Bugow, Stefan; Kellenbrink, Carolin

The parcel hub scheduling problem with limited conveyor capacity and controllable unloading speeds

OR Spectrum

Provided in Cooperation with:
Springer Nature

Suggested Citation: Bugow, Stefan; Kellenbrink, Carolin (2023) : The parcel hub scheduling problem with limited conveyor capacity and controllable unloading speeds, OR Spectrum, ISSN 1436-6304, Springer, Berlin, Heidelberg, Vol. 45, Iss. 2, pp. 325-357, https://doi.org/10.1007/s00291-022-00702-y

OR Spectrum (2023) 45:325–357
https://doi.org/10.1007/s00291-022-00702-y

ORIGINAL ARTICLE

The parcel hub scheduling problem with limited conveyor capacity and controllable unloading speeds

Stefan Bugow[1] · **Carolin Kellenbrink**[1]

Received: 25 November 2021 / Accepted: 21 November 2022 / Published online: 9 January 2023

Abstract

We investigate a specific truck scheduling problem at cross-docks in the postal service industry on an operational level aiming to maximise the number of duly parcels assuming fixed departure times of the outbound trucks. The inbound gates and the conveyors as means of transportation inside the hub constitute the bottleneck resources. As a novel extension, we propose flexible unloading speeds to efficiently utilise the scarce resources. We formalise the problem with a mixed integer program and explicitly incorporate controllable unloading speeds of the inbound trucks. We determine the computational complexity and develop a genetic algorithm to efficiently solve the problem. Our investigation focuses on both the performance of the genetic algorithm and the applicability of the results in a real-world environment by implementing scheduling policies in a simulation model that considers individual parcel interactions. Based on our experimental results, we can state that especially in problem settings with scarce conveyor capacities, our approach to incorporate controllable unloading speeds has the potential of significantly increasing the number of duly parcels.

Keywords Truck scheduling · Parcel hubs · Simulation · MIP · Genetic algorithms · Scheduling policies

1 Introduction

1.1 Decision problems at parcel hubs

In the postal service industry, parcel hubs are widely used to consolidate fright on an intermediate stage of the distribution network. By consolidating parcels, the

✉ Carolin Kellenbrink
carolin.kellenbrink@alumni-uni-hannover.de

Stefan Bugow
stefan.bugow@prod.uni-hannover.de

[1] Institute of Production Management, Leibniz Universität Hannover, Königsworther Platz 1, 30167 Hanover, Lower Saxony, Germany

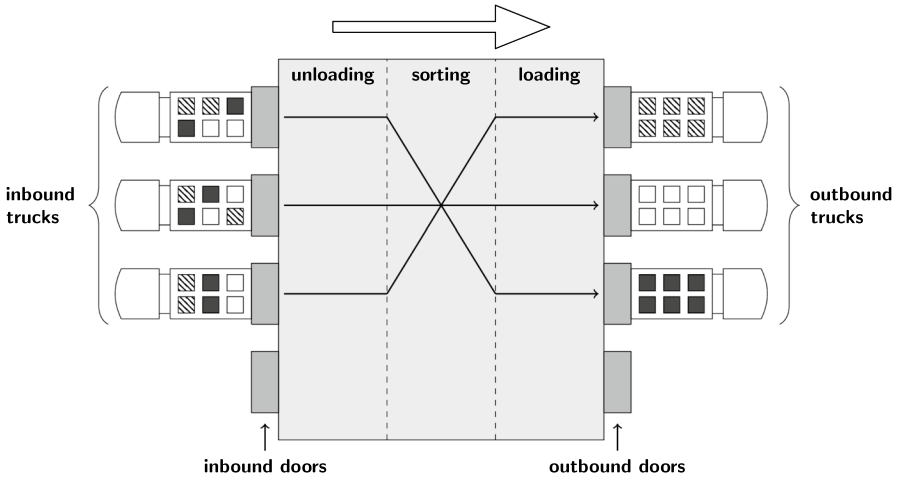


Fig. 1 Schematic layout of a cross-docking terminal

utilisation of trucks is increased and unnecessary stocks can be avoided, cf. Apte and Viswanathan (2000). Parcel hubs are often designed as cross-docking terminals where parcels are unloaded from inbound trucks, sorted according to their destination, transferred and finally loaded onto waiting outbound trucks. Figure 1 illustrates the basic transshipment process in a cross-docking terminal.

In the context of planning the design and operation of cross-docking facilities, a variety of decision problems arise. We will give a rough summary, but refer to Boysen and Fliedner (2010) for an overall overview. On a strategic level, decisions on the location and layout of the hub have to be made. Decisions on the assignment of destinations to outbound docks are relevant on a tactical level. On an operational level, mainly the inner transport scheduling and particularly inbound truck scheduling are key areas to optimise the efficiency of a cross-docking facility, cf. Stephan and Boysen (2011). Our paper focuses on the latter problem. Equivalent to other scheduling problems, the truck scheduling problem at cross-docking hubs deals with allocating tasks of a process to scarce resources over a given time horizon. In this case, the inbound gates as well as the means of transportation inside the hub are those bottleneck resources, see Boysen and Fliedner (2010). This leads to the question of how to efficiently schedule incoming trucks in the specific context of parcel hubs.

1.2 Motivation for the consideration of conveyor capacities and controllable unloading speeds

We consider a parcel hub as shown in Fig. 2. Each outbound door is connected to a single conveyor. At the outbound doors, outbound trucks with a predefined destination and an individual deadline wait to be loaded. Each inbound gate is connected to all conveyors. Since each inbound truck typically contains parcels for a variety of

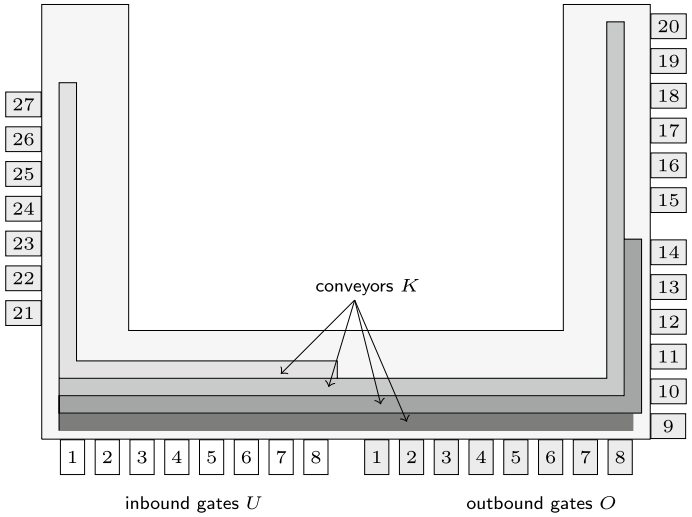


Fig. 2 Exemplary conveyor layout in a parcel hub

destinations, the decision which inbound trucks are unloaded at the same time has significant implications on the workload of the conveyors. If for example several inbound trucks with numerous parcels for the outbound trucks connected to the same conveyor are unloaded at the same time, an overloaded conveyor could slow down the process.

The detailed and general assumptions of our approach are described in Chapter 2. However, the impact of the conveyor capacities on the processes in a parcel hub can be demonstrated using a small illustrative example. We consider a parcel hub with four inbound trucks $i1$–$i4$ and two inbound gates $u1$ and $u2$. They are unloaded with a speed of 10 parcels per minute. Parcels coming from the inbound gates are transferred on two conveyors $k1$ and $k2$ with a capacity of 11 parcels per minute each. Each conveyor is attached to a single outbound gate, at which a single outbound truck is docked. We consider the outbound trucks $o1$ and $o2$ that wait for incoming parcels until their deadline $dl_{o1} = 15$ and $dl_{o2} = 20$. Generally, the objective is to maximise the number of parcels arriving before their deadline. The shipments between the trucks are shown in Table 1.

Under consideration of the limited conveyor capacities, the resulting truck schedule and conveyor utilisation is shown in Fig. 3a, b. Inbound trucks $i1$ and $i3$ are unloaded at the beginning of the planning horizon. We can observe that inbound trucks $i2$ and $i4$ can not be processed concurrently as the resulting workload would violate the limited capacity of conveyor $k1$ of 11 parcels per minute. Since inbound truck $i2$ is scheduled after inbound truck $i4$ in this case, all parcels unloaded from inbound truck $i2$ are tardy. This results in only 275 parcels without delay out of a total number of 400 parcels. This shows that the sole consideration of the conveyor capacity leads to feasible, but not very reasonable results.

Therefore, in parcel hubs, the system flow management may decide to slow down the unloading speed of individual inbound trucks to control the workload on conveyors in such situations, cf. McWilliams et al. (2005). The motivation and contribution

Table 1 Shipment from inbound truck i to outbound truck o

	$o1$	$o2$
$i1$	100	0
$i2$	70	30
$i3$	10	90
$i4$	50	50

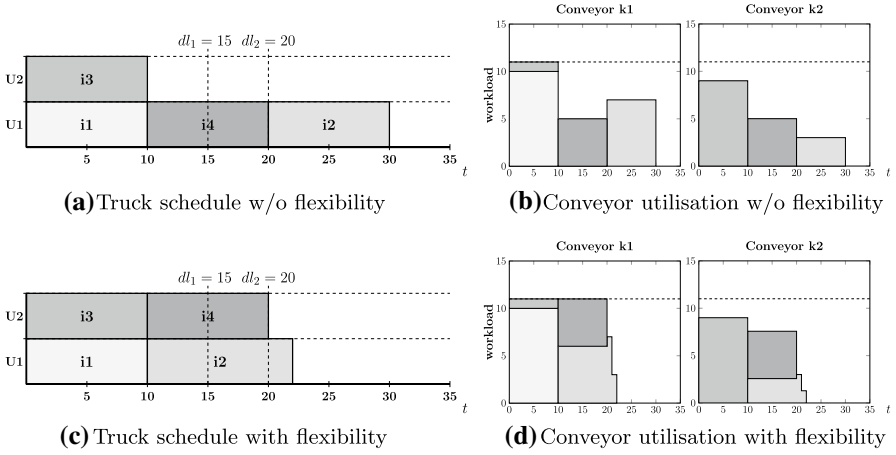


Fig. 3 Comparison of different exemplary truck schedules

of our paper are to allow such a reduction of the unloading speed of the inbound trucks. If we allow a reduction of the unloading speed, the truck schedule and conveyor utilisation in Fig. 3c, d results. Inbound trucks $i2$ and $i4$ are unloaded concurrently as the unloading speed of inbound truck $i2$ is decreased to 6 parcels per minute. Even though the unloading process of truck $i2$ takes longer now, the total number of duly parcels increases significantly to 330 parcels as at least some of the parcels in inbound truck $i2$ are processed before the respective deadlines dl_{o1} and dl_{o2}. After inbound truck $i4$ is fully unloaded, the unloading speed of truck $i2$ is increased for one period, and in the following last period, all remaining parcels are unloaded. The example shows that incorporating controllable unloading speeds of the inbound trucks seems to be a promising approach to efficiently utilise conveyor capacities and increase the number of non-delayed parcels. Thus, we seek to investigate the potentials of controlling unloading speeds in this paper in detail.

1.3 Outline of the paper

The remainder of the paper is structured as follows: In Sect. 2, we present the problem setting and provide a literature review. In Sect. 3, we describe the assumptions and model formulation of the Parcel Hub Scheduling Problem with Limited

Conveyor Capacities and controllable unloading speeds (PHSP-LCC-flex) and discuss the complexity of the problem. In Sect. 4, we introduce a biased random-key genetic algorithm to efficiently solve the problem. We evaluate both the application of the genetic algorithm as well as of the standard solver Gurobi for a set of test instances as part of a performance analysis in Sect. 5. To evaluate how the results of the mathematical model can be applied in a real-world environment, we present a discrete-event simulation model in Chapter 6. We define different scheduling policies that use the results of the mathematical model and test their performance on the instances generated in the previous chapter. The paper ends in Sect. 7 with a short summary of the results and an outlook.

