Available online at www.sciencedirect.com

ScienceDirect

Procedia Computer Science 159 (2019) 2589–2597

Procedia Computer Science

www.elsevier.com/locate/procedia

23rd International Conference on Knowledge-Based and Intelligent Information & Engineering Systems

A Big-Data-Analytics Framework for Supporting Logistics Problems in Smart-City Environments

Alfredo Cuzzocrea[a,*], Massimiliano Nolich[a], Walter Ukovich[a]

[a] *University of Trieste, Piazzale Europa 1, 34127 Trieste, Italy*

Abstract

Containers delivery management is a problem widely studied. Typically, it concerns the container movement on a truck from ships to factories or wholesalers and vice-versa. As there is an increasing interest in shipping goods by container, and that delivery points can be far from railways in various areas of interest, it is important to evaluate techniques for managing container transport that involves several days. The time horizon considered is a whole working week, rather than a single day as in classical drayage problems. Truck fleet management companies are typically interested in such optimization, as they plan how to match their truck to the incoming transportation order. This planning is a relevant both for strategical consideration and operational ones, as prices of transportation orders strictly depends on how they are fulfilled. It is worth noting that, from a mathematical point of view, this is an NP-Hard problem. In this paper, a Decision Support System for managing the tasks to be assigned to each truck of a fleet is presented, in order to optimize the number of transportation order fulfilled in a week. The proposed system implements a hybrid optimization algorithm capable of improving the performances typically presented in literature. The proposed heuristic implements an hybrid genetic algorithm that generate chains of consecutive orders that can be executed by a truck. Moreover, it uses an assignment algorithm based to evaluate the optimal solution on the selected order chains.

Keywords: Decision Support Systems; Hybrid Algorithms; Logistics

1. Introduction

Containers delivery management from ships to factories or wholesalers and vice-versa is a problem widely studied. As there is still a relevant interest in shipping goods by container, and that delivery points can be far from railways in various areas of interest, it is relevant to study heuristics for managing container transport over several days, typically for a week. It is worth noting that this problem is NP-Hard.

* Corresponding author.
E-mail address: alfredo.cuzzocrea@dia.units.it

10.1016/j.procs.2019.09.257

Decision Support Systems for Smart Cities has been proposed in literature, as they can face different applicative problems, such as electro-mobility [3], logistics [1, 10], comfort [16, 13], cyber-security [8], etc. Framework for managing logistics problem has already been proposed [15, 1, 4]. More precisely, truck fleet management companies are typically interested in such optimization, as they plan how to match their truck to the incoming transportation order. This planning is a relevant both for strategical consideration and operational ones, as prices of transportation orders strictly depends on how they are fulfilled.

The main contribution of this paper is to introduce a novel Decision Support System (DSS) capable of managing the container drayage management problem. The DSS optimizes the tasks to be assigned to each truck of a fleet in order to optimize the number of transportation order fulfilled in a week. The proposed system implements a hybrid optimization algorithm capable of improving the performances typically presented in literature. The system implements an hybrid genetic algorithm that generate chains of consecutive orders that can be executed by a truck. Moreover the proposed heuristic uses an assignment algorithm based on the evaluation of optimal solutions for selected order chains.

2. Related Work

The problem of container drayage [20, 19, 18, 17] leads back to one of the following templates. VRP is the problem of minimizing the total travel distance of a number of vehicles, based on various constraints, where every customer must be visited exactly once. To apply this model to the problem of container drayage, the time constraints must be added to the classical formulation and, therefore, VRP with Time Windows (VRPTW) [11] must be considered. As VRP is NP-hard, several heuristics have been proposed to solve.

The problem can also be considered as an assignment Problem (AP). In this case, the problem is to find a one-to-one matching between n tasks and n agents while minimizing the total cost of the assignment [14]. Many variations of the classic AP have been proposed in order to consider different further assumptions. In [4], an heuristic approach to manage wide fleet of vehicles using rolling horizon approach has been proposed.

