Provably High-Quality Solutions for the Meal Delivery Routing Problem

Baris Yildiz[1] and Martin Savelsbergh[2]

[1]Department of Industrial Engineering, Koc University, Istanbul, Turkey
[2]H. Milton Stewart School of Industrial & Systems Engineering, Georgia Institute of Technology, Atlanta, USA

May 16, 2018

Abstract

Online restaurant aggregators with integrated meal delivery networks have become more common and more popular in the past few years. Meal delivery is arguably the ultimate challenge in last mile logistics: a typical order is expected to be delivered within an hour (much less if possible), and within minutes of the food becoming ready. We introduce a novel formulation for a meal delivery routing problem (in which we assume perfect information about order arrivals), and develop a simultaneous column and row generation method for its solution. The analysis of the results of an extensive computational study, using instances derived from real-life data, demonstrates the efficacy of the solution approach, and provides valuable insights into, among others, the (potential) benefits of order bundling, courier shift scheduling, and demand management.

1 Introduction

The transportation sector has been going through a revolutionary transformation that has been necessitated and facilitated by the recent advances in communication and mobile device technologies. Because of the explosive growth of e-commerce and the ever-increasing desire for faster service, current logistics capabilities are stretched to (or beyond) the limit, and the players in the sector are seeking new and creative solutions to address the unprecedented challenges and are exploiting new and emerging business models to drive change. On-demand meal ordering platforms are an example of a daunting transportation problem that has spawned an attractive business opportunity. Instead of restaurant-operated delivery services that fail to scale up to elevated customer expectations of service time, cost, and availability, online restaurant aggregators with integrated delivery networks are being set up to achieve higher efficiency levels that can meet the challenge. Such platforms do not only aim to provide a higher service quality at lower cost, but also to provide more restaurants with the opportunity to offer meal delivery service to their customers. The fast increasing volume of the meal delivery operations on the global scale attests to the strength of the business in a massive and under-penetrated market (Bakker, 2016). According to Morgan Stanley Research, about $210 billion worth of food is ordered for delivery or takeout on an annual basis solely in the US, and online food delivery is expected to grow by 16% annual compound rate in the next 5 years (Morgan Stanley Research, 2017).

Not all news is good, however, for the meal delivery network providers. Consumer resistance to delivery fees and shrinking margins are a fact of the e-business era (Geoffrion and Krishnan, 2001; Bakker, 2016). As a result, there is an urgent need for efficient and effective dispatching technology to support meal delivery operations. However, meal delivery routing problems are quite challenging, not in the least

because of the dynamism and urgency of arriving orders (van Lon et al., 2016). Meal delivery is arguably the ultimate challenge in last mile logistics: a typical order is expected to be delivered within an hour (much less if possible), and within minutes of the food becoming ready (Reyes et al., 2018). Meeting such stringent service quality targets without incurring a prohibitively high cost is only achievable through the adoption of crowd-sourced delivery capacity (Mladenow et al., 2016). Rather than owning a fleet of vehicles and employing drivers to make deliveries, the delivery of meals is crowd-sourced, i.e., handled by contracting individuals to deliver meals. In a market with high order rate fluctuations throughout the day (soaring during the meal times and dropping to very low levels in between) the advantage of having a flexible number of couriers, as in the successful ride-hailing business models for personal transportation, is obvious. However, such a fleet of couriers brings many new operational challenges. Couriers do not necessarily accept all delivery orders offered to them, and do not necessarily complete their planned shifts. Incentives have to be used to induce the desired courier behavior and , e.g., a minimum guaranteed hourly payment if few offered orders are rejected and a planned shift is completed (Reuters, 2016). As a consequence, the study of meal delivery routing problems is not only of value because it satisfies an urgent practical need, but it is also of academic value as meal delivery routing problems exhibit the critical characteristics of the emerging challenges in dynamic routing and scheduling and last mile logistics.

In this paper, we develop an exact solution approach for the Meal Delivery Routing Problem (MDRP) introduced and defined by Reyes et al. (2017), in which we assume *perfect information* about the order arrival stream, i.e., we assume a *clairvoyant* decision maker. Of course, perfect information about the order arrival stream is far from realistic, but the solutions obtained can (1) provide valuable insights into the characteristics of high-quality solutions, which may inform online dispatching algorithm design and (2) provide a benchmark to assess the quality of online dispatching algorithms.

In summary, the main contributions of this study are as follows.

- We present a novel approach for handling (continuous) time in formulations of transportation problems. It is based on the concept of *work-packages* and allows a formulation that can be solved effectively by a combination of row and column generation.
- We conduct an extensive numerical study using instances from the Grubhub MDRP instance set, which contains MDRP instances derived from real-life historic data. The results provide important insights related to online algorithm design and effective dynamic delivery dispatching. In particular, the results show that:
  - Cost and service objectives are well-aligned, so optimizing one objective tends to optimize the other objective as well. Furthermore, Pareto optimal solutions can be obtained efficiently using a hierarchical optimization approach.
  - With properly chosen compensation schemes and courier schedules, minimum pay guarantees can be offered (to increase courier compliance) without a significant increase in cost.
  - Off-line planning of courier schedules aimed at ensuring the right number of couriers are available at the right time can significantly reduce cost and improve service.
  - Order bundling, i.e., having a courier deliver multiple orders in a single trip, does not offer significant cost reduction and service improvement (for this set of MDRP instances).
  - Demand management strategies that seek to influence diners' restaurant choices have considerable potential to improve system performance (in terms of both cost and service).

The remainder of the paper is organized as follows. In Section 2, we briefly discuss related literature. In Section 3, we formally introduce the MDRP and define the notation used throughout the paper. In Section 4, we present a novel mathematical formulation, and in Section 5, we describe an algorithm for

solving it. In Section 6, we discuss the results of an extensive computational study. We conclude, in Section 7, with some final remarks.

Before continuing, we want to emphasize that the meal delivery routing problem defined in Reyes et al. (2017) is motivated by a collaboration with Grubhub, but that it is not an exact representation of the routing problem encountered at Grubhub as certain aspects and constraints have been altered or have not been captured.

2 Literature review

The MDRP falls in the broad class of dynamic vehicle routing problems (dVRP). For a comprehensive coverage of the vast dVRP literature, we refer the reader to Thomas (2010); Pillac et al. (2013), and Psaraftis et al. (2016). Here, we focus on recent studies involving a problem setting similar to the MDRP or a methodology similar to the one we propose for solving instances of the MDRP.

The MDRP is closely related to dynamic delivery problems (dDP), which have emerged recently and represent an important class of dynamic pickup and delivery problems (dDPD); see (Berbeglia et al., 2010) for a detailed survey of the dDPD literature. Motivated by the rising interest of online retailers to offer same-day delivery, researchers have started to investigate various aspects of same-day delivery operations, both from a theoretical (Archetti et al., 2015; Klapp et al., 2016; Ulmer et al., 2016; Reyes et al., 2017) and practical (Azi et al., 2012; Klapp, 2016; Archetti et al., 2016; Dayarian and Savelsbergh, 2017; Voccia et al., 2017) perspective. The defining characteristic of dDPs is that once a vehicle has been dispatched, modifying the vehicle's route is undesirable or impractical, and in the dDP literature typically inadmissible. In the MDRP, couriers have to complete the delivery of the most recently picked up meals, before picking up any new meals at a restaurant. Our formulation and solution approach, however, can easily be extended to relax this requirement when it is unnecessarily restrictive.

