cases, we estimated the worst case marginal insertion cost by inserting the new order into each of the current vehicle routes, running an intra-route optimization and taking the worst assignment. The time needed to obtain the lower and upper bounds (average, minimum and maximum) is shown in the second column of Table 5.

Table 5. Computational results: re-optimization in dynamic settings (running times in seconds).

Group	Marginal costs evaluation	3-order re-optimization VND-RND+F+P	3-order re-optimization S1-RND+F+P
0-40	0.3 (0.0 ; 0.5)	0.5 (0.2 ; 0.8)	0.3 (0.1 ; 0.5)
41-80	5.4 (0.7 ; 8.4)	8.0 (4.4 ; 13.6)	5.0 (2.7 ; 7.7)
81-90	7.3 (3.2 ; 12.0)	16.7 (13.9 ; 21.1)	9.1 (7.8 ; 12.8)
91-100	11.4 (6.9 ; 15.8)	23.8 (20.8 ; 26.6)	13.9 (13.5 ; 17.1)
101-116	11.6 (8.8 ; 19.4)	24.4 (22.5 ; 27.7)	15.2 (14.1 ; 18.1)
All	*9.1 (0.0 ; 19.4)*	*20.4 (0.2 ; 27.7)*	*12.4 (0.1 ; 18.1)*

The algorithm takes a few seconds to estimates the best and the worst marginal costs (about 20 seconds in the worst case), which is fairly compatible with negotiation activities.

A further experiment, whose results appear in the last columns of Table 5, measures the time needed to solve the dynamic MAVRP to suggest the assignment of more than one new simultaneous orders. For each instance in the benchmark, we have chosen at random three new orders to be inserted and repeated the experiment five times, comparing two versions of the tabu search algorithm, namely VND-RND+F+P and S1-RND+F+P: even in this case, and in particular for the S1 exploration strategy, running time seems to be largely compatible with operational settings.

5. Conclusions

In this paper we have presented an integrated tool to support the fleet operations of a freight trucking company. The tool is especially tailored for small companies handling long-haul transportation requests with express deliveries. The tool is based on the interaction between different modules that support the different tasks of the operations management office, like order management (insertion, price negotiation etc.), the setting-up of initial plans for daily operations, reaction to dynamic events (real-time issued orders, vehicle or road network failures, delays in pickup or delivery operations etc.), tracking vehicle activities and status, and other tasks. The tool is developed on the top of a cloud platform that provides real-time and historical data sharing. This allows overcoming the limitations of off-the-shelf applications, each designed for specific tasks and difficult to integrate to each other.

The core of most of the modules is a heuristic routing algorithm that solves MAVRP in both static and dynamic settings: the algorithm implements a two-levels tabu search technique to explore possible order-to-vehicle assignments and possible pickup/delivery operations sequences. Thanks to improved search strategies, neighborhood filtering and parallel exploration, the running times are compatible with operational settings in both static and dynamic cases, so that the algorithm can suitably interact with the tool's modules.

A prototype implementation of the modules described in this paper is currently working at Trans-cel, the small transportation company that promotes this research: the sample application scenarios and computational results reported in this paper show that the proposed modules are able to effectively support the operations management office.

The underlying cloud platform makes the tool open to collaborative decision making. For example, more carriers may register on the platform, each developing its own Planning module that interacts with the other modules, so that selected information on orders can be shared and the vehicle routes for different fleets optimized in a collaborative fashion. This is the object of future research. Further research directions include the development of anticipatory policies for the dynamic and stochastic MAVRP, taking order historical data into account, as well as the automatic detection of profitable future demand based on the heat maps provided by the demand forecast module.

References

Baldacci, R., Mingozzi, A., 2009. A unified exact method for solving different classes of vehicle routing problems. Mathematical Programming, 120 (2), 347-380.

De Giovanni, L., Gastaldon, N., Lauriola, I., Sottovia, F., 2017. A heuristic for multi-attribute vehicle routing problems in express freight transportation. In: Sforza A., Sterle C. (Ed.) Optimization and Decision Science: Methodologies and Applications. ODS 2017. Springer Proceedings in Mathematics & Statistics, 217, Springer, Cham.

Ceselli, A., Righini, D., Salani, M., 2009. A column generation algorithm for a rich vehicle routing problem. Transportation Science, 43(1), 56–69.

Frank, E., Hall, M., 2001. A Simple Approach to Ordinal Classification. Proceedings of the 12th European Conference on Machine Learning. Springer-Verlag, London, 145–156.

Hansen, P., Mlanedovic, N., Pérez, J.A.M., Variable Neighborhood Search: Methods and Applications, Springer-Verlag, 2008, 6:319-360.

Lahyani, R., Khemakhem, M., Semet F., 2015. Rich vehicle routing problems: From a taxonomy to a definition. European Journal of Operational Research, 241(1), 1–14.

Picard, R., Cook, D., 1984. Cross-Validation of Regression Models. Journal of the American Statistical Association, 79 (387), 575-583

Pillac, V., Gendreau, M., Guéret, C., Medaglia, A.L., 2013. A review of dynamic vehicle routing problems. European Journal of Operational Research, 225(1), 1–11.

Psaraftis, H.N., Wen, M., Kontovas, C.A., 2016. Dynamic vehicle routing problems: three decades and counting. Networks, 67(1), 3–31.

TomTom, 2017. TomTom Telematics Webfleet. Retrieved on telematics.tomtom.com/en_us/webfleet/

Toth, P., Vigo, D., 2014. Vehicle Routing: Problems, Methods and Applications. MOS-SIAM Series on Optimization.

Transporeon, 2017. Transporeon logistics software. Retrieved on www.transporeon.com/us/

Vidal, T., Crainic, T.G., Gendreau, M., Prins, C., 2013. Heuristics for multi-attribute vehicle routing problems: A survey and synthesis. European Journal of Operational Research, 231(1), 1–21.

Vidal, T., Crainic, T.G., Gendreau, M., Prins, C., 2014. A unified solution framework for multiattribute vehicle routing problems. European Journal of Operational Research, 234(3), 658–673.

Workwave, 2017. Viamente. Retrieved on www.viamente.com