3. Problem Formulation

The shipping companies, to move the containers in the hinterland, go to transport companies on trucks. These are responsible for transport management of containers between ports/dry-ports to guarantee door-to-door service to companies that require it. Usually, within the company, the plans proceed as follows:

1. in the morning any problems arising during the night are resolved and to check the feasibility of further turns;
2. check if the trips assigned the day before require a change in based on the current state of the fleet;
3. check which trips still need to be covered;
4. in the afternoon we proceed with the new assignments, starting the work of plans for the following day, considering drivers first available.

The planer then works by trying to take the entire week into account but always pays particular attention to the next day.

The transport company in question operates trips defined ABC, where A and C represent the port/interport and B represents the place where the goods come from unloaded/loaded. The movement of the container loaded from the port/interport to the B is defined journey of importing merchandise (or *import*), vice versa the displacement of a empty container is called an export trip (or *export*).

Import travel. In the import travel (Figure 1) the driver must go to the port/interport established to pick up the container containing the goods, transport it and download it to the company that has made the request and in it deposit the emptiness at the port/interport established. It is therefore essential to match the time-frames required by customer orders.

Export travel. On export travel (Figure 1), the driver must pick up the empty container at the port/interport established, go to the company that has made the request to load the goods and transport it to the port/interport established. In this case, it is fundamental to respect the timetables imposed by shipping/railway companies.

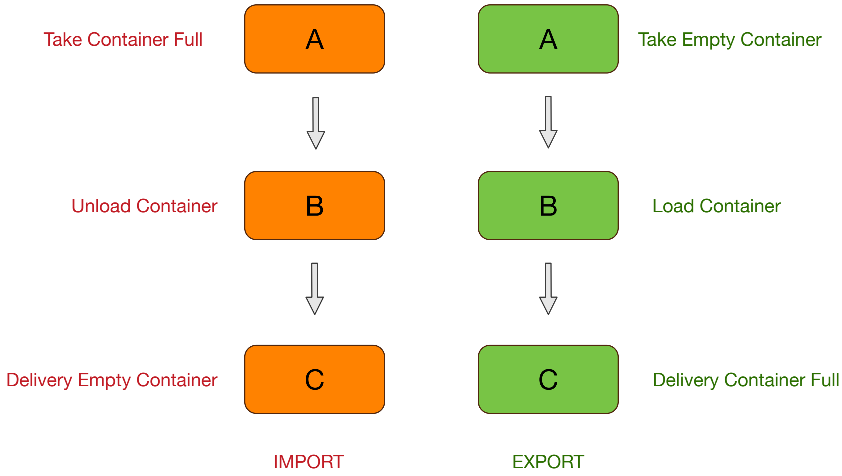

Fig. 1. Import and Export problems.

Other types of travel. The journeys made by the transport company can be further classified types; a distinction is made based on whether the point A and point B coincide:

- if C=A we talk about *round trip*;
- if C≠A one speaks of *one way travel*.

Another distinction is made on the basis of agreements between the maritime company and shipping company:

- we talk of *carrier* journey if the shipping company takes care of the entire land transport of the container;
- we talk instead of a *merchant* journey when land transport is managed by the transport company and the shipping company only puts it the empty container in its terminal is available.

4. Proposed Algorithm

From the analysis of decision making roles inside logistics companies, the planner is the person who deals with managing order-truck assignments. His/her role is related to solve the problems that arose the previous night and plan trips for the next day. In general, he/she tries to take into account the whole week, but always pays particular attention to the next day.

The goal is therefore the realization of a framework that allows to optimize the management of a fleet of vehicles taking into account several days, completing as many orders as possible and minimizing the number of empty trips, as they involve higher costs.

Once an order has been completed then the algorithm must decide what the next order to be assigned to a particular driver must be.