Many solution approaches have been proposed for solving dynamic routing problems. The differences in solution approaches are, in part, a result of the assumptions regarding the information available about future orders. While *dynamic and deterministic* approaches assume that no information is available about future orders, and base decisions only on the current state of the system, *dynamic and stochastic* approaches use probabilistic information about future orders when making decisions. Rolling horizon approaches, using dynamic programming techniques or metaheuristics, have been widely used in deterministic settings, e.g., (Psaraftis, 1980; Gendreau et al., 1999; Ichoua et al., 2000, 2003; Yang et al., 2004; Montemanni et al., 2005; Chen and Xu, 2006). Markov Decision Processes (MDPs), Approximate Dynamic Programming (ADP), and scenario sampling techniques have been used to incorporate stochastic information (Godfrey and Powell, 2002; Bent and Van Hentenryck, 2004; Hvattum et al., 2006; Simao et al., 2009; Novoa and Storer, 2009; Azi et al., 2012; Voccia et al., 2017). In our research on the MDRP, we assume that the decision maker is clairvoyant and has perfect information about future orders. Therefore, we can formulate the MDRP as a single deterministic optimization problem, which we solve using combined column and row generation method.

Column generation and its generalization to integer programs, i.e., branch-and-price, have been successfully used to solve various large-scale optimization problems arising in transportation and telecommunication applications, e.g., (Parker and Ryan, 1993; Barnhart et al., 1994; Park et al., 1996; Barnhart et al., 1998, 2000; Cohn and Barnhart, 2003; Degraeve and Jans, 2007; Desaulniers, 2010). Composite variables (Barnhart et al., 1994) have been used to produce formulations that overcome some of the computational challenges associated with column generation formulations. Careful definition of variables is also critical to our approach for solving the MDRP. Yıldız and Karaşan (2017) recently introduced an innovation to path-based formulations that has inspired many of our formulation ideas. They propose a formulation in which partial columns (path segments) are used instead of whole columns (paths). Con-

sidering path-segments as variables and concatenating path-segments to construct complete paths makes it easy to consider non-simple path solutions and to include certain types of constraints on paths, which are difficult to incorporate in a standard path-based formulations. Yıldız et al. (2016) and Yıldız et al. (2017) propose new applications of path-segment formulations and a branch-and-price algorithm to solve large-scale network design problems arising in different transportation and telecommunications applications. Similar to path segments, we introduce *work-packages*. However, concatenating work-packages to construct courier itineraries requires not only spatial consistency, but also temporal consistency. That is, work-package variables embed spatial as well as temporal information, and both space and time consistency constraints are incorporated in our formulation. Because time is continuous, our formulation has infinitely many variables and infinitely many constraints. We show that a special property of optimal solutions can be exploited to develop a simultaneous column and row generation algorithm to solve instances of the MDRP without explicitly considering each point in time. As such, the approach is similar in spirit to the recently proposed dynamic discretization discovery algorithm for solving a continuous-time network design problem (Boland et al., 2017), which relies on refining partially time-expanded networks.

3 Problem definition

As mentioned above, we consider the problem definition and modeling assumptions introduced in Reyes et al. (2018) and restate them here for the sake of completeness.

We consider a set of meal orders that arrive during a given day. Each order has a placement time, an associated restaurant, an order ready time, and a drop-off location. A set of couriers is available to deliver meals to diners. Each courier has an associated shift and a location where the shift starts. The goal is to determine itineraries for couriers so as to optimize a system performance metric, e.g., average click-to-door time or total courier compensation, while ensuring that all orders are delivered without exceeding a given click-to-door time limit.

Multiple orders can be picked up by a courier at a restaurant and delivered in a single trip. The bundle ready time is the latest ready time of the constituent orders. The time a courier spends traveling between any two locations is assumed to be known and invariant over time. There is a fixed service time for each pick-up and drop-off to account for the time spent by the courier to park his vehicle, walk to the restaurant/delivery location, pick up/drop off one or more orders, and walk back to his vehicle. The service time is independent of the number of orders being picked up. So the earliest pick up time for an order is not smaller than the maximum of the order ready time and the courier arrival time at the restaurant plus half of the service time at a restaurant. The departure time from a restaurant is the pickup time plus half of the service time. Similarly the drop-off time of an order is the arrival time of the courier at the drop-off location plus half of the service time at a drop-off location. The earliest departure time after delivering an order is the drop-off time plus half of the service time.

Couriers only work during their planned shifts, defined by an on-time and an off-time, and cannot be assigned a new delivery after their off-time. However, if a courier has picked an order before his off-time, he is allowed to drop off that order after its off-time. More importantly, it is assumed that couriers make no autonomous decisions while on duty. That is, they always accept any order offered to them, and they wait for (new) orders at their on-location and at the last location of their active assignment.

There are two classes of couriers: minimum-pay (mp) couriers and delivery-only (do) couriers. All couriers get paid a fixed amount for each delivery they complete during their shift. A minimum-pay courier is guaranteed a minimum pay for their shift, i.e., if the earnings from their deliveries fall below this guaranteed amount, their compensation is increased to the minimum guaranteed.

Because there are multiple stakeholders (service provider, restaurants, diners, couriers) in meal delivery routing problems, each with their own goals and concerns, there are several performance metrics to

consider for a MDRP solution:

- Total courier compensation.
- Fraction of couriers receiving guaranteed minimum compensation.
- Click-to-door time (CtD): the difference between the drop-off time of an order and its placement time.
- Click-to-door time overage (CtDO): the difference between the drop-off time of an order and its placement time plus the target click-to-door time.
- Ready-to-door time (RtD): the difference between the drop-off time of an order and its ready time.
- Ready-to-pickup time (RtP): the difference between the pickup time of an order and its ready time.
- Courier utilization (CU): the fraction of the courier duty time that is devoted to driving, picking up orders, and dropping off orders (as opposed to time spent waiting).

There is a maximum allowed click-to-door time and all orders have to be delivered before or at the order placement time plus the maximum allowed click-to-door time.

Next, we provide a formal definition and introduce the notation used throughout the paper.

Let the set of restaurants be denoted by R. The location of a restaurant $r \in R$ is denoted by ℓ_r. Let the set of couriers be denoted by C. Each courier $c \in C$ is characterized by a 3-tuple $\langle e_c, \ell_c, l_c \rangle$, where e_c is the courier's on-time (when the courier goes on duty), ℓ_c is the on-location (where the courier will be at time e_c) and $l_c > e_c$ is the off-time (when the courier goes off duty). Let the set of orders be denoted by O. Each order $o \in O$ is characterized by a 4-tuple $\langle r_o, a_o, e_o, \ell_o \rangle$, where $r_o \in R$ is the restaurant from which the meal is requested, a_o is the order placement time, e_o is the order ready time, and ℓ_o is the drop-off location. Orders from the same restaurant may be combined into *bundles* with multiple drop-off locations. A bundle $b = (o_1, \ldots, o_k), o_i \in O, i = 1, \ldots, k$ is an ordered set where the sequence defines the delivery route. The placement time of a bundle, a_b, is the earliest placement time of any of the orders in the bundle and the ready time of a bundle, e_b, is the latest ready time of any of the orders $O(b)$ in the bundle. Note that the placement time of a bundle is the *earliest* placement time of any of the orders in the bundle. This implies that we assume that the operating environment allows "assignment updates" (see Reyes et al. (2017)). That is, it is allowed to update the set of orders a courier will deliver upon the courier's arrival at the restaurant. This reflects the real-life practice of adding orders to a courier's assignment if additional orders happen to by ready at the restaurant at the time of pickup and those orders fit nicely into the courier's planned trip. By letting the placement time of a bundle be the earliest placement time of any of the orders in the bundle, we mimic this operating practice. The restaurant of a bundle b, r_b, is the restaurant that prepares the meals of the orders the bundle contains. We define s_b and f_b as the first and last order in bundle b. For convenience, the drop-off location of a bundle b is the drop-off location of its last order f_b and denoted as ℓ_b. We define B to be the set of all possible bundles (where we include bundles of cardinality one, i.e., consisting of a single order). The target click-to-door time is denoted by ϱ and the maximum allowable click-to-door time is denoted by ϱ^{max}.

