https://doi.org/10.1016/j.trb.2023.03.007

Joint optimization of parcel allocation and crowd routing for crowdsourced last-mile delivery

Li Wang[a,b], Min Xu[b,*], Hu Qin[a]

[a]School of Management, Huazhong University of Science and Technology, Wuhan, China

[b]Department of Industrial and Systems Engineering, The Hong Kong Polytechnic University, Hung Hom, Hong Kong

Abstract

Urban last-mile delivery providers are facing more and more challenges with the explosive development of e-commerce. The advancement of smart mobile and communication technology in recent years has stimulated the development of a new business model of city logistics, referred to as crowdsourced delivery or crowd-shipping. In this paper, we investigate a form of crowdsourced last-mile delivery that utilizes the journeys of commuters/travelers (crowd-couriers) to deliver parcels from intermediate stations to customers. We consider a logistics service provider that jointly optimizes parcel allocation to intermediate stations and the delivery routing of the crowd-couriers. The joint optimization model gives rise to a new variant of the last-mile delivery problem. We propose a data-driven column generation algorithm to solve the problem based on a set-partitioning formulation. Additionally, a rolling-horizon approach is proposed to address large-scale instances. Extensive numerical experiments are conducted to verify the efficiency of our model and solution approach, as well as the significance of the joint optimization of parcel allocation and the delivery route of the crowdsourced last-mile delivery. The results show that our data-driven column generation algorithm can obtain (near-)optimal solutions for up to 200 parcels in significantly less time than the exact algorithm. For larger instances, the combination of the data-driven column generation algorithm and the rolling-horizon approach can obtain good-quality solutions for up to 1,000 parcels in 15 min. Moreover, compared with crowd-courier route optimization only, the joint optimization of parcel allocation and crowd-routing reduces the total cost by 32%.

Keywords: crowdsourced delivery, last-mile delivery, parcel allocation and crowd routing, data-driven column generation

*Corresponding author

Email addresses: liliw0227@gmail.com (Li Wang), min.m.xu@polyu.edu.hk (Min Xu), tigerqin1980@qq.com (Hu Qin)

1. Introduction

Urban delivery is currently undergoing exciting and challenging times, with ever-growing e-commerce sales. For example, the global retail e-commerce sales grew by more than 16.8% in 2021 (Insider, 2021). According to an industrial report, 131 billion parcels were delivered worldwide in 2020, a figure projected to double in 2026 (Bowes, 2021). As the most expensive component of the order fulfillment cost, last-mile delivery, a concept commonly used in city logistics, has become one of the most challenging problems faced by logistics service providers. Moreover, the growing demand for parcel delivery and accordingly increasing freight traffic have caused negative externalities such as pollution and traffic congestion. It is of particular significance that logistics service providers develop efficient, and cost-effective last-mile delivery solutions in order to reduce costs and achieve operational competence in the era of e-commerce.

In recent years, the advancement of smart mobile and communication technology has stimulated the development of a new business model of city logistics, referred to as crowd-sourced delivery or crowd-shipping, which utilizes the excess capacity of individuals to carry out last-mile deliveries in urban areas (Le et al., 2019). Motivated by the emerging concepts of the sharing economy and collaborative consumption, last-mile crowdsourced delivery has attracted much attention from industry. In fact, multiple platforms have been developed to offer last-mile crowdsourced deliveries. Alnaggar et al. (2021) classified these platforms into four main patterns according to their scheduling and matching mechanisms:

- pure self-scheduling, in which the orders are matched with the crowd-couriers who may log on the platform whenever they want, such as Postmates, DoorDash, and Uber Eats;
- hybrid and centralized scheduling, in which the crowd-couriers make dedicated deliveries for a pre-determined working duration, such as Amazon Flex, Deliv, Instacart, and Shipt;
- en-route matching, where the platform matches the traveler with delivery requests on his/her pre-planned trip, such as Hitch and Roadie;
- bulletin-board type matching, where a crowd-courier picks requests that match his/her schedule and preference, such as Walmart Spark Delivery, DHL MyWays, and Kanga.

Among all the business practices involved, the prominent operational challenge is the en-route matching problem: given time, origin, and destination information about upcoming trips submitted by a set of commuters or occasional drivers, namely crowd-couriers, a crowd-sourced delivery platform needs to match the travelers with delivery requests they can fulfil on their way, such that a set of specific constraints, e.g., the maximum travel (detour) time, capacity, and time windows, are satisfied. After fulfilling the delivery tasks, the crowd-couriers will receive a small amount of compensation according to the extra time they have spent providing the service (Alnaggar et al., 2021). By using ordinary commuters as 'ad hoc couriers', crowdsourced delivery brings significant economic benefits compared with traditional logistics services using full-time drivers and dedicated vehicles.

1.1. Literature Review

Over the past decade, the rise of crowdsourced delivery has motivated a boom in research efforts to explore a variety of strategies dealing with the challenges of crowdsourced delivery platforms. These strategies can be divided into strategic, tactical, and operational decisions. The determination of the locations and capacities of intermediate stations falls in the first category, while the service area and delivery capacity planning belongs to the second group (Yildiz and Savelsbergh, 2019; Nieto-Isaza et al., 2022). Apart from the strategic and tactical strategies, other critical operational decision-makings for crowdsourced delivery involves the parcel matching, vehicle routing, and pricing (Le et al., 2019). In what follows, we will focus on the most relevant studies for the operational decision-making of the crowdsourced delivery. Broadly speaking, these studies can be divided into three groups based on the problem setting.

The first group considers direct end-to-end deliveries by occasional drivers, especially in-store customers, who deliver the online orders. This business model was piloted by Walmart in 2013. Inspired by the real practice, Archetti et al. (2016) for the first time investigated an interesting vehicle routing problem with occasional drivers or in-store customers who were willing to deliver an online order for a small amount of compensation. Each crowd-courier was allowed to deliver one parcel at most. They proposed a multi-start heuristic solution approach that combined a variable neighborhood search and tabu search to solve the problem. Macrina et al. (2017) later studied a more general problem considering time windows and two different scenarios, one allowing multiple deliveries by each occasional driver and another introducing a split delivery strategy. Pugliese et al. (2022b) studied a problem similar to the former scenario explored by Macrina et al. (2017). To solve this problem, the authors proposed a variable neighborhood search approach, where several machine learning techniques were used to explore the most promising areas of the search space. Pugliese et al. (2022a) investigated the same problem with a hybrid approach combining greedy randomized adaptive search procedures and variable neighborhood search. Gdowska et al. (2018) considered a similar setting to Archetti et al. (2016) and developed a stochastic model, taking the probability of the in-store customers accepting or rejecting the assigned delivery tasks into account. Mancini and Gansterer (2022) extended the work of Archetti et al. (2016) by allowing multiple deliveries by each occasional driver in a bidding system where crowd-couriers could submit their bids for bundles they were willing to serve. They proposed two heuristic methods to generate valuable bundles of customers. Given a set of customers and bundles, the bid selection and the routing of company-dedicated vehicles were jointly optimized to minimize the total cost. A large neighborhood search approach was proposed to solve the problem. Arslan et al. (2019) explored a combined operation mode with deliveries performed either by crowd-couriers or by dedicated vehicles. They optimized the routing of the crowd-couriers and the dedicated vehicles from a holistic perspective. A rolling-horizon framework was proposed to handle the dynamic variant and an exact solution approach was developed to solve an offline problem in each optimization run. In a similar vein, Dayarian and Savelsbergh (2020) also addressed a dynamic variant of the problem and proposed two dynamic models with a rolling-horizon framework to deal with the real-time information of in-store customers and online orders. Fatehi and Wagner (2022) studied a labor planning and pricing problem for crowdsourced last-mile delivery systems to satisfy on-demand orders with guaranteed delivery time windows. The authors proposed a novel robust optimization model combining crowdsourcing, robust queueing, and robust routing theories. Ahamed et al. (2021) proposed a novel deep reinforcement learning-based approach to solve a similar end-to-end crowdsourced delivery problem. In addition, Behrend and Meisel (2018) explored the decision making for an integrated item-sharing and crowdsourced delivery platform. They developed mathematical models and heuristics for maximizing the platform's profit and the number of fulfilled requests. Fadda et al. (2018) addressed crowdsourced task assignment problem in which crowds shared the Internet connection of their mobile phone with the dumpsters around them. They formulated this problem as a stochastic programming model and devised a heuristic method to solve it.

The second group considers the crowdsourced end-to-end deliveries with transfers between crowd-couriers at some pre-positioned transfer locations, such as stores and locker stations. For example, Sampaio et al. (2020) addressed a pick-up and delivery problem in which the parcels were picked up either from transfer locations or pick-up locations, and delivered either to customers or transfer locations. Each request was allowed to be transferred once at most. They formulated the problem as a mixed integer programming (MIP) and proposed an adaptive large neighborhood search (ALNS) algorithm to solve it. Recently, Voigt and Kuhn (2021) studied a similar problem but looked at a combined group of couriers comprised of dedicated drivers and crowd-couriers. In addition, the authors incorporated the crowd-couriers' own trips into the delivery system. Again, an MIP and ALNS algorithm were developed to solve the problem. Chen et al. (2018) explored a multi-driver multi-parcel matching problem in which the crowd-couriers again had their own trips, but multiple transfers were allowed for each parcel. They developed an MIP model and proposed two heuristics to solve it. Furthermore, Raviv and Tenzer (2018) explored a dynamic version of a similar problem. They presented a stochastic dynamic programming problem to yield an optimal parcel routing policy under the assumption that the arrivals of parcels and crowd-couriers followed a Poisson distribution. Yıldız (2021) also developed a dynamic programming model for a similar problem but without the distribution assumption regarding the arrival rates of the parcels and crowd-couriers. In addition, the author considered dedicated vehicles to ensure timely delivery.

The first two groups of studies explore the end-to-end crowdsourced delivery system, where the pick-up and delivery locations of a parcel are given in advance. In contrast, a few studies have considered a two-tiered crowdsourced last-mile delivery system that mainly uses crowd-couriers to perform the last leg of the delivery, from the intermediate locations to the customers, with the traditional truck fleet still used to distribute the parcels to the intermediate locations. Janinhoff et al. (2022) introduced a simplified two-tiered crowdsourced last-mile delivery system combining home delivery with out-of-home delivery, where the service provider dropped off parcels at pickup stations and the customers performed the last leg of the delivery process themselves. Kafle et al. (2017) studied a problem setting in which dedicated vehicles functioned as mobile depots. They considered a context in which crowd-couriers needed to submit bids for the delivery requests and, if they won the bid, they would be required to pick up the parcels from the mobile depots and deliver them to the customers. They formulated an MIP model for bid selection and dedicated vehicle routing and scheduling, and then proposed a tabu search method to solve it. Mousavi et al. (2021)

developed a two-stage stochastic programming model to cope with randomly showing up crowd-couriers in a two-tiered crowdsourced last-mile delivery system with mobile depots. Both studies focused on the routing/positioning of the dedicated vehicles and the matching of parcels to crowd-couriers. Macrina et al. (2020) explored a two-tiered delivery system where the crowd-couriers picked up the parcels either from the central depot or from a pre-located intermediate depot to which the parcels were delivered by a conventional fleet. Vincent et al. (2022) extended the study of Macrina et al. (2020) by considering multiple delivery options. The authors developed an MIP model and proposed an ALNS algorithm to solve it. Wang et al. (2016) investigated a similar last-mile delivery system in which parcels were pre-allocated to the nearest POPStation (short for pick own parcel) to the customer locations. The objective was to find the optimal assignment of parcels to crowd-couriers. Perboli et al. (2021) built up a new variant of the bin packing model with time-dependent costs and crowd-couriers for a shared satellite-based last-mile delivery system. The authors assumed the routing cost of each delivery fulfilled by a crowd-courier following a cost-per-stop scheme.

Combining machine learning methods with column generation algorithms is attracting increasing attention in recent years due to its advantages in capturing human behavior and accelerating the solution method (Tahir et al., 2021; Morabit et al., 2021, 2022; Shen et al., 2022; Bayram et al., 2022). Tahir et al. (2021) proposed an algorithm combining machine learning and integral column generation to solve the classical crew-pairing problem. The authors used a deep neural network model to predict the probability of each arc (i.e., flight connection) being selected in a near-optimal solution. The subproblems were reduced on the graph containing only the flight connections with a high probability of reducing the computation time. Morabit et al. (2021) presented a column selection approach based on a binary classification model to solve the crew scheduling problem and the vehicle routing problem with time windows. In each iteration of column generation, the binary classification model was used to predict if the generated columns should be selected for the solution of the restricted master problem. This approach could reduce the computation time spent in reoptimizing the restricted master problem with fewer columns. Morabit et al. (2022) further developed an arc selection approach to solve the same problem in Morabit et al. (2021). The arc selection method was to identify the arcs that had a greater chance of being used or being part of an optimal solution using a binary classification model. Bayram et al. (2022) employed machine learning techniques to capture human behavior in the picking process in warehouse management. A machine learning model was employed to predict the processing times of a group of orders based on historical data from an industrial partner. A set partitioning model was formulated and a classical branch-and-price framework was utilized to solve the model. Processing time prediction was invoked each time a new label was generated in the labeling algorithm for the pricing subproblem.

1.2. Objective and Contributions

For simplicity, few of the aforementioned studies of two-tiered crowdsourced last-mile delivery systems optimized the routing of the crowd-couriers. They assumed that each crowd-courier carried one parcel at most or that the deliveries of multiple parcels assigned to a crowd-courier were independent of another one, e.g., neglecting the distance between the customers. These assumptions weaken their applicability to a real-world two-tiered crowdsourced last-mile delivery system, in which a crowd-courier could deliver multiple parcels at once, and the compensation is paid according to the length of or detour involved in his/her route. To the best of our knowledge, no one has simultaneously addressed parcel allocation to intermediate stations and the routing of crowd-couriers for two-tiered crowdsourced last-mile delivery systems, although doing so is practically relevant and significant. To fill the research gaps, this study investigates a novel joint optimization problem of parcel allocation to intermediate stations and the routing of crowd-couriers (PACR) in a two-tiered crowdsourced last-mile delivery system. We assume that each parcel is associated with a customer location and a deadline. The parcels are allocated to capacitated intermediate stations scattered around an urban area, before being delivered to customers by crowd-couriers. The crowd-couriers have their own original itineraries, characterized by an origin, a destination, their earliest departure time, their latest arrival time, and their maximum travel time. Each crowd-courier is allowed to carry multiple parcels as long as his/her capacity is not violated. Their compensation is calculated based on the additional travel time of the crowd-couriers. A penalty is incurred if a parcel fails to be matched. The objective is to minimize the total amount of compensation paid to the couriers for delivered parcels and the penalties paid for unmatched parcels in the last leg of the delivery by simultaneously determining the optimal allocation of parcels to intermediate stations and the delivery routes of the crowd-couriers.

To achieve the objective, we first formulate a route-based set-partitioning (R-SP) model based on the routes of the crowd-couriers. To solve the R-SP formulation, we propose a novel data-driven column generation (DCG) algorithm framework. The DCG algorithm combines the column generation procedure with a machine learning method to predict the travel time of a match, a combination of an intermediate station, a crowd-courier, and a set of parcels/customers, regardless of the sequence in which the customers are visited. In the existing learning-based column generation method, the machine learning model was embedded in a branch-and-price algorithm either to reduce the size of master problems/subproblems (Morabit et al., 2021, 2022; Shen et al., 2022) or to capture human behavior (Bayram et al., 2022). In contrast, this study embeds the machine learning model in a novel framework designed to select a set of promising matches that are likely to develop into near-optimal routes. This framework takes advantage of the machine learning model in predicting the travel time of crowd-couriers quickly and capturing the crowd-couriers' real routing behavior with historical data. We will first enumerate the matches that are likely to produce feasible routes for the R-SP formulation. In the enumeration procedure, the minimal travel time of the route embedded in a match can quickly be predicted via the machine learning model and further leveraged to enhance the feasibility check. We will then select good-quality matches from the match pool using a column generation procedure for a match-based set-partitioning (M-SP) formulation with the same structure as the R-SP formulation. Because it has the same structure as the R-SP formulation, the good-quality matches are likely to provide good-quality routes. Finally, we will convert the selected good-quality matches into routes by solving traveling salesman problems with time windows (TSPTWs). As such, the R-SP formulation based on the identified feasible and good-quality routes will be solved via an MIP solver to obtain a near-optimal solution to the PACR problem. Based on the characteristics of the time windows of the crowd-couriers and the deadlines of the customers, we propose a rolling-horizon approach to further solve large-scale instances (with over 200 customers). This solves several sub-problems via the proposed DCG algorithm for overlapping sub-periods in temporal sequence, iteratively.

