Crowd-shipping of small parcels in a physical internet

Tal Raviv, Eyal Z. Tenzer
Industrial Engineering Department, Tel Aviv University, 69978 Tel Aviv, Israel
talraviv@eng.tau.ac.il, eyaltenzer@gmail.com

We envision a new logistic process for delivering parcels by crowdsourcing the tasks of the couriers. It is based on a network of automatic service points, which are used as a drop-off, pickup, and intermediate transfer points. The system offers the couriers monetary rewards for stopping by the service points and for transferring parcels between them during their regular trips. We present a stochastic dynamic program that yields an optimal parcel routing policy in polynomial time under some simplifying assumptions. In particular, this policy ignores the capacity constraints of the service points and the couriers. A heuristic policy that is based on the value table of the above dynamic program is devised to relax the simplifying assumptions and adapt it to realistic settings. The economic viability of the proposed method is demonstrated by an extensive simulation study that is based on realistic data of car journeys and small parcel shipments in a metropolitan area. Our experiment shows that even if a small fraction of the drivers is willing to participate, the average delivery time is few hours. Moreover, the rewards paid to the occasional couriers relatively to their engagement time is above the average hourly wage while the average cost of delivering a parcel is significantly lower than the price of parcel delivery service.

Key words: City Logistics, Crowdsourcing, Physical Internet, Shared Mobility, Parcel Delivery, Crowd-shipping, Automatic Parcel Lockers

History: First version - July 2018

1. Introduction

The rapid growth of e-commerce has caused a significant increase in the volume that passes through the parcel delivery industry. Recent advances in mobile computing technology have created new opportunities to redesign the delivery process to make it more efficient and sustainable. Our idea is to utilize the journeys that regular people make with their cars to ship small parcels.

Many authors, e.g., Deutsch and Golany (2017) and Faugere and Montreuil (2017), stress the importance of hyper-connected pickup-and-delivery locker network as a method to increase the efficiency and robustness of B2C delivery systems. Indeed, many courier companies have deployed such networks during the last decade, but we believe that they did not adapt their distribution process to exploit the potential of these networks.

Several organizations, such as Walmart, have recognized the opportunity that the crowd-shipment of small parcels may offer; for example, see Barr and Wohl (2013). Savelsbergh and Van Woensel (2016) note that the challenge of solving the online delivery routing problem in a

crowd-shipment environment is that the arrival of both the parcels and the drivers are stochastic processes.

We envision a novel logistic process for delivering small parcels by people who subscribe as occasional couriers (OCs). A network of service points (SPs) is deployed. Each SP is used as a location where parcels can be dropped off by the senders and picked up by the recipients. The SPs also serve as intermediate transfer points where parcels can be dropped off by an OC and picked up again later by another OC. The SP network constitutes a physical internet (PI) (see Montreuil, Meller, and Ballot (2010)). The SPs are strategically located at accessible sites such as gas stations and can be either automated (e.g., automatic parcel lockers) or manned. The OCs install a location-aware mobile application that offers them monetary rewards for transferring parcels between the SPs during their regular car journeys.

The design and operation of such a logistic network raise many challenges. For example, Bloch and Tzur (2018) studied an SP location problem in the crowd-shipping context. Other strategic decisions are related to the capacity of the SPs, the service level, the pricing policy, and the reward scheme for to the OCs. At the operational level, parcel routing decisions need to be made in real-time, i.e., which parcel should be offered to each OC and to which SP along her route it should be carried to. This paper focuses on providing solutions to these problems.

We devise a method to route the parcels in the system by determining the offers that the system should make to the OCs such that the the operational costs and some service level measures are optimized. Our method is based on a stochastic dynamic program that efficiently solves a simplified version of the problem. Next, the simplifying assumptions are relaxed, and heuristic methods that adapt the solution of the simplified model to more realistic settings are presented.

We conduct a simulation study using realistic data on the movement of people and parcels. The results of our numerical experiment support our hypothesis that such a system can provide a quick delivery service at a low operational cost. The system can manage a large volume of parcel traffic by utilizing the excess capacity of private cars in a small fraction of the journeys that the public already takes.

To demonstrate this concept from the perspective of the OC, let us describe the journey of a fictitious OC named Alice.

7:45 Alice enters her car and reports her destination to a dedicated smart-phone app.

7:51 A voice message is issued by the smart-phone: “You may stop in 300m and pick up three parcels. The reward will be 4 Euros”.

7:52 Alice stops by the SP and collects the parcels from lockers specified by the application. The doors of the lockers opene when the smart-phone is near them.

7:56 Alice is back on the road driving toward her workplace.

8:12 A voice message is issued by the smart-phone: "Dont forget to stop in 300m to drop off the three parcels".

8:13 Alice stops at the SP to drop off the parcels.

8:15 A voice message is issued by the smart-phone: "Thank you for delivering the three parcels. Your account has been credited by 4 Euros. You do not have any more parcels to deliver". Alice continues on her journey.

Several studies consider the possibility of crowdsourcing the delivery process of small parcels. Archetti, Savelsbergh, and Speranza (2016) develop a model in which a shipper can assign some of its delivery tasks to occasional drivers whose origin and destination nearly coincide with those of the parcels. Each driver is assigned a single parcel to be delivered directly from the shipping company to its ultimate destination.

Arslan et al. (2016) consider a new dynamic variant of the pickup-and-delivery problem with time windows where some of the tasks can be matched to a given set of ad-hoc drivers while a third party backup fleet completes the remaining tasks. An ad-hoc driver can stop multiple times along her route to pick up and deliver several parcels. Each parcel is picked up at its origin and dropped off at its ultimate destination. A static version of the problem is formulated as a matching problem with side constraints. The dynamic version is heuristically solved by solving the static problem with a rolling horizon.

Chen, Mes, and Schutten (2017) suggest using a method to harness the spare capacity of journeys that are made by private vehicles to deliver parcels. These scholars develop considered a door-to-door delivery scheme where parcels are assigned to planned trips with the possibility of a small detour. Parcels may be delivered by more than one OC, but the transfers between OCs requires time synchronization; i.e., the OCs have to meet at a specific time and location. The multi-driver multi-parcel matching problem was defined as the problem of assigning parcels to drivers and determining the transfer point so as to minimize the total cost. Parcels can either be delivered by OCs or by a third party courier company at a given, higher cost. The authors formulate a static version of the problem as an integer linear programming (ILP) problem and propose a heuristic algorithm that can be used to solve a dynamic version of the problem.

Lee, Kang, and Prabhu (2016) consider an Uber-like model for parcel delivery where the courier company uses sub-contractors from the crowd to deliver the parcels. Delivery requests are made to the system according to a Poisson process. Each parcel can be either admitted or rejected based on its expected profit. An execution sub-system solves the routing problems of the couriers so as to minimize the total travel cost net of delay and earliness penalties. A Markov decision process (MDP) is used to solve the parcel admission problem.

Voccia, Campbell, and Barrett (2017) introduce the same-day delivery problem where goods that need to be distributed to customers arrive at the depot according to a Poisson process. The objective is to maximize the expected number of requests that can be served on the same day using a dedicated fleet. The vehicles routing decisions are made while considering information about future requests.

Devari (2016) study the possibility of using journeys traveled by the friends of the recipient, based on information gathered from social networks, to perform the last mile delivery. A simulation study is used to the demonstrate that many of the parcels can be delivered by this means.

Chen et al. (2017) introduce a model for a city-wide parcel delivery system based on the journeys of taxis. Once a taxi driver receives a request from a passenger, the system tries to assign him parcels that are located at an SP near his origin to be carried to an SP located near his destination. Each parcel can be transferred to its destination in several legs by different drivers. The authors devise a two-step algorithm. First, a time-dependent routing table is constructed based on historical journey data. The expected delivery to each destination from each SP at each period of the day is calculated using this table. In real time, parcels are opportunistically assigned to drivers that can transfer them to locations with a shorter expected delivery time (net of the transfer time).

McInerney, Rogers, and Jennings (2013) present a crowdsourced parcel delivery model in the context of humanitarian logistics. These scholars develop a machine-learning method to predict the periodic mobility patterns of people based on data obtained from a cellular network and an optimization model to determine the assignment of parcels to OCs in such a manner that minimizes the total delay. The problem is solved using a MDP that takes advantage of the periodicity of the mobility patterns to reduce its state space. In this model, parcels are collected and dropped at SPs and can reach their destination in several legs.

Spiess and Florian (1989) present a model for assigning the passengers of a public transit system. It is assumed that each passenger minimizes her total expected travel time. The route of each passenger is dynamically updated based on the actual order of bus arrivals at the current station of the passenger. This optimization model is used to characterize the behavior of passengers to evaluate the service level provided by a given network and the timetable using a discrete event simulation. From a methodological point of view, the determination of the best potential routes for each passenger is similar to our parcel routing problem. Moreover, in both cases, to account for capacity constraints, some adaptations are made heuristically. However, there are some significant differences between these two problems. First, in our problem, there are capacity constraints on the SPs while the transit stations are assumed to have unlimited capacity. Second, the cost structure in our model is more intricate. In particular, the operator is charged for each stop of the OC and

for each parcel that she transports while in the transit system, all these costs are assumed to be sunk costs.

From a methodological perspective, the work that is most similar to this study is McInerney, Rogers, and Jennings (2013). However, our model makes the assignment decision online based on the actual journeys of OCs who are available for parcel delivery and provide information about their itinerary before starting their journey. Since our context is commercial, we consider the reward to the OCs and minimize a weighted sum of the delay and delivery costs while McInerney, Rogers, and Jennings (2013) ignore these costs. Our model considers the capacity of the OCs' vehicles and the SPs. Due to our proposed reward structure for the OCs, the consolidation of many shipments to the same OC is incentivized. Since in our study we consider an express service (typically same or next day delivery) in a dense metropolitan area, each parcel must be collected and dropped off by an OC in the same journey. In contrast, McInerney, Rogers, and Jennings (2013) consider a service that works in rural areas with low traffic and assume that a parcel can be with the OCs for several days.

Chen et al. (2017) propose a similar framework but make several simplifying assumptions that are not included in our settings. In particular we: 1) consider a rich reward structure for the OCs and explore the trade-off between the costs and the service level; 2) consider the capacity constraints of the OCs and SPs; 3) allow multi-stop journeys of the OCs; and 4) allow for the prioritization of parcels based on their seniority in the system and their priority class.

The main contribution of this paper is the introduction of a logistics business model that utilizes crowd-shipment, using the general public and a PI. We develop effective routing algorithms that can be adapted to realistic settings that support capacity constraints and a complex incentive structure for the OCs. In an extensive simulation study, we demonstrate the economic viability of such a marketplace.