Let t_{ij} be the travel time between any two locations i and j, e.g., restaurant locations, courier on-locations, and order drop-off locations, let v^r be the service time associated with the pickup of an order at a restaurant, i.e., the time a courier needs to park his vehicle, walk to the restaurant, pick up an order, and walk back to his vehicle, and let v^o be the service time associated with the delivery of an order at a customer location, i.e., the time a courier needs to park his vehicle, walk to the customer, drop off an order, and walk back to his vehicle.

Couriers are compensated as follows. A courier $c \in C$ receives p_1 per delivered order, but is guaranteed a minimum compensation of p_2^c per hour, i.e., the courier's compensation is $\max\{p_1 n,\ p_2^c(l_c - e_c)\}$, where n is the number of orders delivered during his shift. A courier $c \in C$ is a *minimum-pay* courier when $p_2^c > 0$ and a *delivery-only* courier when $p_2^c = 0$.

4 Mathematical formulation

Our formulation relies upon the concept of a *work-package*, which represent a possible way to serve a bundle. A work-package w is characterized by a 5-tuple $\langle b_w, s_w, f_w, \sigma_w, \phi_w \rangle$, where b_w denotes the bundle served, s_w denotes the start location, which is either the on-location of a courier or a drop-off location of an order, f_w denotes the end location, which is either the drop-off location of the last order in the bundle or the artificial off-location of a courier (which we will denote by $\bar{\ell}_c$), σ_w denotes the start time and ϕ_w denotes the end time of the work-package. For an order o in work-package w, i.e., $o \in O(b_w)$, the pick-up and drop-off times are denoted by δ_w^o and $\bar{\delta}_w^o$, respectively.

The set of work-packages that can be performed by a courier $c \in C$, respecting the courier's on- and off-time and on-time location, is denoted by $W(c)$. Similarly, the set of bundles that can be served by a courier $c \in C$ is denoted by $B(c)$, and the set of work-packages whose bundle contains order $o \in O$ is denoted by $W(o)$. Let $N = \{\ell_o : o \in O\}$ be the set of drop-off locations (without loss of generality, we assume that orders have unique drop-off locations). For each courier $c \in C$, we define $N^c = N \cup \{\ell_c, \bar{\ell}_c\}$.

For each order $o \in O$, we can define a time interval T_o during which it is possible to start a work-package at location ℓ_o, because there is an earliest possible time the order can be dropped off at ℓ_o (due to its order ready time) and a latest possible time the order can be dropped of at ℓ_o (due to its order placement time and the maximum allowable click-to-door time).

We define the following decision variables:

- $x_w^c = \begin{cases} 1 & \text{if the work-package } w \in W(c) \text{ is performed by the courier } c \in C, \\ 0 & \text{otherwise,} \end{cases}$
- $u_o \geq 0$: drop-off time for an order $o \in O$,
- $z^c \geq 0$: total payment made to courier $c \in C$.

In the remainder, we will refer to these three groups of variables as: work-package variables, time variables, and cost variables, respectively. With these decision variables, we define our formulation P_{COST}

as follows:

$$\min \sum_{c \in C} z^c \tag{1}$$

$$\text{s.t. } z^c \geq \sum_{\substack{w \in W(c) \\ f_w \neq \bar{\ell}_c}} p_1 x_w^c \qquad \forall c \in C, \tag{2}$$

$$z^c \geq p_2^c (l_c - e_c) \qquad \forall c \in C, \tag{3}$$

$$\sum_{c \in C} \sum_{w \in W(c) \cap W(o)} x_w^c = 1 \qquad \forall o \in O, \tag{4}$$

$$\sum_{\substack{w \in W(c) \\ s_w = i}} x_w^c - \sum_{\substack{w \in W(c) \\ f_w = i}} x_w^c = \begin{cases} 1, & \text{if } i = \ell_c, \\ -1, & \text{if } i = \bar{\ell}_c, \\ 0, & \text{otherwise,} \end{cases} \qquad \forall c \in C, i \in N^c \tag{5}$$

$$\sum_{c \in C} \sum_{\substack{w \in W(c) \\ s_w = \ell_o \\ \sigma_w \leq t}} x_w^c + \sum_{c \in C} \sum_{\substack{w \in W(c) \\ f_w = \ell_o \\ \phi_w > t - v^o/2}} x_w^c \leq 1 \qquad \forall o \in O, t \in T_o \tag{6}$$

$$u_o \geq \sum_{c \in C} \sum_{w \in W(c) \cap W(o)} \bar{\delta}_w^o x_w^c \qquad \forall o \in O \tag{7}$$

$$u_o \leq a_o + \varrho^{max} \qquad \forall c \in C, \forall o \in O \tag{8}$$

$$x_w^c \in \{0, 1\} \qquad \forall c \in C, w \in W(c), \tag{9}$$

$$u_o \geq 0 \qquad \forall c \in C, \forall o \in O \tag{10}$$

$$z^c \geq 0 \qquad \forall c \in C, w \in W(c), \tag{11}$$

The objective is to minimize total courier compensation. Constraints (2) and (3) together make sure that a courier is paid the maximum of his guaranteed payment amount and the total of his earnings for completed deliveries. Constraint (4) ensure that each order is served. Constraints (5) ensure spatial consistency for work-packages performed consecutively by a courier. Similarly, Constraints (6) ensure time consistency for work-packages performed consecutively by a courier by enforcing that a work-package can start at a location $\ell_o \in N$ at some time $t \in T^o = [\underline{t}^o, \bar{t}^o]$ only if there is another work-package which ends at ℓ_o before t minus half of the service time needed to drop-off the order (to reach the vehicle after delivery). That is, a work-package has to finish in the interval $[\underline{t}^o, t - \frac{v^o}{2}]$ in order for another work-package to start in $[t, \bar{t}^o]$. Note that we do not limit work-package starting times to a set of discrete time points, i.e., $T_o, o \in O$, represent (continuous) time intervals. This implies that the formulation has an infinite number of variables and constraints. However, in the Section 5.1, we show that it suffices to consider only a finite subset of work-package variables and a finite subset of time consistency constraints to solve P_{COST}. Finally, Constraints (7) and (8) make sure that click-to-door times do not exceed ϱ_{max} for any order.

Note that, inequalities (4)-(10) constitute the polytope of feasible MDRP solutions. Consequently, by replacing the objective function and, possibly, adding inequalities, the formulation can be modified to consider other performance metrics. Below, we will present a few examples to illustrate the flexibility of the formulation.