The contributions of this paper can be summarized as follows:

- We introduce a new variant of the crowdsourced last-mile delivery problem considering parcel allocation and crowd-courier routing in a two-tiered system in which the crowd-couriers are responsible for the last leg of delivery.
- Other than a conventional arc-flow model, we formulate an R-SP model based upon the notion of a route that characterizes both the allocation of parcels and the routing of crowd-couriers.
- We develop a novel DCG algorithm that uses a machine learning model to efficiently identify a subset of feasible and good-quality routes of the R-SP formulation.
- Extensive numerical experiments have been conducted to demonstrate the effectiveness of the proposed solution method and to derive management insights.

The remainder of this paper is organized as follows. The assumptions, notations, and problem description are elaborated in Section 2. An R-SP formulation for the PACR problem is developed in Section 3. Section 4 presents our solution method, a DCG algorithm under a rolling-horizon framework, which is used to solve the R-SP formulation. Extensive numerical experiments and sensitivity analyses are conducted in Section 5. Section 6 presents conclusions and proposed future research.

2. Problem Description

In this section, we define the PACR problem considering a logistics service provider which allocates parcels to intermediate stations scattered around an urban area, and uses a group of crowd-couriers to deliver the parcels from the stations to customer locations for last-mile delivery.

2.1. Sets and Indices

Let $\mathcal{S}$, $\mathcal{K}$, and $\mathcal{V}$ denote the sets of intermediate station locations, crowd-couriers, and customer locations, and $\mathcal{O}$ and $\mathcal{D}$ represent the sets of the origins and destinations of the crowd-couriers. For convenience of notation, $\mathcal{V}$ also refers to the set of parcels. All locations in sets $\mathcal{S}$, $\mathcal{V}$, $\mathcal{O}$, and $\mathcal{D}$ are grouped into set $\mathcal{N}$, while the arcs associated with $\mathcal{N}$ are grouped into set $\mathcal{A}$. To be specific, the arc set $\mathcal{A}$ contains all the arcs corresponding to the visiting sequences of the crowd-couriers, i.e., $\mathcal{A} = \{(o_k, s) \mid k \in \mathcal{K}, s \in \mathcal{S}\} \cup \{(s, i) \mid k \in \mathcal{K}, i \in \mathcal{V}\} \cup \{(i, j) \mid i \in \mathcal{V}, j \in \mathcal{V}\} \cup \{(i, d_k) \mid i \in \mathcal{V}, k \in \mathcal{K}\} \cup \{(o_k, d_k) \mid k \in \mathcal{K}\}$.

2.2. Parameters

Each station $s \in \mathcal{S}$ has a limited capacity, represented by C_s. Each crowd-courier $k \in \mathcal{K}$ is associated with his/her original itinerary, specifying their origin o_k, destination d_k, earliest departure time e_k from the origin, and latest arrival time l_k at the destination, as well as their carrying capacity C_k. The travel time of the arc $(i, j) \in \mathcal{A}$ is denoted by t_{ij}. The maximum acceptable travel time of the crowd-courier $k \in \mathcal{K}$ is T_k, which satisfies $t_{od} \leq T_k \leq l_k - e_k, \forall k \in \mathcal{K}$. Each crowd-courier is allowed to carry multiple parcels but from one intermediate station only, in order to avoid lengthy detours and too many stops for parcel pick-ups. The crowd-couriers receive a certain amount of compensation according to the extra travel time they spend providing the delivery service. Let λ denote the compensation per unit of additional travel time of the crowd-couriers. Each parcel $i \in \mathcal{V}$ should be delivered before the deadline l_i. The weight of parcel i is represented by q_i. A penalty h_i will be incurred if a parcel $i \in \mathcal{V}$ fails to be matched.

Figure 1 illustrates a two-tiered crowdsourced last-mile delivery setup with two intermediate stations, three crowd-couriers, and six customers. The crowd-couriers depart from their origins, pick up the parcels from the intermediate stations, deliver them to customers in sequence, and finish at their destinations. The numbers in square brackets under the courier origin and destination, and customer location, represent the earliest departure time, latest arrival time, and deadline, respectively. The numbers on the arcs denote travel time. It can be seen that the constraints on earliest departure times, latest arrival times, and deadlines are satisfied for all the crowd-couriers.

2.3. Decisions, Objective Function, and Constraints

The objective of the PACR problem is to allocate parcels to intermediate stations and determine routes so that the crowd-couriers can deliver parcels to customer locations along the routes of their original trips so that (i) the parcels, if delivered by the crowd-couriers, arrive at the customer locations on time; (ii) the time-window, capacity, and maximum travel time constraints of the crowd-couriers are respected; (iii) the capacity constraints of the intermediate stations are satisfied; and (iv) the total delivery cost, including the total amount of compensation paid for the delivered parcels and the penalties paid for unmatched parcels, is minimized.

3. Mathematical Formulation

The PACR problem can be formulated into two kinds of models. One is an arc-flow (AF) model based on 3-index variables specifying the arcs traversed by each crowd-courier, whereas the other is an R-SP formulation based on variables corresponding to feasible routes to be defined thereafter. The model AF is presented in Appendix A. It can be solved directly by MIP solvers like CPLEX. Nevertheless, we find from preliminary experiments that the sizes (in terms of the number of parcels) of the instances that can be solved by CPLEX are quite limited. To solve instances of a practical size, we formulate the PACR problem into an R-SP model and propose a DCG algorithm to solve it. The two formulations will be compared in the numerical experiments in Subsection 5.2.

To develop the R-SP formulation, we first define a concept of a route r, denoting that a crowd-courier k_r departs from origin o_{k_r}, picks up parcels $\mathcal{V}_r$ from station s_r, delivers them

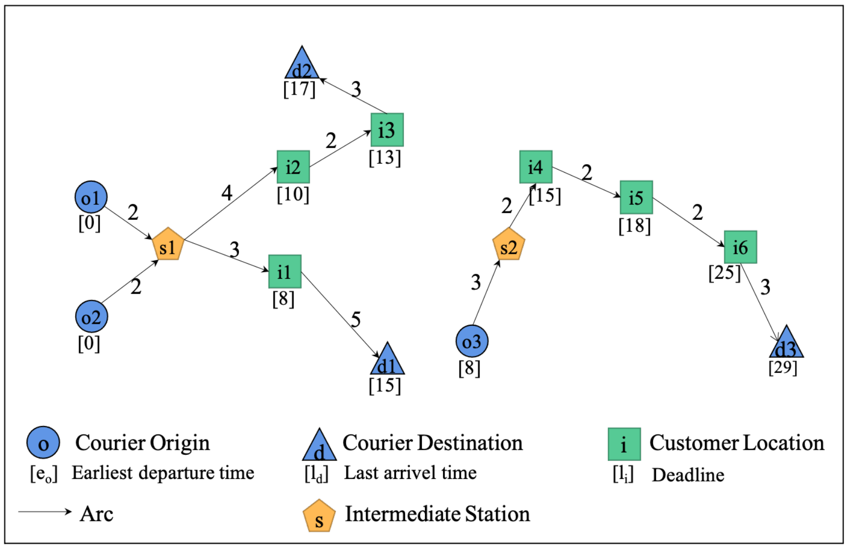


Figure 1: An illustration of a two-tiered crowdsourced last-mile delivery setup

to the customers one by one, and finally arrives at destination d_{k_r}.(see Figure 1). Let t_r, c_r, and q_r denote the travel time, compensation cost, and total parcel weight associated with route r, respectively. Meanwhile, the compensation cost is given by $c_r = \lambda(t_r - t_{o_{k_r} d_{k_r}})$. Let set $\mathcal{R}$ denote the collection of all feasible routes satisfying the following constraints:

(i) Capacity constraint: the total weight of the parcels in $\mathcal{V}_r$ must be less than or equal to C_{k_r}.

(ii) Travel time constraint: the total travel time of crowd-courier k_r does not exceed T_{k_r}.

(iii) Departure/arrival time constraint: let $u_{o_{k_r}}$ be the departure time from the origin, and $u_{d_{k_r}}$ be the arrival time at the destination; then the departure time $u_{o_{k_r}}$ and arrival time $u_{d_{k_r}}$ should satisfy $u_{o_{k_r}} \geq e_{k_r}$ and $u_{d_{k_r}} \leq l_{k_r}$.

(iv) Deadline constraint: crowd-courier k_r arrives at customer location $u_i, \forall i \in \mathcal{V}_r$ no later than deadline l_i.

Let $\mathcal{R}_i$, $\mathcal{R}_k$, and $\mathcal{R}_s$ be the subsets of routes that include parcel $i \in \mathcal{V}$, are delivered by crowd-courier $k \in \mathcal{K}$, and are allocated to station $s \in \mathcal{S}$, respectively. Let x_r be the variable that equals to 1 if route $r \in \mathcal{R}$ is included in the optimal solution. y_i represents a binary variable that equals 1 if parcel $i \in \mathcal{V}$ is not matched to any crowd-courier. The notations used for the mathematical formulations are provided in Table 1 for readability.

Table 1: Notations

Indices/Sets	
$\mathcal{S}$	Set of intermediate station locations
$\mathcal{K}$	Set of crowd-couriers
$\mathcal{V}$	Set of parcels/customer locations
$\mathcal{O}$	Set of origin of crowd-couriers
$\mathcal{D}$	Set of destination of crowd-couriers
$\mathcal{N}$	Set of all the vertices, $\mathcal{N} = \mathcal{S} \cup \mathcal{V} \cup \mathcal{O} \cup \mathcal{D}$
$\mathcal{A}$	Set of arcs connecting any two vertices
$\mathcal{R}$	Set of feasible routes
$\mathcal{R}_i$	Subset of routes including the parcel $i \in \mathcal{V}$
$\mathcal{R}_k$	Subset of routes delivered by crowd-courier $k \in \mathcal{K}$
$\mathcal{R}_s$	Subset of routes allocated to station $s \in \mathcal{S}$
s	Index of intermediate station location
k	Index of crowd-courier
i	Index of vertex
(i, j)	Index of arc
o_k	Index of origin for crowd-courier $k \in \mathcal{K}$
d_k	Index of destination for crowd-courier $k \in \mathcal{K}$
r	Index of route
Parameters	
t_{ij}	Travel time of the arc $(i, j) \in \mathcal{A}$
C_s	Capacity of station $s \in \mathcal{S}$
C_k	Capacity of crowd-courier $k \in \mathcal{K}$
T_k	Maximum travel time of the crowd-courier $k \in \mathcal{K}$
l_k	Latest arrival time at the destination for crowd-courier $k \in \mathcal{K}$
e_k	Earliest departure time from the origin for crowd-courier $k \in \mathcal{K}$
λ	Compensation of unit additional travel time
l_i	Deadline of parcel $i \in \mathcal{V}$
q_i	Weight of parcel $i \in \mathcal{V}$
h_i	Penalty for a parcel $i \in V$ failing to be delivered on time by a crowd-courier
q_r	Total weight of the parcels in route r
t_r	Travel time of route r
c_r	Compensation cost according to the additional travel time of route r
Decision variables	
x_r	Binary variable indicating if route $r \in \mathcal{R}$ being in the optimal solution
y_i	Binary variable indicating if parcel $i \in \mathcal{V}$ is matched to any crowd-courier

308 Then, the R-SP formulation of the PACR problem is given by

[R-SP]

$$\min \sum_{r \in \mathcal{R}} c_r x_r + \sum_{i \in \mathcal{V}} h_i y_i \tag{1}$$

$$\sum_{r \in \mathcal{R}_i} x_r + y_i = 1 \qquad \forall i \in \mathcal{V} \tag{2}$$

$$\sum_{r \in \mathcal{R}_k} x_r \leq 1 \qquad \forall k \in \mathcal{K} \tag{3}$$

$$\sum_{r \in \mathcal{R}_s} q_r x_r \leq C_s \qquad \forall s \in \mathcal{S} \tag{4}$$

$$x_r \in \{0, 1\} \qquad \forall r \in \mathcal{R} \tag{5}$$

$$y_i \in \{0, 1\} \qquad \forall i \in \mathcal{V} \tag{6}$$

The objective function (1) minimizes the sum of the compensation costs paid to the crowd-couriers and the penalty costs of any unmatched parcels. Constraints (2) represent that each parcel is either served by a crowd-courier or left unmatched. Constraints (3) ensure that a crowd-courier can undertake one route at most. Constraints (4) present the capacity constraints of the intermediate stations. The other constraints for a specific route have been implicitly incorporated in the definition of feasible routes. However, due to the large number of routes, the R-SP model cannot be solved directly by MIP solvers. We will thus develop a customized DCG algorithm to solve it in the next section.

4. Solution Method

4.1. DCG Algorithm

Since the number of routes in the R-SP model is huge, instead of enumerating all routes, we need to identify a subset of feasible and good-quality routes for the R-SP formulation. The traditional way to do this is by column generation, in which a time-consuming NP-hard elementary shortest path problem with restricted constraints problem is solved iteratively. For higher computational efficiency, instead of finding feasible and good-quality routes directly, we will identify a set of matches, defined as the combination of an intermediate station, a crowd-courier, and a set of parcels/customers, regardless of the sequence in which the customers are visited, that are likely to produce feasible and good-quality routes. Note that each match may correspond to multiple feasible routes with different visiting sequences for the same set of customers, but only the one associated with the minimum cost has the potential to be included in the optimal solution to the R-SP formulation. Thus, the total number of matches will be much smaller than the number of routes.

As such, we propose a novel DCG algorithm for the R-SP formulation illustrated in Figure 2. We will first enumerate the matches that are likely to produce feasible routes for the R-SP formulation, based on a set of feasible conditions, in Subsection 4.1.1. We will also take advantage of a machine learning model introduced in Subsection 4.2 to quickly predict the minimal travel time of the route embedded in a match. The prediction results will then be leveraged to enhance the feasibility check and eliminate matches that are likely to lead to unfeasible routes. To further check whether a match corresponds to a feasible route, we will need to solve an NP-hard TSPTW for each match, which is time-consuming. Before doing so, in Subsection 4.1.2, we will identify good-quality matches from the match pool using a column generation procedure for an M-SP formulation, based on the match set, with the same structure as the R-SP formulation. Given the predicted travel time, the column generation procedure can be performed quickly. Because of the same structure, the M-SP formulation can be viewed as an approximation of the R-SP formulation. Therefore, good-quality matches are likely to produce good-quality routes. Finally, in Subsection 4.1.3, the selected matches will be converted into routes by solving the TSPTWs. Then, the R-SP formulation based on the identified feasible and good-quality routes will be solved via a MIP solver to obtain a near-optimal solution to the PACR problem. In the following subsections, we will elaborate on the DCG algorithm.