In Section 2, we provide a formal definition of the parcel routing problem in a crowd-sourced PI. In Section 3, we present an efficient stochastic dynamic algorithm that generates an optimal policy for a simplified version of the problem and introduce heuristic methods to adapt this policy to realistic settings. In Section 4, we present and analyze a numerical study in which realistic systems are simulated, and the parcels are routed according to different policies. Since this is one of the first studies to consider a crowd-shipping PI, many directions for further research are discussed in Section 5.

2. Problem definition

In this section, we provide a formal definition of the parcel routing problem in a crowd-sourced PI. A network of N automatic service points (SPs) is given. Each SP_i has a capacity of C_i lockers

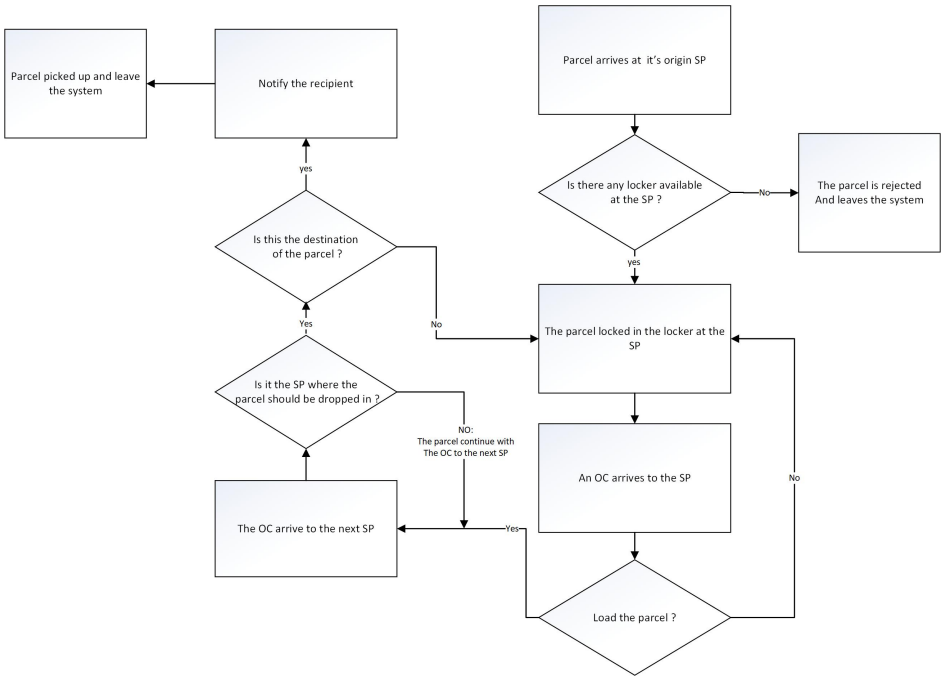

Figure 1 The parcel flow in the system.

that are identical in sizes. The travel time between each pair of SPs (SP_i and SP_j) is denoted by t_{ij}. A Poisson arrival process of OCs is given, and each OC arrives at a predefined sequence of SPs. Each such sequence is characterized by its rate. The system determines when and where each OC is asked to stop and which parcels she should collect and drop off at each SP in her sequence. We assume that whenever asked, an OC is willing to perform the tasks requested by the system subject to the capacity constraint of her vehicle and without deviating or delaying her pre-planned journey. The reward to the OC is calculated as follows: each stop is rewarded by a fixed amount, and an additional reward is paid for transferring each parcel based on its pickup and drop-off SPs.

A Poisson arrival process of parcels to each origin and destination pair of SPs is given. It is assumed that the parcels are identical in size. Once a parcel arrives at its destination, the recipient is notified. The parcel is collected after some identically distributed time. The objective is to route the parcels so as to minimize both the weighted sum of the total time that parcels spend in the system before reaching their destination and the payments to the OCs.

In Figure 1 we present the flow of a parcel through the system. A parcel enters the system at its origin SP. If there is an available locker at this SP, the parcel is admitted and otherwise is rejected from the system. When an OC with some available capacity arrives at the SP, the system determines whether the parcel should be carried by her to some other SP that has available capacity in her sequence. We refer to this SP as the intermediate destination of the parcel. A locker at this SP is reserved for the parcel and became unavailable for other parcels. The parcel is then

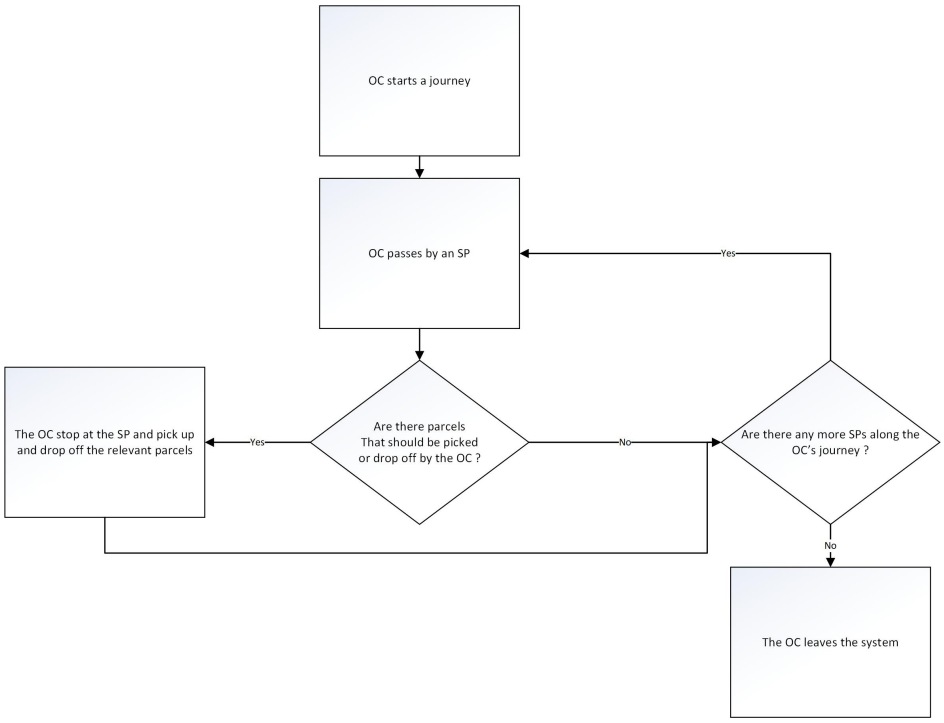

Figure 2 The OC flow in the system.

transported by the OC until she arrives at the intermediate destination and drops it off there. If the intermediate destination coincides with the (final) destination of the parcel, then the recipient is notified. The parcel is then collected by the recipient, and the locker became available again.

In Figure 2 we present the flow of the OC through the system. When the OC starts a journey, she uses her mobile application to notify the system about her planned route. Based on this route, a sequence of SPs is derived. When the OC passes by an SP,the system informs her whether to stop or not and which parcels should be collected or dropped off. After passing by the last SP in the sequence, the OC leaves the system and is rewarded according to her activity.

3. Methodology

In this section, we present a solution method for the parcel routing problem. Our method is based on an exact solution of a simplified version of the problem that ignores the capacity constraints of the SPs and OCs and assumes that the incentives for the OCs are determined separately for each parcel (i.e., there is no additional payment for stopping at an SP). Under these assumptions, the problem is separable and can be solved independently for each parcel destination. We use the solution of the simplified version of the problem to derive a routing policy that is compatible with the capacity constraints (referred to as the greedy policy). Next, we introduce a locally optimal policy that takes into account a more reasonable compensation policy that also considers the effort of stopping at each SP. Finally, we enhance both policies by introducing a variant that assigns a

penalty that is convex in the seniority of parcels in the system. Such a penalty reduces the chance that some parcels will spend a very long time in the system.

In Section 3.1, we devise an exact solution method for the simplified problem. In Section 3.2, we present our greedy routing policy, and in Section 3.3 it is extended to a locally optimal policy. Finally, in Section 3.4, we introduce a variant of the above routing policies, which incorporates a non-linear delay cost component.

3.1. The simplified model

In this section we present an exact and efficient solution method for a simplified version of the problem where the following assumptions are made:

1. The capacity constraints of both the OCs and SPs are non-binding.
2. The reward paid to the OC is determined separately for each parcel based on its pickup and drop-off SPs, regardless of whether other parcels may be collected or dropped off at these SPs by the same OC. That is, there is no special compensation given to an OC for stopping at an SP.
3. The delay cost of a parcel is linear in the time that the parcel spends in the system, meaning that there is a constant penalty charged per parcel from the moment it is dropped off by the sender until the moment that it arrives at its destination SP. The delay penalty per time unit is identical for all the parcels in the system.
4. The handling and stopover time (engagement time) of the OCs at each SP is zero regardless of the number parcels that are picked up or dropped off.
5. The OCs arrive at the beginning of their routes and the parcels arrive to their origins according to time homogeneous Poison processes.
6. The travel times of the OCs between SPs along their routes is known and deterministic.
7. The sequence of the remaining SPs at which the OC is expected to visit after the current one, become known at each SP once the vehicle arrives there. That is, the model is oblivious to information about OCs before their arrival even if they have already visited other SPs in the network.

Under these simplifying assumptions, the routing decisions are separable by parcel, meaning that the policy regarding a parcel is merely determined by its current location and its final destination. The policy derived from this solution is used as the basis for the heuristic model presented below for a more realistic case where assumptions 1-4 above are relaxed.

Assumptions 5 and 6 somewhat divert our policy from the optimum and require further refinements of the model. However, we believe that even under these simplifications, the demonstration of the applicability of the proposed logistic process is valid.

Assumption 7 is sensible because the arrival process of OCs do not represent all the drivers and not even all the registered OCs that follow the sequence but rather only those that are ready to deliver parcels during their current journey. This fact is known only after the OC actually stops at an SP. Moreover, such solutions are likely to be more robust because they do not depend on the future arrivals of OCs that may decide to divert from their planned journey or are delayed by traffic congestion.

Consider the sub-sequence of the remaining SPs in a journey of an OC after visiting a particular SP. Such sub-sequences of several different journeys may coincide, and we refer to them jointly as a *suffix*. The arrival process to a suffix is the unification of several Poisson processes and thus a Poisson process by itself. For example consider the following three SP sequences $1 \rightarrow 2 \rightarrow 3$, $4 \rightarrow 2 \rightarrow 3$, and $2 \rightarrow 3$ that each have an arrival rate of 2 OCs. The arrival rate to the unified suffix $2 \rightarrow 3$ is 6. We use the notion of arrival process to suffixes in the presentation of Algorithm 1 below. Note that while theoretically, the number of suffixes may be exponential in the number of SPs, in practice, when observing the actual movement patterns of drivers, the actual number of such unique suffixes with a non-negligible arrival rate is limited.