2 Truck scheduling with limited conveyor capacity and deadlines

2.1 Problem setting

We investigate a specific truck scheduling problem at cross-docks in the postal service industry. McWilliams et al. (2005) state that parcel hubs can process up to 500.000 parcels each day. With regards to Germany, larger hubs such as the distribution centre in Obertshausen operated by DHL process up to 50.000 parcels per hour, cf. Tripp (2021).

As in other less-than-truckload logistic networks, in the postal service industry, the transported goods are mostly small and have comparatively low value, cf. Jarrah et al. (2009). The exact composition of the set of parcels in the inbound trucks is only roughly known. Under these circumstances, the lack of information would lead to frequent delays if all outbound trucks would have to wait until all parcels of all inbound trucks are unloaded. Accordingly, fixed outbound schedules are applied to ensure a steady flow of trucks in the distribution network. Otherwise, a truck would not arrive in time for the (un)loading process at the next step of the supply chain. Therefore, outbound trucks leave the facility at predetermined deadlines even if they are not fully loaded. Tardy parcels have to either be temporarily stored until the next dispatch period or day, possibly causing a delayed delivery to the customer, or are redirected to costly additional vehicles, e.g. faster vans, cf. Boysen et al. (2013). As both types of cost associated with tardy parcels are not clearly quantifiable, we aim to maximise the number of duly parcels.

The large quantities of parcels require automated or semi-automated conveyor systems to enable an efficient sortation process inside of the hub. In an aggregated view, we consider a parcel hub in so-called line configuration where each conveyor connects all inbound doors with a subset of outbound doors, cf. Haneyah et al. (2014). Of course, the outbound organisation is highly relevant on a tactical level but not part of operational planning. Therefore, the allocation of outbound trucks to outbound doors is assumed to be given. Oftentimes available freight trailers or containers wait at the outbound doors without an actual truck and are only connected with the truck once they are fully loaded. Thus, parcels can always be loaded at the outbound doors and the loading process at the outbound side does not constitute the

bottleneck of the system. Accordingly, each parcel has a known path through the hub as soon as the inbound truck is assigned to a door.

The daily operations at parcel hubs are usually organised into several shifts, cf. Khir et al. (2021). The process of unloading the parcels in each shift starts with the assignment of the first truck to a dock door. Once any inbound truck has been assigned to a dock door, the parcels start leaving the truck in a random sequence. With a given unloading speed, the parcels are unloaded and transferred on the conveyor one after another. If at any time during the process too many parcels are currently on one conveyor, the unloading process for parcels designated for that blocked conveyor is halted until the next parcel leaves the conveyor. After unloading the last parcel of an inbound truck, a new inbound truck is assigned to the dock according to the chosen assignment of trucks to the inbound doors.

A fully loaded outbound truck is replaced with an empty one if the deadline of the last outbound truck is not exceeded yet. As parcels can be intermediately stored at an outbound door to a certain extent, this truck exchange does not influence the unloading process. For this reason, we do not have to take it into account. As the transfer times of parcels through the hub are very short compared to the overall duration of the unloading process and do not differ significantly between pairs of inbound and outbound gates, we neglect them in our approach.

2.2 Related literature for truck scheduling with limited conveyor capacity and deadlines

The truck scheduling problem at cross-docking hubs as an operational planning problem has been extensively studied. For different supply chains, the assumptions for the truck scheduling problem concerning the objective and operational features such as the type of dock doors, mode of internal transportation and storage, inbound and outbound organisation or interchangeability of products may differ greatly. For a detailed classification and review of the problem, we refer to Boysen and Fliedner (2010). For further extensive literature reviews of the field of cross-dock scheduling refer to Van Belle et al. (2012), Ladier and Alpan (2016) and Theophilus et al. (2019).

In this paper, we consider a truck scheduling problem taken from the postal service industry. A key component of the setting is fixed outbound departures mandated by predefined delivery schedules. Similar problem settings arise in supply chains with air cargo terminals where fixed outbound departures defined by flight schedules are relevant, cf. Ou et al. (2010) and Selinka et al. (2016). Tootkaleh et al. (2016) assume fixed deadlines in a setting in which product substitution is possible. In the context of the postal service industry, fixed outbound departures have been considered by Boysen et al. (2013) who seek to minimise contract penalties due to delayed shipments by scheduling inbound trucks.

As the parcels are not interchangeable but individual and should be delivered fast, parcel hub operators seek to transfer shipments directly without temporary storage. Related circumstances are prevalent in the food retail industry where goods are

mostly perishable and thus cannot be stored temporarily in the terminal, cf. Qijun et al. (2009) and Boysen (2010).

Another important aspect of the problem setting is conveyors as means of transportation inside the hub. The means of transportation inside the hub can be seen as a scarce resource. McWilliams et al. (2005) incorporate the effects of congestion inside a parcel hub with conveyors that directly connect inbound and outbound docks into the scheduling problem. They simulate the parcel flows resulting from a given truck schedule and use the information to minimise the time span of the transfer operation. Mathematical approximations for the transfer times are investigated in McWilliams and McBride (2013). Similarly, Clausen et al. (2017) use a combined simulation and optimisation approach where they utilise an optimisation model to balance the workload in the terminal. Another frequently used type of parcel hub is terminals with closed loop sortation systems where parcels are transported on cyclic conveyor belts. Truck scheduling in these systems is investigated by Boysen et al. (2017) and Molavi et al. (2018).

Further research considering limited handling capacities inside the hub irrespective of the employed means of transportation can be found in Li et al. (2004), Carrera et al. (2008) and Serrano et al. (2017). As previously mentioned, for cross-dock terminals with conveyors as the means of transportation for transferring shipments, McWilliams et al. (2005) describe that terminal operators manage the flow of parcels by controlling the unloading speeds of inbound trucks. However, they do not include the decision on unloading speeds in their optimisation approach. In the general context of truck scheduling at cross-docking terminals, Tadumadze et al. (2019) have explored flexibility in unloading speeds by incorporating operational personnel planning into the truck scheduling problem. Similarly, Corsten et al. (2020) combine truck scheduling with tactical shift planning.

The problem setting of truck scheduling including flexibility in the unloading process bears some similarities with the resource-constrained project scheduling problem with flexible resource profiles (FRCPSP) as described by Naber and Kolisch (2014). Another similar problem setting can be found in the field of machine scheduling with controllable processing times, cf. Shabtay and Steiner (2007) and Cheng et al. (1996). In both cases, the authors assume that the resource usage of a job can be altered by shortening or expanding the duration of a job just as we assume to be able to shorten and lengthen the unloading process of the inbound trucks. The main differences of both approaches to the truck scheduling problem are that the resource consumption for each job is assumed to be constant whereas we assume the resource consumption to be varying with time and controllable.

Another field of research with similarities to our problem setting is the energy-constrained scheduling problem (ECSP) with flexible energy demands as investigated for example by Artigues et al. (2013) and Nattaf et al. (2016). In the problem setting, tasks compete for a limited renewable resource and are finished once they receive a sufficient amount of energy. Thus, the processing time of the tasks is flexible as in our case. However, the characteristics of parcel transport are not covered by these approaches.

To our best knowledge, the problem setting of scheduling inbound trucks at parcel hubs with fixed outbound departures considering conveyor capacities and

explicitly modelling flexible unloading speeds has not been explored yet. Therefore, we seek to investigate the effects of controllable unloading speeds in the context of parcel hub truck scheduling.

3 Truck scheduling at parcel hubs with limited conveyor capacity and flexible unloading speeds (PHSP-LCC-flex)

3.1 Model assumptions

In the following, the detailed assumptions of our approach to model the problem setting introduced in Sect. 2.1 are explained. The notation of the **P**arcel **H**ub **S**cheduling **P**roblem with **L**imited **C**onveyor **C**apacity and **Flex**ible Unloading Speeds (PHSP-LCC-flex) is summarised in Table 2.

We follow a time-discrete modelling approach using periods $t \in \mathcal{T}$ with $\mathcal{T} = \{1, \dots, T\}$. An inbound truck $i \in \mathcal{I}$ with $\mathcal{I} = \{1, \dots, I\}$ arrives at the beginning of period at_i. Therefore, this truck can be unloaded in periods $t \in \mathcal{T}_i$ with $\mathcal{T}_i = \{at_i, \dots, T\}$. We define $i \in \mathcal{I}_t \subset \mathcal{I}$ as the set of inbound trucks i that are available in period t with $\mathcal{I}_t = \{i | t \geq at_i\}$.

The cross-docking terminal has U inbound doors. Since we assume negligible transfer times for the parcels through the hub, the decision on which specific door an inbound is assigned to has no effect. Thus, the doors are not modelled separately as it is irrelevant which inbound door is selected for unloading with regards to the duration of the overall process.

Each outbound truck $o \in \mathcal{O}$ with $\mathcal{O} = \{1, \dots, O\}$ is uniquely assigned to one outbound door. Therefore, we do not distinguish between outbound doors and outbound trucks in our approach. We do not model each single parcel as the distribution of parcels inside an inbound truck is not known and the formulation would become mathematically intractable. Instead, we use the assumption that a number of parcels from inbound truck i for outbound truck o $ship_{io}$ is determined and imply a homogeneous distribution inside the truck.

Conveyor $k \in \mathcal{K}$ with $\mathcal{K} = \{1, \dots, K\}$ has a capacity of r_k parcels per period. Each inbound door can access each conveyor. In contrast, each outbound door is assigned to only one conveyor, as exemplarily shown in Fig. 2. We introduce the subset $\mathcal{O}_k \subseteq \mathcal{O}$ containing all outbound doors that are connected to conveyor k.

We define the maximal unloading speed of parcels per period at a single door as x^{max}. The minimal unloading speed is denoted as x^{min}. The objective function of our model aims at maximising the weighted number of duly parcels with a weight w_i for parcels from inbound truck i. The weighting factor w_i for each inbound i factor represents the priority of inbound truck i and can be used for example in case delays for shipments of a certain customer are deemed more severe. For the remainder of the paper, all inbound trucks have the same priority of 1. A deadline in period dl_o represents an outbound truck o leaving at the beginning of this period, meaning that it can be loaded until period $dl_o - 1$. The time periods before the deadline of truck o are defined as $t \in \mathcal{T}_o$.

Table 2 Notation of the PHSP-LCC-flex

Indices and (ordered) sets	
$i \in \mathcal{I}$	Inbound trucks $\mathcal{I} = \{1, \dots, I\}$
$i \in \mathcal{I}_t \subseteq \mathcal{I}$	Inbound trucks available in period t with $\mathcal{I}_t = \{i \mid t \geq at_i\}$
$k \in \mathcal{K}$	Conveyor belts $\mathcal{K} = \{1, \dots, K\}$
$o, \rho \in \mathcal{O}$	Outbound trucks $\mathcal{O} = \{1, \dots, O\}$
$o \in \mathcal{O}_k \subseteq \mathcal{O}$	Outbound trucks designated to conveyor k
$t, \tau \in \mathcal{T}$	Periods $\mathcal{T} = \{1, \dots, T\}$
$t \in T_i \subseteq \mathcal{T}$	Available periods for inbound truck i with $\mathcal{T}_i = \{at_i, \dots, T\}$
$t \in T_o \subseteq \mathcal{T}$	Available periods for outbound truck o
Parameters	
at_i	Arrival time of inbound truck i
dl_o	Deadline of outbound truck o
r_k	Capacity of conveyor k in parcels per period
$ship_{io}$	Number of parcels for outbound truck o in inbound truck i
U	Number of inbound doors
w_i	Weighting factor for inbound i
x^{max}	Maximal unloading speed
x^{min}	Minimal unloading speed
Decision variables	
$x_{it} \geq 0$	Number of parcels that are unloaded in period t from inbound truck i
z_{it}	$= \begin{cases} 1, & \text{if inbound truck } i \text{ is located at a door in period } t \\ 0, & \text{otherwise} \end{cases}$

The model is used to build an optimal inbound schedule, i.e., to answer the question whether an inbound truck i is located at any dock door at the beginning of period t or not, represented by the binary decision variable z_{it}. The continuous variable $x_{it} \geq 0$ states the number of parcels that are unloaded in period t from inbound truck i and thus reflects the unloading speed. Assuming a continuous decision variable rather than an integer variable for the number of unloaded parcels is a slight simplification of the problem as parcels are naturally not dividable. However, as we deal with a large number of parcels for most applications, the deviations are negligible.