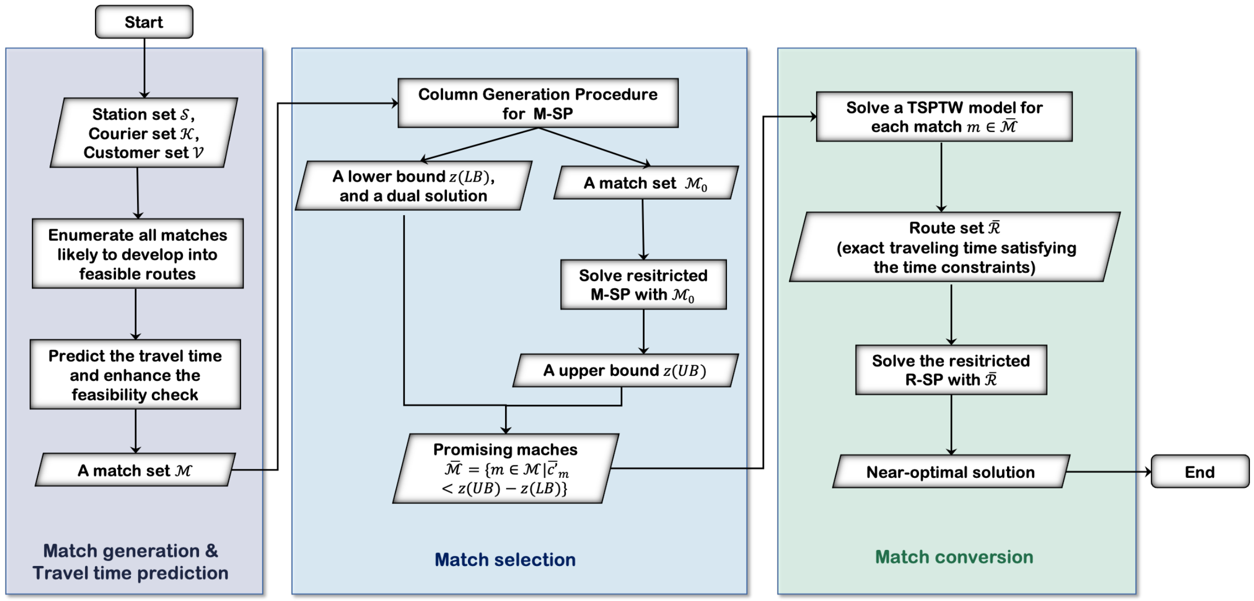

Start
Station set $\mathcal{S}$, Courier set $\mathcal{K}$, Customer set $\mathcal{V}$
Enumerate all matches likely to develop into feasible routes
Predict the travel time and enhance the feasibility check
A match set $\mathcal{M}$
Match generation & Travel time prediction
Column Generation Procedure for M-SP
A lower bound $z(LB)$, and a dual solution
A match set $\mathcal{M}_0$
Solve resitricted M-SP with $\mathcal{M}_0$
A upper bound $z(UB)$
Promising maches $\bar{\mathcal{M}} = \{m \in \mathcal{M} | \bar{c}_m < z(UB) - z(LB)\}$
Match selection
Solve a TSPTW model for each match $m \in \bar{\mathcal{M}}$
Route set $\bar{\mathcal{R}}$ (exact traveling time satisfying the time constraints)
Solve the resitricted R-SP with $\bar{\mathcal{R}}$
Near-optimal solution
End
Match conversion

Figure 2: Flowchart of the proposed DCG algorithm

4.1.1. Match generation

Let match $m = (s_m, k_m, \mathcal{V}_m)$ represent the combination of an intermediate station s_m, a crowd-courier k_m, and a set of parcels $\mathcal{V}_m$. All the locations involved in match m are grouped into set $\mathcal{N}_m$ and $\mathcal{A}_m = \{(i, j) \mid i \in \mathcal{N}_m, j \in \mathcal{N}_m, (i, j) \in \mathcal{A}\}$ is the arc set associated with $\mathcal{N}_m$. Let q_m represent the total weight of the parcels in $\mathcal{V}_m$. The minimal travel time of the route embedded in the match is denoted by t_m and the cost of the match by $c_m = \lambda(t_m - t_{o_{k_m} d_{k_m}})$. The travel time t_m can be derived exactly by solving an NP-hard traveling salesman problem (TSP), which is computationally expensive. Instead, we take advantage of a machine learning model, which will be introduced in the next subsection, to quickly predict t_m. In this subsection, we will present the match generation procedure illustrated in Algorithm 1, used to generate the set $\mathcal{M}$ of matches that are likely to provide feasible routes for the R-SP formulation, based on constraints (i)-(iv).

Firstly, to improve the computational performance, many incompatible matches that cannot produce feasible routes can be neglected based on the following conditions (lines 4-15 in Algorithm 1):

(1) If the total travel time of route $o_k \rightarrow s \rightarrow d_k$ exceeds the maximum travel time T_k or the crowd-courier cannot reach their destination before l_k, all matches including crowd-

courier k and station s can be neglected, since the time constraints would be violated even if there were no parcel delivery.

(2) If the weight of parcel i exceeds the capacity of crowd-courier k, all matches including crowd-courier k and parcel i can be neglected due to the capacity constraint of the crowd-courier.

(3) If the total travel time of a route $o_k \rightarrow s \rightarrow i \rightarrow d_k$ exceeds the maximum travel time T_k or the crowd-courier cannot reach the customer/destination before the deadline l_i/ l_k using route $o_k \rightarrow s \rightarrow i \rightarrow d_k$, any match including crowd-courier k, station s, and any parcel set $\mathcal{V}_m$ containing parcel i can be neglected. If a route where the crowd-courier only delivers parcel i is infeasible, any route where the crowd-courier is delivering parcel i and other parcels will also be infeasible.

Let Ω_{sk} be the set of parcels that cannot be delivered from station s by crowd-courier k and let $\Omega'_{sk} = \mathcal{V} \setminus \Omega_{sk}$ be the complement of $\mathcal{V}$ with respect to Ω_{sk}. Having removed the incompatible pairs of stations and crowd-couriers, the match set $\mathcal{M}$ should be constituted of any possible combination of station $s_m \in \mathcal{S}$, crowd-courier $k_m \in \mathcal{K}$, and any parcel subset $\mathcal{V}_m \subseteq \Omega'_{sk}$ with total weight no larger than C_k. For each compatible pair of station s_m and crowd-courier k_m, to enumerate all the parcel subsets $\mathcal{V}_m \subseteq \Omega'_{sk}$ satisfying the capacity constraint, a parcel set enumeration procedure is designed. First, we find a maximum-sized subset of Ω'_{sk} with total weight no larger than C_k (lines 17-21 in Algorithm 1). Let w_{sk} denote the maximum number of parcels the crowd-courier k can pick up from station s. Then, we can obtain all possible parcel subsets by enumerating all the combinations of the set Ω'_{sk} with number of parcels no larger than w_{sk}.

To check whether match m can provide a feasible route, we could solve an NP-hard TSPTW for $\mathcal{N}_m$ and $\mathcal{A}_m$, which would be very time consuming. In contrast, we employ the predicted minimal travel time of the route embedded in the match. The predicted travel time can be further used to enhance the feasibility check according to the following rules:

(1) The travel time constraint is violated if the predicted travel time $t_m > T_{k_m}$.

(2) The departure/arrival time constraint is violated if the crowd-courier cannot arrive at the destination before l_{k_m}, i.e., $e_{k_m} + t_m > l_{k_m}$.

(3) The deadline constraint is violated if a customer $i \in \mathcal{V}_m$, supposed to be the predecessor of the destination d_{k_m}, would have a delayed delivery even if the crowd-courier departed at the earliest departure time e_{k_m}, i.e., $e_{k_m} + t_m - t_{id_{k_m}} > l_i$.

Matches violating these rules are eliminated from $\mathcal{M}$. Note that the effectiveness of the feasibility check may be affected by the accuracy of the travel time prediction, as some undesired matches might be included in $\mathcal{M}$ or desired matches excluded. However, the numerical experiments in Subsection 5.2.1 show that this has little impact on the solution quality.

Algorithm 1: Match generation procedure

```
Input: Station set $\mathcal{S}$, crowd-courier set $\mathcal{K}$, and customer set $\mathcal{V}$
Output: Feasible match set $\mathcal{M}$
$\mathcal{M} \leftarrow \emptyset$
for $s \in S$ do
    for $k \in \mathcal{K}$ do
        $t \leftarrow t_{o_k s} + t_{sd_k}$
        if $t \leq T_k$ and $e_k + t \leq l_k$ then
            $\Omega_{sk} \leftarrow \emptyset$   ▹ $\Omega_{sk}$ is the set of parcels incompatible with station $s$ and crowd-courier $k$
            for $i \in \mathcal{V}$ do
                if $q_i > C_k$ then
                    $\Omega_{sk} \leftarrow \Omega_{sk} \cup \{i\}$   ▹ Violating constraint (i)
                else
                    $t \leftarrow t_{o_k s} + t_{si} + t_{id_k}$
                    if $t + e_k > l_k$ then
                        $\Omega_{sk} \leftarrow \Omega_{sk} \cup \{i\}$   ▹ Violating constraint (ii) and (iii)
                    else if $e_k + t - t_{id_k} > l_i$ then
                        $\Omega_{sk} \leftarrow \Omega_{sk} \cup \{i\}$   ▹ Violating constraint (iv)
            $\Omega'_{sk} \leftarrow \mathcal{V} \setminus \Omega_{sk}$, $\Omega''_{sk} \leftarrow \mathcal{V} \setminus \Omega_{sk}$, $w_{sk} \leftarrow 0$, $q \leftarrow 0$
            do
                Find the parcel $i^* \in \Omega''_{sk}$ with the minimum weight $q_{i^*}$.
                $q \leftarrow q + q_{i^*}$, $w_{sk} \leftarrow w_{sk} + 1$
                $\Omega''_{sk} = \Omega''_{sk} \setminus \{i^*\}$
            while $q \leq C_k$
            for $\mathcal{V}' \subset \mathcal{V}, |\mathcal{V}'| \leq w_{sk}$ do
                if $\sum_{i \in \mathcal{V}'} q_i \leq C_k$ and $\mathcal{V}' \subseteq \Omega'_{sk}$ then
                    $m \leftarrow (s, k, \mathcal{V}')$
                    ▹ Invoke XGBoost model to predict the travel time of match $m$
                    $t_m \leftarrow XGBoost(m)$
                    if $t_m \leq T_k$ and $e_k + t_m - t_{id_k} \leq l_i, \forall i \in \mathcal{V}'$ then
                        $c_m \leftarrow \lambda(t_m - t_{o_k d_k})$
                        $\mathcal{M} = \mathcal{M} \cup \{m\}$
```

4.1.2. Match selection

Given the match set generated in Subsection 4.1.1, we can develop the following M-SP formulation with the same structure as the R-SP formulation:

[M-SP]

$$\min \sum_{m\in\mathcal{M}} c_m x_m + \sum_{i\in\mathcal{V}} h_i y_i \tag{7}$$

$$\sum_{m\in\mathcal{M}_i} x_m + y_i = 1 \qquad \forall i \in \mathcal{V} \tag{8}$$

$$\sum_{m\in\mathcal{M}_k} x_m \leq 1 \qquad \forall k \in \mathcal{K} \tag{9}$$

$$\sum_{m\in\mathcal{M}_s} q_m x_m \leq C_s \qquad \forall s \in \mathcal{S} \tag{10}$$

$$x_m \in \{0,1\} \qquad \forall m \in \mathcal{M} \tag{11}$$

$$y_i \in \{0,1\} \qquad \forall i \in \mathcal{V} \tag{12}$$

Because it has the same structure, the M-SP model can be viewed as an approximation of the R-SP model. Therefore, good-quality matches are likely to produce good-quality routes. In this section, we will introduce a match selection procedure for obtaining good-quality matches from the match pool by employing the column generation procedure for the M-SP formulation.

The match selection procedure is based on a conclusion drawn by Baldacci et al. (2002) in solving a capacitated p-median problem (CPMP) for an SP formulation. Baldacci et al. (2002) stated that an optimal CPMP solution could be obtained by replacing the whole column set $\mathcal{M}$ in the SP formulation with the subset $\bar{\mathcal{M}}$ defined as follows: $\bar{\mathcal{M}} = \{m \in \mathcal{M} \mid \bar{c}_m < z(UB) - z(LB)\}$, where $z(UB)$ and $z(LB)$ are the cost of a feasible solution of the CPMP and the dual of the linear relaxation of the SP formulation, called DSP. $\bar{c}_m$ represents the reduced cost of column $m \in \mathcal{M}$, corresponding to the feasible solution of DSP with cost $z(LB)$. Fortunately, although the structure of our M-SP formulation is different from that in Baldacci et al. (2002), a similar conclusion can be derived and proved, as detailed below.

For ease of description, we first denote the linear relaxation of the M-SP formulation by LM-SP. The optimal value of the LM-SP will provide a valid lower bound of the M-SP. Let $\pi_i (i \in \mathcal{V})$, $\eta_k (k \in \mathcal{K})$, and $\xi_s (s \in \mathcal{S})$ denote the dual variables associated with constraints (2) - (4), respectively. The dual of the LM-SP, termed DM-SP, is formulated as follows.

[DM-SP]

$$\max \sum_{i\in\mathcal{V}} \pi_i + \sum_{k\in\mathcal{K}} \eta_k + \sum_{s\in\mathcal{S}} C_s \xi_s \tag{13}$$

$$\sum_{i\in\mathcal{V}_m} \pi_i + \eta_{k_m} + q_m \xi_{s_m} \leq c_m \qquad \forall m \in \mathcal{M} \tag{14}$$

$$\pi_i \leq h_i \qquad \forall i \in \mathcal{V} \tag{15}$$

$$\eta_k \le 0 \qquad \forall k \in \mathcal{K} \tag{16}$$

$$\xi_s \le 0 \qquad \forall s \in \mathcal{S} \tag{17}$$

Let $(\boldsymbol{\pi}', \boldsymbol{\eta}', \boldsymbol{\xi}')$ be a feasible solution to DM-SP with cost $z(LB)$. According to the duality theory, any feasible solution of DM-SP provides a valid lower bound for the M-SP. The reduced cost of a match m is given by:

$$\bar{c}_m = c_m - \sum_{i \in \mathcal{V}_m} \pi'_i - \eta'_{k_m} - q_m \xi'_{s_m} \tag{18}$$

Proposition 1. Let $z(UB)$ and $z(LB)$ be the costs of feasible solutions to the M-SP and DM-SP formulations, respectively, and $(\boldsymbol{\pi}', \boldsymbol{\eta}', \boldsymbol{\xi}')$ be a dual solution corresponding to $z(LB)$. Any optimal solution of the M-SP with cost less than $z(UB)$ cannot contain any match $m \in \mathcal{M}$ whose reduced cost is greater than or equal to $z(UB) - z(LB)$.

Proof. Let $\mathbf{x}'$ be a feasible solution of the M-SP formulation with objective value $z(UB)$ and $\mathcal{M}' = \{m \mid x'_m = 1\}$. From Eq. (18), we have

$$\sum_{m \in \mathcal{M}'} \bar{c}_m = \sum_{m \in \mathcal{M}'} c_m - \sum_{m \in \mathcal{M}'} \sum_{i \in \mathcal{V}_m} \pi'_i - \sum_{m \in \mathcal{M}'} \eta'_{k_m} - \sum_{m \in \mathcal{M}'} q_m \xi'_{s_m}.$$

Since $\mathbf{x}'$ is a feasible solution,we have

$$\sum_{m \in \mathcal{M}'} \sum_{i \in \mathcal{V}_m} \pi'_i = \sum_{i \in \mathcal{V}} \pi'_i$$

Let $\mathcal{K}(\mathbf{x}') \subseteq \mathcal{K}$ be the subset of couriers utilized for the solution $\mathbf{x}'$. Then, since $\boldsymbol{\eta}' \le \mathbf{0}$, we have

$$\sum_{m \in \mathcal{M}'} \eta'_{k_m} = \sum_{k \in \mathcal{K}(\mathbf{x}')} \eta'_k \ge \sum_{k \in \mathcal{K}} \eta'_k.$$

Let $\mathcal{M}_s(\mathbf{x}')$ be the subset of matches allocated to station $s \in \mathcal{S}$ for solution $\mathbf{x}'$. Similarly, as $\boldsymbol{\xi}' \le \mathbf{0}$ and $\sum_{m \in \mathcal{M}_s(\mathbf{x}')} q_m \le C_s$, we derive

$$\sum_{m \in \mathcal{M}'} q_m \xi'_{s_m} = \sum_{s \in \mathcal{S}} \sum_{m \in \mathcal{M}_s(\mathbf{x}')} q_m \xi'_s \ge \sum_{s \in \mathcal{S}} C_s \xi'_s.$$

The previous three expressions indicate that the following relationship holds:

$$\sum_{m \in \mathcal{M}'} \sum_{i \in \mathcal{V}_m} \pi'_i + \sum_{m \in \mathcal{M}'} \eta'_{k_m} + \sum_{m \in \mathcal{M}'} q_m \xi'_{s_m} \ge \sum_{i \in \mathcal{V}} \pi'_i + \sum_{k \in \mathcal{K}} \eta'_k + \sum_{s \in \mathcal{S}} C_s \xi'_s,$$

and hence

$$\sum_{m \in \mathcal{M}'} \bar{c}_m \le \sum_{m \in \mathcal{M}'} c_m - \sum_{i \in \mathcal{V}} \pi'_i - \sum_{k \in \mathcal{K}} \eta'_k - \sum_{s \in \mathcal{S}} C_s \xi'_s.$$

Since $z(UB) = \sum_{m\in\mathcal{M}'} c_m$ and $z(LB) = \sum_{i\in\mathcal{V}} \pi'_i - \sum_{k\in\mathcal{K}} \eta'_k - \sum_{s\in\mathcal{S}} C_s\xi'_s$, we obtain

$$z(UB) \geq \sum_{m\in\mathcal{M}'} \bar{c}_m + z(LB). \tag{19}$$

Suppose we have optimal solution $\mathbf{x}^*$ of the M-SP model with optimal value $z^* < z(UB)$ and $\mathcal{M}^* = \{m \in \mathcal{M} \mid x^*_m = 1\}$ while there is a column $m^* \in \mathcal{M}^*$ satisfying $\bar{c_{m^*}} \geq z(UB) - z(LB)$. Noticing that $(\boldsymbol{u}', \boldsymbol{\eta}', \boldsymbol{\xi}')$ is a feasible solution of DM-SP, from constraint (14), the relationship $\bar{c}_m \geq 0, \forall m \in \mathcal{M}$ holds. According to inequality (19), we obtain

$$\begin{aligned} z^* &\geq \sum_{m\in\mathcal{M}^*} \bar{c}_m + z(LB) \\ &= \bar{c}_{m^*} + \sum_{m\in\mathcal{M}^*\setminus\{m^*\}} \bar{c}_m + z(LB) \\ &\geq z(UB) + \sum_{m\in\mathcal{M}^*\setminus\{m^*\}} \bar{c}_m \geq z(UB) \end{aligned}$$

This contradicts our previous statement, $z^* < z(UB)$.