Minimizing average CtD: An important service related metric is average CtD. By replacing the objective (1) with the following expression, we obtain formulation P_{CtD} that minimizes the average CtD:

$$\min \ \frac{1}{|O|} \sum_{o \in O} (u_o - a_o) \tag{12}$$
$$(4) - (10)$$

Ready-to-door time (RtD) focuses on meal freshness. By replacing the order placement time a_o with the ready time e_o in (12), we obtain the formulation P_{RtD} to minimize average RtD.

Minimizing average CtDO: Click-to-door overage is a metric that focuses on service times that are longer than the target ϱ. We obtain formulation P_{CtDO} that minimizes the average CtDO by introducing auxiliary non-negative continues variables $\zeta_o, o \in O$:

$$\begin{aligned}
\min \ & \frac{1}{|O|} \sum_{o \in O} \zeta_o & \\
& \zeta_o \geq u_o - (a_o + \varrho) & \forall o \in O \\
& \zeta_o \geq 0 & \forall o \in O \\
& (4) - (10) &
\end{aligned}$$

Minimizing average RtP: Obviously, meals ordered from restaurants far away from the delivery location, naturally require a larger delivery time, and, thus, are more likely to have a large click-to-door time. Likewise, when the meal preparation time at a restaurant is high, orders placed at that restaurant are more likely to have a large click-to-door time. The ready-to-pickup metric is not affected by these aspects and is therefore, in some sense, a better metric to assess the performance of dispatching optimization technology. By replacing Constraints (7) in P_{RtD} with inequalities

$$u_o \geq \sum_{c \in C} \sum_{w \in W(c) \cap W(o)} \delta_w^o x_w^c \qquad \forall o \in O$$

we obtain formulation P_{RtP} that minimizes the average RtP.

5 Solution method

In the formulations above, dispatch time information is embedded in the variables and time consistency is ensured by means of constraints. As mentioned earlier, this requires infinitely many variables and infinitely many constraints. Of course it is not possible to solve these formulations directly, but we will show that they can be solved efficiently using a simultaneous column and row generation (CR) algorithm. The idea is to start with a finite subset of work-package assignment variables and a finite set of associated time consistency constraints, i.e., a restricted formulation, and iteratively add variables and constraints as needed to reach and prove optimality. In the remainder, we will denote the linear relaxation of P_{COST} as LP_{COST}.

5.1 Column generation

At each iteration of the CR algorithm, we solve a *pricing problem* to identify work-package assignment variables that are not in the restricted formulation, but have a favorable reduced cost, or conclude that

there are no such variables and the algorithm can be terminated. Different from traditional column generation techniques in which the restricted formulation has the complete set of constraints, here the reduced cost of a variable has to be computed while infinitely many time consistency constraints are not included in the formulation. Next, we introduce a *no-delay* property, which is critical to carefully managing the set of variables and constraints and solving LP_{COST}.

Let P^k be the restricted LP_{COST} formulation in the k^{th} iteration of algorithm CR. The set of work-package assignment variables included in P^k is denoted by X^k. For each order $o \in O$, let $T_o^k = \{\sigma_w : x_w^c \in X^k \text{ and } s_w = o\} \cup \{a_o\}$, i.e., T_o^k is the set of start times of the work-packages starting at location l_o included in P^k plus the placement time of order o.

Definition 1. *A set of work-package assignment variables X has the no-delay property if for every work-package assignment variable $x_w^c \in X$, we have that*

- $\sigma_w = a_{s_w}$, *or*
- *there exists a work-package assignment variable $x_{w'}^c \in X$ such that $f_{w'} = s_w$ and $\sigma_w = \phi_{w'} + {}^{v^o}/_2$.*

The no-delay property reflects a dispatching strategy that starts the execution of a work-package as early as possible even if the finish time of the work-package would be the same if the execution of the work-package was started at some later time. In such a dispatch strategy, if a courier has to wait, the courier waits at a restaurant (rather than at a drop off location).

In the next proposition, we will establish that if the set of work-package assignment variables in P^k has the no-delay property, then it is possible to calculate the reduced cost of *any* work-package assignment variable by considering only the time consistency constraints that are in P^k. Later, we will show that requiring that the set of variables in P^k has the no-delay property is not a restrictive assumption.

Proposition 1. *Let $\alpha, \gamma, \lambda, \mu, \eta$ be the dual variables associated with the constraints (2), (4-7) for the restricted problem P^k and assume that X^k has the no-delay property. Then the reduced cost π_w^c of a work-package assignment variable x_w^c is*

$$-p_1\alpha^c - \sum_{o \in b_w} \gamma_o - \lambda_{s_w}^c + \lambda_{f_w}^c - \sum_{t \in T_{s_w}^k : t > \sigma_w} \mu_{s_w t} - \sum_{t \in T_{f_w}^k : t \leq \phi_w - \frac{v^o}{2}} \mu_{f_w t} - \sum_{o \in b_w} \bar{\delta}_w^o \eta_o. \tag{13}$$

Proof. To show that π_w^c is the reduced cost of the variable x_w^c, we need to show that there exists an optimal dual solution in which the variables $\mu_{s_w t}$ and $\mu_{f_w t}$ for $t \in T' = [a_{b_w}, a_{b_w} + \varrho_{max}] \setminus T_{s_w}^k$ are all zero. This is done by showing that the corresponding constraints are redundant for P^k. Specifically, we show that for any time point $t' \in T$, there exists a time point $t^* \in T_{s_w}^k$ such that the left hand side (LHS) value of the time consistency constraint for t^* is at least as large as the LHS of the time consistency constraint for t'.

Let $t' \notin T_{s_w}^k$ and let $\mathcal{T} = \{t \in T_{s_w}^k : t > t'\}$. Furthermore, let $LHS(t)$ be the value of the left hand side of the time consistency constraint for time point t. We consider two cases:

Case 1. If $\mathcal{T} \neq \emptyset$, then consider $t^* = \min\{t \in T_{s_w}^k : t > t'\}$. Observe that in this case there cannot be any work-package variable in X^k that starts at location l_{b_w} in the time interval $[t^* - t']$. Since X^k has the no-delay property, we have $LHS(t^*) \geq LHS(t')$ as desired.

Case 2. If $\mathcal{T} = \emptyset$, then consider $t^* = \max\{T_{s_w}^k\}$. Note that the set $T_{s_w}^k$ always contains the time point a_{s_w} and t^* is well defined. But then there is no work-package that starts at location l_{b_w} after t^*, which implies that $LHS(t^*) \geq LHS(t')$, since $t^* < t'$. Hence, the results follows.

□

The following proposition establishes that an optimal solution to P_{COST} can be found by only considering a set of work-package assignment variables X that has the no-delay property. If w' and w are two consecutive work-packages in a solution (x, u, z) to P_{COST}, i.e., $x^c_{w'} = x^c_w = 1$ and $f_{w'} = s_w$, then we refer to w as the outbound work-package.