The decision is influenced both by current state of the fleet and by knowledge of future assignable orders. For this reason, given the temporal nature of the problem, it was decided to use dynamic programming [15]. Dynamic programming is an optimization method where a complex problem is decomposed into a sequence of more simple problems. This method is similar to the divide and conquer technique, but while the latter generates many identical sub-problems to be solved at each recursive call, the dynamic programming stores solutions from time to time make them available without further calculation. For this reason, greater use of memory is balanced by shorter processing time.

4.1. Formalization of the Method

The main concepts of dynamic programming are:

- period: it is the unit of processing of the problem that is not further decomposable. The decomposition of a complex problem in N periods, sequentially solving one at a time, and the essential characteristic of dynamic programming: each period is resolved as a classic optimization problem, whose solution is used for define the characteristics of the following period. Note that the concept period, however, must not necessarily be temporal;
- status: at each period of the optimization problem is associated with a state; it contains the information necessary to understand what they may be the repercussions that a decision made now may or may have in the future;
- recursive optimization: the recursive optimization procedure allows to solve a problem of N periods finding the solution of a single-period problem including sequentially the following periods and the global optimum is not achieved.

In this paper, the dynamic programming method is applied to the single truck, with the aim of finding the sequence of orders that minimizes the sum of empty trips, so the period is represented from the order while the state is represented by the city to where the driver is at the end of the execution of an order (and therefore also the time of availability at A).

We call the function $f_n(d_n, s_n)$ the cost function of the period, where d_n represents an admissible decision chosen by the D_n set of eligible decisions and s_n the status of the process at period n. The objective function is the choice of the best sequence of decision variables $d_n, d_{n-1}, \ldots, d_0$ to solve the following problems:

$$v_n(s_n) = \min\left[f_n(d_n, s_n) + f_{n-1}(d_{n-1}, s_{n-1}) + \cdots + f_0(d_0, s_0)\right] \tag{1}$$

where $s_{m-1} = t_m(d_m, s_m)$ $(m = 1, 2, \ldots, n)$, $d_m \in D_m$ $m = 0, 1, \ldots, n$.

We define $v_n(s_n)$ as a function of the optimal value; it represents the total cost of all the periods.

In this project, each working day is divided into two periods, morning and afternoon, assuming that a driver is able to complete a maximum of two orders a day and that the optimization has as time horizon the entire working week (5 days).

The following assumptions have been considered:

- consider driver and truck a single entity;
- make no distinction between type of goods, type of container, shipping company;
- consider as the only legal restriction the hours of sleep per day provided for law (9 hours);
- at the beginning of the working week, you already know all the planned orders (time horizon).

The planner can therefore be easily extended to integrate new constraints.

The first action carried out by the planner is order analysis. Data concerning the order for the next working week days are collected, and in particular data about the cities involved in A, B and C points with the respective dates and times are managed. The next phase consists in defining the planning time horizon: the newly-read orders are distributed in K distinct groups each day, based on the day and time slot. The construction of successive order chains to be assigned to each driver is realized through the feasibility analysis: in this phase it is verified whether and which orders can be carried out by a driver starting from the current state s_i. The verification between two o_n and o_m orders $(n \neq m)$ is performed by comparing the expected arrival time at point C of o_n with the time of point A of order o_m. If o_n and planned for the following day, it is added an additional parameter to indicate the hours of sleep.

$$\mathrm{HOUR}_C(o_n) + h + h_d < \mathrm{HOUR}_A(o_m) \tag{2}$$

So if it is worth equation (2), where h is the estimated distance in hours between $\mathrm{CITY}_C(o_n)$ and $\mathrm{CITY}_A(o_m)$ and h_d is the number of sleep hours required by law, then o_m is added to the list of successor candidates of o_n.