Therefore, we must conclude that any optimal solution of the M-SP model with cost less than $z(UB)$ cannot contain any match $m \in \mathcal{M}$ whose reduced cost is greater than or equal to $z(UB) - z(LB)$. □

Proposition 1 states that an optimal solution to the M-SP model must be included in subset $\bar{\mathcal{M}} = \{m \in \mathcal{M} \mid \bar{c}_m < z(UB) - z(LB)\}$. Based on this proposition, we propose a match selection procedure for obtaining the subset $\bar{\mathcal{M}}$. The match selection procedure has two primary steps, as illustrated in Figure 2. In the first step, a valid lower bound as well as the dual solution are obtained using a column generation procedure, and the subset $\mathcal{M}_0 \subseteq \mathcal{M}$ of matches are generated in this procedure. Accordingly, a restricted M-SP formulation is formulated based on the matches in subset $\mathcal{M}_0$. By solving this formulation, we can obtain a valid upper bound of the M-SP model. In the second step, given the lower bound, the corresponding dual solution, and the upper bound, the subset $\bar{\mathcal{M}} \subseteq \mathcal{M}$ of matches whose reduced costs with respect to the given dual solution are less than the gap between the upper and the lower bound will be generated. In the following, we discuss this match selection procedure in detail.

Firstly, we use the column generation procedure to obtain the optimal solution to LM-SP, corresponding to a valid lower bound of M-SP. Let LM-SP($\mathcal{M}_0$) be the restricted linear relaxation of M-SP, obtained from LM-SP by replacing $\mathcal{M}$ with the subset $\mathcal{M}_0 \subseteq \mathcal{M}$. Based on the LP optimality condition, the optimal solution of LM-SP($\mathcal{M}_0$) is also the optimal solution of LM-SP if the reduced cost $\bar{c}_m$ of any match $m \in \mathcal{M}\setminus\mathcal{M}_0$ is non-negative. Therefore, the column generation procedure does not directly solve the LM-SP with a full match set $\mathcal{M}$. Instead, it repeatedly solves LM-SP($\mathcal{M}_0$), generates one or more matches with negative reduced costs, and adds them into $\mathcal{M}_0$. If no such match is found, the procedure terminates and an optimal solution to the current LM-SP($\mathcal{M}_0$), which is also an optimal solution to LM-SP, is obtained. Otherwise, one or more matches with negative reduced costs are added into $\mathcal{M}_0$ and another column generation iteration is invoked.

In the traditional column generation procedure for the VRP, a dynamic programming (label setting/correcting algorithm) or heuristic approach is applied to generate one or more matches with negative reduced costs in each iteration. In this paper, given the c_m of each match $m \in \mathcal{M}$ obtained from the prediction results of the XGBoost model and the current dual solution, we can immediately compute the corresponding reduced cost $\bar{c}_m$ using Eq. (18). In each iteration, we add ϵ matches with minimum negative reduced cost to $\mathcal{M}_0$, where ϵ is a predefined parameter. Using a priority queue, this procedure can be processed very quickly. To control the size of $\mathcal{M}_0$, whenever its size becomes larger than a predefined parameter ϵ_{max}, we remove the $|\mathcal{M}_0| - \epsilon_{max}$ matches with the highest reduced cost from $\mathcal{M}_0$.

As such, we can obtain the valid lower bound $z(LB)$ as well as the corresponding dual solution $(\boldsymbol{\pi}', \boldsymbol{\eta}', \boldsymbol{\xi}')$ associated with LM-SP($\mathcal{M}_0$) via the column generation procedure. In addition, a valid upper bound $z(UB)$ is obtained by solving a restricted M-SP, formulated upon the subset $\mathcal{M}_0$, via an MIP solver. Given valid lower bound $z(LB)$, corresponding dual solution $(\boldsymbol{\pi}', \boldsymbol{\eta}', \boldsymbol{\xi}')$, and valid upper bound $z(UB)$, we can obtain the subset $\bar{\mathcal{M}} = \{m \in \mathcal{M} \mid \bar{c}_m < z(UB) - z(LB)\}$, where $\bar{c}_m$ can be calculated as $\bar{c}_m = c_m - \sum_{i \in \mathcal{V}_m} \pi_i{}' - \eta'_{k_m} - \xi'_{s_m}$. According to Proposition 1, subset $\bar{\mathcal{M}}$ is a set of good-quality matches including the optimal solutions of the M-SP formulation. The size of $\bar{\mathcal{M}}$ is much smaller than that of $\mathcal{M}$, as is verified in Subsection 5.2.3.

4.1.3. Match conversion

The previous subsection introduces the match selection procedure used to obtain a good-quality match set $\bar{\mathcal{M}}$. Since $\bar{\mathcal{M}}$ contains the optimal solution of the M-SP model, we can obtain the optimal solution by solving the restricted M-SP model with $\bar{\mathcal{M}}$. However, the optimal solution to the M-SP model might deviate from that of the R-SP model, since the prediction error in the match cost c_m may affect the feasibility of the time constraints as well as the accuracy of the objective value. In addition, we are not able to obtain the routes of the crowd-couriers since the machine learning model focuses on predicting the travel time rather than the visiting sequence. Therefore, to reduce the negative influence of prediction error, we propose a match conversion procedure for transforming the good-quality match set $\bar{\mathcal{M}}$ into a feasible and good-quality route set $\bar{\mathcal{R}}$. To this end, we will solve a TSPTW for each match $m \in \bar{\mathcal{M}}$. By solving the following TSPTW, we can check whether the match satisfies time constraints (ii)-(iv) while also obtaining the explicit route and exact routing cost.

[TSPTW]

$$\min \sum_{(i,j) \in \mathcal{A}'_m} t_{ij} x_{ij} \tag{20}$$

$$Eq.\ (B.2) - (B.5) \tag{21}$$

$$u_j \geq u_i + t_{ij} - M(1 - x_{ij}) \qquad \forall (i,j) \in \mathcal{A}'_m \tag{22}$$

$$u_{s_m} - t_{o_{k_m} s_m} \geq e_{k_m} \tag{23}$$

$$u_{d_{k_m}} \leq l_{k_m} \tag{24}$$

$$u_i \leq l_i \qquad \forall i \in \mathcal{V}_m \tag{25}$$

$$u_{d_{k_m}} - u_{s_m} \leq T_{k_m} - t_{o_{k_m} s_m} \tag{26}$$

$$x_{ij} \in \{0, 1\} \qquad \forall (i, j) \in \mathcal{A}'_m \tag{27}$$

If the TSPTW model is feasible for match m, i.e., the time constraints are satisfied, we can convert each match $m \in \bar{\mathcal{M}}$ into a feasible route. The obtained feasible routes are grouped into $\bar{\mathcal{R}}$. Correspondingly, we can derive a restricted R-SP model by replacing route set $\bar{\mathcal{R}}$ with $\mathcal{R}$. By solving the restricted R-SP model via an MIP solver, we can obtain a near-optimal solution to the R-SP model, including the crowd-couriers' routes. The performance of the match conversion is verified in the experiment section (Subsection 5.2.3). Note that the match conversion procedure can be implemented in parallel to speed up the computation, since the TSPTW of each match can be solved independently.

4.2. Travel Time Prediction Based on Machine Learning

This subsection introduces the machine learning model used to predict the minimal travel time of the route embedded in a match. Previous research has proposed approximations for the tour length of the TSP and the vehicle routing problem (VRP) (Beardwood et al., 1959; Daganzo, 2005). However, such approximations are effective only when the number of customers in a given route is large (Shen and Qi, 2007). In general, a crowd-courier may not be willing to deliver too many parcels at once. Therefore, we utilize the machine learning model to predict the travel time of the route of a crowd-courier. Liu et al. (2021) also employed a machine learning method to predict the travel time for a food delivery problem. Integrating the machine learning model into the optimization formulation, to ensure its tractability, they chose relatively simple machine learning models, e.g., a linear regression model. In contrast, since the travel time prediction is independent of the optimization formulation in our work, we can adopt more complex machine learning models with better performance and accuracy.

This paper employs a machine learning model based on XGBoost for fast prediction. The mapping of a given match to the minimal travel time of its embedded route can be considered as a regression problem, which can be solved by XGBoost very efficiently. XGBoost is a scalable end-to-end tree boosting system pioneered by Chen and Guestrin (2016). It has been wildly utilized by data scientists to achieve state-of-the-art results in both industrial and academic fields. The specific internal formulations of XGBoost can be found in Chen and Guestrin (2016).

Note that a variety of machine learning models can be applied to predict the travel time, such as linear regression, a tree model, the support vector product, or a neural network. However, in this paper, we focus on incorporating the machine learning model into the optimization algorithm rather than exploring the performance of different machine learning models. Therefore, we only focus on feature engineering and label collection.

In this work, we consider two cases of labels associated with match m, depending on whether the crowd-couriers strictly follow the routes recommended by the platform or not. Specifically, if the crowd-couriers strictly perform the routes provided by the platform, the labels are supposed to be the objective value of the TSP formulation. In practice, the crowd-couriers' travel routes might deviate from the planned delivery sequences as the crowd-couriers often have their own routing preferences and their real routing behavior may be intricate to be modeled explicitly. In this case, the labels will be obtained directly from the actual routes recorded in the historical data set. Liu et al. (2021) discussed this case in food delivery. They utilized the machine learning method to predict couriers' travel time based on the industrial partner's historical data. They demonstrated the importance of learning behavioral aspects from operational data. Based on the practical delivery data, the experimental results showed that the machine learning model had significantly smaller errors than the TSP solution, even with a small number of locations in a route. A similar method was also discussed by Bayram et al. (2022) focusing on the picking process in warehouse management.

The features can be classified into the number of locations, visiting area, distance, dispersion, and their interactive terms (Liu et al., 2021; Cavdar and Sokol, 2015; Chien, 1992; Kwon et al., 1995). In the literature, the traveler departs from the depot, visits a set of customers and returns to the depot at the end. In our work, since the crowd-couriers depart from and end up at different locations, the route structure is different from that in the literature. Correspondingly, we incorporate the location information of the destination of the crowd-courier into the features, as summarized in Table 2.

4.3. Rolling-Horizon Approach

Owing to the characteristics of the time windows of the crowd-couriers and the deadlines of the customers, a rolling-horizon framework is designed to further solve large-scale instances of the problem. The rolling-horizon approach is widely used in planning problems when the same group of decisions are repeated over time. Instead of dealing with the problem over the whole planning horizon, we will divide the time horizon into a series of smaller periods and solve them in sequence.

Figure 3 illustrates the rolling-horizon approach to the PACR problem in detail. The problem corresponding to the whole planning horizon H is solved by repeatedly addressing several sub-problems in temporal sequence. A sub-problem involves the parcels and crowd-couriers with deadlines/latest arrival times no later than the end of a sub-period t_{end}. Let T denote the length of the first sub-period, which is called the basic sub-period. At each iteration, the DCG algorithm described in Subsections 4.1.1 - 4.1.3 is invoked to solve the corresponding sub-problem. After that, the sub-period rolls forward by Δt, which is called the forward period. At the same time, the determined routes associated with parcels whose deadlines are before $t_{\text{end}} - T + \Delta t$ will be fixed. We also update the available capacity of the stations according to the fixed routes. Taking the gth iteration as an example, the parcels and crowd-couriers involved in the gth sub-problem are grouped into $\mathcal{V}_{\text{iter}_g}$ and $\mathcal{K}_{\text{iter}_g}$, respectively. First, we solve the sub-problem by invoking the DCG algorithm described in Subsection 4.1.1 - 4.1.3, deriving the determined routes $\mathcal{R}_{\text{iter}_g}$ and the unmatched parcels. Next, we fix the determined routes associated with parcels whose deadlines are before $t_{\text{end}_g} - T + \Delta t = g\Delta t$, grouping them in the set $\mathcal{R}_{\text{fix}_g} = \mathcal{R}_{\text{fix}_{g-1}} \cup \{r \in \mathcal{R}_{\text{iter}_g} \mid \exists i \in \mathcal{V}_r, s.t., l_i < g\Delta t\}$. As a result, the parcels and crowd-couriers related to $\mathcal{R}_{\text{fix}_g}$ are fixed/occupied and will not be involved in the subsequent decision processes. The fixed parcels and occupied crowd-couriers are grouped into sets $\mathcal{V}_{\text{fix}_g} = \bigcup_{r\in\mathcal{R}_{\text{fix}_g}} \mathcal{V}_r$ and $\mathcal{K}_{\text{fix}_g} = \{k_r \in \mathcal{K} \mid r \in \mathcal{R}_{\text{fix}_g}\}$, respectively. At the same time, we update the capacities of the stations according to the fixed routes $\mathcal{R}_{\text{fix}_g}$. Afterwards, the next sub-period is extended forward an additional Δt, and the corresponding sub-problem includes the unfixed parcels/unoccupied crowd-couriers whose deadlines/last arrival times are before $t_{\text{end}_{g+1}} = t_{\text{end}_g} + \Delta t$, i.e., $\mathcal{V}_{\text{iter}_{g+1}} = \{i \in \mathcal{V} \setminus \mathcal{V}_{\text{fix}_g} \mid l_i < t_{\text{end}_{g+1}}\}$ and $\mathcal{K}_{\text{iter}_{g+1}} = \{k \in \mathcal{K} \setminus \mathcal{K}_{\text{fix}_g} \mid l_k < t_{\text{end}_{g+1}}\}$. In this way, the process is iterated as the planning horizon rolls forward. As the rolling-horizon proceeds, more and more decision variables become fixed, until eventually we reach the end of the planning horizon.