In order to reduce the time horizon used in the model, we assume the period after the last deadline, i.e. $T = \max_{o \in \mathcal{O}}(dl_o)$, to be a dummy period. In the last (dummy) period, the capacity restrictions of the conveyors and of the doors are relaxed. Thus, all inbound trucks that are not (fully) unloaded until the deadline are scheduled to be unloaded in this period. Of course, the resulting schedule is most likely not directly applicable to a real-world-setting. However, with regard to the objective function value, all parcels unloaded in or after the last deadline are tardy, no matter how late they are unloaded. This aspect does not have an influence on the realisability of the plan. In a post-processing step, we can always allocate the inbound truck to specific doors and arbitrarily schedule the remaining inbound trucks in the periods after the last deadline

to generate a viable plan. This entails the basic assumption that the time window until the following dispatching period is sufficiently large to process all trucks scheduled in the dummy period.

According to the classification scheme of truck scheduling problems at cross-docking hubs by Boysen and Fliedner (2010), we can describe the problem as $[E|a_j, t_j = 0, \tilde{d}_o, \text{fix}, \bar{r_k}, p_i = \text{var}|\sum U_o]$ where $p_i = var$ refers to controllable unloading speeds. We add $\bar{r_k}$ to signify limited conveyor capacities.

3.2 Mathematical model

We now describe the model the **P**arcel **H**ub **S**cheduling **P**roblem with **L**imited **C**onveyor **C**apacity and **Flex**ible Unloading Speeds (PHSP-LCC-flex) formally. Afterwards, the verbal explanation of the equations follows.

Model PHSP-LCC-flex

$$\max\ Z = \sum_{o\in\mathcal{O}} \sum_{t\in\mathcal{T}_o} \sum_{i\in\mathcal{I}_t} x_{it} \cdot w_i \cdot \frac{ship_{io}}{\sum_{\rho\in\mathcal{O}} ship_{i\rho}} \tag{1}$$

subject to

$$\sum_{i\in\mathcal{I}_t} z_{it} \leq U \quad \forall t \in \mathcal{T}\backslash\{T\} \tag{2}$$

$$\sum_{i\in\mathcal{I}_t} \left(x_{it} \cdot \frac{\sum_{o\in\mathcal{O}_k} ship_{io}}{\sum_{o\in\mathcal{O}} ship_{io}} \right) \leq r_k \quad \forall k \in \mathcal{K}, \forall t \in \mathcal{T}\backslash\{T\} \tag{3}$$

$$\sum_{t\in\mathcal{T}_i} x_{it} = \sum_{o\in\mathcal{O}} ship_{io} \quad \forall i \in \mathcal{I} \tag{4}$$

$$x_{it} \leq x^{max} \cdot z_{it} \quad \forall i \in \mathcal{I}, \forall t \in \mathcal{T}_i\backslash\{T\} \tag{5}$$

$$x_{it} \geq x^{min} \cdot z_{it} \quad \forall i \in \mathcal{I}, \forall t \in \mathcal{T}_i \tag{6}$$

$$\sum_{\tau=at_i}^{t} z_{i\tau} \leq t \cdot (1 + z_{it} - z_{it+1}) \quad \forall i \in \mathcal{I}, \forall t \in \mathcal{T}_i\backslash\{at_i, T\} \tag{7}$$

$$\sum_{\tau=t}^{T} z_{i\tau} \leq (T - t + 1) \cdot (1 + z_{it} - z_{it-1}) \quad \forall i \in \mathcal{I}, \forall t \in \mathcal{T}_i\backslash\{at_i, T\} \tag{8}$$

$$z_{it} \in \{0,1\} \quad \forall i \in \mathcal{I}, \forall t \in \mathcal{T}_i \tag{9}$$

$$x_{it} \geq 0 \quad \forall i \in \mathcal{I}, \forall t \in \mathcal{T}_i \tag{10}$$

The Objective Function (1) assures the maximisation of the duly parcels. The value of $\frac{ship_{io}}{\sum_{\rho \in \mathcal{O}} ship_{i\rho}}$ specifies the fraction of parcels from inbound truck i that are designated for outbound truck o.

The limitations due to the doors are represented by Restrictions (2). Constraints (3) model the conveyor capacity. Both limitations are excluded for the last (dummy) period T.

Constrains (4) assure that over all periods between the arrival at_i of the inbound truck i and the dummy period T all parcels of each inbound truck i are unloaded. Parcels can only be unloaded from an inbound truck i if this truck is located at a door in the respective period t ($z_{it} = 1$), see Restrictions (5). At the same time, these restrictions limit the unloading of parcels to a maximum speed x^{max} for each inbound truck and each period except the dummy period. The minimal speed is addressed in Equations (6).

Constraints (7) restrict $z_{i\tau}$ to 0 for all *previous* periods $\tau \leq t$ if the truck i was newly located at a door in period $t+1$ ($z_{it} = 0$ and $z_{i,t+1} = 1$). Analogously, Constraints (8) force $z_{i\tau}$ to 0 for *subsequent* periods $\tau \geq t$ if the truck was removed from the door in the previous period $t-1$ ($z_{i,t-1} = 1$ and $z_{it} = 0$). In combination, the constraints define an uninterrupted presence of the inbound truck at a door. Note that there may be periods in which the unloading of the truck is halted ($x_{it} = 0$) even if it is located at the door. The formulation bears similarities to the on/off event-based formulation of the resource-constrained project scheduling problem, cf. Koné et al. (2011). Constraints (9) and (10) define the variable types.

Please note that when using $x^{min} = x^{max}$, the unloading speeds of all trucks are fixed to the given value. In this case, we receive a model that does not allow any flexibility in the unloading process.

3.3 Computational complexity of the problem

To determine the $\mathcal{NP}$-hardness of the PHSP-LCC-flex, we can refer to the problem of scheduling jobs on a single machine minimising the total tardiness of all jobs which was proven to be $\mathcal{NP}$-hard by Du and Leung (1990). A similar approach can be found in Boysen et al. (2013) for a truck scheduling problem at cross-docks with fixed departure times. The single machine job scheduling problem minimising total tardiness is defined as follows:

For a given set of jobs $j \in J$ with processing times p'_j and due dates d'_j, find a schedule that minimises the objective function

$$\sum_{j \in \mathcal{J}} \max\{0, C_j - d'_j\} \tag{11}$$

where C_j is the completion time of job j.

According to the Graham Notation found in Graham et al. (1979), the problem is represented by the tuple $1||\sum T_j$. Representing an instance of $1||\sum T_j$ as an instance of the PHSP-LCC-flex first involves assuming a hub with only a single gate reflecting the single machine and thus $U = 1$. Each inbound truck i represents a job j. We set the maximum and minimum unloading speed to the standard unloading speed $x^{max} = x^{min} = 1$. This assumption leads to the situation that each inbound truck that is located at a door in a period is unloaded with a speed of exactly 1 parcel per minute and thus to $x_{jt} = z_{jt}$. Additionally, this can be easily shown mathematically by inserting the values in Restrictions (5) and (6).

The single inbound door in combination with an unloading speed of 1 parcel per period assures that Restriction (3) is never binding for any $r_k \geq 1$ and can be omitted as Restriction (2) is stricter in any case.

We assume that each job and inbound truck, respectively, are available in the first period, i.e., $at_i = 0$ and $\mathcal{T}_i = \mathcal{T}$ as well as $\mathcal{I}_t = \mathcal{I}$. We define $ship_{jo}$ as

$$ship_{jo} = \frac{\max\left\{0, \left\lceil \frac{o - d'_j}{p'_j} \right\rceil\right\}}{\sum_{\rho} \max\left\{0, \left\lceil \frac{\rho - d'_j}{p'_j} \right\rceil\right\}} \cdot p'_j. \tag{12}$$

Together with the definition of $ship_{jo}$, the equation $\sum_{o \in \mathcal{O}} ship_{jo} = p'_j$ holds. Therefore, Restriction (4) assures that each inbound truck and job respectively is scheduled for exactly p'_j time units. For Restrictions (7) and (8), there are no adjustments necessary. With this interpretation of the PHSP-LCC-flex, all restrictions of the single machine problem are considered.

Concerning the objective function, for each period t' of the machine scheduling problem, an outbound truck o is created that is available in exactly this period, i.e., $o = t'$ and $\mathcal{T}_o = \{t'\}$.

We assume

$$w_j = -\sum_{t'} \max\left\{0, \left\lceil \frac{t' - d'_j}{p'_j} \right\rceil\right\}. \tag{13}$$

If we then minimise the objective function value given in Eq. (1) multiplied with minus 1 instead of maximising the original value, we get:

$$min\ Z = \sum_{t' \in \mathcal{T}'} \sum_{j \in \mathcal{J}} z_{jt'} \cdot \max \left\{ 0, \left\lceil \frac{t' - d'_j}{p'_j} \right\rceil \right\} \tag{14}$$

To assure that Objective Function (14) equals Objective Function (11), the following equation has to hold:

$$\sum_{t' \in \mathcal{T}'} z_{jt'} \cdot \max \left\{ 0, \left\lceil \frac{t' - d'_j}{p'_j} \right\rceil \right\} = \max\{0, C_j - d'_j\} \tag{15}$$

As $z_{it'}$ equals 1 exactly in periods $C_j - p'_j + 1$ to C_j, this can be written as

$$\sum_{t'=C_j-p'_j+1}^{C_j} \max \left\{ 0, \left\lceil \frac{t' - d'_j}{p'_j} \right\rceil \right\} = \max\{0, C_j - d'_j\}, \tag{16}$$

which is always true.

With the stated reduction, any instance of $1||\sum T_j$ can be represented as an instance of the PHSP-LCC-flex as stated above. As $1||\sum T_j$ is proven to be strongly $\mathcal{NP}$-hard, the PHSP-LCC-flex is also strongly $\mathcal{NP}$-hard as well.

4 Heuristic for the PHSP-LCC-flex

4.1 Genetic algorithm

As the truck scheduling problem at parcel hubs constitutes an operational planning problem, finding a good solution in a short time frame is often of high importance. As heuristic approaches are able to generate adequate solutions comparably fast, they enjoy great popularity in literature, cf. Theophilus et al. (2019). Especially genetic algorithms are frequently used. Thus, we develop a genetic algorithm to solve the PHSP-LCC-flex heuristically.

Genetic algorithms are metaheuristics inspired by the principles of natural selection and were first introduced by Holland (1975). Genetic algorithms mimic the process of natural selection by using a population of individuals (solutions) and subjecting them to variations through genetic operators, such as recombination and mutation, to generate individuals for subsequent populations over several generations (iterations). In each generation, a number of individuals is discarded through the selection operator. As solutions of higher quality are selected for the following

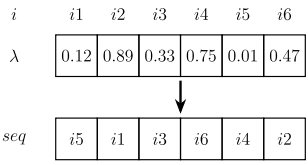


Fig. 4 Exemplary sequence in random-key representation

generation, the population drifts towards promising sections of the solution space. The algorithm avoids local optima through this usage of a population of solutions, cf. Rothlauf (2011).