Proposition 2. *If P_{COST} is feasible, then it has an optimal solution (x^*, u^*, z^*) in which for all consecutive work-packages w' and w, outbound work-package w starts at time $\sigma_w = max\{\phi_{w'} + {}^{v^o}/_2, a_{b_w}\}$.*

Proof. Let $(\bar{x}, \bar{u}, \bar{z})$ be an optimal solution to P_{COST} which does not satisfy the condition stated in the proposition. Then there exits some outbound work-package w with $\sigma_w > \max\{\phi_{w'} + {}^{v^o}/_2, a_{b_w}\}$. Note that σ_w cannot be less than $\phi_{w'} + {}^{v^o}/_2$ because of the time consistence constraints, and cannot be less than $a_{b_{w'}}$ because couriers cannot start a work-package before the bundle placement time. Now consider a work-package w^* that is identical w except for its start time $\sigma_{w*} = \max\{\phi_{w'} + {}^{v^o}/_2, a_{b_w}\}$. But then we can construct an alternative optimal solution $(\bar{u}, x^*, \bar{z})$ with

$$x^{*c}_{\tilde{w}} = \begin{cases} 1, \text{ if } \tilde{w} = w*, \\ 0, \text{ if } \tilde{w} = w, \\ \bar{x}^c_w, \text{ otherwise.} \end{cases} \qquad \forall c \in C, \tilde{w} \in W(c)$$

Observe that replacing $\bar{x}$ with x^* only affects Constraints (6) and (7). When $o = w_{b_s}$, inequalities (6) remain satisfied since we have $\sigma_{w*} = \max\{\phi_{w'} + {}^{v^o}/_2, a_{b_w}\}$ and inequalities (7) hold because $\sigma_{w*} < \sigma_w$ implies $\phi_{w*} < \phi_w$. When $o \neq w_{b_s}$, we only need to consider Constraints (6), which remain satisfied, again, because we have $\phi_{w*} < \phi_w$. Clearly, replacing $\bar{x}$ with x^* does not change the objective function value, so the modified solution $(x^*, \bar{u}, \bar{z})$ is also optimal and has one fewer work-package that violates the condition stated in the proposition. Repeating the same steps as many times as needed, we can construct an alternative optimal solution that satisfies the requirements of the proposition. □

Since LP_{COST} is a minimization problem, in the pricing problem we look for negative reduced cost columns that are not yet in the model. Propositions (1) and (2) allow us to develop an enumeration algorithm (PS) to identify such columns. Before presenting the pseudo-code for PS in Algorithm 1, we define additional notation and introduce concepts that are used in PS to improve its computational efficiency.

Let drop-off time function $g^c : O \times N \times Z_{\geq 0} \times B \mapsto Z_{\geq 0}$ return the drop-off time of order $o \in O$, when courier $c \in C$ starts at location $\ell \in N^c$ at time $t \in Z_{\geq 0}$ to perform a work-package serving bundle $b \in B$, where $g^c(o, \ell, t, b) = \infty$ when:

- the order o is not included in bundle b,
- the start time t is before the bundle placement time, i.e., $t < a_b$,
- the start time t is before the courier on-time, i.e., $t < e_c$,
- the pick-up time from the restaurant is later than the courier off-time, i.e., $\max\{t + t_{\ell\ell_{r_b}} + {}^{v^r}/_2, e_b\} + {}^{v^r}/_2 > l_c$, or
- the drop-off time of any order $o \in O(b)$ is later than $a_o + \varrho^{max}$.

Let ϵ_o be the earliest possible start time of a work-package with start location ℓ_o. We have that $\epsilon_o = e_o + {}^{v^r}/_2 + t_{\ell_{r_o}\ell_o} + v^o$, since the earliest drop-off time for order o occurs when a courier reaches restaurant r_o exactly ${}^{v^r}/_2$ time units before the order ready time, so that the order is picked up at the

order ready time. Furthermore, let $\upsilon^c(o,b)$ be the latest possible start time of a work-package with start location ℓ_o serving bundle b and carried out by courier $c \in C$, assuming that $g^c(\bar{o}, \ell_o, t^*, b) \neq \infty$ for all $\bar{o} \in O(b)$. We say there exists a *plausible transition* from order $o \in O$ to bundle $b \in B$ when $\max\{\epsilon_o, e_c\} < \upsilon^c(o,b)$ for some $c \in C$. The set of all bundles for which there exists a plausible transition from order $o \in O$ is denoted by $B(o)$. For a pair $(o,b), o \in O, b \in B(o)$, we define the set of plausible start times for work-packages that can be performed by courier $c \in C$, starting at ℓ_o and serving bundle b as $T(c,o,b) = [\max\{\epsilon_o, e_c\}, \upsilon^c(o,b)]$.

Algorithm 1: PS Algorithm

Input: X^k
Output: X^{k+1}

$X^{k+1} = X^k$;
foreach $o \in O$ **do**
  $\bar{T}_o^k = \{\phi_w + {}^{\upsilon^o}\!/_2 : x_w^c \in X^k, \text{for some } c \in C \text{ and } f_w = o\}$;
foreach $o \in O$ **do**
  foreach $b \in B(o)$ **do**
    foreach $c \in C(b) \cap C(o)$ **do**
      foreach $t \in T(c,o,b) \cap \left(\bar{T}_o^k \cup \{a_b\}\right)$ **do**
        Set $\bar{w}$ such that: $b_{\bar{w}} = b, s_{\bar{w}} = \ell_o, d_{\bar{w}} = \ell_b, \sigma_{\bar{w}} = t, \phi_{\bar{w}} = g^c(f_{b_{\bar{w}}}, \ell_o, t, b)$;
        foreach $\bar{o} \in b$ **do**
          $\delta_{\bar{w}}^{\bar{o}} = g^c(\bar{o}, \ell_o, t, b)$
        if $\pi_{\bar{w}}^c < 0$ **then**
          add $x_{\bar{w}}^c$ to X^{k+1};
Return X^{k+1}

When PS is called at the k^{th} iteration of CR, it first generates the sets of time points $\bar{T}_o^k, o \in O$, which contains the time points at which a work-package can start without violating the no-delay property. It then considers only these time points plus the bundle placement time a_b when looking for new work-package assignment variables to include in the model to serve bundle b. Note that by Proposition 2, for any location $l_o, o \in O$, it suffices to consider work-packages that start at one of the time points in $\bar{T}_o^k$ and the bundle ready time a_b when looking for a work-package to serve a bundle $b \in B$. Note too that if at the start of PS the set of work-package assignment variables has the no-delay property, then so will the extended set of work-package assignment variables after PS finishes. Therefore, the reduced costs of the newly identified work-package assignment variables can be calculated as specified in Proposition 1. Observe that PS does not consider work-packages that start at a courier's on-location or end at a courier's artificial off-location. As we discuss in Section 5.3, work-package assignment variables associated with the first and last assignment of a courier are included in the initial set of variables X^0.

5.2 Row generation

If PS adds any new work-package assignment variables to the restricted model, we have to generate additional rows as well, because the sets $T_o^k, o \in O$ change. To add the required rows, we analyze the extended set of work-package assignment variables to determine the new time points that have been introduced and add the corresponding time consistency constraints.

Let X^{k+1} be the set of work-package assignment variables after PS is executed in the k^{th} iteration of CR. We start by generating the sets of time points $T_o^{k+1} = \{\sigma_w : x_w^c \in X^{k+1} \text{ for some } c \in C \text{ and } f_w = o\} \cup \{a_o\}$ for all $o \in O$. Then, for each $o \in O$ and $t \in T_o^{k+1}$, we check whether the corresponding time consistency constraint is missing, and, if so, add it to P^{k+1}.

