Computers & Industrial Engineering 99 (2016) 16–28

Contents lists available at ScienceDirect

Computers & Industrial Engineering

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

Robust cross-dock scheduling with time windows

Anne-Laure Ladier[a,*], Gülgün Alpan[b,c]

[a] Univ Lyon, INSA Lyon, DISP EA 4570, F-69621 Villeurbanne, France
[b] Univ. Grenoble Alpes, G-SCOP, F-38000 Grenoble, France
[c] CNRS, G-SCOP, F-38000 Grenoble, France

ARTICLE INFO

Article history:
Received 2 February 2016
Received in revised form 1 July 2016
Accepted 2 July 2016
Available online 7 July 2016

Keywords:
Logistics
Robustness
Cross-docking
Cross-dock truck scheduling
Project scheduling

ABSTRACT

Cross-docking is a logistic technique that helps to accelerate the goods flow and to reduce inventory costs; but it requires a perfect coordination of the inbound and outbound trucks. The truck scheduling problem has been studied by many authors, but mainly in a deterministic case. And yet, many uncertainties can arise in the process: if a truck is delayed, or the process times change, does the truck schedule remain feasible and stable? This article proposes robust models for the truck scheduling model with time windows. The reformulations of the original model are based on classical techniques in robust optimization (*minimax* and minimization of the expected regret) but also on techniques from robust project scheduling (resource redundancy and time redundancy). The numerical study carried out to compare the nine different models shows that the methods based on resource redundancy give good results in the cross-docking case. Minimizing the average number of trucks docked at a given door is a good way to ensure robustness in the schedule, but it also increases storage.

1. Introduction

The tremendous growth of e-commerce has made the consumers even more volatile and impatient than before; in a tense economical context with fierce competition, companies must have efficient and fast supply chains. Since it proposes a "just-in-time" approach to logistics, the concept of cross-docking can help coping with this challenge. In a cross-docking platform or cross-dock, pallets are transferred directly from inbound trucks to outbounds trucks with as few intermediate storage as possible. The goods stay typically less than 24 h in the platform.

An overview of a variety of research questions related to cross-docking is given in the exhaustive literature review by Van Belle, Valckenaers, and Cattrysse (2012). Among the operational questions dealing with the internal management of the platform, emerge four different types of problems. Truck-to-door assignment problems seek to assign the trucks present at a given time to the platform doors, usually with the objective of minimizing the distance traveled by the goods. Truck scheduling problems focus on the temporal dimension rather than the spatial one: they seek to schedule the trucks without taking distances into account. Truck-to-door scheduling problems combine the two approaches. Finally, some articles focus solely on the management of the internal operations (pallet routing, storage location).

In this article we focus on the truck scheduling problem. According to the review by Van Belle et al. (2012), several authors propose different models to schedule the trucks in a cross-docking platform: we can mention for example Li, Lim, and Rodrigues (2004), Álvarez Pérez, González-Velarde, and Fowler (2009), Boysen, Fliedner, and Scholl (2010), Boysen (2010), Boloori Arabani, Fatemi Ghomi, and Zandieh (2010, 2011). The interested reader can refer to Van Belle et al. (2012) for a detailed review of these different articles. Posterior to Van Belle et al.'s review, Ladier and Alpan (2014) propose an integer program and three heuristics to solve a cross-dock truck scheduling problem in which the transportation providers express their wishes in advance regarding the time at which they would like to arrive to and leave from the platform. The objective is to schedule the inbound and outbound trucks, as well as the pallet movements, in order to minimize the number of pallets transiting through storage and to maximize the transportation providers' satisfaction regarding the presence time window allocated to their trucks. The two objectives are combined in a weighted sum.

A common point of the truck scheduling articles previously mentioned is that they deal with a deterministic environment, where all data are certain and reliable. This observation led Boysen and Fliedner (2010), in their review of cross-dock truck scheduling problems (which also includes a research agenda listing

---

* Corresponding author.
E-mail addresses: anne-laure.ladier@insa-lyon.fr (A.-L. Ladier), gulgun.alpan@g-scop.fr (G. Alpan).

http://dx.doi.org/10.1016/j.cie.2016.07.003
0360-8352/

the main issues left to be addressed in this area), to note the following:

> "Arrival times of trucks are typically bound to heavy inaccuracies, because traffic congestion or engine failures delay inbound trucks [...]. Thus, the following research questions need to be answered in this context: [...] How to derive robust plans, *i.e.* plans which remain feasible in spite of (shorter) delays?"