For sake of simplicity, two time interval per day ($K = 2$), called morning and afternoon, are considered in the following. Starting from the time horizon previously defined, a scan cycle is performed for each Monday morning order (group 1), verifying for each of these the feasibility with the orders of Monday afternoon (group 2). In this way we obtain a list of all the possible and feasible combinations. If the feasibility condition is not met, the order is also memorized as it is compared with the orders of Tuesday morning (group 3). If we assume to perform a planning based on four periods, at the end of the cycle we will obtain a set of lists, the longest will have size 4 and the shortest will have size 1. At this point, we proceed with the recursive optimization by applying the backward induction method (backward induction). The method produces the best sequences that can be obtained without containing joint orders (see Figure 2).

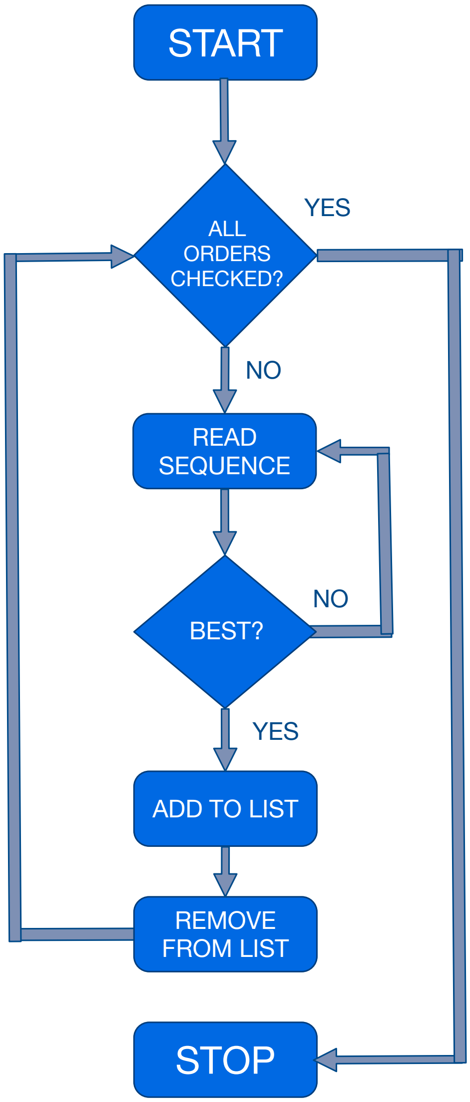

Fig. 2. Backward induction algorithm

To obtain results in a format suited for planning, the output is composed by a class containing an irregular three-dimensional matrix (jagged array) $v_{i,j,k}$, where i represents a list, j represents a sub-list and k a single order. Each execution of the recursive optimization generates the matrix just mentioned.

At the end of the recursive optimization a set of P three-dimensional matrices are obtained, where P is the stage number chosen for the planning. As P varies, the number of sequence groups of orders vary accordingly (see Table 1).

The last phase of the program consists in assigning to the drivers of the groups created.

The assignment has been performed by applying the unbalanced version of the Hungarian algorithm [12].

P	Groups
1	10
2	5
3	4
4	3
5	2
6	2

Table 1. Number of groups of sequences based on the number of stages (K=2)

5. Experimental Results

5.1. One-Stage Planning

In Figure 3 is illustrated the behavior of the proposed DSS for a one-stage plan. As you can see from the map the orders assigned to the driver are very localized. This happens because and at the end of the execution of an order the program tends to look for the next order in such a way as to minimize the distance $d(o1; o2)$ to pick up the new container.