5.3 Initial set of columns

Recall that PS only considers work-packages that start and end in some location $\ell \in N$. However, there are work-packages that start at a courier's on-location and finish at a courier's artificial off-location. Therefore, we include the following three sets of work-package assignment variables in X^0:

- *Buds:* For each bundle $b \in B(c)$, we create the variable x_w^c, where $b_w = b$, $s_w = \ell_c$, $f_w = \ell_b$, $\sigma_w = \max\{l_c, a_b\}$, and $\phi_w = \max\{\sigma_w + t_{\ell_c \ell_{r_b}} + \frac{v^r}{2}, e_b\} + \frac{v^r}{2} + t_{\ell_{r_b} \ell_b} + \frac{v^o}{2}$;
- *Leaves:* For each bundle $b \in B(c)$, we create the variable x_w^c, where $b_w = \emptyset$, $s_w = \ell_b$, $f_w = \bar{\ell}_c$, and $\sigma_w = \phi_w = e_c$; and
- *Nulls:* For each courier, we create the variable x_w^c, where $b_w = \emptyset$, $s_w = \ell_c$, $f_w = \bar{\ell}_c$, $\sigma_w = \phi_w = e_c$.

The variables in the group *Buds* represent the possible first work-packages for a courier, the variables in the group *Leaves* represent the potential last work-packages for a courier, and the variables in the group *Nulls* represent "do nothing" itineraries for a courier, indicating that no work-packages are assigned to a courier during his shift. Note that by including these variables from the start, PS can be restricted to work-packages that start and end at an order drop-off location.

In addition, we use a simple greedy heuristic (GH) to try and construct a feasible solution to P_{COST} and add the associated work-package assignment variables as well. The main idea is to assign work-packages to couriers in order of non-decreasing bundle placement time. A work-package is assigned to the courier that can complete it the earliest. Algorithm 2 presents the pseudo-code for GH.

Unfortunately, finding a feasible solution to P_{COST} is nontrivial and GH may fail to do so. If that happens, we proceed as follows. Let F be the set of orders that are not served in the solution produced by GH. Consider the modification of P_{COST} in which:

- we define auxiliary binary variables $y_o \in \{0, 1\}$ for $o \in F$,
- we add the term $\sum_{o \in F}(p_1 + \epsilon)y_o$ to the objective (for some $\epsilon > 0$), and
- we change Constraints (4) to

$$\sum_{c \in C} \sum_{w \in W(c) \cap W(o)} x_w^c = \begin{cases} 1 - y_o, & \text{if } o \in F, \\ 1, & \text{otherwise} \end{cases} \qquad \forall o \in O.$$

Note that the solution produced by GH can be easily converted to a feasible solution to the modified problem $\overline{P_{COST}}$ by setting $y_o = 1$ for all $o \in F$. It is also clear that P_{COST} is feasible if and only if $\overline{P_{COST}}$ has an optimal solution in which the auxiliary variables are all equal to zero. Therefore, if the solution produced by GH is not feasible for P_{COST}, we switch to using $\overline{P_{COST}}$ until all auxiliary variables have zero values or we prove that no such solution exists, in which case we conclude the problem is infeasible and stop.

5.4 Strengthening

Two groups of valid inequalities can be added to strengthen the linear relaxation.

Algorithm 2: GH

Input: R, C, O

Output: X

Let $\bar{O} = O, X = \emptyset$;
foreach $c \in C$ **do**
    Build the empty list $path^c$
while $\bar{O} \neq \emptyset$ **do**
    Let o^* be the order in $\bar{O}$ with the minimum placement time;
    set $\bar{w} = null$,$c^* = null$, $bestTime = \infty$;
    foreach $c \in C : l_c < e_{o^*} + \varrho_{max} + {}^{v^o}/_2$ **do**
        if $path^c$ *is empty* **then**
            $\sigma = max\{l_c, a_{o^*}\}$;
            if $\sigma + t_{\ell_c \ell_{r_{o^*}}} + {}^{v^r}/_2 > e_c$ **then**
                Next c
            $\phi = max\{\sigma + t_{\ell_c \ell_{r_{o^*}}} + {}^{v^r}/_2, e_{o^*}\} + {}^{v^r}/_2 + t_{\ell_{r_{o^*}} \ell_{o^*}} + {}^{v^o}/_2$;
            if $\phi > e_{o^*} + \varrho_{max}$ **then**
                Next c
            if $\phi < bestTime$ **then**
                $bestTime = \phi$, $c^* = c$;
                Set $\bar{w}$ such that: $b_{\bar{w}} = \{o^*\}, s_{\bar{w}} = \ell_c, f_{\bar{w}} = \ell_{o^*}, \sigma_{\bar{w}} = \sigma, \phi_{\bar{w}} = \phi$;
        else
            Let $\hat{w}$ be the last work-package in the list $path^c$ and $\hat{o} = d_{b_{\hat{w}}}$;
            if $\phi_{\hat{w}} + {}^{v^o}/_2 > e_c$ **then**
                Next c
            $\sigma = max\{\phi_{\hat{w}} + {}^{v^o}/_2, a_{o^*}\}$;
            if $\sigma + t_{\ell_{\hat{o}} \ell_{r_{o^*}}} + {}^{v^r}/_2 > e_c$ **then**
                Next c
            $\phi = max\{\sigma + t_{\ell_{\hat{o}} \ell_{r_{o^*}}} + {}^{v^r}/_2, e_{o^*}\} + {}^{v^r}/_2 + t_{\ell_{r_{o^*}} \ell_{o^*}} + {}^{v^o}/_2$;
            if $\phi > e_{o^*} + \varrho_{max}$ **then**
                Next c
            if $\phi < bestTime$ **then**
                $bestTime = \phi$, $c^* = c$;
                Set $\bar{w}$ such that: $b_{\bar{w}} = \{o^*\}, s_{\bar{w}} = \ell_{\hat{o}}, f_{\bar{w}} = \ell_{o^*}, \sigma_{\bar{w}} = \sigma, \phi_{\bar{w}} = \phi$;
        if $c^* \neq null$ **then**
            add $\bar{w}$ to the end of the list $path^{c^*}$, add $x^{c^*}_{\bar{w}}$ to the set X.
    $\bar{O} = \bar{O} \setminus \{o^*\}$;
return X

Courier time limit: Preliminary computational experience has shown that the LP relaxation bound deteriorates when there are fewer courier hours. Therefore, the following (knapsack-like) constraint can be effective to strengthen the LP relaxation bound when there are time periods in which the total number of courier hours is a limiting factor:

$$\sum_{w \in W(c)} \kappa_w x_w^c \leq K_c \qquad \forall c \in C, \tag{14}$$

where $\kappa_w = \min\{\phi_w, l_c\} - \sigma_w$ is the time required to perform work-package w and K_c is the time courier c has available (his shift duration).

Disaggregated time consistency inequalities: Disaggregation of the time consistency constraints by courier results in the following inequalities:

$$\sum_{\substack{w \in W(c) \\ f_w = \ell_o \\ \sigma_w \leq t}} x_w^c - \sum_{\substack{w \in W(c) \\ s_w = \ell_o \\ \phi_w \leq t}} x_w^c \geq 0 \qquad \forall c \in C, o \in O, t \in T_o \tag{15}$$

Preliminary computational experience has shown that starting with Constraints (6) and adding inequalities (15) on the fly can be an effective strategy for improving the LP relaxation bound without increasing the solution time too much. Note that the addition of inequalities (15) can easily be handled in the pricing problem.