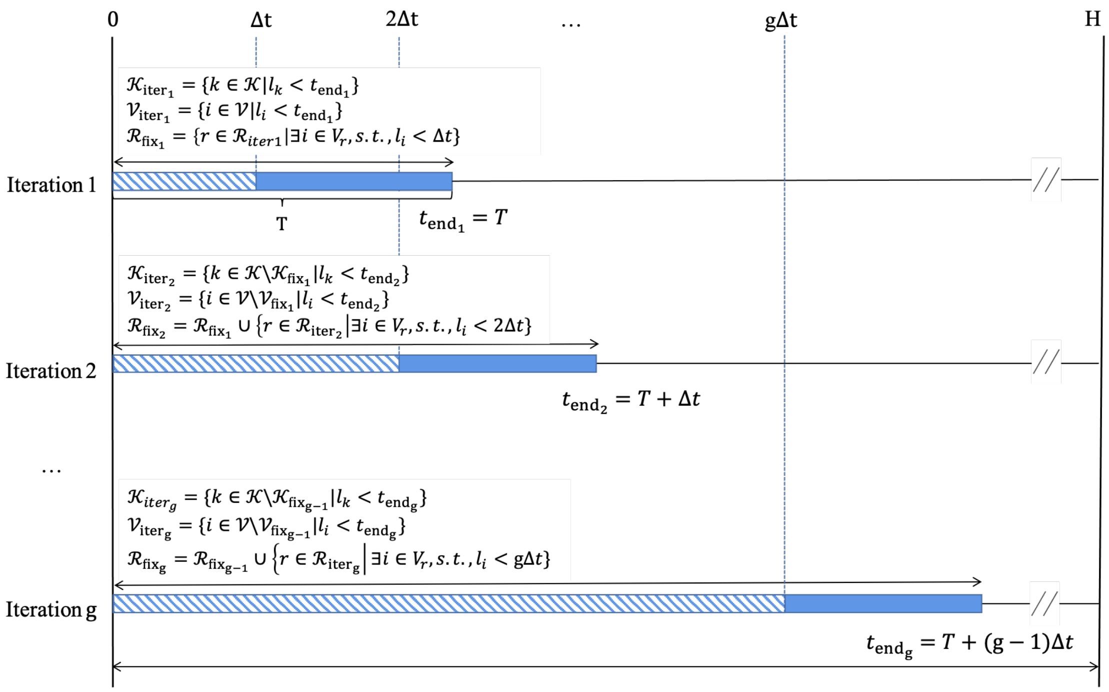

0
Δt
2Δt
…
gΔt
H
$\mathcal{K}_{\text{iter}_1} = \{k \in \mathcal{K} | l_k < t_{\text{end}_1}\}$
$\mathcal{V}_{\text{iter}_1} = \{i \in \mathcal{V} | l_i < t_{\text{end}_1}\}$
$\mathcal{R}_{\text{fix}_1} = \{r \in \mathcal{R}_{iter1} | \exists i \in V_r, s.t., l_i < \Delta t\}$
Iteration 1
T
$t_{\text{end}_1} = T$
$\mathcal{K}_{\text{iter}_2} = \{k \in \mathcal{K}\backslash\mathcal{K}_{\text{fix}_1} | l_k < t_{\text{end}_2}\}$
$\mathcal{V}_{\text{iter}_2} = \{i \in \mathcal{V}\backslash\mathcal{V}_{\text{fix}_1} | l_i < t_{\text{end}_2}\}$
$\mathcal{R}_{\text{fix}_2} = \mathcal{R}_{\text{fix}_1} \cup \{r \in \mathcal{R}_{\text{iter}_2} | \exists i \in V_r, s.t., l_i < 2\Delta t\}$
Iteration 2
$t_{\text{end}_2} = T + \Delta t$
…
$\mathcal{K}_{iter_g} = \{k \in \mathcal{K}\backslash\mathcal{K}_{\text{fix}_{g-1}} | l_k < t_{\text{end}_g}\}$
$\mathcal{V}_{\text{iter}_g} = \{i \in \mathcal{V}\backslash\mathcal{V}_{\text{fix}_{g-1}} | l_i < t_{\text{end}_g}\}$
$\mathcal{R}_{\text{fix}_g} = \mathcal{R}_{\text{fix}_{g-1}} \cup \{r \in \mathcal{R}_{\text{iter}_g} | \exists i \in V_r, s.t., l_i < g\Delta t\}$
Iteration g
$t_{\text{end}_g} = T + (g-1)\Delta t$

Figure 3: Illustration of rolling-horizon framework

Two factors determine the trade-off between overall computational efficiency and solution quality in this framework. One is the basic sub-period T, and the other is the forward period Δt. As T increases, a larger subset of parcels and crowd-couriers are included in a sub-problem, leading to a better solution, but also increased computation time and memory consumption. In contrast, more decision variables are fixed after solving each sub-problem with the increase of Δt, resulting in a smaller subset of parcels and crowd-couriers to be optimized in subsequent iterations. Moreover, the number of iterations will decrease with the increase of Δt. Therefore, an increase of Δt leads to a lower complexity of computation and a lower quality of solution. As a result, preliminary experiments are necessary to devise suitable values of Δt and T so as to strike a good balance between the computational efficiency and solution quality.

5. Numerical Experiments

The algorithm is coded in Java 8 and runs on an Intel Xeon W-2265 3.50 GHz Processor, with 128 RAM and Windows 10 Professional. We use CPLEX 20.1 as the linear programming solver for the column generation procedure and the MIP solver for the restricted M-SP/R-SP models and TSP(TW) models. In this section, we first describe the instance generation and parameter setting. We will then evaluate the performance of the DCG algorithm and the rolling-horizon approach for large instances. Next, we will explore the benefit of the joint optimization of parcel allocation and delivery routes. Finally, we will conduct a sensitivity analysis of the impacts of station dispersion and crowd-courier availability on the performance of the crowdsourced last-mile delivery system.

5.1. Instance Generation and Parameter Setting

We generate several instances with different numbers of parcels ranging from 10 to 150 and three intermediate stations within a 20 km × 20 km square region. The number of crowd-couriers is set to be half the number of parcels. The coordinates of the station locations, customer locations, and origins and destinations of the crowd-couriers are randomly generated in the region. The distance between vertices i and j, i.e., d_{ij}, is computed as their Euclidean distance. By assuming a constant speed of 50 km/h, the travel time between i and j, i.e., t_{ij}, can be obtained readily. The capacity of each intermediate station is assumed to be half the total weight of the parcels. The latest arrival time $l_k, \forall k \in \mathcal{K}$ is randomly and uniformly selected from [0, 720]. The earliest departure time $e_k, \forall k \in \mathcal{K}$ is computed as $e_k = l_k - (t_{o_k d_k} + t^k_{flex})$, where t^k_{flex} is the departure time flexibility of courier k, set to 30. The maximum travel time limitation T_k is set to $\min\{l_k - e_k, 1.5t_{o_k d_k}\}, \forall k \in \mathcal{K}$. The capacity of each crowd-courier C_k is 3. The crowd-couriers' compensation per unit additional travel time λ is set to 1. The deadline l_i of each parcel $i \in \mathcal{V}$ is uniformly and randomly selected from [0, 720]. The weight of each parcel q_i is set to 1. The penalty for an unmatched parcel, h_i, is $1.5t_{s^*_i i}$, where $t_{s^*_i i}$ is the travel time from the customer to their nearest intermediate station s^*_i. The problem instances used in the numerical experiments can be found on GitHub (https://github.com/liliw27/PACR.git).

Since the historical data of our problem is not available, in the experiment tests, we assume that the crowd-couriers will follow the routes planned by the platform so that the labels of the training set and test set are collected as the objective value of the TSP formulation presented in Appendix B.

The training set and test set are generated using a randomly generated instance with 200 parcels. The total numbers of samples in the training set and test set are 2,000,000 and 400,000, respectively. To obtain a sample set with a balanced number of parcels, we set the ratio of the numbers of matches including 2 to 4 parcels to $C^2_{200} : C^3_{200} : C^4_{200}$. The training parameters of XGBoost, including the maximum tree depth (to control tree depth), minimum child weight (to control splitting), number of rounds, and learning rate (to control the step shrinkage), are set to 6, 4, 200, and 0.1, respectively. The training objective function is the squared loss function. The evaluation metrics are the mean absolute error (MAE), root mean square error (RMSE), and mean absolute percentage error (MAPE). The MAE, RMSE, and MAPE of the training result for the test set are 1.77, 2.28, and 0.05, respectively. The importance score, the number of times a feature is used to split the data across all trees, is illustrated in Appendix C.

All the above parameters will remain unchanged throughout the numerical experiments unless stated otherwise.

5.2. Performance of the DCG Algorithm

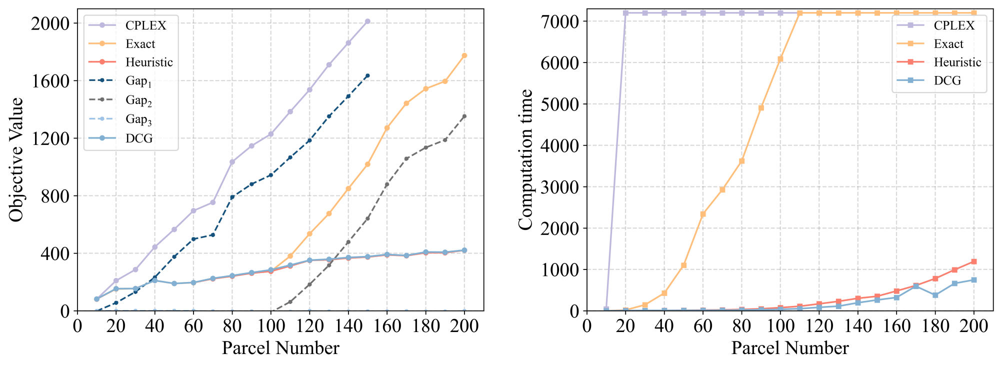
In this section, we will conduct numerical experiments to evaluate the performance of the proposed DCG algorithm by comparing it with three benchmark approaches. The first one is to use the CPLEX optimizer to solve the AF model in Appendix A. The second and third one are an exact and a heuristic algorithm for the R-SP model, respectively. We will also provide a set of results to demonstrate the efficacy of the critical algorithm components, i.e., the match selection and match conversion procedures, in the proposed DCG algorithm.

5.2.1. Comparison with benchmark approaches

To evaluate the performance of the proposed algorithm, we will compare the results obtained by the proposed DCG algorithm for solving the R-SP model against the results of using CPLEX for solving the AF model, an exact algorithm, and a heuristic algorithm. In the exact algorithm, the R-SP model is formulated with the routes obtained by solving the TSPTWs for all possible matches. Note that the so-called 'all possible matches' are obtained using Algorithm 1 as designed in Subsection 4.1.1, but the version disregarding the XGBoost prediction and corresponding feasibility check, i.e., excluding lines 25-27 of Algorithm 1. In the heuristic algorithm, the travel time of a match is obtained by employing a heuristic method instead of a machine learning model. Various approximation algorithms and heuristics have been developed to solve the TSP, such as the simulation-optimization methods, including constructive heuristics, local search, and Lin-Kernighan heuristic, as well as stochastic approximation methods (Beardwood et al., 1959; Daganzo, 2005; Perboli et al., 2018, 2017). For more details on heuristic algorithms for TSP, readers may refer to Applegate et al. (2011). In this study, we use a well-known and classical heuristic method, 2-opt, to solve the TSP model and obtain the travel time of the matches. Furthermore, the match selection procedure introduced in Subsection 4.1.2 is employed to reduce the computation time of the exact and heuristic algorithms.

The comparison results for instances with number of parcels (#Parcel) ranging from 10 to 200 are tabulated in Table 3. We report the objective value (Obj) and computation time (Time) of each method. The best objective values and optimal objective values are highlighted in bold and with an asterisk, respectively. For ease of comparison, we also represent the relative gaps RelGap$_1$, RelGap$_2$, and RelGap$_3$, computed as $\frac{obj^{*}-obj}{obj} * 100\%$, $\frac{obj^{*}-obj_1}{obj_1} * 100\%$, and $\frac{obj^{*}-obj_2}{obj_2} * 100\%$ respectively, where obj^{*}, obj, obj_1, and obj_2 represent the objective values obtained by the DCG algorithm, CPLEX, the exact algorithm, and the heuristic algorithm. As can be seen, the DCG algorithm has a much better performance than CPLEX. For instance, with 10 parcels, CPLEX solves the instance to optimality in 45 sec while the DCG algorithm obtains a near-optimal solution with a 1.2% relative gap in much less time (1 sec). For the instances with 20 to 150 parcels, the DCG algorithm can on average obtain an objective value 68% lower than that of CPLEX, in a few minutes. The objective value obtained by CPLEX is more than five times that obtained by the DCG algorithm in the instance with 150 parcels. For the instances with 160 parcels or more, CPLEX fails to obtain any valid solution due to an out-of-memory error, while the DCG algorithm manages to obtain good-quality solutions within 12 min. As for the exact algorithm, the results show that it only solves instances with up to 100 parcels optimally within a time limit of 2 h. For these ten instances, the DCG yields comparable solutions in much less computation time (11.67 sec vs. 1,393.40 sec). Out of the ten instances, the DCG can solve three instances to optimality. For the other seven instances with no more than 100 parcels, the average gap to the optimal objective is below 2%. When there are 100 or more parcels, the exact algorithm fails to solve these instances to optimality in 2 h, whereas the DCG algorithm can obtain much higher quality solutions (43.0% decrease in objective value on average) in a few minutes. The objective value obtained by the exact algorithm is more than four times that obtained by the DCG algorithm when the number of parcels is 200. Compared with the heuristic algorithm, the DCG algorithm can obtain comparable solutions with an average relative gap of 0.91% in much less computation time (34.9% decrease in the computation time on average). The advantage in computational efficiency of the DCG algorithm over the heuristic algorithm becomes more apparent when the number of parcels is large.

Based on the data in Table 3, we further visualize the solution quality and computational efficiency in Figure 4. Figure 4a shows the variations in the objective values and the gaps between the CPLEX/exact/heuristic algorithm and the DCG algorithm. The objective value increases with the increase in the number of parcels for all four methods. To be specific, the curve CPLEX is always higher than the curves Exact, Heuristic, and DCG. The curves Exact and DCG almost coincide when the number of parcels is between 10 and 100, and the curve Exact is always higher when the number of parcels is greater than 100. The curve Heuristic overlaps with that of DCG over the whole range of parcel number. Overall, this suggests that the solution quality of CPLEX is the worst, while that of the heuristic and DCG algorithms is the best. The gap between CPLEX and the DCG algorithm (curve $\boldsymbol{Gap}_1$) drastically increases with the increase in the number of parcels, implying a significant advantage of the DCG algorithm over CPLEX in larger instances. The gap between the exact algorithm and the DCG algorithm (curve $\boldsymbol{Gap}_2$) is almost zero when the number of parcels is between 10 and 100, implying that near-optimal solutions can be achieved by the DCG algorithm. When the number of parcels increases further to 200, the gap significantly increases, from 63 to 1,353, implying that, the larger the instance, the more significant an improvement in solution quality can be obtained by the DCG algorithm. The gap between the heuristic algorithm and the DCG algorithm (curve $\boldsymbol{Gap}_3$) remains zero under all values of parcel number, implying that the heuristic algorithm and the DCG algorithm have similar performance in terms of solution quality. Figure 4b shows the variation in the computation times of the four methods as the number of parcels increases. As we can see, the computation times of CPLEX and the exact algorithm increase drastically with the increase in the number of parcels. CPLEX and the exact algorithm run for over 2 h when there are more than 20 and 100 parcels, respectively. This indicates that the computation time of CPLEX and the exact algorithm increases exponentially with the increase of the problem size. In contrast,



(a) Objective value and gap (b) Computation time

Figure 4: Variations in gap and computation time with an increase in the number of parcels

the computation times of the heuristic and DCG algorithms appear much less sensitive to the problem size while the DCG algorithm shows a visible advantage with less computation time when the number of parcels exceeds 100. In particular, the time of the DCG algorithm only increases slightly, from 1 sec to 748 sec, when the number of parcels increases from 10 to 200, while, for the heuristic algorithm, it increases from 1 sec to 1,194 sec over the same range of parcel number. In summary, the outstanding performance of the proposed DCG algorithm in terms of solution quality and computational efficiency demonstrates its potential to be implemented in a real-world crowdsourced last-mile delivery system.

5.2.2. Performance on real-world instances

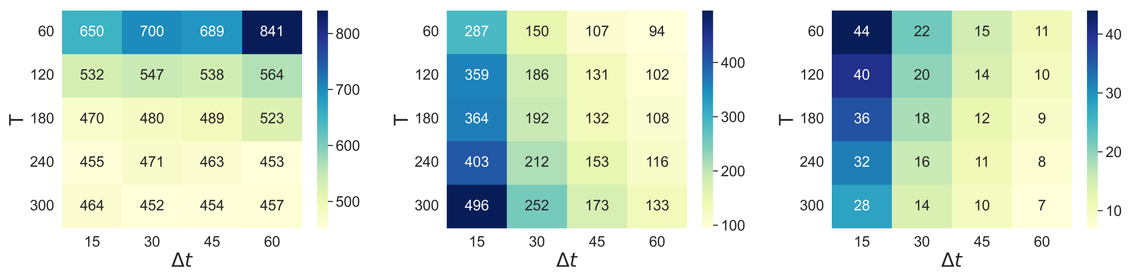
We also test the performance of the proposed DCG algorithm on a group of instances based on real-world instances in Mousavi et al. (2021) for the stochastic mobile depot and crowd-shipping problem. The instances were generated according to the 2016 version of the Transportation Tomorrow Survey (TTS) from the data management group (DMG), a comprehensive travel survey conducted every five years in the Toronto Area. The City of Toronto has 625 traffic zones, each of which could be a customer's location, intermediate station, and origin or destination of a crowd-courier. Mousavi et al. (2021) considered the instances with the number of customers ranging from 10 to 40 and the number of intermediate stations ranging from 10 to 30. The number of customers was assumed to be larger than the number of intermediate stations. The capacity of an intermediate station was set to 10. Note that some modifications have been made to the instances in this study considering the differences in problem settings. For example, we set the earliest departure time and latest arrival time of each crowd-courier, as well as the deadline for each customer, based on the aforementioned instance generation method in Subsection 5.1. The results are shown in Table 4. Similar to the test results on the randomly generated instances, it shows that the proposed DCG algorithm can obtain (near-)optimal solutions with the least computation times in most real-world instances. Particularly, the proposed DCG algorithm solves the first seven out of nine instances in the least time on average among the three approaches. As for the last two instances, the average gap obtained by the DCG algorithm to the optimal objective value remains below 3%.