A policy for routing a parcel in the network prescribes decisions that should be made about a parcel whenever an OC passes by its current SP. The decisions are whether or not the OC should pick up the parcel and if so, to which SP along the remaining part of her journey the parcel should be transferred to. Note that under the above assumptions these two decisions can be made simultaneously. The same type of decisions should be made regarding each of the parcels at the SP based on their destinations and the SPs in the suffix of the OC.

The above policy can be encoded in a compact way using a value table that represents the expected remaining cost for delivering each parcel from any SP in the system to any possible destination SP. If the immediate cost of delivering a parcel with an OC to a one of SPs along her sequence plus the expected remaining cost at this location is less than the expected cost at the current location, then the parcel is picked up by the OC. If the OC has several such cost-saving SPs along her sequence, then the one that maximizes the savings is selected. Note that there is no need to store the optimal decision with respect to any of the (possibly large number of) OC suffixes of each SP since the decision can be made easily by comparing the value at the current location and the value at each SP in the suffix plus the cost of transferring the parcel to it.

Next, we introduce some additional notations that are needed for the presentation of the parcel routing optimization model:

n The number of SPs in the network. We assume, without a loss of generality, that SP_n is the destination of the parcel.

t_{ij} The travel time of an OC from SP_i to SP_j.

r_{ij} The reward payable to the OC for transferring a single parcel from SP_i to SP_i.

h The delay cost of a parcel per time unit

$\mathcal{C}_i$ A collection of the OC suffixes that begin at SP_i. Each member of this collection represents a unique ordered set (sub-sequence) of SPs that are visited by some OCs that pass by SP_i, regardless of the SPs that they visited prior to arriving at SP_i. For convenience we include SP_i as the first stop in each such suffix.

$\mathcal{C}_i^k$ The k^{th} OC suffix in $\mathcal{C}_i$.

λ_i^k The arrival rate of the OCs (who are willing to actively serve as such) to the suffix $\mathcal{C}_i^k$.

We formulate the optimal routing problem as a stochastic dynamic program (DP). The states in this DP are divided into two categories: SP states and OC states. The SP state represents the location of a parcel while it is idle at an SP and no OC passes by it. The OC states are related to a parcel at the moment when an OC with a particular suffix passes by its SP. The action set of the OC states is $\mathcal{C}_i^k$, which represents SPs to which the parcel should be transferred. Recall that $\mathcal{C}_i^k$ include SP_i which represents leaving the parcel at its current SP. The action sets of the SP states are singletons (the parcel waits until an OC arrives at the SP). We include the SP states to simplify the calculation of the values of the OC states. The relation between the values of these states are defined by the Bellman equations, following (1) and (2).

$$v^*(S_i^k) = \min_{j \in \mathcal{C}_i^k} \{r_{ij} + ht_{ij} + v^*(S_j^0)\}. \tag{1}$$

$$v^*(S_i^0) = \frac{h}{\sum_{k \in K_i^*} \lambda_i^k} + \sum_{k \in K_i^*} \frac{\lambda_i^k}{\sum_{l \in K_i^*} \lambda_i^l} v^*(S_i^k) \tag{2}$$

where

$$K_i^* \equiv \{k \in \mathcal{C}_i : \arg\min_{j \in \mathcal{C}_i^k} \{r_{ij} + ht_{ij} + v^*(S_j^0)\} \setminus \{i\} \neq \emptyset\}.$$

The boundary conditions are defined by value of the SP state at the destination.

$$v^*(S_n^0) = 0.$$

The value of an OC state in (1) represents the expected remaining cost for a parcel located at SP_i at the moment when an OC with suffix $\mathcal{C}_i^k$ passes by it. SP (j) in this suffix, is selected such that the total expected cost is minimized. This cost consists of the reward paid to the OC (r_{ij}), the delay cost while the parcel is traveling with the OC (ht_{ij}), and the future expected remaining cost at the drop-off location $[v(S_j^0)]$. Since $r_{ii} = 0$ and $t_{ii} = 0$, if the optimal decision is to leave the parcel at its current SP (i), then suffix k has no effect on the parcel. In such a case, $v(S_i^k) = v(S_i^0)$.

Based on the solution described in (1), we define the set K_i^* as the collection of all the suffixes to which the parcel is transferred from SP_i by the OC. The values of the SP states are calculated in (2) as the sum of the delay cost multiplied by the expected waiting time until the next relevant OC and the expected value of the parcel being idle at the SP for which it will be transferred.

Alternative way to describe K_i^*, which is based on the values of the OC states, $v^*(S_i^k)$, is

$$K_i^* = \underset{K \subset \mathcal{C}_i}{\arg\min} \left\{ \frac{h}{\sum_{k \in K} \lambda_i^k} + \sum_{k \in K_i} \frac{\lambda_i^k}{\sum_{l \in K} \lambda_i^l} v^*(S_i^k) \right\}. \tag{3}$$

Note that K_i^* is the set of suffixes that serves a parcel at SP_i (with a particular destination) in an optimal solution. Indeed, in (3), a subset of the suffixes that minimizes the expected SP state is selected. Recall that the value of an SP state consists of the following: (a) the expected delay cost, which is incurred until the next arrival of an OC that will transfer it to a better SP and (b) the expected value of the arrival of the next such OC.

To solve equations (1) and (2) simultaneously, one should determine the order by which the states can be evaluated, such that all the relevant values of the other states are available when needed. We accomplish this by defining a set of SP states with known values and initializing it with the destination SP_n, whose value is defined by the boundary condition. The values of the rest of the SP states are initialized as very large numbers. At each iteration, the optimal policy and values are tentatively calculated for all the yet unknown OC states, based on the current SP states values using equation (1). Then, the unknown SP state values are tentatively calculated using (2) and the current OC state values.

In Proposition 1, we show that the SP states among the unknown states with the smallest calculated values admit their actual value in the optimal solution and thus can be added to the set of known SPs states. The process is iterated until the values of all the states are known. With respect to an optimal solution v^* of DP (1)-(2) and a particular SP_g, $g \neq n$, let $Q_g = \{j \in N : v^*(S_j^0) \geq v^*(S_g^0)\}$ and $\bar{Q}_g \equiv N \setminus Q_g$. Consider a tentative solution for the DP with values denoted by v such that $v(S_i^0) = v^*(S_i^0)$ for all $i \in \bar{Q}_g$ and $v(S_i^0) > v^*(S_i^0)$ for all $i \in Q_g$. In addition, the tentative values $v(S_i^k)$ for $k > 0$ are determined as in (1), where $v(S_i^0)$ is used instead of $v^*(S_i^0)$. That is,

$$v(S_i^k) = \min_{j \in \mathcal{C}_i^k} \{r_{ij} + h t_{ij} + v(S_j^0)\}. \tag{4}$$

Next, we define

$$v'(S_i^0) = \min_{K \subset \mathcal{C}_i} \left\{ \frac{h}{\sum_{k \in K} \lambda_i^k} + \sum_{k \in K} \frac{\lambda_i^k}{\sum_{l \in K} \lambda_i^l} v(S_i^k) \right\}, \tag{5}$$

and

$$K_i = \underset{K \subset \mathcal{C}_i}{\arg\min} \left\{ \frac{h}{\sum_{k \in K} \lambda_i^k} + \sum_{k \in K_i} \frac{\lambda_i^k}{\sum_{l \in K} \lambda_i^l} v(S_i^k) \right\}.$$

Proposition 1 *For v and v' defined as above, $v'(S_g^0) = v^*(S_g^0)$.*

Proof: We begin by proving the following claims

1. $v(S_i^k) \geq v^*(S_i^k)$ for all $i \in Q_g$ and $v(S_i^k) = v^*(S_i^k)$ for all $i \in \bar{Q}_g$
2. For all suffixes $k \in K_g^*$, $v(S_g^k) = v^*(S_g^k)$.
3. $K_g = K_g^*$

Claim 1: The values of the OC states are calculated in (4) based on the problem parameters and the values of the SP states; $v(S_i^0)$. $v(S_i^k)$ is non decreasing in $v(S_i^0)$. By our construction of v, $v(S_i^0) \geq v^*(S_i^0)$; thus, the obtained OC state values satisfy $v(S_i^k) \geq v^*(S_i^k)$. Now, if the optimal decision at state S_g^k is to send the parcel with the OC, then the parcel will be transferred to $SP_{i'}$ such that $v(S_{i'}^0) < v(S_g^0)$. In this case, $i' \in \bar{Q}_g$, which implies that $v(S_{i'}^0) = v^*(S_{i'}^0)$ and thus equation (4) coincides with the optimality (1). That is, $v(S_i^k) = v^*(S_i^k)$. Note that in the minimization problem solved in (4), some SPs that are not $\bar{Q}_g$ may be considered but these SPs cannot be the optimal solution both in (4) and (1) and thus do not affect the value of $v(S_i^k)$.

Claim 2: Consider a suffix $k \in \mathcal{C}_g$ such that $k \in K_g^*$. Such a suffix includeS at least one SP in $\bar{Q}_g$ and the optimal decision is to transfer the parcel to some SP in $\bar{Q}_g \cap \mathcal{C}_g^k$. The last statement is true because in an optimal solution, the parcels are only transferred to SPs with lower $v^*(S_i^0)$. In our tentative solution, $v(S_i^0) \geq v^*(S_i^0)$ for $i \in Q_g$, while $v(S_i^0) = v^*(S_i^0)$ for $i \in \bar{Q}_g$, and $v^*(S_i^0) \geq v^*(S_j^0)$ for all $i \in Q_g, j \in \bar{Q}_g$. Therefore, $v(S_i^0) \geq v^*(S_j^0)$ for all $i \in Q_g, j \in \bar{Q}_g$. Now, by (4), $v(S_g^k) = v^*(S_g^k)$ for $k \in K_g$ because the SP that minimizes this expression must be one from $\bar{Q}_g$.

Claim 3: To prove that $K_g = K_g^*$, note that according to Claim 1, for subset $K \subseteq \mathcal{C}_g$ the value of

$$\frac{h}{\sum_{k\in K}\lambda_i^k} + \sum_{k\in K}\frac{\lambda_i^k}{\sum_{l\in K}\lambda_i^l} v(S_g^k) \geq \frac{h}{\sum_{k\in K}\lambda_i^k} + \sum_{k\in K}\frac{\lambda_i^k}{\sum_{l\in K}\lambda_i^l} v^*(S_g^k)$$

but by 2, we know that

$$\frac{h}{\sum_{k\in K_g^*}\lambda_i^k} + \sum_{k\in K}\frac{\lambda_i^k}{\sum_{l\in K_g^*}\lambda_i^l} v(S_g^k) = \frac{h}{\sum_{k\in K_g^*}\lambda_i^k} + \sum_{k\in K_g^*}\frac{\lambda_i^k}{\sum_{l\in K_g^*}\lambda_i^l} v^*(S_g^k)$$

Therefore, K_g^* is an optimal solution of (5) and thus $K_g = K_g^*$.