4.2 Solution representation

The performance of a genetic algorithm is highly dependent on an adequate representation of the solutions of the problem setting. For the PHSP-LCC-flex, we employ a random-key representation that encodes solutions as a vector λ of $|I|$ real values taken from the range between 0 (reflecting a high priority) and 1 (reflecting a low priority) as exemplarily shown in Fig. 4. The random keys are first used to create a sequence of trucks by sorting them in non-descending order according to their random-key values. This sequence is then used to generate a feasible truck schedule according to a decoding scheme.

The advantage of a random-key representation lies in its feature to guarantee that only feasible solutions are generated throughout the execution of the algorithm, cf. Mendes et al. (2009). As the decoding scheme is designed to always generate feasible truck schedules, solutions can be altered through recombination and mutation without ever leading to infeasible solutions. Thus, additional computational effort for repair procedures of infeasible solutions can be avoided.

With the given sequence of trucks, we can start the decoding scheme as shown in Algorithm 1. The decoding scheme always creates a feasible truck schedule with regard to the restrictions of the problem setting concerning conveyor capacities and the availability of gates. Further, it allows flexibility in the unloading speed. We assume that the unloading speed does not have a lower limit and thus $x^{min} = 0$. Each inbound truck is assigned to an empty inbound door in the order of the given sequence *seq* as early as possible. Then, as many parcels as possible are unloaded in the following periods until the inbound truck is empty. The number of unloaded parcels in a period t^c is either limited by the remaining conveyor capacity $r^{rem}_{kt^c}$ in relation to the share of parcels r^{share}_{ik} that are designated to this conveyor, the maximum unloading speed x^{max} or the number of remaining parcels in the inbound truck $load_i$. After the truck has been fully unloaded, the next truck in the sequence is scheduled. Once each inbound is fully unloaded in this manner, the decoding procedure stops. The procedure always generates a feasible, but not necessarily optimal, truck schedule for the given sequence. The fitness of the resulting truck schedule is determined similarly to the Objective Function (1) of the PHSP-LCC-flex.

Algorithm 1 Decoding procedure for a solution of the PHSP-LCC-flex in random-key representation

Input: Instance $P = (\mathcal{I}, \mathcal{O}, \mathcal{K}, \mathcal{T}, U, r_k, x^{max}, ship_{io}, at_i)$
Solution λ in random-key notation
Output: Feasible truck schedule

Initialize earliest possible starting time at gates $ES_u := 0 \quad \forall u = 1, ..., U$
Initialize remaining conveyor capacity $r_{kt}^{rem} := r_k \quad \forall k \in \mathcal{K}, t \in \mathcal{T}$
Initialize inbound load $load_i := \sum_{o \in \mathcal{O}} ship_{io} \quad \forall i \in \mathcal{I}$
Initialize number of unloaded parcels per period $x_{it} := 0 \quad \forall i \in \mathcal{I}, t \in \mathcal{T}$
Decode random-key encoded solution λ into inbound truck sequence seq
for $i \in seq$ **do**
  Select gate u' with earliest possible starting time $ES_{u'}$
  Compute fraction of parcels per conveyor $r_{ki}^{share} = \frac{\sum_{o \in \mathcal{O}_k} ship_{io}}{\sum_{o \in \mathcal{O}} ship_{io}}, k \in \mathcal{K}$
  Compute current period $t^c = \max(at_i, ES_{u'})$
  while $load_i > 0$ **do**
    Schedule maximal number of unloaded parcels in period t^c
    $$x_{it^c} = \min \left\{ \min_{k \in \mathcal{K}} \left(\frac{r_{kt^c}^{rem}}{r_{ki}^{share}} \right), x^{max}, load_i \right\}$$
    Update conveyor utilisation $r_{kt^c}^{rem} = r_{kt^c}^{rem} - x_{it^c} \cdot r_{ki}^{share}, \quad \forall k \in \mathcal{K}$
    Update loading status $load_i = load_i - x_{it^c}$
    Update current period $t^c = t^c + 1$
  end
  $ES_{u'} = t^c$
end

In the decoding procedure, deadlines are not considered. This is no limitation of our approach as the whole framework assures that inbound trucks with many parcels for urgent outbound trucks are prioritised in the course of the algorithm. However, the performance of the procedure is limited by another aspect. Unloading flexibility is limited since the unloading speed is chosen as high as possible. Thus, intended extension of the unloading time of an inbound truck, e.g. due to a reduction of the unloading speed (maybe even to zero), is generally disregarded. The procedure is myopic in its nature and does not give the option of delaying the unloading process of a specific inbound truck to reserve capacities for other inbound trucks that might have closer deadlines.

To combat these limitations, we propose an LP-based improvement procedure that we apply at the end of the genetic algorithm. It utilises the feasible truck schedule by fixing the periods in which an inbound truck is located at an inbound door but recalculates the number of unloaded parcels for all trucks using a reduced formulation of the problem setting. To enforce the inbound gate assignment in the reduced formulation, the parameter x_{it}^{up} is introduced which reflects the maximum number of unloaded parcels for each inbound per period. The

parameter x_{it}^{up} is set to the maximum number of unloaded parcels x^{max} for those periods in which inbound i is unloaded for the given inbound gate assignment and to 0 for the rest of the periods as given by the following formula.

$$x_{it}^{up} = \begin{cases} x^{max}, & x_{it} > 0 \\ 0, & else \end{cases} \tag{17}$$

Since the decoding procedure only generates feasible inbound gate assignments, the number of concurrently assigned trucks never exceeds the number of available gates. Thus, the reduced formulation only has to encompass restrictions concerning conveyor capacity, complete unloading of the inbound trucks and the maximum number of unloaded parcels in each period as all other restrictions are already fulfilled. The model for the reduced formulation of the PHSP-LCC-flex is formally defined as follows.

Reduced Model for the PHSP-LCC-flex

$$\max\ Z = \sum_{o\in\mathcal{O}} \sum_{t\in\mathcal{T}_o} \sum_{i\in\mathcal{I}_t} x_{it}^{re} \cdot w_i \cdot \frac{ship_{io}}{\sum_{\rho\in\mathcal{O}} ship_{i\rho}} \tag{18}$$

subject to

$$\sum_{i\in\mathcal{I}_t} \left(x_{it}^{re} \cdot \frac{\sum_{o\in\mathcal{O}_k} ship_{io}}{\sum_{o\in\mathcal{O}} ship_{io}} \right) \leq r_k \quad \forall k \in \mathcal{K}, \forall t \in \mathcal{T}\backslash\{T\} \tag{19}$$

$$\sum_{t\in\mathcal{T}_i} x_{it}^{re} = \sum_{o\in\mathcal{O}} ship_{io} \quad \forall i \in \mathcal{I} \tag{20}$$

$$x_{it}^{re} \leq x_{it}^{up} \quad \forall i \in \mathcal{I}, \forall t \in \mathcal{T}_i\backslash\{T\} \tag{21}$$

$$x_{it}^{re} \geq 0 \quad \forall i \in \mathcal{I}, \forall t \in \mathcal{T}_i \tag{22}$$

The objective function (18) maximises the number of non-delayed parcels. Constraints (19) restrict the number unloaded parcels for each conveyor k to the conveyor capacity. Constraints (20) ensure that each inbound truck is fully unloaded. The given truck schedule is enforced by Restrictions (21). Lastly, Eq. (22) define x_{it}^{re} as a positive continuous variable.

With the LP-based improvement procedure, we can address the limitations of the decoding procedure regarding the number of unloaded parcels. However, intended extensions of unloading processes are still not possible since the starting times and ending times are fixed based on Restriction (17). This means that even with the improvement procedure, only a subsection of the overall solution space of the

problem is covered by the random-key representation of the solution. Consequentially, for some instances, an optimal solution might not be covered by the representation. However, we can still use the procedure to find good solutions quickly.

4.3 Elements of the genetic algorithm: initialisation, crossover, mutation and selection

The algorithm is initialised by generating N^{ind} randomly generated individuals. As we use a *biased random-key* approach, each individual receives a vector λ of $|I|$ uniformly distributed random numbers taken from the range between 0 and 1.

In the definition of the crossover operator, we favour individuals with higher fitness. The $N^{elite} < \frac{N^{ind}}{2}$ fittest individuals of the current population are designated as the *elite*. Each time two parents are selected for the crossover operator, the first parent λ^{p1} has to be taken from *elite*. Thereby, individuals can be selected more than once. The second parent λ^{p2} is randomly selected from the rest of the current population. Based on the parents λ^{p1} and λ^{p2}, we apply the crossover operator to generate a single offspring λ^{o}. We generate a uniformly distributed random number q from the interval [0, 1] ($q \sim \mathcal{U}(0,1)$) for each random key λ_i^o of the solution and compare it with the selection probability p^s. If the generated value q is smaller than p^s, the respective random key λ_i^o of the offspring receives the random key of the first (elite) parent λ_i^{p1}. Otherwise, λ_i^o is set to the respective random key of the second parent λ_i^{p2}. Applying the crossover operator with selection probability $p^s = 0.75$ is illustrated in Fig. 5. In total, N^{cross} individuals are generated in this way.

The biased approach has a strong tendency to converge to a local optimum. Thus, we use a mutation operator that introduces completely new solutions. In each generation, N^{mut} randomly generated mutants are created. Each random-key value λ_i of such an individual is set to a random value between 0 and 1. In this way, we introduce new random-key values in each generation and the diversity of the population is greatly enhanced to counteract premature convergence to a local optimum.

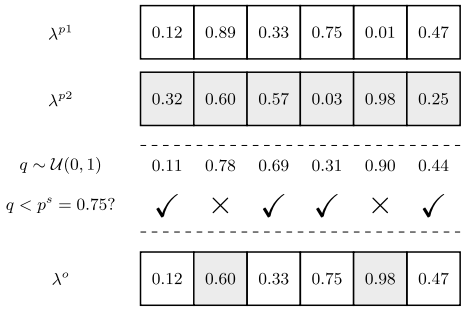


Fig. 5 Crossover in random-key representation with $p^s = 0.75$

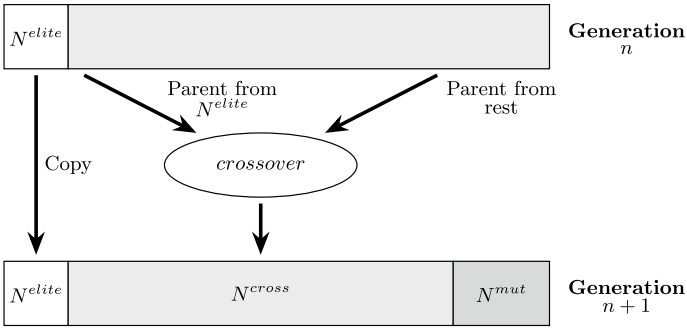


Fig. 6 Selection between generations

Figure 6 illustrates the creation of new generations. The N^{elite} individuals from *elite* are directly copied to the next generation. The majority of the new generation consists of N^{cross} offspring generated by the crossover operator. The remainder are N^{mut} randomly generated individuals. Therefore, the total number of individuals per generation is $N^{ind} = N^{elite} + N^{cross} + N^{mut}$. Generally, the procedure leads to a fast convergence to local optima through the preferential treatment of good solution and still allows for a high degree of population diversity due to the rigorous mutation operator.