5.2.3. Performance of match selection and match conversion procedures

This subsection examines the computational efficiency of the critical algorithm components, i.e., the match selection and match conversion procedures. To this end, we compare the results obtained by the DCG algorithm, including both the match selection and match conversion procedures, against two benchmark methods. The first, referred to as the 'MS' method, is similar to the DCG algorithm, but only includes the match selection procedure and disregards the match conversion procedure. The other is referred to as the 'w/o MS' approach, and excludes the match selection procedure as well. Kindly note that, without the match conversion procedure, both the MS and w/o MS methods solve the M-SP model with the match cost c_m instead of the routing cost c_r in the objective function. Therefore, to calculate the actual objective values of the MS and w/o MS methods, we need to transform the optimal solutions to the M-SP model into feasible solutions to the R-SP model by solving TSPTWs corresponding to the optimal matches. Theoretically, the TSPTWs might be infeasible, although we have never encountered this situation in our computational experiments. To maintain the feasibility of the corresponding solutions, we solve the M-SP model again after removing the matches with infeasible TSPTWs. The results are tabulated in Table 5. The objective value and computation time are reported in columns 'Obj' and 'Time'. Column 'ColNum' shows the number of matches/routes utilized to solve the M-SP/R-SP model, i.e., $|\bar{\mathcal{M}}|$ for the MS method, $|\mathcal{M}|$ for the w/o MS method, and the $|\bar{\mathcal{R}}|$ for the DCG algorithm. Column 'MatRed' indicates the fraction of the number of columns reduced by using the MS method rather than the w/o MS method, i.e., $\frac{|\mathcal{M}|-|\bar{\mathcal{M}}|}{|\mathcal{M}|}$. Column 'RelGap' reports the calculation $\frac{obj^{*}-obj_3}{obj_3} * 100\%$, where obj^{*} and obj_3 are the objective values of the DCG algorithm and MS method, respectively.

To evaluate the significance of the match selection procedure, we analyze the results of the comparison of the MS and w/o MS methods. As we can observe, when the number of parcels is no more than 160, the solutions obtained by the MS and w/o MS methods are the same. The primary reason is that, in such cases, the match selection procedure will retain all the columns belonging to the optimal solutions for the M-SP model. In terms of computation time, the MS method takes 79% less time than the w/o MS method on average. That is, the MS method is more computationally efficient than the w/o MS method, especially for larger instances. When the number of parcels is less than 50, the computation time of the MS method is slightly larger than that of the w/o MS method. This is because the extra computation time induced by the match selection procedure is slightly larger than the time saved by using the match set reduction to solve the restricted M-SP model. Moreover, it can be seen that the match selection procedure helps to reduce the size of $|\mathcal{M}|$ by over 95% on average. For the instances with 170 or more parcels, the w/o MS procedure fails to obtain any valid solution due to an out-of-memory error.

To evaluate the performance of the match conversion procedure, we compare the results of the DCG algorithm against the MS method, approaches with and without the conversion procedure, respectively. Table 5 shows that the DCG algorithm always obtains a better solution than the MS method, except in the first instance. The match conversion procedure reduces the objective value by 12% on average. Moreover, the objective value gap between the MS method and the DCG algorithm increases with the number of parcels, suggesting



(a) Objective value (b) computation time (sec) (c) Iteration number

Figure 5: Influence of Δt and T

that the match conversion procedure significantly improves the solution quality for larger instances. The performance increment over the MS method also indicates that, even if a prediction error exists, the match selection procedure can still derive a high-quality match set likely to produce high-quality routes. However, since the TSPTW needs to be solved for each selected match, the match conversion procedure leads to a slight increase in computational time (1 min on average). Nevertheless, it is worth putting up with a slight sacrifice in computational efficiency in order to attain a significant improvement in solution quality.

5.3. Performance of Rolling-Horizon Approach

To examine the influence of the pre-specified forward period Δt and basic sub-period T on the performance of the rolling-horizon method in terms of solution quality and computational efficiency, we solve the PACR problem under different values of Δt and T in ten randomly generated instances with 200 parcels. The objective values, computation times, and number of iterations obtained under different values of Δt and T are illustrated in Figure 5. As expected, a larger Δt deteriorates the solution quality while reducing the computation time. The reason is that, as Δt increases, a larger subset of decision variables is fixed in each iteration, and a shorter overlapping time interval is re-optimized, resulting in a deteriorated quality of solution. Moreover, as a direct result of a larger Δt, a smaller number of parcels and crowd-couriers are included in the sub-problem in each iteration, and fewer iterations are processed in the rolling-horizon method, leading to a lower computational complexity. In contrast, when the length of the base sub-period T increases, a larger subset of decision variables and more information about the parcels and crowd-couriers are included in each sub-problem, resulting in a better quality of solution but more computational effort.

To evaluate the performance of the rolling-horizon approach for large-scale instances, we compare the results obtained by employing the DCG algorithm in the rolling-horizon framework (DCG-RH) to those obtained from the single DCG algorithm. The latter can also be considered a special case of the rolling time horizon approach with $T = H$. Also, to strike a good balance between the solution quality and computational effort, we set T to $\min\{\bar{T}, \frac{Hn}{|V|}\}$, where $\bar{T}$ and n are two predetermined parameters for the upper bound of T and the average number of parcels included in the sub-problem of each iteration, respectively. In the following experiments, we set Δt, $\bar{T}$, and n to 45 min, 300 min, and 100, respectively. The results obtained from solving instances with 200 to 1,000 parcels are summarized in

Table 6. Column 'RelGap' is calculated from $\frac{obj^r - obj^*}{obj^*} * 100\%$, where obj^r represents the objective value obtained by the DCG-RH approach. When the number of parcels is 200, the solution quality of the DCG-RH approach and the DCG are very close (361 vs. 355), but the DCG-RH approach takes only half of the computation time of the DCG algorithm. As can be observed, when the number of parcels is 400 or above, the DCG-RH approach obtains much better solutions (68% decrease in cost on average), taking less computation time (48% decrease in computation time, on average). In particular, when the number of parcels is 1,000, the objective value obtained by the DCG-RH approach is reduced by more than 80% and it uses only 25% of the computation time consumed by the DCG algorithm. Overall, for larger instances, the DCG-RH approach brings significant improvements not only in solution quality but also in computational efficiency. According to the results represented in Figure 5, the expectation was that the solution quality obtained by the DCG algorithm ($T = H$) would be better than that of the DCG-RH approach. However, Table 6 shows that the DCG-RH approach significantly outperforms the DCG algorithm when solving large-scale instances. The main reason is that, to avoid an out-of-memory error in the DCG algorithm for large-scale instances, we set the maximal value of $|\mathcal{M}|$ to 10×10^7.

5.4. Benefit Analysis of Joint Optimization

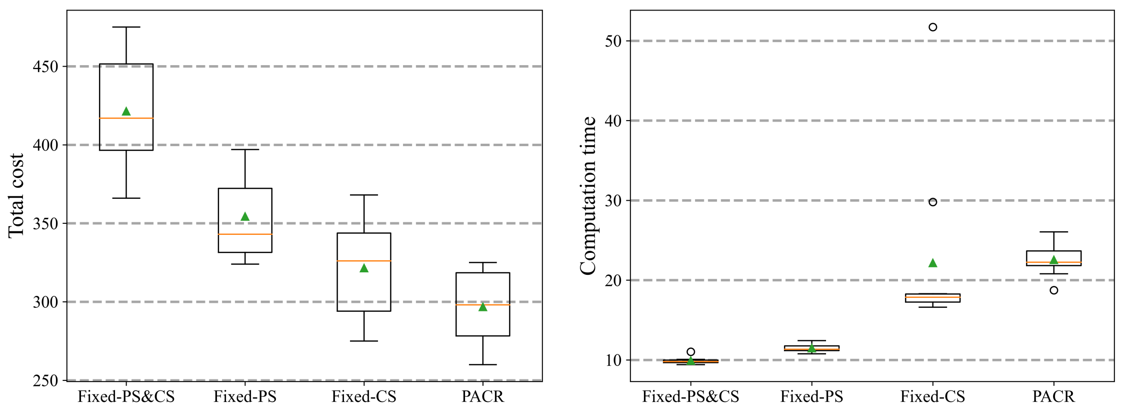
In this subsection, we evaluate the benefit of the joint optimization of the PACR problem by comparing it with three other kinds of schemes:

(1) Fixed-PS: In this scheme, each parcel is pre-allocated to the nearest station subject to the capacity constraint. In other words, the parcel allocation decisions are pre-confirmed, and the PACR problem is reduced to a crowd-courier routing problem. A similar scheme can found in Wang et al. (2016).

(2) Fixed-CS: In this scheme, each crowd-courier can only picks up parcels from the fixed station nearest to his/her origin.

(3) Fixed-PS&CS: In this scheme, we further simplify the problem by combining the former two schemes. To be specific, each parcel is pre-allocated to its nearest station, and each crowd-courier only picks up parcels from the fixed station nearest to his/her origin.

For each scheme, we test ten random instances with 100 parcels, generated using the method discussed in Subsection 5.1. Figure 6 presents the results for the total cost and computation time, using a boxplot. As can be observed, the joint optimization scheme we propose in this paper performs significantly better than the other three schemes in terms of total cost, because it considers a larger search space. On average, the joint optimization scheme reduces the objective value by 32%, 16%, and 8% compared to the Fixed-PS&CS, Fixed-PS, and Fixed-CS schemes, respectively. The comparison of the time spent by the four schemes reveals the additional complexity introduced by the extra decisions that have to be made in the joint optimization scheme. As special cases of the PACR problem considered in this paper, the other three schemes simplify the problem in different ways, thus consuming less computational effort. However, compared to the Fixed-CS scheme, which produces the lowest cost among the three benchmark schemes, the computation time of the PACR problem is more stable and robust even though their average computation times are similar.



(a) Total Cost

(b) Computation time

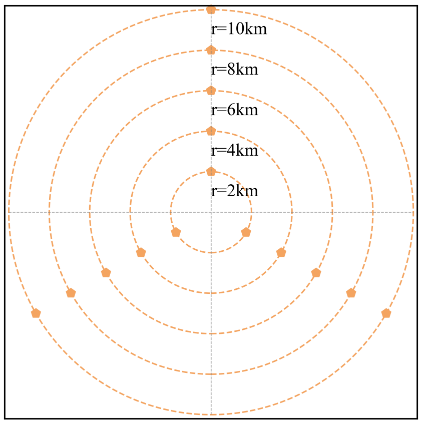
Figure 6: Performance of joint optimization

5.5. Sensitivity Analysis

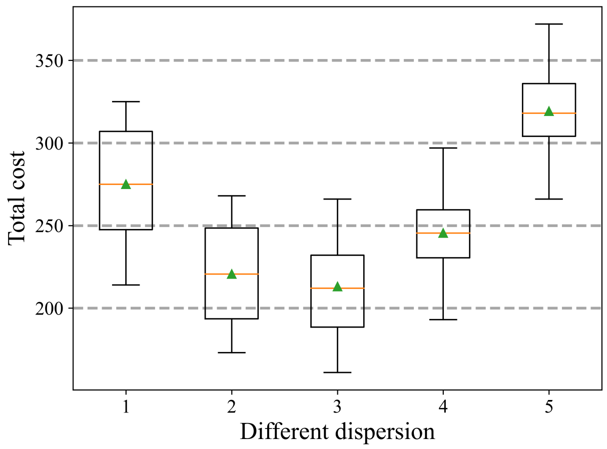
In this section, we first investigate how the degree of dispersion of the stations, measured by the average distance between each station and the central point of all the stations, affects the performance of the crowdsourced last-mile delivery system. Then, we analyze the impact of crowd-courier availability on the system's performance, with their availability measured by three metrics: the ratio of the number of crowd-couriers to the number of parcels (C/P ratio), the capacity of the crowd-couriers (CourCap), and the departure time flexibility of the crowd-couriers (DTFlex).

5.5.1. Analysis of the impact of the degree of dispersion of the stations

To evaluate the impact of the degree of dispersion of the stations, we set up scenarios with five different degrees of dispersion. As shown in Figure 7a, for simplicity, we locate three stations equally spaced out on circles with radii of 2 km, 4 km, 6 km, 8 km, and 10 km. The respective degrees of dispersion are coded as 1, 2, 3, 4, and 5, respectively. It can be seen that the distribution of the stations becomes more dispersed with the increase in radius. Figure 7b shows the total cost under the different degrees of dispersion based on averaging the results of 10 random instances with 100 parcels for each dispersion scenario. Interestingly, as the degree of dispersion increases, the total cost first drops and then rises. Specifically, the cost decreases from 275 to 213 when the degree of dispersion increases from 1 to 3. Then, it increases to 319 as the degree of dispersion increases from 3 to 5. The main reason is that the stations clustered in the center are far away from the customers located on the boundary or in the corners of the area, leading to longer routes and higher costs of delivery for those parcels. Similar results are found when the stations are distributed around the boundary. In contrast, stations lying between the center and boundary are relatively closer to any vertex, resulting in lower costs of delivery between the stations and the customers. Therefore, it is recommended that service providers should avoid to place their stations in the center or on the boundary of the delivery area.



(a)



(b)

Figure 7: Impact of station dispersion

5.5.2. Analysis of the impact of crowd-couriers' availability

In this subsection, we analyze the impact of crowd-courier availability, i.e., C/P ratio, CourCap, and DTFlex, on the system's performance. We vary each of the parameters within its feasible range while keeping the other two parameters set to the values given in Subsection 5.1. Under a particular parameter setting, we report the average results for 10 randomly generated instances with 100 parcels. Specifically, we report the total cost (TotCost), the cost of the compensation paid to the crowd-couriers (ComCost), the penalty cost for unmatched parcels (PenCost), the average compensation paid to each crowd-courier (AvgCost/C), the average compensation paid for each matched parcel (AvgCost/P), the number of matched parcels (MatP), the number of matched crowd-couriers (MatC), and the average number of parcels matched to each crowd-courier (AvgPNum).

Table 7 summarizes the results under different C/P ratios ranging from 0.1 to 1. According to Table 7, the total cost decreases as the number of available crowd-couriers increases, with the rate of decrease becoming smaller when the C/P ratio exceeds 0.6. In particular, the total cost drops sharply by 370 when the C/P ratio increases from 0.1 to 0.2, but only by 30 when the C/P ratio increases from 0.6 to 0.7. This suggests that encouraging more crowd-couriers to participate in the last-mile delivery service could produce considerable profits, but with a C/P ratio exceeding 0.6, the profit improvement would become relatively limited. The penalty cost of unmatched parcels follows a similar downward trend, and is almost eliminated once the C/P ratio exceeds 0.6. In comparison, the compensation cost shows a trend of first rising and then falling. To be specific, as the C/P ratio grows from 0.1 to 0.4, the compensation cost increases from 126 to 320, with more than three times the number of parcels being matched to the crowd-couriers. As the C/P ratio grows further, up to 1.0, the compensation cost decreases slightly to 216. This may be attributed to the fact that the chance of matching parcels to crowd-couriers that have to make fewer detours rises as more crowd-couriers become available, resulting in a lower compensation cost. Even though an increase in the C/P ratio could bring massive profits to the last-mile delivery service provider, it would reduce the appeal to the crowd-couriers since they receive less than half the compensation when the C/P ratio increases from 0.1 to 1.0. Moreover, the probability of a crowd-courier matching with a parcel drops from 96% to 36% as the C/P ratio increases from 0.1 to 1. The decrease in the matching probability might thus be another factor that dampens the crowd-couriers' enthusiasm for engaging in last-mile delivery services.

The results for different departure time flexibility are summarized in Table 8. It can be observed that the total cost displays a downward trend with the increase in departure time flexibility, with the decrease rate becoming smaller when the departure time flexibility exceeds 20 min. To be specific, the total cost falls from 943 to 538 when the departure time flexibility increases from 5 min to 10 min. In contrast, it only decreases from 369 to 362 when departure time flexibility increases from 20 min to 25 min (the cost reduction falls from 405 to 7). The results indicate that encouraging crowd-couriers with more flexible departure times could significantly reduce the delivery costs of the crowdsourced last-mile delivery system, but that cost reduction would become limited for departure time flexibility of more than 20 min. In other words, from an operational perspective, a 20 min departure time flexibility might be sufficient for the crowd-couriers to generate effective and profitable routes. The reduction of the total cost is mainly attributable to a reduction in the penalty costs paid for unmatched parcels, while the cost of the compensation paid to the crowd-couriers displays an opposite trend. This is because more parcels and crowd-couriers can be matched. As can be seen, about three times as many parcels and twice as many crowd-couriers are matched when the departure time flexibility increases from 5 min to 20 min. Nevertheless, the increment in the compensation is always much smaller than the reduction in the penalty cost, resulting in a decrease in the total cost. Another interesting finding is that the crowd-couriers may be able to obtain more than 2.5 times the compensation if they adjust their departure time flexibility from 5 min to 20 min. The primary reason is that each crowd-courier can then deliver more parcels on average and travel longer routes, enabled by their larger detour tolerance, which depends on their departure time flexibility.

