Computers & Operations Research 77 (2017) 127–140

Contents lists available at ScienceDirect

Computers & Operations Research

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

The pickup and delivery problem with time windows and handling operations

Marjolein Veenstra [a,*], Marilène Cherkesly [b], Guy Desaulniers [c], Gilbert Laporte [d]

[a] University of Groningen, P.O. Box 800, 9700 AV Groningen, The Netherlands
[b] Université du Québec à Montréal and GERAD, C.P. 8888, succursale Centre-ville, Montréal, Canada H3C 3P8
[c] École Polytechnique de Montréal and GERAD, C.P. 6079, succursale Centre-ville, Montréal, Canada H3C 3A7
[d] HEC Montréal and CIRRELT, 3000 chemin de la Côte-Sainte-Catherine, Montréal, Canada H3T 2A7

ARTICLE INFO

Article history:
Received 15 January 2016
Received in revised form
5 July 2016
Accepted 20 July 2016
Available online 30 July 2016

Keywords:
Vehicle routing with pickups and deliveries
Handling operations
Column generation
Branch-price-and-cut
Valid inequalities

ABSTRACT

This paper introduces the pickup and delivery problem with time windows and handling operations. In this problem, the loading compartment of a vehicle is modeled as a linear LIFO stack. When an item is picked up, it is positioned on top of the stack. When it is on top of the stack, it can be delivered without additional handling. Otherwise, items on top must be unloaded before the delivery and reloaded afterwards, which requires time. We define two rehandling policies. For both policies, rehandling is only allowed at delivery locations and there is no specific reloading order for the rehandled items. Under the first policy, only compulsory rehandling is allowed. Under the second policy, in addition to compulsory rehandling, preventive rehandling is allowed. For each policy, we propose a branch-price-and-cut algorithm with an ad hoc dominance criterion for the labeling algorithm used to generate routes. Computational results are reported on benchmark instances for the pickup and delivery problem with time windows.

1. Introduction

In pickup and delivery problems, a fleet of vehicles based at a depot is used to complete a set of requests. A request consists of transporting an item (which can consist of multiple units) from a specific location, where the item is loaded, to a specific location, where it is unloaded. A time window is given for each pickup or delivery location, specifying the time interval during which service must start. We consider a fleet of homogeneous vehicles of limited capacity, where the compartment is rear-loaded and operated in a last-in-first-out (LIFO) fashion. The compartment is modeled as a linear LIFO stack. This implies that when an item is picked up, it is positioned on top of the stack. Therefore, an item is accessible for delivery if it is on top of the stack. Otherwise, the items on top must be unloaded before the delivery of the item and reloaded afterwards, which requires supplementary time. We define a rehandling operation as the unloading and reloading operations of an item at a pickup or delivery location. A handling operation can refer to a rehandling operation, loading an item at its pickup location, or unloading an item at its delivery location. We indicate this problem in the remainder as the pickup and delivery problem with time windows and handling operations (PDPTWH). Let 0, i^+ and i^- denote the depot, and the pickup and delivery locations corresponding to request i, respectively. Fig. 1 illustrates two routes, where route (a) does not require rehandling, whereas in route (b) item 2 needs to be rehandled before delivering item 1.

We introduce and analyze two different rehandling policies. Because an item needs to be delivered at a delivery location, the customer will allow items to be rehandled if its item is not on top of the stack. On the other hand, at a pickup location, because other vehicles from different suppliers may wait to load or unload, the customer might not allow items to be rehandled. Therefore, rehandling operations are only allowed at delivery locations for both policies. We assume that it is not possible to stop at a random location in the route to do the rehandling, which implies that eventual rehandling operations begin at the same time as the service. Therefore, rehandling operations must start within the time window of the delivery location where rehandling occurs. For both policies, there is no specific reloading order for the rehandled items. We define two items i and j to be at the same level if the most recent handling operation for both items occurred at the same location. Item i is said to be on top of item j if the most recent handling operation for item i occurred after the most recent

* Corresponding author.
E-mail addresses: marjolein.veenstra@rug.nl (M. Veenstra), marilene.cherkesly@gerad.ca (M. Cherkesly), guy.desaulniers@gerad.ca (G. Desaulniers), gilbert.laporte@cirrelt.ca (G. Laporte).

http://dx.doi.org/10.1016/j.cor.2016.07.014
0305-0548/

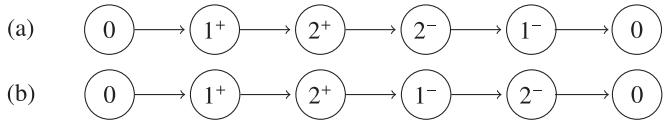

Fig. 1. Route (a) does not require rehandling and route (b) requires one rehandling operation before delivering item 1.

handling operation for item j. Under the first rehandling policy, called policy 1, only compulsory rehandling is allowed, i.e., all and only the items on top of the delivered item must be rehandled. Note that policy 1 forbids the rehandling of items that are on the same level as, or below, the delivered item. The second rehandling policy, called policy 2, is a generalization of policy 1. Under policy 2, compulsory rehandling must be done and preventive rehandling is allowed, i.e., all items can be rehandled at once. Fig. 2 depicts a route and its corresponding vehicle configuration under policy 1, while Fig. 3 depicts the same route and an example of a corresponding vehicle configuration under policy 2. In Fig. 2, two items are rehandled upon delivering item 1, namely items 2 and 3. Item 4 is rehandled upon delivering item 2 and again upon delivering item 3. In Fig. 3, two items are rehandled upon delivering item 1, namely items 2 and 3 and two items are rehandled upon delivering item 2, namely items 3 and 4. Since items 2 and 3 are on the same level when delivering item 2, rehandling item 3 is done preventively. Since preventive rehandling operations are not allowed under policy 1, the vehicle configuration in Fig. 3 is infeasible under policy 1. Because each rehandling operation requires additional time, it may happen that the time windows are respected under policy 2, but not under policy 1. A vehicle route is feasible if (i) the capacity of the vehicle is always respected, (ii) the time windows are respected, (iii) the pickup location of a request is visited before its corresponding delivery location, and (iv) the rehandling policy is respected. We denote by PDPTWH-1 and PDPTWH-2 the PDPTWH under policies 1 and 2, respectively. The goal of the PDPTWH is to compute feasible routes that first minimize the number of vehicles and then the total travel costs.