5 Numerical analysis

5.1 Test design

To investigate the performance of the mathematical model and the genetic algorithm, we use three distinct data sets of different sizes. An overview over the different instance sets is shown in Table 3 and explained in the following. One class of instances represents a parcel hub that is comparably small, with $|\mathcal{I}| = 19$ inbound trucks, $|\mathcal{O}| = 27$ outbound destinations and $|\mathcal{K}| = 4$ connecting conveyor belts. The second data set represents a medium-sized hub with $|\mathcal{I}| = 36$ inbound trucks and $|\mathcal{O}| = 55$ outbound destinations connected by $|\mathcal{K}| = 5$ conveyor belts. The parameters used for the small and medium class of instances are based on data for parcel hubs we actually observed in practice. The third instance class represents a large parcel hub with $|\mathcal{I}| = 100$ inbound trucks, $|\mathcal{O}| = 100$ outbound destinations and $|\mathcal{K}| = 15$ connecting conveyor belts. For the large class of instances we do not have any direct data to use as a reference from practice but hubs of similar size are described in Boysen et al. (2017). Generally, we mainly seek to investigate the computational performance of our solution methods with this more synthetic instance set. For all instances, each conveyor transports parcels to a specified subset of outbound trucks.

Table 3 Instance data

Parameter	Small	Medium	Large
Inbound trucks $\lvert\mathcal{I}\rvert$	19	36	100
Outbound destinations $\lvert\mathcal{O}\rvert$	29	55	100
Conveyor belts $\lvert\mathcal{K}\rvert$	4	5	15
Parcels in inbound trucks l_i	650		
Maximal speed x^{max}	65		
Latest deadline T^{max}	36		
Parcel heterogeneity factor α	{0.4,0.6,0.8}		
Deadline time window factor μ	{0.1,0.2,0.3}		
Conveyor scarcity factor σ	{0.9,1,1.1}		
Gate scarcity factor β	{0.9,1,1.1}		

For all types of hubs, we consider a broad variety of different instances. For all instances, we define the number of parcels in inbound truck i as $l_i = 650$ and a weighting factor $w_i = 1\ \forall i \in \mathcal{I}$. We set the maximal speed x^{max} to 65 $\frac{parcels}{period}$ and a minimal speed x^{min} to 0 $\frac{parcels}{period}$, i.e., full flexibility. The individual parcels are randomly assigned to a number of destinations represented by outbound trucks o according to the parameter $\alpha \in]0, 1]$. The parameter determines the heterogeneity of the parcels inside the inbound trucks with regard to their destination. For a heterogeneity $\alpha = 1$, each inbound truck potentially includes parcels for all destinations. A heterogeneity $\alpha = 0.5$ refers to the case where the parcels are designated for 50% of destinations, etc. To achieve this property, at first a random subset $\mathcal{O}_i^{sub} \subset \mathcal{O}$ with $\lceil|\mathcal{O}| \cdot \alpha\rceil$ elements is generated

$$\mathcal{O}_i^{sub} = \text{sample}(\mathcal{O}, \lceil|\mathcal{O}| \cdot \alpha\rceil) \qquad \forall i \in \mathcal{I}. \tag{23}$$

Then, a preliminary number of parcels $ship_{io}^{raw}$ per destination o is randomly determined:

$$ship_{io}^{raw} = \text{rand}(0.5, 1.5) \cdot \frac{l_i}{|\mathcal{O}_i^{sub}|} \qquad \forall o \in \mathcal{O}_i^{sub}, \forall i \in \mathcal{I}. \tag{24}$$

Afterwards, the preliminary values are adjusted to equal the total number of parcels l_i for each inbound truck. Thus, for each inbound truck, we decrease or increase the number of parcels for a random destination until the total number of parcels $\sum_{o \in \mathcal{O}_i^{sub}} ship_{io}$ equals l_i.

The referenced time horizon is set to $T^{max} = 36$. Thus, with a period length of 5 min, the referenced planning horizon has a length of 180 min. To set the deadline dl_o of each outbound destination, we use a parameter $\mu \in [0, 1]$. Lower values of μ represent less scattered deadlines.

$$dl_o = \lceil\text{rand}(1 - \mu, 1 + \mu) \cdot T^{max}\rceil \qquad \forall o \in \mathcal{O} \tag{25}$$

Note that the actual time horizon can be longer than the referenced time horizon T^{max} due to late deadlines.

The arrival time at_i is set to the first period of the time horizon for 70% of inbounds and randomly taken between period t2 and t5 for the remaining 30% of inbounds.

In the numeric study, we seek to investigate hubs with different levels of resource scarcity, namely the conveyor scarcity and gate scarcity. We first determine a lower bound for the conveyor capacity r_k^{LB} under the assumption that all parcels $\sum_{i\in\mathcal{I}} l_i$ are equally distributed over the complete time horizon T^{max} as well as over all conveyors $|\mathcal{K}|$ and are unloaded in time:

$$r_k^{LB} = \frac{\sum_{i\in\mathcal{I}} l_i}{|\mathcal{K}| \cdot T^{max}} \tag{26}$$

We then introduce the parameter $\sigma > 0$ to set the conveyor scarcity. Note that it is not essential to use a integer value for this parameter.

$$r_k = \lceil r_k^{LB} \cdot \sigma \rceil \tag{27}$$

We apply the same idea of an evenly distribution of the unloading process over the whole planning horizon T^{max} to compute the minimal number of gates U^{LB}.

$$U^{LB} = \frac{\sum_{i\in\mathcal{I}} l_i}{T^{max} \cdot x^{max}} \tag{28}$$

Using the gate scarcity $\beta > 0$, we can compute the number of gates of an instance:

$$U = \lceil U^{LB} \cdot \beta \rceil \tag{29}$$

If we set the scarcity parameters σ or β to values lower than one, the available resources are certainly insufficient and tardy parcels are inevitable. Accordingly, if the scarcity parameters σ or β are greater than one, we can potentially find a solution where all parcels reach their destinations before the deadline. Our analysis focuses on the influence of the scarcity of both gate and conveyor capacities. Thus, we set the conveyor scarcity factor σ and the gate scarcity factor β to the values $\{0.9, 1, 1.1\}$ for all instance classes and assess the impact on performance and the potentials of a higher degree of flexibility.

We generate five instances with different random seeds for each combination of the given parameters and each class hub size. Thus, the total number of instances we consider for each hub size is $3^4 \cdot 5 = 405$. With 3 classes of instances, we receive $3 \cdot 405 = 1215$ instances in total.

5.2 Performance analysis of the genetic algorithm and the MIP model

In this section, we evaluate the computational performance of the PHSP-LCC-flex solved with the standard solver Gurobi and the genetic algorithm. Both the results for the standard solver Gurobi 9.5 as well as the genetic algorithm were calculated

on the university's computer cluster with a 2.93 GHz Intel Westmere-EP Xeon X5670 processor and 48 gigabytes of RAM. Both were implemented in Python.

For the genetic algorithm, we assume a population size N^{ind} of 50 individuals. 20% of individuals are taken from the *elite* part of the population ($N^{elite} = 10$) and 70% of the individuals of the following generation are generated through the crossover operator ($N^{cross} = 35$). The remaining 10% of the population consists of randomly generated mutants ($N^{mut} = 5$). The selection probability equals $p^s = 0.75$ for all instances. We apply the LP-based improvement procedure to the *elite* part of the population in the last iteration of the algorithm. We present the results for a time limit of $\bar{t} = 1$, $\bar{t} = 10$ and $\bar{t} = 30$ s. For Gurobi, we set the time limit to $\bar{t} = 30$ and $\bar{t} = 3600$ s.

In Fig. 7, we present the influence of the instance parameters on the solutions. We refer to the results of the genetic algorithm with a time limit of 30 s, as this approach is able to generate results with a reasonable solution quality for all instance classes. Each bar in this figure shows the average share of duly parcels over 135 instances for each instance parameter under consideration. Please note that we refer to the *share* of duly parcels instead of the total number of duly parcels as the interpretation of

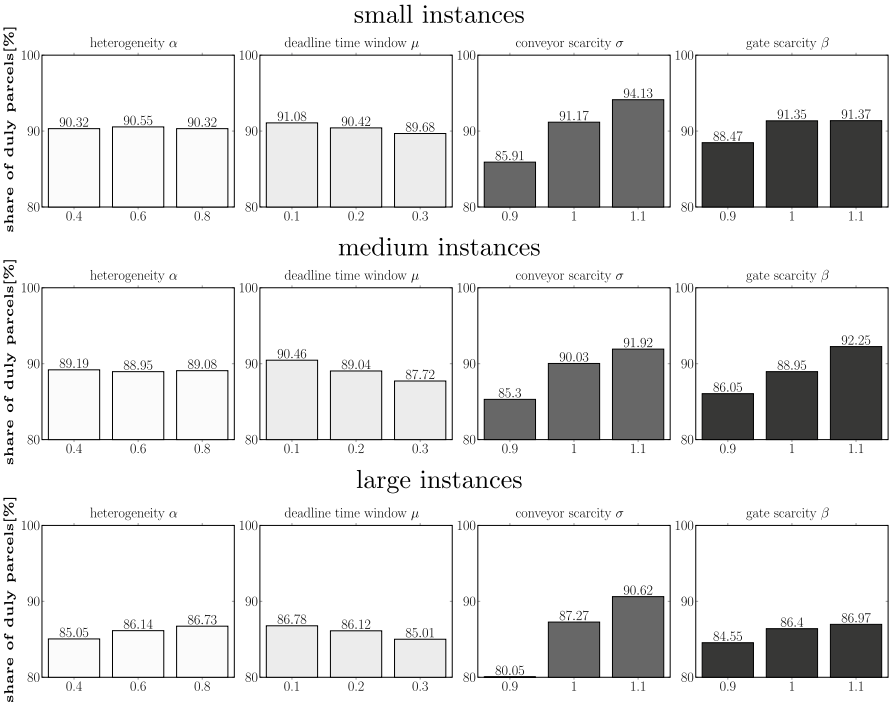


Fig. 7 Influence of instance parameters

this value is easier. Of course, this conversion has no influence on the behaviour of the results.

The heterogeneity α of parcels in the truck does not have a large influence on the share of duly parcels with around 90% for all instance classes. Only for the large instances, a higher heterogeneity has a positive influence on the punctuality of the parcels. The deadline time window μ influences the tardiness slightly, but for all instance classes in the same manner: A higher variability, i.e. higher values of μ, makes it harder to load the parcels in time. The conveyor scarcity σ, which is the main focus of our paper, has the highest influence on the performance of the cross-dock. For low values for σ, the conveyor capacities are particularly scarce and the share of duly parcels decreases. The gate scarcity β has a similar influence as a higher scarcity, i.e. low values for β, decreases the share of duly parcels. As both the conveyor scarcity σ and the gate scarcity β influence the performance of the cross-docking hub, we will focus the following analysis on these aspects.

Table 4 depicts the numeric results for the instances corresponding to small, medium and large hubs. For the small and medium instances, each row represents the average results over 45 instances with the respective values for σ and β. Note that regarding the large instance class, for two instances only trivial solutions with gap $= \infty$ could be found. Thus, we excluded those instances.

Gurobi^{3600} refers to results obtained using Gurobi with a time limit of 3600 s and is used as the benchmark. We present the average share of non-delayed parcels ("obj"), the average relative optimality gap displayed by the solver after the time limit ("gap"), the average computation time ("time") and the share of instances solved to optimality within the time limit ("*%opt*"). For Gurobi^{30}, we show the share of duly parcels found by Gurobi after a time limit of 30 s ("obj"), the gap to the solution found by Gurobi^{3600} ("gap^{Gurobi}"), the gap to the best found solution by all methods ("gap^{best}") and the share of instances for which the best solution has been found by Gurobi^{30} ("*%best*") . For the genetic algorithm, GA^{1}, GA^{10} and GA^{30} show the results for a time limit of $\bar{t} = 1$, $\bar{t} = 10$ and $\bar{t} = 30$ s, respectively. We use the same metrics as for Gurobi^{30}. The entries in bold font refer to the averages over all instances of the respective instance class.