The impact of the crowd-courier's capacity on the system's performance is tabulated in Table 9. It shows that a capacity increase can significantly reduce the total cost by 702 (as much as 71%), although the reduction's slope becomes close to zero when the capacity exceeds 3. This suggests that the platform could easily attract crowd-couriers with larger capacities by offering higher compensation to them, as long as the additional compensation cost did not exceed the cost reduction brought about by using high-capacity crowd-couriers. The penalty costs for unmatched parcels are virtually eliminated, as about 98% of the parcels can be matched when the capacity is 5. The compensation cost increases from 250 to 348 as the capacity increases from 1 to 2, with almost twice the number of parcels matched. Then, the compensation cost gradually decreases to 277 as the capacity grows further to 5 because parcels with similar customer locations can be grouped into a single route of a crowd-courier with greater capacity, resulting in a lower compensation cost. Due to batching similar customers, the service provider only pays 50% of the compensation for each matched parcel when the capacity increases from 1 to 5. Nevertheless, this also shows that crowd-couriers with more capacity can obtain more compensation, whereas they have less probability of matching to parcels and less capacity utilization.

6. Conclusions

In this paper, we investigate a crowdsourced last-mile delivery problem in which a group of crowd-couriers is utilized to deliver parcels from intermediate stations to customers. We explore the joint optimization of the parcel allocation to the stations and the delivery routes of the crowd-couriers. We propose two mathematical formulations for the PACR problem: a conventional arc-flow formulation and a route-based formulation. To solve the route-based formulation, we design a novel data-driven column generation algorithm, based on the notion of a match, under a rolling-horizon framework. The data-driven column generation algorithm contains three key components: First, we enumerate all possible matches that are likely to produce feasible routes, based on a compatibility condition and the cost estimated by the machine learning method. Second, we propose a match selection procedure to select a subset of the high-potential matches that are likely to produce high-quality routes, by applying the column generation method to a match-based set-partitioning formulation. Third, for each match in the high-potential subset, a traveling salesman problem with time windows is solved to convert the matches into good-quality routes with exact travel times, and to ensure the time constraints are met. A rolling time horizon can be employed to further improve the computational efficiency for large instances (≥ 200 customers).

Moreover, extensive numerical experiments are conducted to verify the effectiveness of the proposed algorithm. The results show that the data-driven column generation algorithm can find a (near-)optimal solution much faster than using CPLEX for the arc-flow formulation and an exact algorithm for the route-based set-partitioning formulation. The proposed rolling-horizon framework can handle significantly larger instances with more than 200 parcels within a few minutes. We also compare the proposed joint optimization model with three special cases of the problem. The results indicate that the joint optimization of parcel allocation and delivery routing significantly decreases the costs of the crowdsourced last-mile delivery system. This finding demonstrates the necessity of joint optimization and hence validates the significance of this study.

Finally, the effects of the degree of dispersion of the stations and crowd-courier availability on the performance of the crowdsourced delivery system are analyzed. We find that significant cost savings can be obtained by locating the stations between the boundary and the centre of the city. In addition, large profits can be obtained with an increase in the crowd-courier availability, represented by the number of available crowd-couriers, their departure time flexibility, and their capacity. However, the profit improvements tend to be limited after crowd-courier availability exceeds a certain threshold.

Future research can explore several further directions. First, we assume that the crowd-couriers must accept the parcel delivery tasks matched to them, whereas the crowd-couriers might reject the matched tasks in reality. Incorporating the choice behavior of the crowd-couriers would offer more alignment with reality. Second, the dedicated vehicles of the service company can be considered to ensure service quality. Third, the location problem of the intermediate stations under uncertainty, such as uncertain demand, is another topic worth exploring. Fourth, designing a reasonable compensation scheme that can strike a good balance between attracting more crowd-couriers and reducing the cost of service provider is significant for the viability of this new business model. Lastly, other the operational aspect, the economic, social, and environmental implications of this kind of crowdsourced delivery pattern are also important issues to be investigated in the future.

Acknowledgements

The authors gratefully thank Dr. Kianoush Mousavi, Prof. Merve Bodur, and Prof. Matthew J. Roorda for providing us the problem instances generated based on the data of transportation tomorrow survey (TTS) from the data management group (DMG) in the City of Toronto. The work described in this paper was supported by National Natural Science Foundation of China (No. 71901189) and the Research Grants Council of the Hong Kong Special Administrative Region, China (Project No. PolyU 15221821), and the Research Committee of The Hong Kong Polytechnic University (UAKQ). This research was also supported by National Natural Science Foundation of China (Grant nos. 71971090, 71821001, 7210010522).

References

Ahamed, T., Zou, B., Farazi, N.P., Tulabandhula, T., 2021. Deep reinforcement learning for crowdsourced urban delivery. Transportation Research Part B: Methodological 152, 227–257.

Alnaggar, A., Gzara, F., Bookbinder, J.H., 2021. Crowdsourced delivery: A review of platforms and academic literature. Omega 98, 102139.

Applegate, D.L., Bixby, R.E., Chvátal, V., Cook, W.J., 2011. The traveling salesman problem, in: The Traveling Salesman Problem. Princeton University Press.

Archetti, C., Savelsbergh, M., Speranza, M.G., 2016. The vehicle routing problem with occasional drivers. European Journal of Operational Research 254, 472–480.

Arslan, A.M., Agatz, N., Kroon, L., Zuidwijk, R., 2019. Crowdsourced delivery—a dynamic pickup and delivery problem with ad hoc drivers. Transportation Science 53, 222–235.

Baldacci, R., Hadjiconstantinou, E., Maniezzo, V., Mingozzi, A., 2002. A new method for solving capacitated location problems based on a set partitioning approach. Computers & Operations Research 29, 365–386.

Bayram, V., Baloch, G., Gzara, F., Elhedhli, S., 2022. Optimal order batching in warehouse management: A data-driven robust approach. INFORMS Journal on Optimization .

Beardwood, J., Halton, J.H., Hammersley, J.M., 1959. The shortest path through many points, in: Mathematical Proceedings of the Cambridge Philosophical Society, Cambridge University Press. pp. 299–327.

Behrend, M., Meisel, F., 2018. The integration of item-sharing and crowdshipping: Can collaborative consumption be pushed by delivering through the crowd? Transportation Research Part B: Methodological 111, 227–243.

Bowes, P., 2021. Pitney Bowes Parcel Shipping Index. https://www.pitneybowes.com/us/shipping-index.html.

Cavdar, B., Sokol, J., 2015. A distribution-free tsp tour length estimation model for random graphs. European Journal of Operational Research 243, 588–598.

Chen, T., Guestrin, C., 2016. Xgboost: A scalable tree boosting system, in: Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 785–794.

Chen, W., Mes, M., Schutten, M., 2018. Multi-hop driver-parcel matching problem with time windows. Flexible Services and Manufacturing Journal 30, 517–553.

Chien, T.W., 1992. Operational estimators for the length of a traveling salesman tour. Computers & Operations Research 19, 469–478.

Daganzo, C., 2005. Logistics systems analysis. Springer Science & Business Media.

Dayarian, I., Savelsbergh, M., 2020. Crowdshipping and same-day delivery: Employing in-store customers to deliver online orders. Production and Operations Management 29, 2153–2174.

Fadda, E., Perboli, G., Tadei, R., 2018. Customized multi-period stochastic assignment problem for social engagement and opportunistic iot. Computers & Operations Research 93, 41–50.

Fatehi, S., Wagner, M.R., 2022. Crowdsourcing last-mile deliveries. Manufacturing & Service Operations Management 24, 791–809.

Gdowska, K., Viana, A., Pedroso, J.P., 2018. Stochastic last-mile delivery with crowdshipping. Transportation Research Procedia 30, 90–100.

Insider, 2021. Worldwide ecommerce continues double-digit growth following pandemic push to online. https://www.emarketer.com/content/worldwide-ecommerce-continues-double-digit-growth-following-pandemic-push-online.

Janinhoff, L., Klein, R., Scholz, D., 2022. Multitrip vehicle routing with delivery options: A data-driven application to the parcel industry. Available at SSRN 4130046 .

Kafle, N., Zou, B., Lin, J., 2017. Design and modeling of a crowdsource-enabled system for urban parcel relay and delivery. Transportation Research Part B: Methodological 99, 62–82.

Kwon, O., Golden, B., Wasil, E., 1995. Estimating the length of the optimal tsp tour: An empirical study using regression and neural networks. Computers & Operations Research 22, 1039–1046.

Le, T.V., Stathopoulos, A., Van Woensel, T., Ukkusuri, S.V., 2019. Supply, demand, operations, and management of crowd-shipping services: A review and empirical evidence. Transportation Research Part C: Emerging Technologies 103, 83–103.

Liu, S., He, L., Max Shen, Z.J., 2021. On-time last-mile delivery: Order assignment with travel-time predictors. Management Science 67, 4095–4119.

Macrina, G., Pugliese, L.D.P., Guerriero, F., Laganà, D., 2017. The vehicle routing problem with occasional drivers and time windows, in: International Conference on Optimization and Decision Science, Springer. pp. 577–587.

Macrina, G., Pugliese, L.D.P., Guerriero, F., Laporte, G., 2020. Crowd-shipping with time windows and transshipment nodes. Computers & Operations Research 113, 104806.

Mancini, S., Gansterer, M., 2022. Bundle generation for last-mile delivery with occasional drivers. Omega 108, 102582.

Morabit, M., Desaulniers, G., Lodi, A., 2021. Machine-learning–based column selection for column generation. Transportation Science 55, 815–831.

Morabit, M., Desaulniers, G., Lodi, A., 2022. Machine-learning-based arc selection for constrained shortest path problems in column generation. arXiv preprint arXiv:2201.02535 .

Mousavi, K., Bodur, M., Roorda, M.J., 2021. Stochastic last-mile delivery with crowd-shipping and mobile depots. Transportation Science .

Nieto-Isaza, S., Fontaine, P., Minner, S., 2022. The value of stochastic crowd resources and strategic location of mini-depots for last-mile delivery: a benders decomposition approach. Transportation Research Part B: Methodological 157, 62–79.

Perboli, G., Brotcorne, L., Bruni, M.E., Rosano, M., 2021. A new model for last-mile delivery and satellite depots management: The impact of the on-demand economy. Transportation Research Part E: Logistics and Transportation Review 145, 102184.

Perboli, G., Gobbato, L., Maggioni, F., 2017. A progressive hedging method for the multi-path travelling salesman problem with stochastic travel times. IMA Journal of Management Mathematics 28, 65–86.

Perboli, G., Rosano, M., Saint-Guillain, M., Rizzo, P., 2018. Simulation–optimisation framework for city logistics: an application on multimodal last-mile delivery. IET Intelligent Transport Systems 12, 262–269.

Pugliese, Di Puglia, L., Ferone, D., Festa, P., Guerriero, F., Macrina, G., 2022a. Solution approaches for the vehicle routing problem with occasional drivers and time windows. Optimization Methods and Software , 1–31.

Pugliese, L.D.P., Ferone, D., Festa, P., Guerriero, F., Macrina, G., 2022b. Combining variable neighborhood search and machine learning to solve the vehicle routing problem with crowd-shipping. Optimization Letters , 1–23.

Raviv, T., Tenzer, E.Z., 2018. Crowd-shipping of small parcels in a physical internet.

Sampaio, A., Savelsbergh, M., Veelenturf, L.P., Van Woensel, T., 2020. Delivery systems with crowd-sourced drivers: A pickup and delivery problem with transfers. Networks 76, 232–255.

Shen, Y., Sun, Y., Li, X., Eberhard, A., Ernst, A., 2022. Enhancing column generation by a machine-learning-based pricing heuristic for graph coloring, in: Proceedings of the AAAI Conference on Artificial Intelligence, pp. 9926–9934.

Shen, Z.J.M., Qi, L., 2007. Incorporating inventory and routing costs in strategic location models. European Journal of Operational Research 179, 372–389.

Tahir, A., Quesnel, F., Desaulniers, G., El Hallaoui, I., Yaakoubi, Y., 2021. An improved integral column generation algorithm using machine learning for aircrew pairing. Transportation Science 55, 1411–1429.

Vincent, F.Y., Jodiawan, P., Redi, A.P., 2022. Crowd-shipping problem with time windows, transshipment nodes, and delivery options. Transportation Research Part E: Logistics and Transportation Review 157, 102545.

Voigt, S., Kuhn, H., 2021. Crowdsourced logistics: The pickup and delivery problem with transshipments and occasional drivers. Networks .

Wang, Y., Zhang, D., Liu, Q., Shen, F., Lee, L.H., 2016. Towards enhancing the last-mile delivery: An effective crowd-tasking model with scalable solutions. Transportation Research Part E: Logistics and Transportation Review 93, 279–293.

Yıldız, B., 2021. Express package routing problem with occasional couriers. Transportation Research Part C: Emerging Technologies 123, 102994.

Yildiz, B., Savelsbergh, M., 2019. Service and capacity planning in crowd-sourced delivery. Transportation Research Part C: Emerging Technologies 100, 177–199.

Appendix A. Arc-flow formulation

To formulate the PACR problem, the following decision variables are used:

- x_{ij}^k : a binary variable that equals 1 if crowd-courier $k \in K$ traverses arc $(i, j) \in \mathcal{A}$, and 0 otherwise;
- y_i : a binary variable that equals 1 if parcel $i \in \mathcal{V}$ is not matched to any crowd-courier;
- z_s^k: the total weight of the parcels picked up from intermediate station $s \in \mathcal{S}$ by crowd-courier $k \in \mathcal{K}$;
- u_i: the arrival time at vertex $i \in \mathcal{N} \setminus \mathcal{S}$;
- u_s^k: the arrival time of crowd-courier $k \in \mathcal{K}$ at station $s \in \mathcal{S}$.