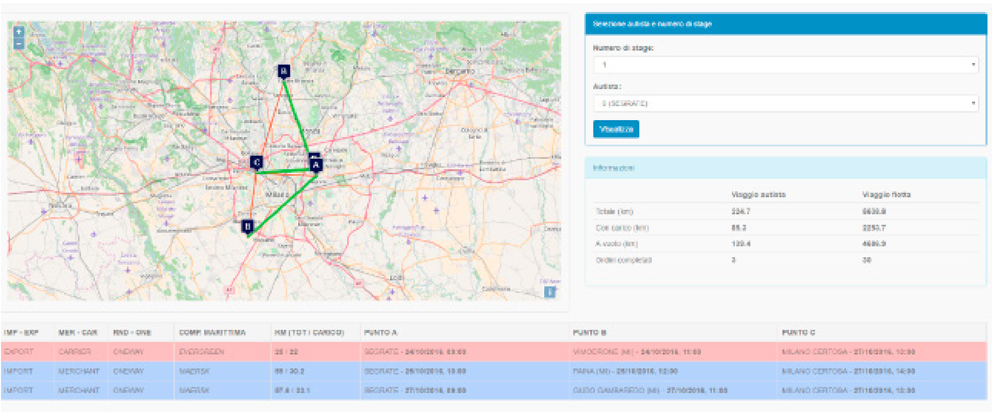

Fig. 3. One stage example

5.2. Four-Stage Planning

Figure 4 shows the behavior of the proposed DSS for a four-stage planning.

As you can see from the map the orders assigned to the driver are more distributed throughout the territory; this happens because one has been assigned to the driver chain of orders involving a greater number of ports/interports.

5.3. Fleet Planning

The analyzes were made through 10 random generations of 100 orders each, where these can be both long (within 800 km) and short (within 200 km).

Through various random executions, distinguishing between oneway and roundtrip orders, it was found that the algorithm works better in the first case as the cities A and C are distributed in a less localized manner.

As seen from the graph in Figure 5 and in Table 2 the number of total km traveled by the fleet decreases with increasing of the number of stages, while the number of km traveled without a container increases. This is due to the fact that with the increase of the stage the drivers execute orders more distant than their path, increasing the coverage radius.