Finally, according to Claim 3, we know that $K_g = K_g^*$, and according to Claim 2, we know that $v(S_g^k) = v^*(S_g^k)$ for all $k \in K_g$. Therefore, for S_g^0, equation (5) coincides with the optimality equation (2) and thus $v(S_g^0) = v^*(S_g^0)$. $\square$

The calculation of $v'(S_i^0)$ [and equivalently, $v^*(S_g^0)$] using (5) requires finding a subset $K_i \subset \mathcal{C}_i$ that minimizes this expression. This can be accomplished in polynomial time using the characterization of the optimal solution provided by Proposition 2 below.

Proposition 2 *There exists an optimal solution $K \subset \{1, \ldots, \mathcal{C}_i^k\}$ of (5) such that K consists of the indexes of the suffix with the smallest $v'(S_i^k)$ among the suffixes in $\mathcal{C}_i^k$*

In order to prove Proposition 2, we first introduce the following two lemmas:

Lemma 1 *Consider a subset of indexes of suffixes $K \subset \mathcal{C}_i$ and an additional sequence index $\bar{k} \in \mathcal{C}_i \setminus K$; then, the solution of the following optimization problem is*

$$\min_{x\in[0,1]} \left\{ \frac{h}{x\lambda_i^{\bar{k}} + \sum_{k\in K}\lambda_i^k} + \sum_{k\in K} \frac{\lambda_i^k}{x\lambda_i^{\bar{k}} + \sum_{l\in K}\lambda_i^l} v'(S_i^k) + \frac{x\lambda_i^{\bar{k}}}{x\lambda_i^{\bar{k}} + \sum_{l\in K}\lambda_i^l} v'(S_i^{\bar{k}}) \right\}. \tag{6}$$

is either $x=0$, $x=1$ or $x\in[0,1]$.

Proof: The proof of this claim follows from the fact that the sign of the first partial derivative of the expression minimized in (6) with respect to x is independent of the value x and thus is monotone. Clearly, if the result is positive, then the minimum is obtained at $x=0$, but if it is negative at $x=1$ or if it is zero, then any value of x is optimal. Let us first simplify the expression by using the following notation $\Lambda = \sum_{k\in K}\lambda_i^k$, $A = \sum_{k\in K}\lambda_i^k v'(S_i^k)$, $\lambda = \lambda_i^{\bar{k}}$ and $v = v'(S_i^{\bar{k}})$. With this new notation we can rewrite the function that is minimized in (6) as

$$f(x) = \frac{h}{x\lambda+\Lambda} + \frac{A + x\lambda v}{x\lambda+\Lambda}.$$

Next,

$$f'(x) = -\frac{\lambda(A+\lambda v x)}{(\Lambda+\lambda x)^2} - \frac{h\lambda}{(\Lambda+\lambda x)^2} + \frac{\lambda v}{\Lambda+\lambda x}$$

Next, by solving $f'(x) > 0$,

$$-\frac{\lambda(A+\lambda v x)}{(\Lambda+\lambda x)^2} - \frac{h\lambda}{(\Lambda+\lambda x)^2} + \frac{\lambda v}{\Lambda+\lambda x} > 0$$

$$\frac{\lambda v}{\Lambda+\lambda x} > \frac{\lambda(A+\lambda v x)}{(\Lambda+\lambda x)^2} + \frac{h\lambda}{(\Lambda+\lambda x)^2}$$

$$\frac{v}{\Lambda+\lambda x} > \frac{(A+\lambda v x)}{(\Lambda+\lambda x)^2} + \frac{h}{(\Lambda+\lambda x)^2}$$

$$(\Lambda+\lambda x)v > A + \lambda v x + h$$

$$\Lambda v - A - h > 0$$

That is, the sign of $f'(x)$ independent of x, meaning that $f(x)$ is monotone as we claimed above. □

Lemma 2 *Consider the case when the set $\mathcal{C}_i$ contains two suffixes $\mathcal{C}_i^{k_1}$ and $\mathcal{C}_i^{k_2}$ such that $v'(S_i^{k_1}) = v'(S_i^{k_2})$; then, either there exists an optimal solution $K \subset \{1,...,|C_i|\}$ of (5) in which both k_1 and k_2 are contained, or there exists one in which neither of the two is included.*

Proof: Consider a set $K \subset \{1, ..., |C_i|\}$ such that $k_1 \notin K$ and $k_2 \in K$. We can write the value of this solution as

$$\frac{h}{x(\lambda_i^{k_1} + \lambda_i^{k_2}) + \sum_{k \in K} \lambda_i^k} + \sum_{k \in K} \frac{\lambda_i^k}{x(\lambda_i^{k_1} + \lambda_i^{k_2}) + \sum_{l \in K} \lambda_i^l} v'(S_i^k) + \frac{x(\lambda_i^{k_1} + \lambda_i^{k_2})}{x(\lambda_i^{k_1} + \lambda_i^{k_2}) + \sum_{l \in K} \lambda_i^l} v'(S_i^{\bar{k}}) \quad (7)$$

when $x = \frac{\lambda_i^{k_2}}{\lambda_i^{k_1} + \lambda_i^{k_2}}$

However, by Lemma 1, we know that (7) is minimized over the interval $x \in [0, 1]$ when either $x = 0$ or $x = 1$. The former is the value of the solution $K \setminus \{k_2\}$, and the latter is the value of the solution $K \cup \{k_1\}$. □

Next we are ready to prove Proposition 2:

Proof: Assume by contradiction that the claim is wrong and let K be an optimal solution (2) with a minimal number of suffixes that violates it, i.e., minimal number of suffixes that are not included in K, while the suffixes with larger values are included in this set. Consider suffix $k_1 \notin K$ and suffix $k_2 \in K$ such that $v(S_i^{k_1}) < v(S_i^{k_2})$. Next, let us construct an auxiliary instance of the problem in which suffix k_1 is replaced by k_1' such that $\lambda_{k_1'} = \lambda_{k_1}$ but $v(S_i^{k_1'}) = v(S_i^{k_2})$. Clearly the value of the optimal solution of this modified problem cannot be lower (better) then the solution of the original problem since we increased the value of one of the suffixes. However, we know that the solution K for the auxiliary problem admits the same value for the original problem, and thus, it must be optimal for the auxiliary problem as well.By Lemma 2 either $K \cup \{k_1'\}$ or $K \setminus \{k_2\}$ are also optimal for the auxiliary problem. Finally, the values of the solutions for the original problem are not higher (worse) then the value for the auxiliary one. Therefore, we could have an optimal solution with a smaller number of suffixes that violate the claim of the proposition, which contradicts the definition of K. □

Using the results of Proposition 1 and Proposition 2, we design a Dijkstra-like algorithm that solves the dynamic program (1) and (2) in polynomial time. See Algorithm 1.

In lines 1-3 of Algorithm 1, the values of the SP states are initialized, and the set Q, which includes the yet to be calculated SP states, is constructed. The main loop of the algorithm repeats a set of operations described below until Q is empty; that is, the values of all the SP states are known. In line 5, the values of all the OC states are calculated based on the current value of the SP states. Next, in lines 6-9, the tentative values of all the unknown SP states are calculated based on the current values of the OC states. In lines 10-11, the SPs with the smallest tentative value among the SP states with unknown values are selected and removed from the set of unknown SP-states. In line 12, these tentative values are registered as the actual values of the selected SP states.

The complexity of the algorithm is dominated by the loop in lines 4-13 that iterates through the SP states and the nested loop in lines 6-9. In the inner loop, the set of $O(m)$ suffixes is partitioned

Let $v(S_n^0) = 0$;
Let $Q = \{1, \ldots, N\} \setminus \{n\}$;
for $i \in Q$ **do** Let $v(S_i^0) = \infty$;
while $Q \neq \emptyset$ **do**
  for $i \in Q, k = 1, \ldots, |\mathcal{C}_i|$ **do** $v(S_i^k) = \min_{j \in \mathcal{C}_i^k} \{ht_{ij} + r_{ij} + v(S_j^0)\}$;
  for $i \in Q$ **do**
    Sort the suffixes in $\mathcal{C}_i$ in non-decreasing order of $v(S_i^k)$;
    Let $v'(S_i^0) = \min_{k' \in \{1,\ldots,|\mathcal{C}_i|\}} \left\{ \frac{h}{\sum_{k=1}^{k'} \lambda_i^{[k]}} + \sum_{k=1}^{k'} \frac{\lambda_i^{[k]}}{\sum_{l=1}^{k'} \lambda_i^{[l]}} v(S_i^{[k]}) \right\}$;
  end
  Let $I_{min} = \arg\min_{i \in Q} v'(S_i^0)$;
  Let $Q = Q \setminus I_{min}$;
  for $i \in I_{min}$ **do** Let $v(S_i^0) = v'(S_i^0)$;
end

Algorithm 1: Dijkstra-like algorithm for the parcel routing problem

to $O(n)$ subsets, and each one is sorted separately, which no more difficult than sorting the entire set. Therefore, the complexity of the inner loop is $O(m \log m)$. The entire process is repeated up to $n-1$ times; thus, the overall complexity is $O(nm \log m)$. Recall that the algorithm calculates the value of all the states for a given destination. Therefore, calculating the entire value table for all the possible destinations can be accomplished in $O(n^2 m \log m)$.

The output of the DP algorithm consists of the values of both the SP and OC states. However, for the implementation of the routing policy, we suggest saving only the values of the SP states for each pair of current location and destination. The routing decision can then be made online by quickly solving the relevant optimality equation (1) each time an OC is available near an SP. Such a practice allows the model to react (heuristically) to situations that were not anticipated when the values were calculated, e.g., an OC with a sequence that was not observed in the historical data used to forecast the arrival rate of the OCs.

3.2. Greedy loading policy

In this section, and those that follow, we show how the optimal policy of the simplified model can be heuristically applied in realistic settings where most of the assumptions of this model are relaxed. The idea of the greedy algorithm is to select the most attractive parcels that the OC can serve while enforcing her capacity constraint as well as the capacity constraint of the SPs on the rest of her journey.

Let $L = \emptyset$;
while $|L| \leq C$ **do**
  for $p \in P$ **do**
    Calculate j_p^* and Δ_{i,j_p^*} ;
    if $\Delta_{i,j_p^*} \leq 0$ **then** Let $P = P \setminus \{p\}$;
  end
  if $P = \emptyset$ **then** Break;
  Let p be the parcel with the highest Δ_{i,j_p^*} Let $L = L \cup \{p\}$;
  Update the remaining reservable capacity of $SP_{j_p^*}$;
end

Algorithm 2: Greedy policy for loading and unloading parcels

Let v_{ij} be the value of a parcel with destination j that is located at SP_i under the simplified model as pre-calculated by Algorithm 1.