The PDPTWH arises in the transportation of heavy, dangerous or large items in a less-than-truckload setting. To our knowledge, the PDPTWH has not previously been studied, but several variants of this problem have been investigated, such as the pickup and delivery problem (see Berbeglia et al. [2], Parragh et al. [14,15], and Savelsbergh and Sol [18], for surveys), the pickup and delivery problem with time windows (PDPTW), the pickup and delivery problem with time windows and LIFO loading (PDPTWL), which prohibits rehandling operations, the traveling salesman problem with pickups and deliveries and handling costs (TSPPD-H), where two types of items are considered, those transported from the depot to customers and those transported from customers to the depot, and the pickup and delivery traveling salesman problem with handling costs (PDTSPH), where only compulsory rehandling is allowed and the reloading sequence is given.

Ropke et al. [17] proposed a branch-price-and-cut algorithm for the PDPTW that solves instances with up to 96 requests to optimality within two hours, on a computer equipped with an AMD Opteron 250 processor (2.4 GHz). State-of-the-art algorithms for the PDPTWL were developed by Cherkesly et al. [5,6]. Cherkesly et al. [5] proposed branch-price-and-cut algorithms that can solve instances with up to 75 requests to optimality within one hour, on a computer equipped with an Intel Core i7-3770 processor (3.4 GHz), while Cherkesly et al. [6] developed a population-based metaheuristic that solves instances with up to 300 requests within three hours, on a computer equipped with an Intel (R) Xeon(R) X5675 processor (3.07 GHz). For the instances with known optimal values, the average optimality gap obtained with their algorithm ranges between 0.17% and 0.34%. Battarra et al. [1] proposed two exact algorithms to solve the TSPPD-H under different handling policies: the first one is a branch-and-cut approach, while the second one combines Benders decomposition and branch-and-cut. The tests were run on a computer equipped with an AMD Athlon 64 × 2 Dual processor (2.20 GHz), and instances with up to 25 customers were solved within two hours. Erdoğan et al. [10] developed heuristics for the TSPPD-H that can solve instances with up to 200 customers. The experiments were performed on a computer equipped with an Intel Core 2 Quad processor (2.83 GHz). The combination of tabu search and exact dynamic programming performs best, resulting in an average percentage deviation of 0.07% from the best known solutions. The largest instances were solved in approximately one hour on average. Veenstra et al. [19] proposed a heuristic for the PDTSPH, but the authors did not report optimality gaps for the larger instances.

This work is rooted in two different streams of research, namely, pickup and delivery routing with LIFO loading, and pickup and delivery routing with handling operations. Ropke and Cordeau [16] developed a branch-price-and-cut algorithm for the PDPTW, in which several families of valid inequalities are introduced. Cherkesly et al. [5] developed three branch-price-and-cut algorithms for the PDPTWL. They proposed an ad hoc dominance criterion and a labeling algorithm for the elementary shortest path problem with pickups and deliveries, time windows, capacity, and LIFO constraints. Cherkesly et al. [4] introduced the pickup and delivery problem with multiple stacks (PDPTWMS) and implemented two branch-price-and-cut algorithms. They adapted the hybrid branch-price-and-cut algorithm of Cherkesly et al. [5] for the PDPTWL to the PDPTWMS. Battarra et al. [1] proposed three handling policies for the TSPPD-H. Under the first policy, all items delivered at the depot are positioned on top of the items delivered at the customers, whereas under the second policy all items delivered at customers are positioned on top of the items delivered at the depot. The third policy is a hybrid between the first two. We extend these ideas to develop rehandling policies for our problem where items are transported from specific pickup locations to specific delivery locations. We propose branch-price-and-cut algorithms based on those of Ropke and Cordeau [16] and Cherkesly et al. [4,5].

The goal of this paper is to model the PDPTWH and to develop

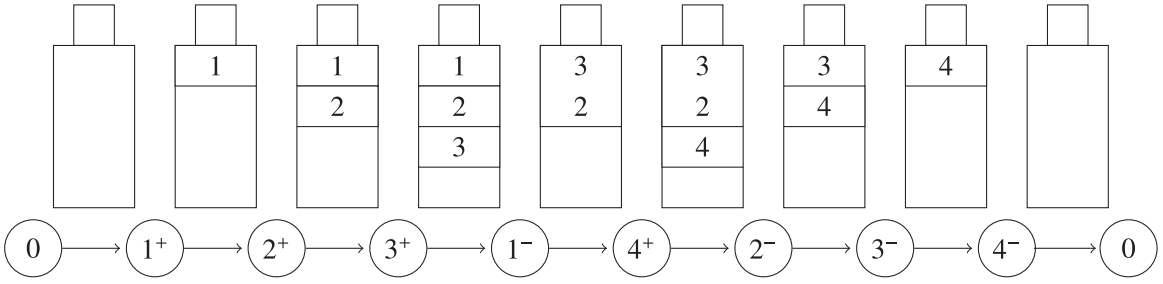

Fig. 2. Example of a route with its corresponding vehicle configuration under policy 1. There is no separation line between items at the same level.