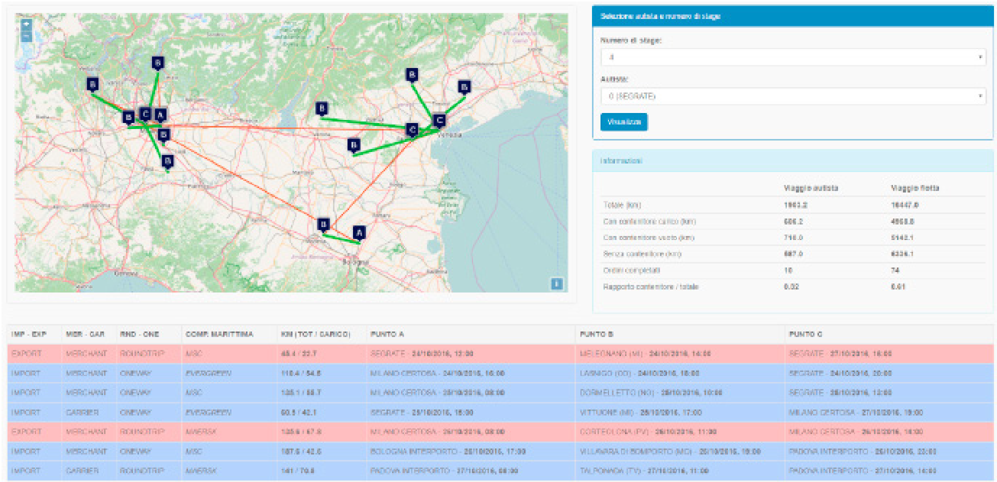

Fig. 4. Four stage example

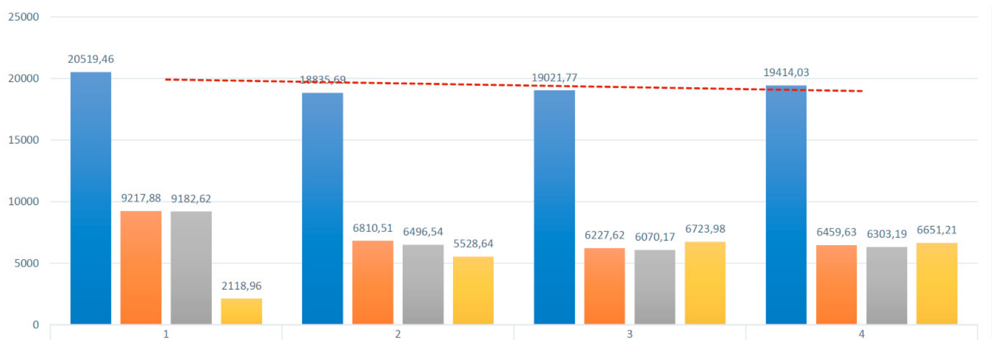

Fig. 5. Fleet assignment analysis

Nr. Stage	1	2	3	4
Total km	20519	18835	19021	19414
km Container Full	9217	6810	6227	6459
km Container Empty	9182	6496	6070	6303
km Without Container	2118	5528	6723	6651

Table 2. Performance of the proposed method using from 1 up to 4 stages.

The proposed method has been compared to the algorithm presented in [4], showing a mean improvement of around 5% for the tested set of orders.

6. Conclusions and Future Work

In this paper, classes of orders such as import/export and merchant/carrier, in addition to the owner of the container were considered. This information for how or were not used for realization of the plans as we wanted to concentrate on improving the coverage of orders and reduction of kilometers traveled. In more general cases, the elements considered can be increased, starting with the characteristics of the driver himself (age, qualifications, experience ...), from the type of containers (20 feet, 40 feet, HiCube etc.), from the type of goods transported, etc. However, the modular structure of the framework allows it to be easy extended.

For what concerns the making the chains, from the various analyzes conducted, one thinks of a point weak of the algorithm is the assignment of chains to drivers: when one chain is assigned to a driver, all sequences are eliminated they contain chain orders; in this way it can easily happen that if the orders are not well distributed, many ”long” chains are eliminated, hence a lower number is assigned to the fleet. It is also noted that the 3-stage plan is the one that provides less satisfactory results: this is probably caused by the chosen weekly subdivision criterion, that is, and 3 groups containing sequences up to a maximum of 3 plus a group of sequences from one order.

For what concerns input quality, all simulations were done with an order input and random cities created by the Generator; not having general knowledge of the distribution of orders in the practical field, this input can not therefore be considered real.

For what concerns processing capabilities, the results were obtained by setting the 4 to maximum number of stages in order to limit the computational effort required. The assignment of one eight of 10 drivers on one 50 input of 100 orders considering 4 stages is processed in the order of seconds (on average 2 seconds); if the number of orders is increased to 300, with one 6 stage plan, the computation time is estimated at about 4 hours.

Future work is related to several aspects. From a side, we plan to devise more experiments in the context of real-life settings, for obtaining a better assessment of our framework’s capabilities. From another side, we plan to extend our framework with innovative capabilities that may confer *flexibility* (e.g, [5, 2]), *scalability* (e.g, [7]) and *privacy-preservation* (e.g, [6, 9]) to it.

References

[1] E. Cascetta, A. Carteni, F. Pagliara, and M. Montanino. A new look at planning and designing transportation systems: A decision-making model based on cognitive rationality, stakeholder engagement and quantitative methods. *Transport policy*, 38:27–39, 2015.

[2] Georgios Chatzimilioudis, Alfredo Cuzzocrea, Dimitrios Gunopulos, and Nikos Mamoulis. A novel distributed framework for optimizing query routing trees in wireless sensor networks via optimal operator placement. *J. Comput. Syst. Sci.*, 79(3):349–368, 2013.

[3] M. Clemente, M. P. Fanti, G. Iacobellis, M. Nolich, and W. Ukovich. A decision support system for user-based vehicle relocation in car sharing systems. *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, 48(8):1283–1296, Aug 2018.