Consider a vehicle that passes by SP_i with the available capacity C and let S be the set of SPs in the suffix of the OC with available capacity including SP_i (which is included even if it is full). Note that SP_i is always contained in S and thus it cannot be an empty set. Let P be the set of parcels currently located at SP_i and let $d(p)$ be the ultimate destination of each parcel $p \in P$. For each such parcel we find the $SP_{j_p^*}$ such that

$$j_p^* \in \arg\max_{j \in S} = \{v_{i,d(p)} - v_{j,d(p)} - r_{ij} - ht_{ij}\}.$$

We define $\Delta_{i,j_p^*} = v_{i,d(p)} - v_{j_p^*,d(p)} - r_{ij_p^*} - ht_{ij_p^*}$. Note that it is always possible to leave a parcel at its current SP;therefore, $\Delta_{i,j_p^*} \geq 0$. Our greedy algorithm iteratively selects the parcel with maximal Δ_{i,j_p^*} among the remaining parcels and assigns it to be transported by the OC to $SP_{j_p^*}$. This process is repeated until the capacity of the OC is exploited, or there are no more parcels with a positive Δ_{i,j_p^*} in SP_i. The set of parcels that should be transported by the OC is returned in L. Note that the algorithm recalculates j_p^* and Δ_{i,j_p^*} at each iteration since it may be affected by the availability of the SPs. A pseudo-code of this procedure is presented in Algorithm 2. The complexity of this algorithm is $O(C|P||S|)$; in practice, since $|S|$ and C are relativity small, the decision is made in a small fraction of a second.

3.3. Locally optimal loading policy

The greedy policy ignores the stopping cost and bases the loading decisions on the handling cost of each parcel separately. In this section, we introduce a loading decision mechanism that considers the stopping cost, i.e., the rewards that are paid to the OCs for each stop along their journey regardless

of the number of parcels they handle. Such a reward scheme is likely to increase consolidation of parcels and economize the delivery process.

Under the locally optimal loading policy, an optimization problem is solved each time an OC approaches an SP to decide whether she should stop at the SP, which parcels she should pick up and which other SPs she should stop at on her journey. All these decisions are made to optimize the total value gain (δ_p) of the transported parcels net of handling and stopping payments and considering the stopping decisions that were already made due to parcels that were picked up at previous SPs. Note that previous stopping decisions are not reconsidered or changed; therefore, the costs due to these decisions are regarded as sunk costs. We formulate this problem as a binary linear program using the following notations:

Parameters

SP_i The next SP in the OC's journey

S The set of all the SPs in the OC's journey after visiting SP_i. If this set is empty, there is no point in picking up a parcel at SP_i.

F The set of SPs along the remaining journey (a subset of S) at which the OC is already committed to stop at due to the parcels that are already on board.

I An indicator that equals 0 if the OC is already committed to stop at SP_i and 1, otherwise.

C The remaining capacity of the OC after dropping off the parcels destinated to SP_i but before potentially picking up any new parcel at this location.

R_j The available capacity of SP_j ($j \in S$). Recall that a locker is available if it is empty and not reserved for other parcels.

$d(p)$ The final SP destination of parcel p.

Δ_{pj} The potential net value gain from transferring parcel p from SP_i to SP_j is defined as

$$\Delta_{pj} = v_{i,d(p)} - v_{j,d(p)} - r_{ij} - ht_{ij}.$$

Note that this value gain is net of the handling and delay costs but not of the stopping costs at the SPs.

P The set of parcels located at SP_i with $\Delta_{pj} > 0$ for some $j \in S$. Note that these are the only parcels that should be considered for transfer using an OC that is about to visit the SPs in S.

K The reward paid to the OC for stopping at each SP regardless of the number of parcels picked up or dropped off there (the stopping cost).

Decision variables

x_{pj} A binary variable that equals "1" if parcel $p \in P$ is transfered by the OC to SP_j ($j \in S$).

y_j A binary variable that equals "1" if the OC stops at SP_j ($j \in S \setminus F$).

$$\max \sum_{p \in P} \sum_{j \in S} x_{pj} \Delta_{pj} - K \left(I + \sum_{j \in S \setminus F} y_j \right) \tag{8}$$

subject to

$$\sum_{j \in S} x_{pj} \leq 1 \qquad \forall p \in P \tag{9}$$

$$\sum_{p \in P} \sum_{j \in S} x_{pj} \leq C \tag{10}$$

$$\sum_{p \in P} x_{pj} \leq R_j \qquad \forall j \in S \tag{11}$$

$$x_{pj} \leq y_j \qquad \forall p \in P, j \in S \setminus F \tag{12}$$

$$x_{pj} \in \{0,1\} \qquad \forall p \in P, j \in S \tag{13}$$

$$y_j \in \{0,1\} \qquad \forall j \in S \setminus F \tag{14}$$

In the objective function (8) the total value gain net of all the cost components is maximized, assuming that the OC will stop at SP_i. If $I = 0$, that is the OC is already committed to stop at SP_i, then the stopping cost at this SP is omitted; otherwise, the stopping cost is reduced from the net gain. Constraint (9) stipulates that each parcel is transferred to at most one of the SPs along the remainder of the OC's journey. Constraint (10) limits the number of the parcels that are picked up to the available capacity of the OC's vehicle. Constraint (11) limits the number of parcels that are transferred to each SP to its available capacity. Constraint (12) requires that parcels are transferred only to SPs where the OC stops. Constraints (13) and (14) define the domain of the decision variables.

If the optimal value of the program (8)-(14) is positive, it worth calling the OC the stop at the current SP and ask her to pick the parcels as prescribed by the values of x_{pj}. The program is solved each time an OC passes by an SP unless this is the last SP in the journey, or she is already at full capacity.

While the complexity status of the local loading problem is not clear, in practice, we can solve reasonably large instances of the above problem in a fraction of a second using a commercial solver. Thus, it can be embedded in the system and used in real-time. This phenomenon is not surprising

since, for a fixed value of y_i, the problem is equivalent to the bipartite matching problem. Note that the dimension of the y variable is the number of SPs in the suffix of the OC for which stopping decisions have not yet been made. In practice, this number is very seldom more than four or five.

3.4. Value inflation protocol

We demonstrate in Section 4 that the greedy policy and the locally optimal loading policy work very well in terms of average delivery time. However, a small fraction of the parcels has a very long delivery time and even seems to "get stuck" in the system. For example, if a parcel arrives at an SP that is highly accessible to its destination (i.e., with a small value) but the capacities of the OCs are binding, newer parcels with higher values may be repeatedly prioritized over older parcels. Indeed, the value gain is not affected by the seniority of the parcel in the system.

To reduce the variability of the delivery time, we propose to inflate the value of parcels over time to increase the value gain from transferring them toward their destination. Let $\mathcal{I}(\tau)$ be a monotone increasing function, which is referred to as the *value inflation protocol*. Let $\tilde{v}_{ij}(\tau) = \mathcal{I}(\tau)v_{ij}$, where τ represents the total time that the parcel has already spent in the system. These inflated values can be used by the greedy and locally optimal algorithms instead of the original values to prioritize senior parcels over those that have recently arrived. In our experiment, we use an exponential inflation protocol $\mathcal{I}(\tau) = \alpha^{\tau}$, where $\alpha \geq 1$.

4. Numerical experiment.

In this section, we present a numerical study that simulates a realistic system, and the parcels are routed based on the greedy policy and the locally optimal policy both with and without value inflation. The dynamics of the simulated system are described in Figure 1 and Figure 2. In the simulation model, our simplifying assumptions are relaxed. In particular, we enforce the capacity constraints of the SPs and the OCs, simulate the handling and stopover time at the SPs and consider both of these components of the OC's reward. The simulation was coded in Python 2.7. The code and our data are available upon request from the authors.

Section 4.1 describes the use cases of our study in detail. Section 4.2 explains the measure key performance indicators (KPIs) and reports the results for a motivating example. Section 4.3 explores the conditions for the stability of the system, and section 4.4 presents a comprehensive full factorial experiment that allows for a comparison of the different loading policies and a sensitivity analysis of our parameters and their underlying assumptions.

4.1. The simulation study setup

For our simulation, we used a set of 50 SPs selected using a process proposed by Bloch and Tzur (2018), who is located in central Israel. We created a set of OC sequences between these SPs based

on actual movement patterns collected from a vehicle sharing system. From these sequences, we generated the set of suffixes for each SP and determined its arrival rate proportionally to the number of journeys that follow this suffix per day. Based on this pattern, we created three types of OC supply Poisson processes with the rates of 2000, 4000 and 8000 OCs per day in the entire system. These rates represent a small fraction of the 4.8 million journeys traveled by private vehicles daily in this region Dori (2016). We believe that the above rates are very conservative lower bounds of the willingness of drivers to participate as OCs once the system is up and running. We assume that each OC is ready to carry up to 50 parcels at a time if offered. The reward scheme is based on a payment per parcel and a payment per stop. Since all the journeys in this region are relatively short, the payment to the OC was assumed is fixed, i.e., not affected by the distance. The payment for handling each parcel is €0.35 or €0.47, and the payment per stop is €1.20. The handling time per parcel is one minute (including both the loading and unloading operations), while the delay caused by each stop of the OC is two minutes. The traveling time for each sequence was collected from Google Map and assumed to be deterministic.

Similarly, we created the arrival process of the parcels for their origins and destinations. Since we could not obtain actual data on the shipment of parcels, we artificially created a process in which most of the parcels are sent from SPs located in employment zones to SPs that are uniformly distributed over the network. We identify the characteristics of each zone using the vehicle movement patterns during the day. Zones to which most of the journeys take place in the morning are considered to be employment zones, and the remainder of the zones are considered to be residential. Based on these patterns, we created the Poisson arrival processes of 4000, 8000 and 16000 parcels per day (system-wide). We check two levels of SP capacities; one with 400 lockers per SP and one where the SPs have unlimited capacity. We also simulate a system where the OCs and SPs have unlimited capacity as well as zero handling and stopover times in order to validate our simplified model. Indeed, under these conditions, the expected costs predicted by the DP coincide with the those observed in the simulation.