In addition, let M represent a positive large number. With the above notation, the arc-flow formulation of the PACR problem is presented as follows:

[AF]

$$\min \ \lambda \sum_{k\in\mathcal{K}} \Big(\sum_{(i,j)\in\mathcal{A}} t_{ij} x_{ij}^k - t_{od}^k \Big) + \sum_{i\in\mathcal{V}} h_i y_i \tag{A.1}$$

$$\sum_{k\in\mathcal{K}} \sum_{i\in\mathcal{S}\cup\mathcal{V}} x_{ij}^k + y_j = 1 \qquad \forall j\in\mathcal{V} \tag{A.2}$$

$$\sum_{j\in\mathcal{S}\cup\{d_k\}} x_{o_k j}^k = 1 \qquad \forall k\in\mathcal{K} \tag{A.3}$$

$$\sum_{i\in\mathcal{V}\cup\{o_k\}} x_{id_k}^k = 1 \qquad \forall k\in\mathcal{K} \tag{A.4}$$

$$x_{o_k s}^k = \sum_{j\in\mathcal{V}} x_{s,j}^k \qquad \forall k\in\mathcal{K}, s\in\mathcal{S} \tag{A.5}$$

$$\sum_{i\in\mathcal{S}\cup\mathcal{V}} x_{ij}^k = \sum_{i\in\mathcal{V}\cup\{d_k\}} x_{ji}^k \qquad \forall k\in\mathcal{K}, j\in\mathcal{V} \tag{A.6}$$

$$\sum_{i\in\mathcal{S}\cup\mathcal{V}} \sum_{j\in\mathcal{V}} q_j x_{ij}^k \le C_k \qquad \forall k\in\mathcal{K} \tag{A.7}$$

$$\sum_{k\in\mathcal{K}} z_s^k \le C_s \qquad \forall s\in\mathcal{S} \tag{A.8}$$

$$z_s^k \ge \sum_{i\in\mathcal{S}\cup\mathcal{V}} \sum_{j\in\mathcal{V}} q_j x_{ij}^k - M(1 - x_{o_k s}) \qquad \forall s\in\mathcal{S}, k\in\mathcal{K} \tag{A.9}$$

$$z_s^k \le C_k x_{o_k s}^k \qquad \forall s\in\mathcal{S}, k\in\mathcal{K} \tag{A.10}$$

$$z_s^k \le \sum_{i\in\mathcal{S}\cup\mathcal{V}} \sum_{j\in\mathcal{V}} q_j x_{ij}^k \qquad \forall s\in\mathcal{S}, k\in\mathcal{K} \tag{A.11}$$

$$u_j \ge u_i + t_{ij} - M\Big(1 - \sum_{k\in\mathcal{K}} x_{ij}^k\Big) \qquad \forall i\in\mathcal{V}, j\in\mathcal{V} \tag{A.12}$$

$$u_s^k \ge u_{o_k} + t_{o_k s} - M(1 - x_{o_k s}^k) \qquad \forall k\in\mathcal{K}, s\in\mathcal{S} \tag{A.13}$$

$$u_j \ge u_s^k + t_{sj} - M(1 - x_{sj}^k) \qquad \forall k\in\mathcal{K}, s\in\mathcal{S}, j\in\mathcal{V} \tag{A.14}$$

$$u_{d_k} \ge u_i + t_{id_k} - M(1 - x_{id_k}^k) \qquad \forall k\in\mathcal{K}, i\in\mathcal{V} \tag{A.15}$$

$$u_{o_k} \ge e_k \qquad \forall k\in\mathcal{K} \tag{A.16}$$

$$u_{d_k} \le l_k \qquad \forall k\in\mathcal{K} \tag{A.17}$$

$$u_i \le l_i \qquad \forall i\in\mathcal{V} \tag{A.18}$$

$$u_{d_k} - u_{o_k} \le T_k \qquad \forall k\in\mathcal{K} \tag{A.19}$$

$$x_{ij}^k \in \{0,1\} \qquad \forall (i,j)\in\mathcal{A}, k\in\mathcal{K} \tag{A.20}$$

$$y_i \in \{0,1\} \qquad \forall i\in\mathcal{V} \tag{A.21}$$

$$z_s^k \in \mathbb{R}_+ \qquad \forall s\in\mathcal{S}, k\in\mathcal{K} \tag{A.22}$$

$$u_i \in \mathbb{R}_+ \qquad \forall i\in\mathcal{N}\setminus\mathcal{S} \tag{A.23}$$

$$u_s^k \in \mathbb{R}_+ \qquad \forall s\in\mathcal{S}, k\in\mathcal{K} \tag{A.24}$$

The objective function (A.1) minimizes the compensation cost paid for the additional travel time of the crowd-couriers and the penalty cost of unmatched parcels. Constraints (A.2) represent that each parcel is either served by a crowd-courier or left unmatched. Constraints (A.3) impose that each crowd-courier departs from origin o_k and goes to a station or directly to destination d_k, while constraints (A.4) ensure that each crowd-courier ends up at destination d_k, having traveled there from origin o_k or a customer location. Constraints (A.5) and (A.6) are the flow preservation constraints of the stations and customer locations, respectively. Inequalities (A.7) and (A.8) represent the capacity constraints of the crowd-couriers and the intermediate stations, respectively. Constraints (A.9) - (A.11) impose that z_s^k is the total weight carried by crowd-courier k from intermediate station s if crowd-courier k visits station s, and 0 otherwise. Constraints (A.12) - (A.17) ensure that the crowd-courier's earliest departure time and latest arrival time are respected. To be more specific, constraints (A.12) require that, if customer j is served right after customer i, the arrival time at customer j is not earlier than the arrival time at customer i plus travel time t_{ij}. Similarly, constraints (A.13) - (A.15) stipulate the arrival time of crowd-courier k at intermediate station s, when having traveled from the crowd-courier's origin o_k, the arrival time at customer j, when having traveled from intermediate station s, and the arrival time at the crowd-courier's destination d_k, when having traveled from customer i, respectively. Constraints (A.18) enforce the deadlines of the parcels. Constraints (A.19) impose the maximum acceptable travel times of the crowd-couriers.

The model AF can be solved directly by MIP solvers like CPLEX. Nevertheless, after some preliminary experiments, we find that the sizes (in terms of number of parcels) of the instances that can be solved by CPLEX are quite limited. To solve instances of a practical size, we reformulate the PACR problem into an R-SP model and propose a DCG algorithm to solve it. The two formulations will be compared based on numerical experiments in Subsection 5.2.

Appendix B. TSP formulation for label collection

[TSP]

$$\min \sum_{(i,j)\in\mathcal{A}'_m} t_{ij}x_{ij} \tag{B.1}$$

$$\sum_{i\in\{s_m\}\cup\mathcal{V}_m} x_{ij} = 1 \qquad \forall j\in\mathcal{V}_m \tag{B.2}$$

$$\sum_{j\in\{d_{k_m}\}\cup\mathcal{V}_m} x_{ij} = 1 \qquad \forall i\in\mathcal{V}_m \tag{B.3}$$

$$\sum_{j\in\mathcal{V}_m} x_{s_m j} = 1 \tag{B.4}$$

$$\sum_{i\in\mathcal{V}_m} x_{i d_{k_m}} = 1 \tag{B.5}$$

$$\sum_{(i,j)\in\mathcal{A}(\mathcal{N}')} x_{ij} \le |\mathcal{N}'| - 1 \qquad \forall \mathcal{N}' \subset \mathcal{N}_m \setminus \{o_{k_m}\} \tag{B.6}$$

$$x_{ij} \in \{0, 1\} \qquad \forall (i, j) \in \mathcal{A}'_m \tag{B.7}$$

The set $\mathcal{A}(\mathcal{N}')$ denotes the set of arcs associated with $\mathcal{N}'$. For ease of presentation, we define the arc set $\mathcal{A}'_m = \mathcal{A}_m \setminus \{(o_{k_m}, s_m)\}$, excluding the arc (o_{k_m}, s_m) which is indeed traversed by crowd-courier k_m, with $x_{o_{k_m}s_m} = 1$ and not needing to be optimized.

Appendix C. Feature importance score of XGBoost model

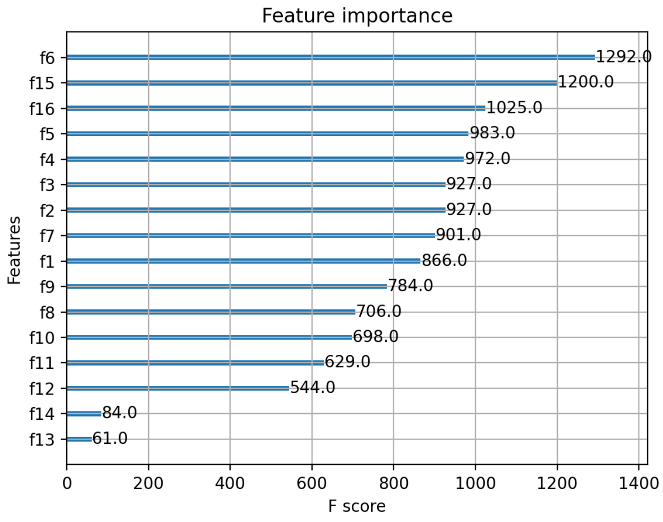


Figure C.1: Feature importance score of XGBoost model

Table 2: Definitions of the prediction features

Features	Definition
f$_1$: $\bar{d}_s$	The average distance between the station and the customer locations
f$_2$: d_s^{max}	The longest distance between the station and the customer locations
f$_3$: d_s^{min}	The shortest distance between the station and the customer locations
f$_4$: $\bar{d}_d$	The average distance between the customer locations and the destination of the crowd-courier
f$_5$: d_d^{max}	The longest distance between the customer locations and the destination of the crowd-courier
f$_6$: d_d^{min}	The shortest distance between the customer locations and the destination of the crowd-courier
f$_7$: lat_1	The maximum latitudinal difference between a pair of customer locations
f$_8$: lat_2	The maximum latitudinal difference between a pair of customer locations (including the station and the destination of the crowd-courier)
f$_9$: lat_3	The average latitudinal difference between a pair of customer locations
f$_{10}$: lat_4	The average latitudinal difference between a pair of customer locations (including the station and the destination of the crowd-courier)
f$_{11}$: lng_1	The maximum longitudinal difference between a pair of customer locations
f$_{12}$: lng_2	The maximum longitudinal difference between a pair of customer locations (including the station and the destination of the crowd-courier)
f$_{13}$: lng_3	The average longitudinal difference between a pair of customer locations
f$_{14}$: lng_4	The average longitudinal difference between a pair of customer locations (including the station and the destination of the crowd-courier)
f$_{15}$: R_1	The area of the smallest rectangle covering the customer locations
f$_{16}$: R_2	The area of the smallest rectangle covering the customer locations, the station, and the destination of the crowd-courier

Table 3: Comparison of the proposed algorithm, CPLEX, exact algorithm, and heuristic algorithm

#Parcel	DCG		CPLEX		Exact		Heuristic		RelGap_1	RelGap_2	RelGap_3
	Obj	Time	Obj	Time	Obj	Time	Obj	Time			
10	83	1	82*	45	82*	2	82*	1	1.2%	1.2%	1.2%
20	154	6	210	7,200	152*	19	154	6	-26.7%	1.3%	0.0%
30	155*	8	287	7,200	155*	146	155*	8	-46.0%	0.0%	0.0%
40	211*	9	444	7,200	211*	428	211*	10	-52.5%	0.0%	0.0%
50	190*	10	566	7,200	190*	1,103	190*	13	-66.4%	0.0%	0.0%
60	197	12	696	7,200	196*	2,342	197	18	-71.7%	0.5%	0.0%
70	226	14	754	7,200	223*	3,905	223*	23	-70.0%	1.4%	1.4%
80	245	19	1,036	7,200	241*	2,327	241*	35	-76.4%	1.7%	1.7%
90	266	27	1,147	7,200	262*	3,621	262*	50	-76.8%	1.5%	1.5%
100	285	36	1,229	7,200	275*	6,089	275*	78	-76.8%	3.6%	3.6%
110	318	52	1,385	7,200	381	7,200	312	111	-77.0%	-16.5%	1.9%
120	352	85	1,537	7,200	536	7,200	351	170	-77.1%	-34.3%	0.2%
130	359	113	1,711	7,200	676	7,200	355	230	-79.0%	-46.9%	1.1%
140	371	196	1,863	7,200	850	7,200	367	303	-80.1%	-56.4%	1.1%
150	377	263	2,013	7,200	1,019	7,200	374	352	-81.3%	-63.0%	0.8%
160	392	323	-	-	1,272	7,200	389	481	-	-69.2%	0.8%
170	385	595	-	-	1,443	7,200	383	613	-	-73.3%	0.5%
180	409	380	-	-	1,544	7,200	404	781	-	-73.5%	1.2%
190	408	664	-	-	1,596	7,200	404	993	-	-74.4%	1.0%
200	422	748	-	-	1,775	7,200	421	1,194	-	-76.2%	0.2%

Table 4: Comparison of the proposed algorithm, exact algorithm, and heuristic algorithm on real-world instances

(#Parcel,#Station)	DCG		Exact		Heuristic	
	Obj	Time	Obj	Time	Obj	Time
(10, 10)	25	0.6	25	2.2	25	0.9
(20, 10)	42	5.9	42	29.9	42	5.7
(20, 20)	25	6.1	25	46.5	25	5.7
(30, 10)	60	6.1	60	92.0	60	6.6
(30, 20)	36	6.4	36	193.9	36	8.6
(30, 30)	33	7.3	33	288.2	33	10.7
(40, 10)	75	7.7	75	284.1	75	10.7
(40, 20)	44	10.2	43	566.4	43	16.8
(40, 30)	39	12.4	38	771.3	38	23.1

Table 5: Performance of match selection and match conversion procedures

#Parcel	w/o MS			MS			DCG			MatRed	RelGap
	Obj	MatNum	Time	Obj	MatNum	Time	Obj	MatNum	Time		
10	83	124	0.5	83	123	1.1	83	123	1.5	0.8%	0.0%
20	159	2,043	0.3	159	597	0.3	154	597	5.7	70.8%	-3.1%
30	164	14,643	0.5	164	1,067	0.6	155	1,067	8.1	92.7%	-5.5%
40	227	43,058	0.9	227	1,555	1.0	211	1,555	8.6	96.4%	-7.0%
50	197	114,931	3.4	197	1,730	2.3	190	1,730	9.7	98.5%	-3.6%
60	227	226,641	8.8	227	3,713	4.4	197	3,713	11.7	98.4%	-13.2%
70	264	366,238	13.5	264	3,769	6.1	226	3,769	14.2	99.0%	-14.4%
80	278	615,984	23.8	278	3,637	10.5	245	3,637	18.9	99.4%	-11.9%
90	304	994,867	50.5	304	5,280	17.3	266	5,280	27.2	99.5%	-12.5%
100	322	1,618,806	75.6	322	6,445	26.8	285	6,445	36.3	99.6%	-11.5%
110	378	2,193,646	111.9	378	8,087	36.1	318	8,087	51.6	99.6%	-15.9%
120	405	2,962,344	149.4	405	12,706	54.0	352	12,706	85.3	99.6%	-13.1%
130	394	4,348,944	392.8	394	12,756	72.7	359	12,756	112.6	99.7%	-8.9%
140	429	5,782,371	603.5	429	33,244	108.7	371	33,244	195.6	99.4%	-13.5%
150	424	7,292,741	700.2	424	49,633	140.6	377	49,633	263.4	99.3%	-11.1%
160	462	9,310,156	1,011.3	462	57,690	172.0	392	57,690	323.1	99.4%	-15.2%
170	-	-	-	447	134,454	248.1	385	134,454	595.2	-	-13.9%
180	-	-	-	498	34,738	297.1	409	34,738	380.5	-	-17.9%
190	-	-	-	516	105,913	466.0	408	105,913	663.8	-	-20.9%
200	-	-	-	538	106,638	467.3	422	106,638	748.5	-	-21.6%

Table 6: Performance of proposed rolling-horizon approach

#Parcel	DCG-RH				DCG		RelGap
	T	Δt	Obj	Time	Obj	Time	
200	300	45	361	213	355	420	1.69%
400	180	45	647	439	2,563	1,522	-74.76%
600	120	45	940	582	4,740	1,624	-80.17%
800	90	45	1,290	836	6,710	2,685	-80.77%
1,000	72	45	1,680	855	8,448	2,893	-80.11%

Table 7: Impacts of C/P ratio

C/P ratio	TotCost	ComCost	PenCost	AvgCost/C	AvgCost/P	MatP	MatC	AvgPNum
0.1	1011	126	885	13.0	4.5	28.0	9.6	2.9
0.2	741	262	479	14.1	4.8	55.0	18.6	3.0
0.3	547	315	232	12.5	4.2	74.6	25.2	3.0
0.4	425	320	105	10.8	3.7	86.8	29.6	2.9
0.5	384	316	68	10.3	3.5	90.4	33.7	2.9
0.6	312	297	14	8.8	3.1	96.4	33.6	2.9
0.7	282	273	9	7.9	2.8	97.4	34.4	2.8
0.8	256	250	6	7.0	2.5	98.0	35.6	2.8
0.9	230	225	5	6.3	2.3	98.4	36.0	2.7
1	219	216	3	5.9	2.2	98.6	36.4	2.7

Table 8: Impacts of departure time flexibility

DTFlex	TotCost	ComCost	PenCost	AvgCost/C	AvgCost/P	MatP	MatC	AvgPNum
5	943	70	873	4.0	2.0	35.6	17.4	2.0
10	538	191	347	7.1	2.6	72.2	26.6	2.7
15	393	280	113	9.2	3.2	87.4	30.6	2.9
20	369	307	62	9.8	3.4	90.8	31.6	2.9
25	362	300	62	9.5	3.3	90.4	31.6	2.9
30	367	313	54	10.0	3.5	91.0	31.6	2.9
35	354	311	43	9.8	3.4	92.2	32.0	2.9
40	355	313	42	9.8	3.4	92.2	32.0	2.9
45	345	313	32	9.7	3.4	93.6	32.4	2.9
50	335	303	32	9.3	3.3	93.2	32.8	2.8
55	328	303	25	9.2	3.2	94.2	33.0	2.9
60	325	302	23	9.2	3.2	94.4	33.0	2.9

Table 9: Impacts of the crowd-courier's capacity

CourCap	TotCost	ComCost	PenCost	AvgCost/C	AvgCost/P	MatP	MatC	AvgPNum
1	994	250	744	5.8	6.6	37.8	42.8	1.0
2	533	348	185	8.8	4.5	78.2	39.8	2.0
3	352	317	35	9.9	3.4	93.6	32.0	2.9
4	297	279	18	11.1	3.1	97.0	27.0	3.9
5	292	277	15	12.0	3.0	98.0	23.0	4.2