5.5 Obtaining a high-quality IP solutions

A branch and price (BP) algorithm can be developed to solve P_{COST} by embedding the CR algorithm in a branch and bound scheme. The branching scheme proposed by Yıldız and Karaşan (2017) for path-segment formulations can be adapted in a straightforward way to obtain a BP algorithm for solving P_{COST}. However, a much simpler scheme is already able to obtain integer solutions of proven high quality.

Preliminary computational experience has shown and that the well-known heuristic approach that solves an IP with the columns generated during the solution of the LP relaxation is very effective and produces near-optimal solutions in almost all cases (which also shows that the LP relaxation bound of P_{COST} is very tight).

This approach can be further enhanced by detecting and including additional columns that have not been generated by the CR algorithm (which focuses on solving LP_{COST}), but that are helpful when searching for high-quality IP solutions.

Since the itineraries of the couriers are composed of work-packages, the timing of the work-packages has to be such that they can be concatenated to form a valid itinerary. However, in the LP relaxation these strict timing requirements may be "relaxed" as certain complete itineraries may not be needed in an optimal solution to the LP relaxation. Therefore, we propose a heuristic column generation algorithm (HCG) that inspects the solution of LP_{COST} to detect potentially useful columns that are missing in the solution.

The idea behind HCG is to create and explore concatenations of individual work-packages and add concatenated work-packages to the formulation in the hope that it allows the creation of beneficial courier itineraries. The steps of HCG are explained below.

Using the solution x^* to LP_{COST}, we generate a graph $G_c^* = (V_c^*, A_c^*)$ for each courier $c \in C$ with $V_c^* = \{v \in B : x_w^{c*} > 0, \text{ for some } w \in W(c),\ b_w = v\}$, and A_c^* containing arcs (v_1, v_2) for all $v_1, v_2 \in V_c^*$ for which (f_{v_1}, v_2) is a plausible transition for courier c. Once G_c^* is build, one can enumerate all possible

(feasible) sequences of work-packages and add associated variables x_w^c to the formulation. If the number of possible sequences in G_c^* is prohibitively large, we can restrict A_c^* to contain only those transitions that have been utilized in the LP solution. That is, A_c^* contains the arc (v_1, v_2) if there exists some $\bar{w} \in W(c)$, such that $s_{\bar{w}} = v_1, f_{\bar{w}} = v_2$ and $x_{\bar{w}}^{c*} > 0$.

6 Computational experiments

In order to test the computational efficiency of our solution approach and to gain insight into the structure of high-quality solutions to MDRP instances, we have conducted a comprehensive numerical study using the Grubhub MDRP instance set (`https://github.com/grubhub/mdrplib`), which is derived from real-world historic data. When analyzing the solutions to the instances, we seek to answer, among others, questions like:

- What are the characteristics of the high-quality solutions when minimizing cost?
- What are the characteristics of the high quality solutions when minimizing average click-to-door time?
- How much does order bundling help in reducing cost or average click-to-door time?
- How much does optimizing courier shifts help in reducing cost or average click-to-door time?
- What is the potential of demand management strategies to improve performance critical metrics measures?

Before proceeding with the analysis of the results of our numerical experiments, we first present relevant details about MDRP instances used in our study and the configuration of the CR algorithm used to solve them.

6.1 Instances

The Grubhub MDRP instances are designed to resemble realistic daily order arrival patterns and courier shifts in metropolitan areas. Instance names encode important information about the instance characteristics. We restate the naming conventions for convenience and the sake of completeness.

- The first digit represents the random seed value. Instances with the different seed values represent different metropolitan areas.
- Size reduction information:
  - o100: original data (i.e., no size reduction);
  - o50: 50% reduction in the number of orders and the number of courier hours (sampling from the order set); and
  - r50: 50% reduction in the number of orders and the number of courier hours (sampling from the restaurant set, deleting all orders placed at the restaurant).
- Courier schedules variations:
  - s1: historical courier shifts; and
  - s2: optimized courier shifts (having about the same – never more – courier hours).

- Travel speed variations:
  - t100: original travel times; and
  - t75: short travel times (original travel times multiplied by 0.75).
- Meal preparation times variations:
  - p100: original meal preparation times; and
  - p125: long meal preparation times (original meal preparation times are multiplied by 1.25).

We have used 24 instances with seed value 0. We have used the service times (four minutes for pick-up and four minutes for drop-off), target CtD (40 minutes), maximum CtD (90 minutes), and per-hour and per-delivery courier compensation rates ($15 and $10, respectively) provided with the instances. Table 1 lists the instance characteristics, where the columns *#ord*, *#rest* and *#cor* indicate the number of orders, the number of restaurants, and the number of couriers, the column *c.hours* lists the total number of courier hours available, the column *Delivery Time* lists the minimum, maximum, and average travel time from the restaurant to a drop-off location, and the column *Preparation Time* lists the minimum, maximum, and average meal preparation times.

Table 1: Problem Instances

					Delivery Time			Preperation Time		
Instances	#ord	#rest	#cour	c.hours	min	max	mean	min	max	mean
0r50t75s1p100	242	54	61	151.48	1	14	5.83	1	55	17.93
0r50t75s1p125	242	54	61	151.48	1	14	5.83	1	68	22.03
0r50t75s2p100	242	54	73	146.50	1	14	5.81	1	55	17.93
0r50t75s2p125	242	54	73	146.50	1	14	5.81	1	68	22.03
0r50t100s1p100	242	54	61	151.48	1	19	7.60	1	55	17.93
0r50t100s1p125	242	54	61	151.48	1	19	7.60	1	68	22.03
0r50t100s2p100	242	54	73	146.50	1	19	7.57	1	55	17.93
0r50t100s2p125	242	54	73	146.50	1	19	7.57	1	68	22.03
0o50t75s1p100	252	93	61	151.48	1	14	5.91	1	55	16.60
0o50t75s1p125	252	93	61	151.48	1	14	5.91	1	68	20.35
0o50t75s2p100	252	93	72	146.50	1	14	5.93	1	55	16.60
0o50t75s2p125	252	93	72	146.50	1	14	5.93	1	68	20.35
0o50t100s1p100	252	93	61	151.48	1	19	7.73	1	55	16.60
0o50t100s1p125	252	93	61	151.48	1	19	7.73	1	68	20.35
0o50t100s2p100	252	93	72	146.50	1	18	7.73	1	55	16.60
0o50t100s2p125	252	93	72	146.50	1	18	7.73	1	68	20.35
0o100t75s1p100	505	116	113	303.00	1	14	5.66	1	55	17.04
0o100t75s1p125	505	116	113	303.00	1	14	5.66	1	68	20.90
0o100t75s2p100	505	116	117	293.00	1	14	5.69	1	55	17.04
0o100t75s2p125	505	116	117	293.00	1	14	5.69	1	68	20.90
0o100t100s1p100	505	116	113	303.00	1	19	7.38	1	55	17.04
0o100t100s1p125	505	116	113	303.00	1	19	7.38	1	68	20.90
0o100t100s2p100	505	116	117	293.00	1	19	7.41	1	55	17.04
0o100t100s2p125	505	116	117	293.00	1	19	7.41	1	68	20.90
Average	-	-	-	-	1	16.4	6.69	1	61.5	19.14

6.2 Implementation

The algorithm is implemented using Java. All experiments were run on 64-bit machine with an Intel Xeon E5-2650 v3 processor at 2.30 GHz running Linux and using CPLEX 12.6 for solving LPs and IPs. The time limit for solving the IP (after the LP relaxation has been solved and additional columns are

included) is set to two hours for 0o50 and 0r50 instances and to eight hours for 0o100 instances. If after solving an instance, the optimality gap is greater than 5%, we resolve the instance now initiating CR with the best-known IP solution. A two-phase approach is used when bundling of orders is enabled. In Phase 1, we solve the instance considering only individual orders. In Phase 2, when bundles of orders are considered, we provide the solution obtained in Phase 1 as an initial feasible solution.