[4] M. Clemente, M. P. Fanti, M. Nolich, G. Stecco, and W. Ukovich. Modelling and solving the multi-day container drayage problem. In *2017 13th IEEE Conference on Automation Science and Engineering (CASE)*, pages 231–236, Aug 2017.

[5] Alfredo Cuzzocrea. Combining multidimensional user models and knowledge representation and management techniques for making web services knowledge-aware. *Web Intelligence and Agent Systems*, 4(3):289–312, 2006.

[6] Alfredo Cuzzocrea and Elisa Bertino. Privacy preserving OLAP over distributed XML data: A theoretically-sound secure-multiparty-computation approach. *J. Comput. Syst. Sci.*, 77(6):965–987, 2011.

[7] Alfredo Cuzzocrea, Rim Moussa, and Guandong Xu. Olap*: Effectively and efficiently supporting parallel OLAP over big data. In *Model and Data Engineering - Third International Conference, MEDI 2013, Amantea, Italy, September 25-27, 2013. Proceedings*, pages 38–49, 2013.

[8] Alfredo Cuzzocrea, Massimiliano Nolich, and Walter Ukovich. An innovative architecture for supporting cyber-physical security systems. In *ICCSA*, volume 5, pages 658–667, 2018.

[9] Alfredo Cuzzocrea and Vincenzo Russo. Privacy preserving OLAP and OLAP security. In *Encyclopedia of Data Warehousing and Mining, Second Edition (4 Volumes)*, pages 1575–1581. 2009.

[10] M. P. Fanti, G. Iacobellis, M. Nolich, A. Rusich, and W. Ukovich. A decision support system for cooperative logistics. *IEEE Transactions on Automation Science and Engineering*, 14(2):732–744, April 2017.

[11] H. Hashimoto, T. Ibaraki, S. Imahori, and M. Yagiura. The vehicle routing problem with flexible time windows and traveling times. *Discrete Applied Mathematics*, 154(16):2271–2290, 2006.

[12] H. W. Kuhn. Variants of the hungarian method for assignment problems. *Naval Research Logistics Quarterly*, 3(4):253, 1956.

[13] Massimiliano Nolich, Daniele Spoladore, Sara Carciotti, Raol Buqi, and Marco Sacco. Cabin as a home: A novel comfort optimization framework for iot equipped smart environments and applications on cruise ships. *Sensors*, 19(5), 2019.

[14] D. W. Pentico. Assignment problems: A golden anniversary survey. *European Journal of Operational Research*, 176(2):774–793, 2007.

[15] H. Simao, J. Day, A. George, T. Gifford, J. Nienow, and W. Powell. An approximate dynamic programming algorithm for large-scale fleet management: A case application. *Transportation Science*, 43(2):178–197, 2009.

[16] Daniele Spoladore, Sara Arlati, Sara Carciotti, Massimiliano Nolich, and Marco Sacco. Roomfort: An ontology-based comfort management application for hotels. *Electronics*, 7(12), 2018.

[17] Xiaoqing Sun, Manish Garg, Zahir Balaporia, Kendall Bailey, and Ted Gifford. Optimizing transportation by inventory routing and workload balancing: Optimizing daily dray operations across an intermodal freight network. *INFORMS Journal on Applied Analytics*, 44(6):579–590, 2014.

[18] Z. Xue, C. Zhang, W.-H. Lin, L. Miao, and P. Yang. Container drayage problem with flexible orders and its near real-time solution strategies. *A tabu search heuristic for the local container drayage problem under a new operation mode*, 62:136–150, 2014.

[19] R. Zhang, J.-C. Lu, and D. Wang. Container drayage problem with flexible orders and its near real-time solution strategies. *Transporta- tion Research Part E: Logistics and Transportation Review*, 61:235–251, 2014.

[20] R. Zhang, W. Y. Yun, and H. Kopfer. Heuristic-based truck scheduling for inland container transportation. *OR Spectrum*, 32(3):787–808, 2010.