This article therefore aims at answering the research question proposed by Boysen and Fliedner (2010). The truck arrival times, which are the decision variables in the truck scheduling problems, are indeed subject to uncertainties. It is also the case of other parameters such as the time needed to unload a pallet, or the time needed to transfer it. Ladier, Alpan, and Greenwood (2014), modeling them using probability distributions, show that truck schedules are very sensitive to these sources of uncertainty. Hence, our goal in this article is to generate a truck schedule that remains feasible and stable when facing these uncertainties. Our article builds upon the models presented in Ladier and Alpan (2014) in order to propose robust formulations for the cross-dock truck scheduling problem. The underlying logic behind each formulation can be applied to different base models as well. The robustness of each formulation is measured with the robustness indicators proposed in Ladier et al. (2014). Robustness has a cost, however – making a cross-docking schedule more robust is likely to increase the stock level. A trade-off between feasibility, stability and stock increase should therefore be found to get a satisfying schedule.

Section 2 defines the concept of robustness and explores related literature; Section 3 reviews the model presented in Ladier and Alpan (2014). In Section 4 we propose different reformulations of the model and explain why they might be more robust than the original version. Using the methodology detailed in Section 5, these assumptions are experimentally tested in Section 6 to compare the performances of the different formulations. Concluding remarks and perspective for future works are given in Section 7.

2. Robustness literature review

In their review of the scheduling and project scheduling literature, Herroelen and Leus (2005) identify different methods used to cope with uncertainty: reactive scheduling, stochastic scheduling, fuzzy scheduling, proactive robust scheduling and sensitivity analysis.[1] We can reuse many of these ideas in order to create robust cross-dock schedules. Section 2.1 therefore reviews several interesting tracks found in the robust scheduling literature, but also in the more specific branch of robust project scheduling. The next subsection reviews the articles dealing with uncertainty in the cross-docking literature.

2.1. Robust scheduling in the literature

In our context of mathematical programming for scheduling, an option to add robustness as a performance measure is to reformulate the objective function in order to capture the robustness idea. This can be done in many different ways, reviewed by Sabuncuoglu and Goren (2009) in their state-of-the-art focusing on robustness and stability in a manufacturing environment. They propose an organized list of different objective functions used to ensure stability and robustness. Based on their work and after adding other measures proposed in more recent papers, we can list (not exhaustively) the main possible objective functions for robust scheduling.

Objective functions based on realized performance.
The idea is to ensure that the performance level achieved by the schedule remains high when facing a disruption. For a minimization problem, this can be done by (1) minimizing the expected realized performance, (2) minimizing the worst-case performance (*minimax method*: the worst-case performance is the max of the performances obtained for all the scenarios considered; this criteria is called *absolute robustness* by Kouvelis & Yu (1997)), (3) minimizing the performance of the schedule in the most probable scenario, (4) minimizing the expected deviation of the realized schedule's performance from the initial deterministic performance, and (5) minimizing the variance of realized performance measure, etc.

Objective functions based on regret.
We call regret the difference between the realized and the optimal performance, *i.e.* the performance that would have been realized if the disruptions were known in advance and used as data. The idea is to ensure that the performance level achieved is close to what it would have been with a full information. It is usually done by minimizing the expected regret, or minimizing the regret in the worst case (*minimax regret method*; this criteria is called *absolute deviation* or *relative deviation* by Kouvelis & Yu (1997)).

Objective functions based on slacks.
These measures are proposed by Hazir, Haouari, and Erel (2010) in the context of robust project scheduling. They are based on the slack of some project tasks, *i.e.* the amount of delay that a task can take without delaying the completion time of the total project. A slack is therefore a buffer time that can protect a specific task against delay or disruptions, when placed right after the task in a Gantt chart. Using a simulation experiment, Hazir et al. (2010) show that two performance measures have a high correlation with indicators on the project punctuality: the maximum weighted slack (where the weight of a slack is the number of immediate successors, in the Gantt chart, of the task protected by the slack, or its total number of successors), and the maximum ratio between the total project buffer size and the total project completion time.

Objective functions based on realized performance and based on regret are not specific to robust scheduling, and are largely used in robust optimization in general. The interested reader can refer, for example, to Nikulin (2006) for an extended annotated bibliography of robustness in combinatorial optimization and scheduling theory, or to Gabrel, Murat, and Thièle (2013) for a more recent review of the literature regarding robust optimization.

Slack-based measures, on the contrary, are very specific to project scheduling. They follow the idea emphasized by Davenport and Beck (2000), who show that redundancy-based techniques are a way to proactively ensure the robustness of a schedule. For slack-based indicators, the redundancy is applied on *time*, since the idea is to keep reserve time or buffer time periods. Davenport and Beck (2000) note that *resource* redundancy (keeping some resources in standby) is another way to ensure robustness in scheduling. However, resource redundancy is not usually used in project management, since keeping idle resources is expensive.

Time redundancy is much more frequently used in project scheduling. It was originally proposed in 1990 by Chiang and Fox (1990) (and later Gao (1996)) who developed the concept of temporal protection. The "protected" duration of each activity equals its original duration augmented with the duration of breakdowns

---

[1] Sensitivity analysis checks the effect of parameter changes, it is thus quite far from the aim of this study.