The simulation used a value table for its decisions (in all the proposed methods) that was calculated by our DP using the same handling cost and OC arrival process, while the delay cost was set to €1.20 or €4.80 per parcel/day. This cost represents the urgency of delivering the parcels. Tables created with higher delay costs are expected to reduce the delivery time but increase the sum of the rewards paid to the OCs. Recall that the delay cost is also directly used by our loading policies to calculate Δ_{ij}. Note that the stopping cost is not considered directly by the DP and thus does not affect the value table. However, the stopping cost is taken into account by the locally optimal loading policy. For the value exponential inflation protocol, we use $\alpha = 1.5$. We also test the policies using constant values, i.e., $\alpha = 1$. Each simulation was run for 50 simulated days divided

Table 1 Experimental settings

Factor	Value / Levels	Relevant to
Parcel rate	4000, 8000, 16000	Simulation
OC rate	2000, 4000, 8000	DP, Simulation
delay cost (€)	1.2, 4.8	DP, Simulation under greedy policy
Handling cost (€)	0.37,0.45	DP, Simulation
Stopping cost (€)	1.2	Simulation (under locally optimal policy)
Travel time matrix	Nominal travel times from Google Map	DP, Simulation
Exponential value inflation α	1, 1.5	Simulation
Parcel loading and unloading times (min.)	0.5 per operation	Simulation
Time until the parcel is picked by its recipient	Exponential random variable with mean 12 hours	Simulation
stopover time (min.)	2	Simulation
SP Capacity (parcels)	400, unlimited	Simulation
OC Capacity (parcels)	50	Simulation
Loading policies	Greedy, Locally optimal	Simulation
Simulation time (days)	50	Simulation
Simulation warm-up time (days)	5	Simulation

into blocks of one day. We run a full factorial experiment with 288 cases. The different parameter values (levels of the factors) are summarized in Table 1.

We collected the statistics and inspected the dynamics of various measures such as the rate of parcels delivered within 24 hours and the rate of rejected parcels. Few of the tested configurations, with a high arrival rate of parcels and low arrival rate of OCs, exhibit poor service levels, and some seem to deteriorate over time in terms of the delivery times and parcel rejection ratio. These cases are reported and discussed in Section 4.3 but omitted from further analysis. All the other configurations seem to stabilize in terms of the above measures within 2-3 days. However, we decided to set the simulation warm-up time to 5 days, to be on the safe side. Thus, the statistics presented below are based on the performance during the last 45 days of the simulation. The collected key performance indicators (KPIs) are listed in 2 and explained in the next section.

4.2. KPIs and a motivating example

Since the process is viable only if it can provide good value to all the involved parties, namely, the customers, the OCs and the operator, we begin by defining KPIs that are represent their merits and costs. In Table 2, we report the values of these KPIs incorporating the two loading policies under realistic conditions where the arrival rate of OCs to the system is 8000 per day, and the total arrival rate of parcels to all the SPs is 16,000. The compensation to OCs is €0.47 for handling each

parcel and €1.25 for each stop. The exponential value inflation rate is $\alpha = 1.5$, and the delay cost is €1.20 per day. The OC capacity is 50 parcels, and the SP capacity is 400.

In the first row of Table 2, the average delivery time of parcels is reported. This time is measured as the difference between the arrival time of the parcel at its destination SP and the time it was dropped off at its origin. The percentage parcels delivered within the first 24 hours (resp., three days) is presented in the second (resp., third) row of the table. It appears that under the above conditions, both policies result in an excellent service level, where the average delivery time is approximately three hours, and almost all the parcels are delivered within 24 hours. In practice, this represents a premium service that can be classified as a *same-day delivery*. The service level provided by the two policies is similar with, perhaps, a slight advantage when using the greedy loading policy, since its average delivery time is shorter.

The average cost per parcel is presented in the fourth row and is calculated as the ratio between the sum of the rewards to the OCs (stopping and handling costs) over an extended period and the number of delivered parcels during this period. In an equivalent system that does not use crowdsourcing, these costs correspond to the transportation cost and the labor costs of the courier. We observe that the mean cost per parcel is significantly lower under the locally optimal policy. However, under both policies, the cost per parcel is low compared to the costs of employing a fleet of professional couriers and dedicated vehicles to provide such a premium service.

Due to the capacity limitation of the SPs and the high arrival rate of parcels in our experiment, some parcels were rejected at their origins. The rejection rates under the two policies,which are presented in the fifth row of the table, are approximately 1%. We note that in practice, when there is no available capacity at the drop-off location, the senders may be redirected by the system to a nearby SP where she can use the service. Moreover, since a location-based smart-phone app can manage the shipping process, the sender is guided directly to SPs with available capacity. However, this user behavior is not modeled in our simulation and rejected senders are assumed to abandon the system.

In the sixth row of the table, we present the number of legs in the average journey of a parcel. Each leg represents a section of the parcel route in which a different OC transported the parcel. We see that under both policies, a parcel is handled on average by approximately two OCs, which demonstrates the effectiveness of the PI concept.

In the seventh row, we present the ratio of the active OCs, which is the share of the journeys in which the OCs handle parcels out of the total number of journeys in which they are willing to do so. It appears that the tendency of the locally optimal policy to consolidate shipments results in fewer active OCs. Naturally, each active OC under the locally optimal policy handles more parcels

Table 2 Comparison between the two policies under good conditions

Average measure values	Locally optimal	Greedy
Average delivery time (hours)	3.28	2.84
% Same-day delivery	99.45%	99.33%
% Three-days delivery	100.00%	100.00%
Average cost per parcel (€)	1.11	1.6
% Parcel rejected	1.04%	1.21%
Average number of OC legs per parcel	1.94	1.98
% active OC	32.61%	59.43%
Average number of parcels handled by an active OC	11.81	6.62
Average number of stops per active OC	2.35	2.24
Average engagement time per active OC (mins)	19	12
Average payment to an active OC €	7.37	5.28

and makes more stop at more SPs along her journey. These data are presented in the eighth and ninth rows.

In the last two rows of the table, we present the average engagement time of an active OC and her average reward. We note that under both loading policies, the payment per hour is more than €23, which is comparable to the average hourly salary in the EU, (see Eurostat (2017)). In fact, in many labor markets, even much smaller rewards may be sufficient. For example, an average Uber or Lyft driver in the USA earns approximately $17 (€14.50) per hour.

From the example above, we get a sense that when the conditions of the system are favorable, the Heuristics delivery policies allow us to provide a good level of service with competitive costs. Next, we seek to understand the entire picture by dividing the experiment into different parts. We will first examine the non-binding capacity constraint cases described in 4.3.

4.3. Conditions for stability of the system

Next, we discuss and analyze the conditions under which the proposed system may reach a reasonable steady state over time. Due to the capacity constraints of the SPs, the system may deteriorate into a gridlocked state when many of the SPs are at full capacity most of the time, and thus, it is difficult for parcels to move toward their destinations. In such cases, the system becomes jammed and cannot accept new parcels. In this section, we identify the conditions under which such a gridlock is likely to occur.

The gridlock phenomenon is demonstrated in Figure 3 and Figure4, where we plot the dynamics of the rejection rates and the three-days delivery ratios for two systems with the same cost parameter under a locally optimal loading policy with an exponential value inflation rate $\alpha = 1.5$. In both systems, the demand for parcel delivery is 16,000 per day, but one is facing a supply of 2000 potential OCs per day while the other is facing a supply of 8000 per day. It is apparent from the graphs that the system with 2000 potential OCs per day quickly deteriorates to a very poor

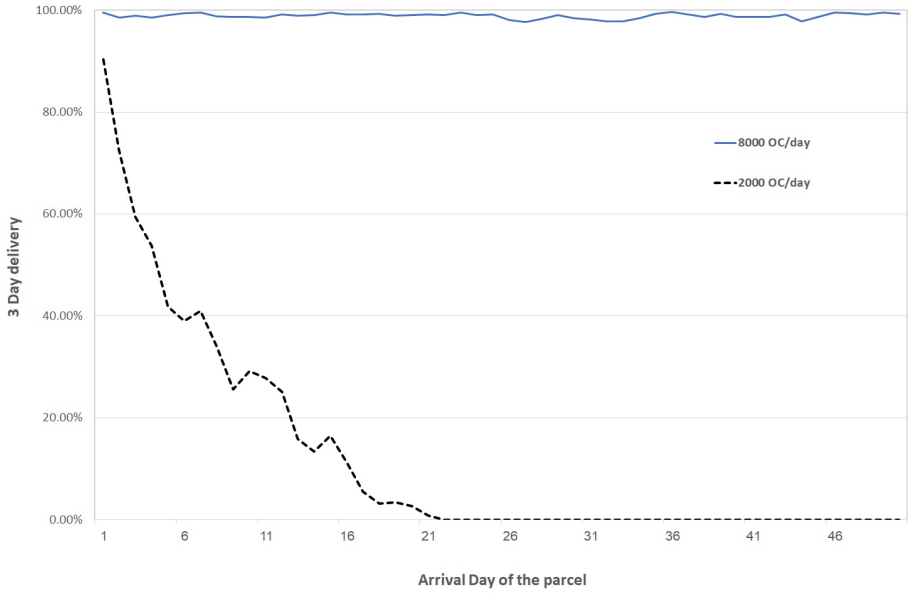

Figure 3 **Dynamics of the 3-day delivery KPI for systems with the parcel arrival rate of 16,000/day and OC arrival rates of 2000 and 8000/day.**

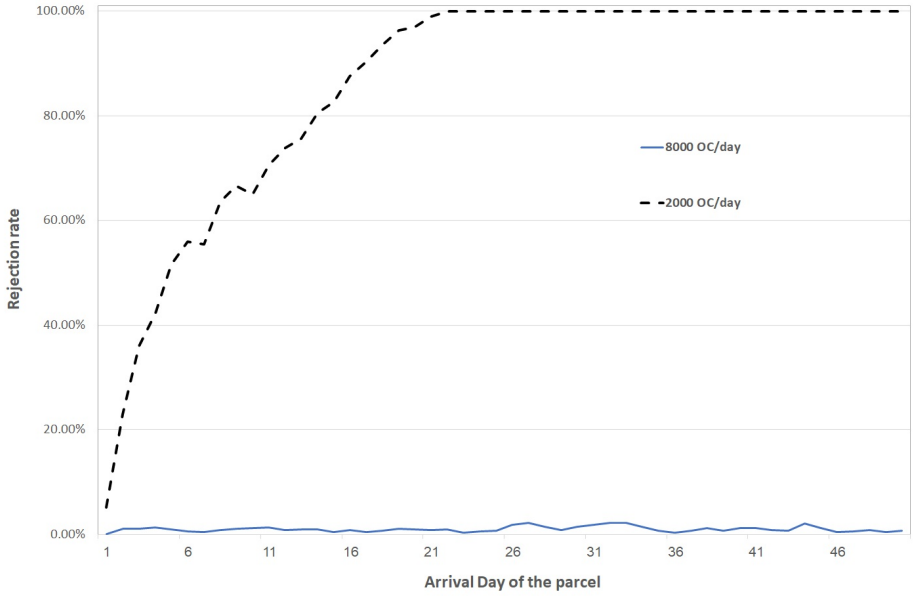

Figure 4 **Dynamics of the rejection rate KPI for systems with the parcel arrival rate of 16,000/day and OC arrival rates of 2000 and 8000/day.**

service level. Indeed, starting from 22^{nd} /day, the rejection rate is almost 100% and very, very few parcel are delivered. On the other hand, the system with 8000 potential OCs per day stabilizes with a negligible rejection rate, and almost all the parcels are delivered within three days.