Gurobi could only find a proven optimal solution for the PHSP-LCC-flex for 155, i.e. 38.3 %, of the 405 small instances and could not solve any of the medium or large instances to optimality within the time limit of 3600 s. Especially for instances with gate scarcity factor $\beta = 0.9$ and thus the gate capacity constituting the bottleneck of the system, we observe a comparatively large optimality gap for the small and medium instances. For the large instances, the solutions found by Gurobi are generally poor as we observe an average optimality gap of 81.2 % even with a time limit of 3600 s. For a time limit of 30 s, only trivial solutions with an objective function value of 0 can be found for the majority of these large instances. For the small and medium instances, we can observe a comparatively small deviation of 1.3 and 1.4 % between the solutions with a time limit of 30 s and 3600 s.

Comparing the results of the genetic algorithm with those generated by Gurobi shows that the genetic algorithm is able to find better solutions on average for all instance classes if we apply a time limit of 30 s for both approaches. We can observe for instances where Gurobi is able to prove optimality or shows an insignificant remaining

Table 4 Numerical results for the genetic algorithm and Gurobi

			Gurobi^{3600}				Gurobi^{30}				GA^{1}				GA^{10}				GA^{30}			
size	σ	β	obj	gap	time	% opt	obj	gap^{Gurobi}	gap^{best}	% best	obj	gap^{Gurobi}	gap^{best}	% best	obj	gap^{Gurobi}	gap^{best}	% best	obj	gap^{Gurobi}	gap^{best}	% best
small	0.9	0.9	84.9	1.4	3600	0.0	83.0	2.3	2.3	0.0	82.1	3.3	3.4	0.0	83.9	1.2	1.3	0.0	84.0	1.0	1.0	11.1
	1.0	0.9	89.6	1.9	3600	0.0	88.4	1.4	1.5	8.9	88.3	1.5	1.6	0.0	89.4	0.2	0.4	20.0	89.6	0.0	0.2	33.3
	1.1	0.9	90.9	1.9	3600	0.0	90.6	0.3	0.4	46.7	90.5	0.5	0.6	8.9	90.9	-0.1	0.0	64.4	90.9	-0.1	0.0	64.4
	0.9	1.0	86.6	0.2	2793	35.6	84.9	1.9	1.9	2.2	83.6	3.4	3.4	0.0	84.8	2.0	2.0	0.0	85.0	1.8	1.8	0.0
	1.0	1.0	91.8	0.2	3123	28.9	90.0	2.0	2.0	0.0	89.5	2.6	2.6	0.0	90.4	1.5	1.5	0.0	90.6	1.3	1.3	0.0
	1.1	1.0	95.7	0.7	3493	4.4	94.5	1.2	1.2	2.2	94.2	1.6	1.6	0.0	94.9	0.8	0.8	0.0	95.0	0.7	0.7	4.4
	0.9	1.1	86.3	0.0	1258	88.9	85.4	1.0	1.0	11.1	83.6	3.0	3.0	0.0	84.5	2.1	2.1	0.0	84.8	1.8	1.8	0.0
	1.0	1.1	92.1	0.0	1132	95.6	91.3	0.9	0.9	11.1	90.1	2.1	2.1	0.0	90.7	1.5	1.5	0.0	91.0	1.2	1.2	0.0
	1.1	1.1	95.7	0.0	1030	91.1	95.2	0.5	0.5	17.8	94.3	1.5	1.5	0.0	94.8	0.9	0.9	0.0	94.8	0.9	0.9	0.0
∅			**90.4**	**0.7**	**2625**	**38.3**	**89.2**	**1.3**	**1.3**	**11.1**	**88.5**	**2.2**	**2.2**	**1.0**	**89.4**	**1.1**	**1.2**	**9.4**	**89.5**	**1.0**	**1.0**	**12.6**
medium	0.9	0.9	83.2	5.0	3600	0.0	82.1	1.7	3.5	0.0	82.6	0.7	2.9	0.0	84.5	-1.5	0.7	11.1	85.0	-2.1	0.1	32.0
	1.0	0.9	87.5	2.3	3600	0.0	87.6	0.7	1.6	28.9	87.7	-0.3	1.4	0.0	88.7	-1.5	0.3	11.1	88.9	-1.7	0.0	93.3
	1.1	0.9	87.5	2.0	3600	0.0	88.1	0.3	1.0	31.1	88.0	-0.6	1.0	0.0	88.8	-1.6	0.1	8.9	88.9	-1.7	0.0	95.6
	0.9	1.0	85.6	2.4	3600	0.0	84.5	1.7	1.9	6.7	83.7	2.2	2.8	0.0	85.1	0.6	1.2	6.7	85.5	0.1	0.7	17.8
	1.0	1.0	89.8	4.0	3600	0.0	88.6	1.5	2.7	4.4	89.5	0.3	1.8	0.0	90.8	-1.2	0.3	15.6	91.1	-1.5	0.0	75.6
	1.1	1.0	91.4	3.9	3600	0.0	91.2	0.6	1.4	11.1	91.7	-0.2	0.8	0.0	92.3	-1.0	0.1	15.6	92.4	-1.1	0.0	93.3
	0.9	1.1	87.0	0.3	3600	0.0	85.5	1.8	1.8	0.0	84.1	3.4	3.4	0.0	85.1	2.2	2.2	0.0	85.5	1.8	1.8	0.0
	1.0	1.1	92.8	0.6	3600	0.0	90.9	2.1	2.1	2.2	90.7	2.3	2.4	0.0	91.6	1.4	1.4	2.2	91.8	1.2	1.2	0.0
	1.1	1.1	96.9	0.5	3600	0.0	94.9	2.1	2.1	6.7	94.9	2.1	2.1	0.0	95.6	1.3	1.3	0.0	95.8	1.2	1.2	2.2
∅			**89.1**	**2.3**	**3600**	**0.0**	**88.1**	**1.4**	**2.0**	**10.1**	**88.1**	**1.1**	**2.1**	**0.0**	**89.2**	**-0.1**	**0.9**	**7.9**	**89.4**	**-0.4**	**0.6**	**45.5**
large	0.9	0.9	50.2	86.5	3600	0.0	0.0	100.0	100.0	0.0	78.0	-72.6	1.7	0.0	78.6	-74.0	0.9	2.3	79.3	-75.5	0.0	97.7
	1.0	0.9	73.4	21.7	3600	0.0	0.0	100.0	100.0	0.0	84.6	-15.8	1.8	0.0	85.4	-16.9	0.9	0.0	86.2	-18.0	0.0	100.0
	1.1	0.9	79.1	12.8	3600	0.0	0.0	100.0	100.0	0.0	87.2	-10.4	1.1	0.0	87.7	-11.0	0.5	0.0	88.2	-11.6	0.0	100.0
	0.9	1.0	48.8	99.2	3600	0.0	0.0	100.0	100.0	0.0	79.1	-86.8	1.3	0.0	79.7	-88.1	0.7	4.4	80.2	-89.3	0.0	95.6
	1.0	1.0	55.6	76.5	3600	0.0	0.0	100.0	100.0	0.0	86.5	-67.5	1.4	0.0	87.1	-68.7	0.6	4.4	87.6	-69.7	0.0	95.6
	1.1	1.0	70.5	48.5	3600	0.0	0.0	100.0	100.0	0.0	90.6	-41.3	0.8	0.0	91.0	-42.0	0.4	6.7	91.3	-42.5	0.0	93.3
	0.9	1.1	47.1	103.1	3600	0.0	0.0	100.0	100.0	0.0	79.8	-92.1	1.1	2.3	80.2	-93.0	0.6	13.6	80.7	-94.2	0.1	81.8
	1.0	1.1	46.0	122.3	3600	0.0	0.3	99.3	99.7	0.0	87.2	-113.2	0.9	2.2	87.7	-114.4	0.4	20.0	88.0	-115.1	0.1	80.0
	1.1	1.1	42.0	160.3	3600	0.0	0.0	100.0	100.0	0.0	91.6	-150.0	0.7	0.0	92.0	-151.0	0.3	6.7	92.3	-151.7	0.0	93.3
∅			**57.0**	**81.2**	**3600**	**0.0**	**0.0**	**99.9**	**100.0**	**0.0**	**85.0**	**-72.2**	**1.2**	**0.5**	**85.5**	**-73.2**	**0.6**	**6.5**	**86.0**	**-74.2**	**0.0**	**93.0**

optimality gap, such as the small instances with $\beta = 1.1$, the gap GA^{30} is only between 0.9 and 1.8 %. Thus, we can conclude that the genetic algorithm is generally capable of finding solutions very close to the proven optimum. Especially for instances with the gate scarcity factor $\beta = 0.9$, the genetic algorithm performs better than Gurobi. Additionally, the genetic algorithm is able to provide adequate solutions for the large instances where Gurobi only generates solutions of poor quality. Concerning the influence of the time limit of the genetic algorithm, we can see a steady increase in the solution quality for all instances as expected. Particularly, for instances with scarce conveyor capacities ($\sigma = 0.9$), the difference between the solutions found after 1 s and those found after 30 s is more pronounced.

6 Practical application using a simulation model

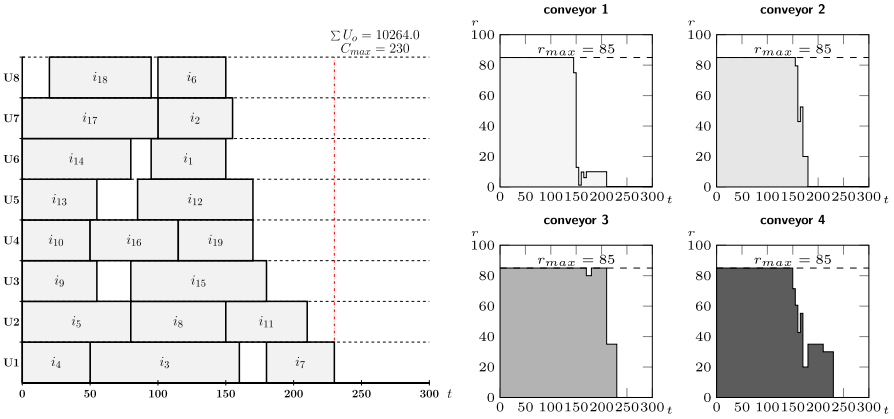
6.1 Discrete-event simulation model of the parcel hub

To examine the applicability of the PHSP-LCC-flex in real-world environments, we consider the transshipment process for the hub illustrated in Fig. 2. The U-shaped hub has eight inbound doors $u1$ to $u8$ and 27 outbound doors $o1$ to $o27$ connected by four conveyors $k1$ to $k4$. Suppose 19 inbound trucks $i1$–$i19$ contain 650 parcels each. Thus, $19 \cdot 650 = 12350$ parcels are transferred in total. We assume that each inbound truck is unloaded with a fixed rate of 13 parcels per minute and that a maximum of 85 parcels can be transported on each conveyor per minute.

Figure 8 shows an optimised inbound truck schedule of the PHSP-LCC-flex. For the following analysis, we denote the number of duly parcels for outbound truck o as U_o. We receive a result of $\sum_o U_o = 10264$ non-delayed parcels with a makespan of 230 min as shown in Fig. 8a. With regards to the conveyor utilisation we can observe in Fig. 8b that in the results of the optimisation model the conveyors are used to the fullest extend for the majority of the time. Further, all the dock doors are used to unload inbound trucks.

However, the mathematical model does not consider parcel interactions but aggregates the individual parcels into a homogenised parcel flow as modelling detailed interactions would render the model mathematically intractable. In a real-world environment, the parcel flow originating from an inbound truck is mostly heterogeneous since the parcels leave the trucks in a random sequence. Thus, we may observe an accumulation of transfers for a specific conveyor, especially when interacting with the parcel flows from concurrently unloaded inbound trucks. This can lead to the blocking of said conveyor and delays in the overall process.