For our settings, where there is an SP capacity of 400 and an OC capacity of 50, we observe that under both policies and all cost parameter combinations, the girdlock occurs when the ratio between the supply of OCs and the parcel demand rate is $1:8$. When the ratio is $1:4$, the system stabilizes with a rejection rate of $3-12\%$, which seems too high for a parcel delivery service. When the ratio between the supply of OCs and the parcel demand rate is $1:2$, the rejection ratio is less than 2%, and it is nearly zero when the ratio is $1:1$.

FOR hypothetical settings where the SP capacity is not limited, the system never rejects a parcel. However, assuming that OC capacity is still limited and the supply of OCs is insufficient, the system can reach a state in which it cannot provide a service for some of the origins and destinations. In such cases, increasing numbers of parcels are piled up in the SPs. In Figure 5, we demonstrate this phenomenon by plotting the number of parcels in the system versus time in our 50-days experiment for two systems with the same settings as above but without SP capacity constraints. While the system with the supply of 8000 OCs a day seems to stabilize after 2-3 days, the number of parcels in the system that face a supply of 2000 OCs a day continues to grow over time. This result implies that the service level in the congested SPs (and possibly in other SPs from which parcels are routed via the congested ones) is deteriorating over time. Thus, in highly congested systems, a different routing policy that avoids congested SPs may help increase the throughput. Such a policy is not within the scope of this paper; please see the discussion in Section 5.

While the uncapacitated system is deteriorating at a slower rate compared to the capacitated SP case, it does not seem to reach a steady state. Therefore, increasing the SP capacity alone cannot compensate for the small supply of OCs. Note that even when the total capacity of all the potential OCs is significantly larger than the number of parcels, the system is not necessarily stable. For example, the total capacity of the 2,000 potential OCs in our experiment is 100,000 per day, and parcel demand is only 16,000 per day. This result occurs because the routes of many of the potential OCs do not align with the origins and destinations of the parcels.

In our experiment, systems with a supply-demand ratio of 1:2 or better are always stable. We conclude this study by performing stability tests for all the settings in our full factorial experiments. We consider a system to be stable if there is no statistically significant trend in its same-day delivery ratio during the 50-day simulation period, excluding the first five-days warm-up period.

A model that ignores SP capacity can be useful only when the capacity constraint is not likely to be binding. For example, if the SPs are manned facilities with a large capacity, and the ratio between the demand and the supply of OCs is sufficiently small.

We note that in real-life, it is not likely that the operator will face the issues of gridlock and instability; indeed, once the service level starts to deteriorate, demand will decrease. Moreover,

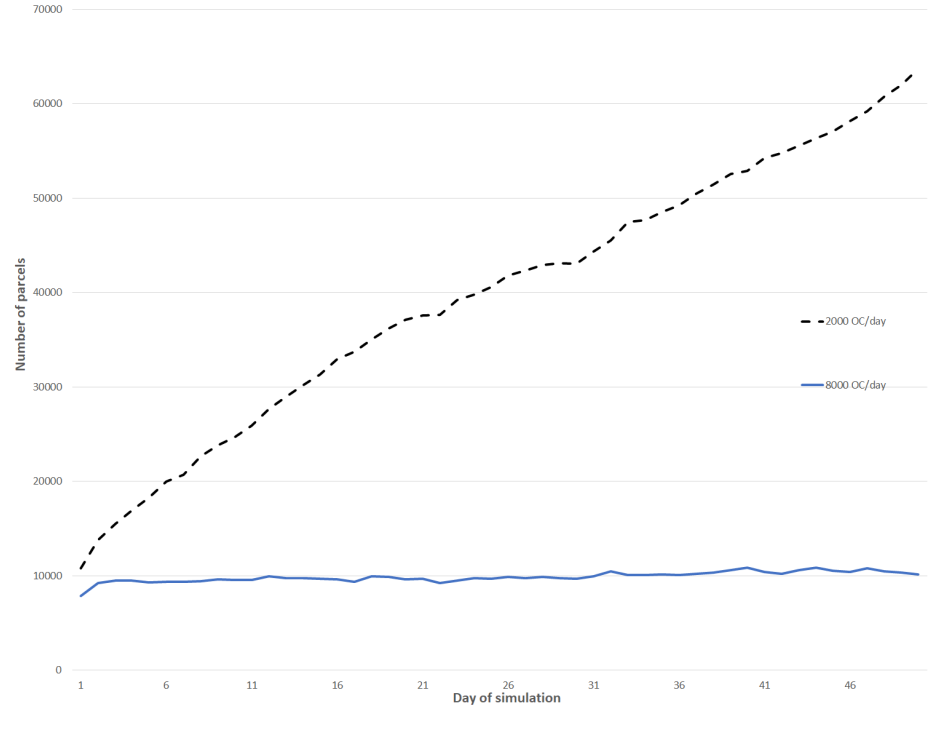

Figure 5 **The accumulation of parcels in the system with the parcel arrival rate of 16,000/day and OC arrival rates of 2000 and 8000/day under no SP capacity constraints .**

when parcels pile up in the SPs, more parcels are offered to the OCs to deliver at each stop. As a result, the incentives that the OCs can collect increase, which is likely to increase their supply. The operator should set the SP capacity, the loading policy, the incentives of the OC, and the price of the service in a way that will induce the most profitable equilibrium. These strategic decisions are beyond the scope of this paper, but the model that we devise can be used to examine many possible scenarios.

In practice, the operator is likely to gradually update the above decisions based on the actual performance of the system. For example, the capacity of each SP can be either increased or decreased based on its actual historical occupancy. Similarly, the price of the delivery service can be increased based on the actual payments made to OCs for each particular origin and destination.

4.4. Comparing the policies and a sensitivity analysis

In this section, we examine the effect of the loading policy, the exponential value inflation rate (α) and the delay costs on the performance of the system in terms of the KPIs defined above. In addition, we perform a sensitivity analysis of the other parameters to test the robustness of our results.

Here, we focus on a system with capacity constraints on the SPs. Recall that when the ratio between the potential OC arrival rate and the parcel arrival rate is lower than $1:2$, the capacitated

system rejects a large share of the parcels. Hence, we limit the discussion to cases where this ratio is at least $1:2$. The analysis below is based on 144 cases that satisfy the above conditions.

In Table 3, we compare the KPIs of systems with low and high levels of the handling cost component of the OC compensation calculation. The presented KPIs are averaged over all the other levels in our full factorial experiment. Recall that the value table is calculated by the DP based on the delay and handling cost parameters, and hence, the differences in the service level KPIs. However, it is apparent from the table that the differences are not substantial. Therefore, the operator is free to use OC compensation to balance the demand for the service and the supply of OCs. Indeed, by increasing the handling cost and charging higher service fees from the customers, the operator can increase the supply of OCs and reduce the demand for the service. This, is turn, will surely improve the service level, but the evaluation of this effect is outside the scope of our model.

Table 3 Comparison of the impact of two handling costs on the different KPIs

Handling cost (€/parcel)	**0.37**	**0.45**
Shipping time (hours)	4.32	4.47
Same-day delivery ratio	98.59%	98.43%
Three-days delivery ratio	99.79%	99.79%
Parcel rejection rate	0.30%	0.25%
Cost per parcel (€)	1.58	1.79
Active OC rate	43.83%	43.42%
Stops per active OC	2.24	2.25
Parcels per active OC	7.28	6.91
Payment to active OC (€)	5.29	5.98
Active OC engagement time (mm:ss)	8:15	7:55

The exponential inflation rate (α) and the delay cost (h) are two parameters that affect the routing decision but do not represent any actual cost; therefore, they can be used by the operator to control the trade-off between the service level and operational costs. In Tables 4 and 5, we examine four combinations of h and α levels with respect to the two loading policies. Each block in these tables represents the average KPIs for 12 cases (with different levels of supply and demand rates as well as the different handling costs).

As expected, we observe that higher values of α and h shorten the delivery time while increasing the cost per parcel. The effect of these parameters is sensitive to the settings of the system, and the operator should fine-tune them. This process of updating α and h can be done behind the scenes, on a daily basis without directly affect the senders and the OCs.

In our settings, it seems that the locally optimal policy is always a better choice than using the greedy policy. Indeed, for any combination of α and h, under the greedy policy at least one combination (not necessarily the same on every time) provides a comparable service level at a

significantly lower cost under the locally optimal policy. For example, when comparing the KPIs of the greedy policy with $\alpha = 1$ and $h = 1.2$ to the locally optimal policy with $\alpha = 1.5$ and $h = 1.2$, we see that the three- and same-day delivery ratio of the latter are slightly larger, and the parcel rejection rate is slightly lower. At the same time, the cost per parcel is almost 30% lower under the locally optimal policy.

Table 4 Effect of the delay cost and α on the KPIs of the greedy loading policy

Delay cost (h) in €	1.2		4.8	
Exponential inflation rate (α)	1.0	1.5	1.0	1.5
Parcel rejected rate	0.18%	0.19%	0.41%	0.41%
Same-day delivery rate	98.19%	98.53%	98.88%	98.89%
Three-days delivery rate	99.77%	99.78%	99.77%	99.77%
Average cost per parcel (€)	1.76	1.79	2.08	2.09

Table 5 Effect of the delay cost and α on the KPIs of the locally optimal loading policy

Delay cost (h) in €	1.2		4.8	
Exponential inflation rate (α)	1.0	1.5	1.0	1.5
Parcel rejected rate	0.16%	0.17%	0.34%	0.35%
Same-day delivery rate	97.37%	98.26%	99.05%	98.90%
Three-days delivery rate	99.83%	99.80%	99.84%	99.77%
Average cost per parcel (€)	1.19	1.25	1.64	1.66

Finally, we demonstrate economies of scale in the operation of a crowd-sourced PI by examining the same-day delivery ratio and the cost per parcel when the rate of parcels and number of OCs are increased proportionally; the ratio remains unchanged at $2:1$. For the purpose of the demonstration, we present these values for the locally optimal policy with $\alpha = 1.5, h = 1.2$ and a handling cost of €0.45 in Figure 6. The figure clearly shows that as the scale of the system is increased the service level can be improved without increasing the cost per parcel. Similar phenomena are observed under different settings and loading policies. Note that only labor and transportation costs are considered here. However, the inclusion of other cost components such as infrastructure, management, and marketing can only enhance the economies of scale.

We believe that economies of scale occur because of several factors such as a higher frequency of visits of OCs at SPs, more opportunities to send parcels in routes that consist of fewer legs and more opportunities for shipment consolidation. Note that the increased volume of parcels may require SPs with increased capacity. However, it turns out that there are also economies of scale for the SP capacity. To demonstrate this, in Figure 7, we examine the average number of parcels per SP at the end of each day. We compare two extreme settings: one with 2000 OC/day and 4000

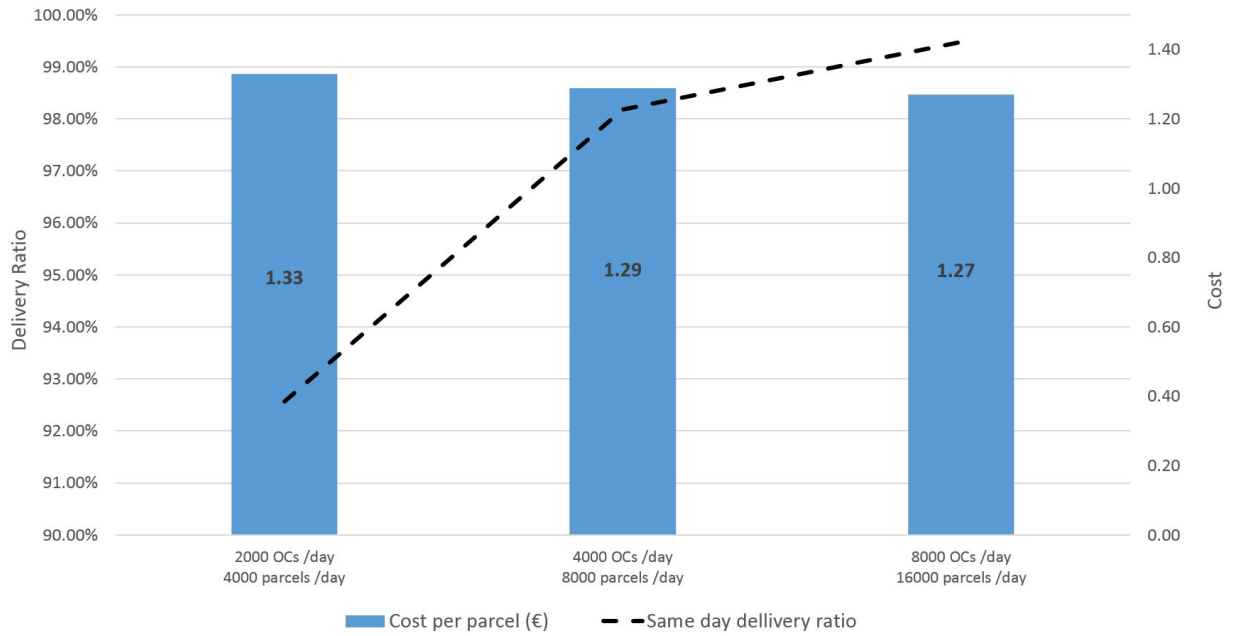

Figure 6 **Economies of scale**

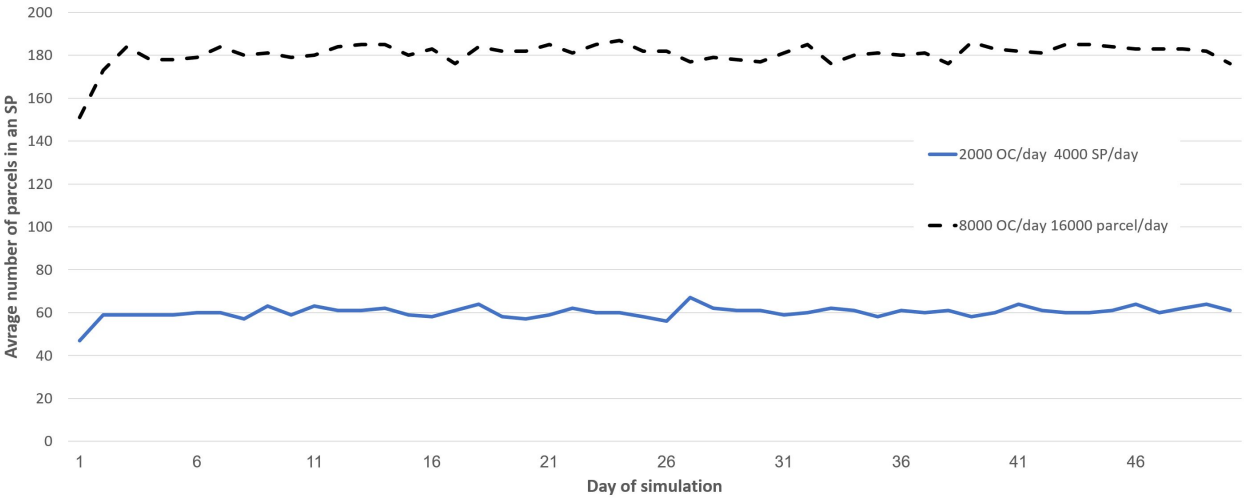

Figure 7 **The influence of economies of scale on the SP's capacity**

parcels/day and one with rates that are four times higher. Note that the utilization of the SPs is only about three times higher in the latter case.

5. Conclusions.

In this study, we introduce a model for crowd-shipping using a PI of SPs. We devised a routing algorithm that is shown to perform well and demonstrate the economic viability of this model using

a simulation study. This is the first study on this particular logistic model, and many questions remain to be addressed.

For example, in our model, we consider a single class of parcels, all with the identical size and priority. Moreover, the arrival process of both the OCs and parcels is time-homogeneous. Extensions of our model that relax these simplifying assumptions are worth studying.

Our online parcel routing algorithms respect the capacity constraints of the current and next SPs along the route of each parcel, but they do not consider the state of other SPs that may serve the parcel in future legs of the journey.It would be intresting for future research to develop a routing algorithm that considers the bottleneck resources in the system, which can be achieved by updating the values of the SPs based on their current load and predictions derived from historical data. A more straightforward approach may be to reserve some "safety capacity" in each SP for new parcels and for parcels that should be delivered only to this SP.

The crowd-shipping PI is a complex marketplace that matches the demand produced by shippers and the supply offered by the OCs while considering the capacity of the SPs. The operator can control the demand by setting the price and the quality of the offered service. Simultaneously, she can control the supply by tuning the incentives of the OCs. All these parameters can be varied dynamically and spatially to react to the current state of the system. In addition, the operator can reduce the load on congested SPs by offering incentives to recipients that can quickly pick up their parcels from these SPs after the delivery or even be ready to pick the parcel up at a different location. To attempt any of these further analyses mentioned above, the formulation of a pricing/incentives/service level model and the development of efficient algorithms to solve it will be required.

The initialization of the proposed crowd-shipping PI may be challenging. Indeed, without a large community of OCs, the system cannot offer high-quality service at a reasonable cost. At the same time, without sufficient demand, no profitable delivery tasks can be offered to the OCs, which would make it impossible to build the OC community. We propose to overcome this problem by initiating the system with the aid of professional couriers (PC) that transfer the parcels within the network. As the market is created the system gradually offers tasks to the OCs. In the long run, it may be economical to utilize both OCs and PCs. The PCs can help in increasing the reliability of the system at times and in locations where the OCs cannot meet the demand or when SPs become temporarily congested. The optimal routing of parcels and PCs in such a hybrid system is an additional interesting direction for future research.

Acknowledgments

This research was supported by the Israeli ministry of science and technology.

References

Archetti C, Savelsbergh M, Speranza MG, 2016 *The vehicle routing problem with occasional drivers. European Journal of Operational Research* 254(2):472–480.

Arslan A, Agatz N, Kroon LG, Zuidwijk RA, 2016 *Crowdsourced delivery-a pickup and delivery problem with ad-hoc drivers. SSRN Electronic Journal* 1–29.

Barr A, Wohl J, 2013 *Exclusive: Walmart may get customers to deliver packages to online buyers. REUTERS–Business Week* URL `https://www.reuters.com/article/us-retail-walmart-delivery/exclusive-wal-mart-may-get-customers-to-deliver-packages-to-online-buyers-idUSBRE92R03820130328?irpc=932`, accessed 24.6.2018.

Bloch A, Tzur M, 2018 *The service point location in crowd delivery system. working paper* .

Chen C, Zhang D, Ma X, Guo B, Wang L, Wang Y, Sha E, 2017 *Crowddeliver: planning city-wide package delivery paths leveraging the crowd of taxis. IEEE Transactions on Intelligent Transportation Systems* 18(6):1478–1496.

Chen W, Mes M, Schutten M, 2017 *Multi-hop driver-parcel matching problem with time windows. Flexible services and manufacturing journal* 1–37.

Deutsch Y, Golany B, 2017 *A parcel locker network as a solution to the logistics last mile problem. International Journal of Production Research* 0(0):1–11, URL `http://dx.doi.org/10.1080/00207543.2017.1395490`.

Devari A, 2016 *Crowdsourced last mile delivery using social network*. Ph.D. thesis, State University of New York at Buffalo.

Dori O, 2016 *Driving to work in tel aviv at seven miles per hour: the numbers behind israel's traffic woes. Haaretz on-line edition* URL `https://www.haaretz.com/israel-news/business/the-numbers-behind-israel-s-traffic-woes-1.5435662`.

Eurostat, 2017 *Statistics explained - hourly labour costs*. URL `http://ec.europa.eu/eurostat/statistics-explained/index.php?title=Hourly_labour_costs`, accessed 24.6.2018.

Faugere L, Montreuil B, 2017 *Hyperconnected pickup & delivery locker networks*.

Lee S, Kang Y, Prabhu VV, 2016 *Smart logistics: distributed control of green crowdsourced parcel services. International Journal of Production Research* 54(23):6956–6968.

McInerney J, Rogers A, Jennings NR, 2013 *Learning periodic human behaviour models from sparse data for crowdsourcing aid delivery in developing countries. arXiv preprint arXiv:1309.6846* .

Montreuil B, Meller RD, Ballot E, 2010 *Towards a physical internet: the impact on logistics facilities and material handling systems design and innovation. Progress in material handling research* 305–327.

Savelsbergh M, Van Woensel T, 2016 *50th anniversary invited articlecity logistics: Challenges and opportunities. Transportation Science* 50(2):579–590.

Spiess H, Florian M, 1989 *Optimal strategies: a new assignment model for transit networks. Transportation Research Part B: Methodological* 23(2):83–102.

Voccia A Stacy, Campbell M Ann, Barrett T W, 2017 *The same-day delivery problem for online purchases. Transportation Science* Published online in Articles in Advance 19 May 2017.