To answer the question how the results of the optimisation model can be used in a real-world setting, we use a discrete-event model for simulating the parcel hub with a custom-made simulation framework in Python. We define system states, events and an event list to model the transfer of the parcels through the hub. The main entities of the model are the parcels, which are generated once a truck is assigned to an inbound door. As a benchmark policy for the comparison

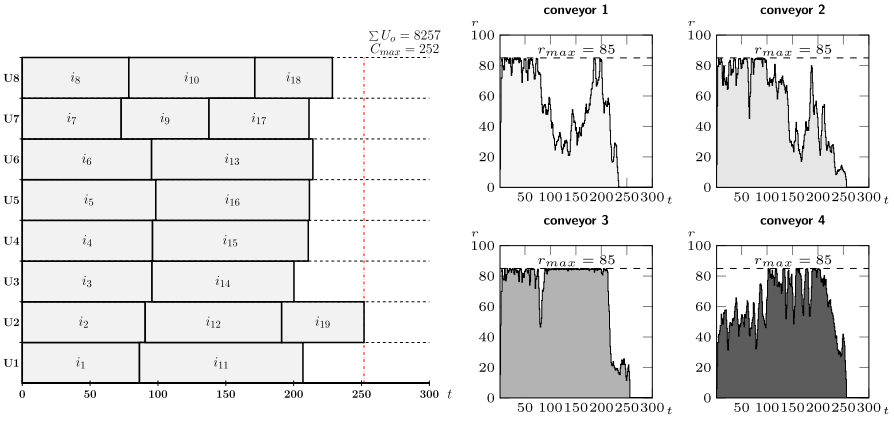


(a) Optimised schedule **(b)** Optimised conveyor utilisation

Fig. 8 Exemplary optimisation results

and reference case of the operations at the parcel hub, we assume a simple first-come first-serve scheduling policy for the assignment of the inbound trucks.

Figure 9a shows the resulting schedule, the total makespan C_{max} of 252 min until all parcels are unloaded and the number of duly parcels $\sum_o U_o = 8257$ for a FCFS policy. Thus, the results differ significantly from those of the deterministic model. The resulting conveyor utilisation is shown in Fig. 9b. We can observe that conveyor $k3$ constitutes the bottleneck of the system since it is operated close to its maximum capacity most of time, whereas the other conveyors have fluctuating capacity usage throughout the planning horizon. The unloading time of the individual trucks is significantly slowed down by the conveyors operating close



(a) FCFS schedule **(b)** FCFS conveyor utilisation

Fig. 9 Exemplary simulation results

to their maximum capacity. For example, although loaded with the same number of parcels, unloading truck $i9$ takes much longer than that of truck $i19$. This indicates that the trucks compete for the scarce conveyor capacity with the result of slowing down their unloading process, eventually leading to delays.

As observable in the example, a FCFS scheduling policy does neither consider limited conveyor capacities nor outbound deadlines. For that reason, conveyors may for instance be occupied by parcels with a loose deadline or by parcels that are already too late to reach their corresponding outbound truck in time. In addition, by finding a better schedule for the inbound trucks, the utilisation of the remaining conveyors could be improved and would lead to a higher share of duly parcels for the corresponding outbound trucks.

Our goal is to develop a more sophisticated decision support system for controlling the assignment of inbound trucks to inbound doors with the help of the results of PHSP-LCC-flex. To achieve this, we define scheduling policies that use the results of the PHSP-LCC-flex.

6.2 Scheduling policies

The result of the PHSP-LCC-flex is a truck schedule that includes the time periods in which each inbound truck is unloaded at a gate and the number of parcels that are unloaded in each period. We use the information provided by these truck schedules to develop different scheduling policies. We define three distinct scheduling policies that are based on the results of the PHSP-LCC-flex and use two simple scheduling rules as a reference for the assignment of the inbound trucks to door:

Basic scheduling policies

- *First-Come First-Serve (FCFS)*: Sort the inbound trucks in non-descending order according to their arrival time and assign them to a dock once it is free.
- *Sort by Priority (Prio)*: Sort the inbound trucks according to their urgency-based priority value $prio_i = \sum_{o \in O} dl_o \cdot ship_{io} / \sum_{\rho \in \mathcal{O}} ship_{i\rho}$ in ascending order and assign them once a gate is free. Thus, inbound trucks containing many parcels for outbound trucks with early deadlines are prioritised.

Scheduling policies based on the PHSP-LCC-flex model

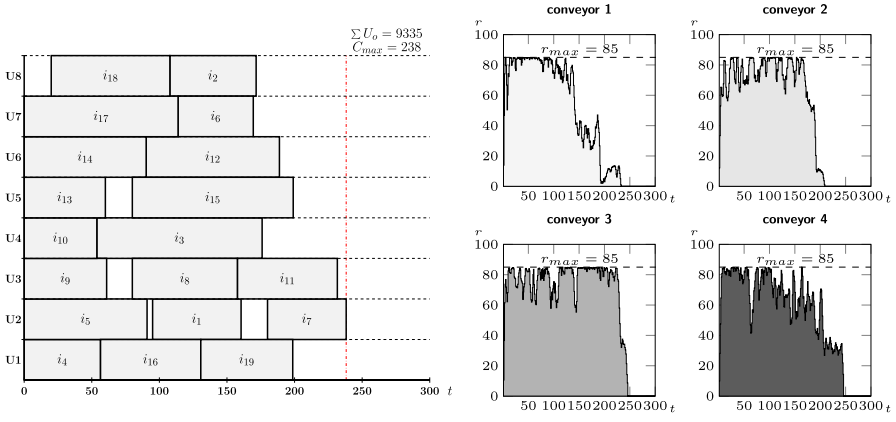
- *Starting Times of the PHSP-LCC-flex (Flex)*: Use the starting times of the solution of the PHSP-LCC-flex and assign the inbound trucks once a dock is free after their starting time.
- *Starting Times and Unloading Speeds of the PHSP-LCC-flex (Flex$^+$)*: Use the starting times **and** the calculated unloading speeds of the solution of the PHSP-LCC-flex and assign the inbound trucks once a dock is free after their starting time.

Please note that the unloading speed refers to the actual start period and thus a fixed sequence of speeds is implemented. As an example, assume an inbound truck that should—according to the solution of the mathematical model—start in period 5 with a speed of 40 parcels in this period and then accelerates to a speed of 65 parcels per period in period 6 and the following. If this truck actually starts in period 7 in the simulation, the fixed speed of the first unloading period 7 is set to 40 parcels per period and the speed is than accelerated to 65 parcels per period in its second unloading period, namely period 8.

- *Starting Times and Priorities of the PHSP-LCC-flex (Flex^{prio})*: Use the starting times of the solution of the PHSP-LCC-flex and assign the inbound trucks once a dock is free after their starting time. In case a conveyor reaches its maximal capacity, prioritise those trucks that would have had to unload the highest number of parcels until the current time period according to the solution of the PHSP-LCC-flex. The priority of truck i at simulation time t' is formally calculated according to the following formula:

$$prio_{it'} = \sum_{t=1}^{t'} x_{it}^{opt} \tag{30}$$

To illustrate the effect of implementing scheduling policies based on the results of the PHSP-LCC-flex, we apply the Flex^{+} scheduling policy to the example given above. The resulting truck schedule and conveyor utilisation is shown in Fig. 10. We observe that conveyors cannot be fully utilised, cf. Fig. 10b. Variations in the workload induced by random accumulations of parcels for specific conveyors cause delays and result in a reduced number of non-tardy parcels $\sum U_o = 9335$ and a longer makespan of $C_{max} = 238$ compared to the results of the model with $\sum U_o = 10264$ and $C_{max} = 230$ as shown in Fig. 8. Thus, the potentials observed



(a) Simulated Schedule with Flex^{+} **(b)** Simulated conveyor utilisation

Fig. 10 Schedule and conveyor utilisation for Flex^{+}

in the deterministic setting cannot be fully exploited in the real-world application. However, if we compare the results to those of the simple FCFS policy, we observe a significant improvement as we can increase the number of parcels arriving on time from 8257 to 9335.

In Sect. 6.3, we present more extensive numerical results for the different scheduling policies to investigate their performance for the test instances illustrated in Sect. 5.1.

6.3 Numerical results for the scheduling policies using the simulation model

We employ the simulation model to assess the applicability of the optimised inbound schedules using the stated scheduling policies. We apply results of the genetic algorithm using a time limit of 30 s (GA^{30}) as input values to get the information on starting times and speed, respectively, required for implementing the scheduling policies. For each instance, we conduct 100 simulation runs with different random unloading sequences for the parcels leaving the inbound trucks and apply all scheduling policies in each simulation run. Thus, the unloading sequence is the same for each scheduling policy for an individual run. We present the average number of non-delayed parcels over all simulation runs.

In Table 5, each line represents the averages for $3 \cdot 3 \cdot 5 = 45$ instances for given scarcity factors σ and β. Thus, each row shows the averages for 45 instances. We present the results for the FCFS policy (FCFS), the priority-based scheduling policy (Prio), using the starting times generated with the PHSP-LCC-flex (Flex), employing both the starting times and unloading speeds from the PHSP-LCC-flex (Flex^{+}) and using a combination of the starting times of the PHSP-LCC-flex together with the prioritisation of inbound trucks according to the planned number of unloaded parcel until the current period (Flex^{prio}).

The results show that the share of non-delayed parcels is increased with an average of 3.7% for the small instances, of 2.1% for the medium instances and 0.9% for the large instances when using the Flex policy compared to the implementation of the FCFS policy. The Flex^{+} policy only yields minor improvements with regards to the number of non-delayed parcels for the large instances. For the small and medium instances, applying the Flex^{+} policy can even be detrimental compared to the Flex policy. An explanation for this is that the unloading speeds of the Flex^{+} scheduling policy are highly dependent on the current state of the system with regard to the gate and conveyor utilisation. Thus, the results are only directly applicable if the inbound truck schedule is executed as planned in the model. Smaller variations of the plan due to, e.g., the discretisation of the parcel flow or small delays induced by the random accumulation of parcels for a certain conveyor in the simulation model have implications on the state of the system. As a consequence, we observe desynchronisation of the unloading speeds and the system state.

With an improvement of 5.4, 3.3 and 2.4%, respectively, for the small, medium and large instances compared to the FCFS policy, Flex^{prio} generates the best results. The higher performance compared to Flex policy can be explained by the indirect usage of the unloading speeds provided by the PHSP-LCC-flex. Flex^{prio} is not as

Table 5 Percentage of duly parcels for the scheduling policies

Size	σ	β	FCFS	Prio	Flex	Flex^{+}	Flex^{prio}
Small	0.9	0.9	70.9	71.1	75.0	74.5	75.9
	1.0	0.9	77.5	78.3	80.8	80.5	81.3
	1.1	0.9	82.6	83.7	85.1	85.0	85.5
	0.9	1.0	72.9	73.3	75.9	75.9	77.7
	1.0	1.0	79.5	79.9	82.3	82.1	83.5
	1.1	1.0	86.6	87.4	89.3	89.0	90.0
	0.9	1.1	73.2	73.2	75.9	76.5	78.5
	1.0	1.1	80.8	81.7	82.9	83.8	85.5
	1.1	1.1	86.9	88.1	89.7	90.1	91.3
			79.0	**79.6**	**81.9**	**81.9**	**83.3**
Medium	0.9	0.9	71.6	71.8	73.8	73.1	74.4
	1.0	0.9	78.0	78.9	79.2	79.2	79.5
	1.1	0.9	83.0	83.9	83.0	83.0	83.3
	0.9	1.0	72.6	72.8	75.1	74.4	76.3
	1.0	1.0	78.9	79.5	81.2	80.5	81.8
	1.1	1.0	84.8	85.5	86.1	86.0	86.3
	0.9	1.1	73.8	74.1	75.9	75.6	78.2
	1.0	1.1	81.3	82.2	83.7	83.3	85.2
	1.1	1.1	88.7	89.4	90.4	90.4	91.5
			79.2	**79.8**	**80.9**	**80.6**	**81.8**
Large	0.9	0.9	68.1	68.0	68.9	69.0	69.9
	1.0	0.9	74.7	75.0	75.3	75.3	75.6
	1.1	0.9	80.0	80.6	79.9	79.9	80.1
	0.9	1.0	69.7	69.7	70.4	70.7	72.1
	1.0	1.0	76.6	76.6	77.3	77.3	78.3
	1.1	1.0	81.9	82.4	83.2	83.0	83.5
	0.9	1.1	70.7	70.7	71.1	71.9	73.6
	1.0	1.1	77.8	78.0	78.3	78.4	80.0
	1.1	1.1	83.9	84.1	84.9	84.8	85.7
			75.9	**76.1**	**76.6**	**76.7**	**77.7**

susceptible to smaller variations as the Flex^{+} policy as it only prioritises certain trucks instead of assuming predefined and fixed unloading speeds for each parcel as an input. Thus, deviations between the system state in the simulation and the optimised truck schedule become less severe.

For further insights on the opportunities of employing the scheduling policies in specific circumstances, refer to Fig. 11. We present the results of the simulations study with fixed values for the instance parameters α, μ, σ and β, respectively. We only present the results of the medium instances as a reference case as the other instance classes show the similar tendencies. Each bar illustrates the average improvement for a specific scheduling policy compared to the FCFS scheduling

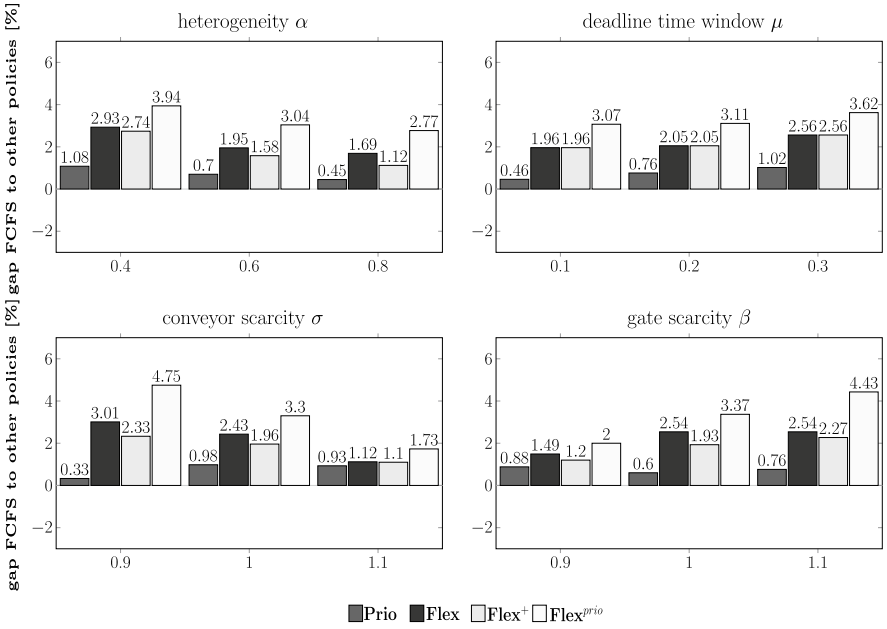


Fig. 11 Comparison FCFS and advanced scheduling policies

policy with the given parameter value for α, μ, σ and β, respectively, thus 135 instances each.

We can observe that Flex^{prio} performs best for all instance parameters. Especially in instances with a low level of heterogeneity α with regards to the parcel destinations, the Flex, Flex^{+} and Flex^{prio} policies perform well. Varying levels of deadline tightness μ have a lower influence on the performance of the Flex and Flex^{+} policy. For a very high scarcity of the conveyors, i.e. low values of σ, Flex^{prio} shows the highest improvement. Similar results can be seen for instances with high values of β and thus a comparably high number of gates. In both cases, the conveyor capacities are more likely to constitute the bottleneck of the system. Accordingly, if the conveyor capacity σ increases and gate capacity β decreases, the effect is less pronounced.

To reiterate, the main focus of our approach is to efficiently use limited conveyor capacities by allowing flexibility. Thus, the most relevant instances are those instances where the conveyors are actually the limiting factor of the system, hence, especially σ= 0.9 and $\beta = 1.1$. We can observe that for those instances the number of non-delayed parcels when applying Flex^{prio} is significantly higher compared to FCFS with 4.75% and 4.43%, respectively. Naturally, the performance of Flex^{prio} is much lower for instances where the conveyor capacities are not the bottleneck, such as those with $\sigma = 1.1$. This shows that allowing flexibility in the unloading process is a promising approach if the conveyor capacities are actually the limiting factor of the system.

7 Conclusions and outlook

In this paper, we considered a specific truck scheduling problem for parcel hubs where limited conveyor capacities and outbound deadlines are assumed. We presented an extension to the problem setting by incorporating controllable unloading speeds and formalise the problem with a mathematical model. As the problem is proven to be $\mathcal{NP}$-hard, we present a genetic algorithm to provide an efficient solution method. The model formulation and the genetic algorithm were investigated in a performance analysis using a problem-specific set of instances. Further, we investigated the applicability of the results in a real-world environment by implementing a simulation model that considers individual parcel interactions.

Based on our experimental results, we can state that a standard solver has difficulties to generate good solutions for the medium and large instance class. In contrast, the genetic algorithm was shown to be efficient for all instances. Our simulation study further showed that especially in problem settings with scarce conveyor capacities σ, high gate capacities β and high parcel heterogeneity α, incorporating controllable unloading speeds by integrating the results of the PHSP-LCC-flex has the potential of significantly increasing the number of non-tardy parcels compared to a simple FCFS policy.

For future research, it would be interesting to have a more detailed modelling approach with regards to its granularity that disaggregates the conveyors into several sorters. Even though the computational complexity would surely increase, a more detailed view on the hub might offer further insights. Another approach would be to utilise the simulation model in a combined simulation-optimisation procedure to estimate the objective function value. This would incorporate parcel interactions in the solution procedure and has the potential to improve the applicability of the generated truck schedules.

Funding Open Access funding enabled and organized by Projekt DEAL.

Data availability statement The test instances that support the findings of this study are available at https://doi.org/10.25835/0090537.

References

Apte UM, Viswanathan S (2000) Effective cross docking for improving distribution efficiencies. Int J Logist 3(3):291–302

Artigues C, Lopez P, Hait A (2013) The energy scheduling problem: industrial case-study and constraint propagation techniques. Int J Prod Econ 143(1):13–23

Boysen N (2010) Truck scheduling at zero-inventory cross docking terminals. Comput Oper Res 37(1):32–41

Boysen N, Fliedner M (2010) Cross dock scheduling: classification, literature review and research agenda. Omega 38(6):413–422

Boysen N, Briskorn D, Tschöke M (2013) Truck scheduling in cross-docking terminals with fixed outbound departures. OR Spectrum 35(2):479–504

Boysen N, Fedtke S, Weidinger F (2017) Truck scheduling in the postal service industry. Transp Sci 51(2):723–736

Carrera S, Chami K, Guimaraes R, et al (2008) Negotiation models for logistic plarform planning and scheduling. In: 11th international workshop on project management and scheduling, pp 43–46

Cheng T, Chen Z, Li CL (1996) Parallel-machine scheduling with controllable processing times. IIE Trans 28(2):177–180

Clausen U, Diekmann D, Pöting M et al (2017) Operating parcel transshipment terminals: a combined simulation and optimization approach. J Simul 11(1):2–10

Corsten H, Becker F, Salewski H (2020) Integrating truck and workforce scheduling in a cross-dock: analysis of different workforce coordination policies. J Bus Econ 90(2):207–237

Du J, Leung JYT (1990) Minimizing total tardiness on one machine is np-hard. Math Oper Res 15(3):483–495

Graham RL, Lawler EL, Lenstra JK, et al (1979) Optimization and approximation in deterministic sequencing and scheduling: a survey. In: Annals of discrete mathematics, vol 5. Elsevier, pp 287–326

Haneyah S, Schutten JM, Fikse K (2014) Throughput maximization of parcel sorter systems by scheduling inbound containers. In: Efficiency and innovation in logistics. Springer, pp 147–159

Holland JH (1975) Adaption in natural and artificial systems: an introductory analysis with application to biology, control, and artificial intelligence. The University of Michigan Press, Detroit

Jarrah AI, Johnson E, Neubert LC (2009) Large-scale, less-than-truckload service network design. Oper Res 57(3):609–625

Khir R, Erera A, Toriello A (2021) Two-stage sort planning for express parcel delivery. IISE Trans 53(12):1353–1368

Koné O, Artigues C, Lopez P et al (2011) Event-based milp models for resource-constrained project scheduling problems. Comput Oper Res 38(1):3–13

Ladier AL, Alpan G (2016) Cross-docking operations: current research versus industry practice. Omega 62:145–162

Li Y, Lim A, Rodrigues B (2004) Crossdocking-jit scheduling with time windows. J Oper Res Soc 55(12):1342–1351

McWilliams DL, McBride ME (2013) Exploring mathematical approximation for the time spans of transfer operations in parcel transshipment terminals. Comput Ind Eng 64(1):342–356

McWilliams DL, Stanfield PM, Geiger CD (2005) The parcel hub scheduling problem: a simulation-based solution approach. Comput Ind Eng 49(3):393–412

Mendes JJ, Gonçalves JF, Resende MG (2009) A random key based genetic algorithm for the resource constrained project scheduling problem. Comput Oper Res 36(1):92–109

Molavi D, Shahmardan A, Sajadieh MS (2018) Truck scheduling in a cross docking systems with fixed due dates and shipment sorting. Comput Ind Eng 117:29–40

Naber A, Kolisch R (2014) Mip models for resource-constrained project scheduling with flexible resource profiles. Eur J Oper Res 239(2):335–348

Nattaf M, Artigues C, Lopez P et al (2016) Energetic reasoning and mixed-integer linear programming for scheduling with a continuous resource and linear efficiency functions. OR Spectrum 38(2):459–492

Ou J, Hsu VN, Li CL (2010) Scheduling truck arrivals at an air cargo terminal. Prod Oper Manag 19(1):83–97

Qijun Q, Zhang Z, Song X et al (2009) Application research of cross docking logistics in food cold-chain logistics. In: 2009 international conference on information management. Innovation management and industrial engineering. IEEE, pp 236–240

Rothlauf F (2011) Design of modern heuristics: principles and application. Springer, Berlin

Selinka G, Franz A, Stolletz R (2016) Time-dependent performance approximation of truck handling operations at an air cargo terminal. Comput Oper Res 65:164–173

Serrano C, Delorme X, Dolgui A (2017) Scheduling of truck arrivals, truck departures and shop-floor operation in a cross-dock platform, based on trucks loading plans. Int J Prod Econ 194:102–112

Shabtay D, Steiner G (2007) A survey of scheduling with controllable processing times. Discret Appl Math 155(13):1643–1666

Stephan K, Boysen N (2011) Cross-docking. J Manag Control 22(1):129

Tadumadze G, Boysen N, Emde S et al (2019) Integrated truck and workforce scheduling to accelerate the unloading of trucks. Eur J Oper Res 278(1):343–362

Theophilus O, Dulebenets MA, Pasha J et al (2019) Truck scheduling at cross-docking terminals: a follow-up state-of-the-art review. Sustainability 11(19):5245

Tootkaleh SR, Ghomi SF, Sajadieh MS (2016) Cross dock scheduling with fixed outbound trucks departure times under substitution condition. Comput Ind Eng 92:50–56

Tripp C (2021) Dienstleisternetze in der distributionslogistik. In: Distributions-und handelslogistik. Springer, pp 359–421

Van Belle J, Valckenaers P, Cattrysse D (2012) Cross-docking: state of the art. Omega 40(6):827–846