Because the size of the instances used in the numerical experiments is non-trivial, techniques to enhance the efficiency of the CR algorithm are critical. To this end, we introduce a Selective Column Inclusion (SCI) scheme, which controls the number of columns added after a pricing problem is solved. More specifically, we modify PS as follows:

- As soon as a negative reduced cost column is found for a bundle, we move on to the next bundle (i.e., we break the loop over the time points – Line 7 in Algorithm 1 – and the loop over the couriers – Line 6 in Algorithm 1).
- If the number of columns generated by the pricing problem, c, exceeds 1000, then we sort the columns in order of non-increasing reduced costs and only add the top $\max\{1000, c\alpha\}$ columns, where $\alpha = 0.02$ for 0o100 instances and $\alpha = 0.1$ for the others.

In Table 2, we show the impact of this enhancement as well as the impact of including additional columns before solving the IP (as this also turned out to be quite important) when solving P_{COST} assuming all couriers are minimum-pay couriers. We report the number of columns generated in the LP solution phase (*#cols*) within the one hour time limit (when a single column generation iteration takes more then one hour we wait for it to finish), the number of column generation iterations completed during the LP solution (k), the average cpu time – in seconds – required to complete a single column generation iteration (*cg-time*), and the optimality gap (*gap*) – we put NA if the algorithm fails to solve LP relaxation in the given time limit.

Table 2: Computational Enhancements with Selective Column Inclusion and IP Column Generation

	No Enhancement				SCI				SCI + HCG			
Instances	#cols	k	cg-time	gap	#col	k	cg-time	gap	#col	k	cg-time	gap
0o50t75s1p100	5181692	1	2827	NA	22109	7	5.42	0.046	22504	7	5.42	0.034
0o50t75s1p125	4217509	1	2785	NA	27279	13	4.00	0.037	27715	13	4.00	0.025
0o50t75s2p100	6711228	1	4175	NA	28375	11	3.90	0.031	28761	11	3.90	0.016
0o50t75s2p125	4374564	1	2745	NA	28453	11	4.18	0.034	28836	11	4.18	0.016
Average	5121248.3	1.0	3133.0	NA	26554	10.5	3.95	0.037	26954	10.5	3.95	0.023

We see that without SCI, it is not even possible to solve the LP relaxation of P_{COST}; on average, a single column generation iteration takes about 3000 seconds without SCI, and there are cases where CR fails to complete a single column generation iteration in one hour. On the other hand, with SCI, each column generation iteration can be completed in a few seconds and the number of column generation iterations required to solve the LP relaxation is small, less than 11 on average. The benefit of using HCG is also evident. Including only a small number of additional columns reduces the average optimality gap from 3.7% to 2.3%, which is significant.

6.3 Solution analysis

Next, we turn our attention to the analysis of the solutions to the 24 instances in our tests set. We conducted 448 experiments to analyze different settings.

6.3.1 Cost minimization

Since couriers are compensated differently depending on whether they have minimum pay guarantee or not, the total courier compensation amount varies as the proportion of minimum-pay couriers changes. Obviously, from a myopic cost perspective its always better to have fewer minimum-pay couriers. However, for strategic as well as operational reasons, such as ensuring the availability of a large enough courier pool and increasing courier compliance rates, meal delivery companies offer a minimum pay guarantees to some portion of their couriers. Therefore, a relevant question to answer is "How much does it cost to have minimum-pay couriers?" In our experiments, we considered four different proportions of minimum-pay couriers, $\gamma \in \{55, 70, 80, 100\}$, to (partially) answer this question.

Detailed results of these experiments can be found in Table 8 in Appendix A, but critical information is captured in Figure 1. However, before discussing Figure 1, we want to emphasize that the optimality gap for the solutions, on average, is less than two percent, and, thus, any insights derived from these solutions should be meaningful.

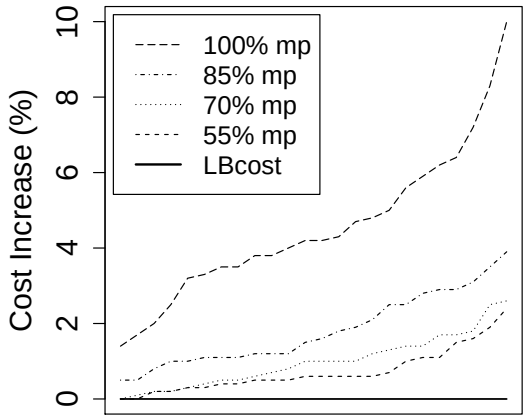


Figure 1: Cost increase due to the introduction of minimum-pay couriers.

The baseline in Figure 1 represents the situation in which none of the couriers have a minimum pay guarantee, in which case the total courier compensation is simply the number of orders times the compensation per delivery of \$10. The graphs for each of the proportions of minimum-pay couriers shows the increase in cost relative to the baseline. More specifically, for each of the minimum-pay courier proportions, we have listed the solutions in order of non-decreasing percentage cost increase. We see that a cost very close to the baseline (minimum possible) cost is achieved for most of the instances when the proportion of minimum-pay couriers is 70% or less; on average, the cost of the solutions for $\gamma = 70\%$ is only one percent higher than the baseline. Moreover, the cost does not increase drastically when the proportion of minimum-pay courier grows; even if all couriers have a minimum pay guarantee, the cost is only 4.6 % above the baseline, on average, and never more than 9%. Therefore, for these set of instances, offering a minimum pay guaranty to a large portion of the couriers is probably worth it, given the associated benefits.

These results can, in part, be explained by considering the relations between courier utilization levels, expected work package execution times, and courier compensation schemes. Observe that the guaranteed pay rate, i.e., \$15 per hour, is 1.5 times the payment for a delivery, i.e., \$10 per delivery. Thus, if a courier completes more than 1.5 deliveries per hour, the minimum pay guarantee becomes inactive. This implies that if the "right" work packages are assigned to couriers, in the sense that the work load is

distributed "evenly" among the couriers, it may be possible to get minimum cost solutions that are quite close to the baseline. Examining the results in Table 8 reveals that, on average, a courier is busy about 36 minutes per hour. Furthermore, the expected work package duration is $2 \times 6.69 + 2 \times 4 = 21.38$ minutes, as the average travel time between the restaurant and the order drop-off location is 6.69 minutes, and the pick-up and drop-off of an order takes 4 minutes. This suggests that in the 36 minutes, a courier completes $36/21.38 = 1.68$ orders, which is slightly more than 1.5, which means that the minimum pay guarantee becomes inactive.

Another question of interest is the effect of order bundling on cost. Since the compensation to couriers is not affected by the bundling of orders, the effect of bundling of orders on cost may be limit. This is confirmed by the results of our experiments, as can be seen in Table 3, where we report the effect of allowing bundles of at most two orders. The average cost improvement is less than 0.5 percent.

Table 3: Cost benefits of order bundling.

	minimum-pay proportion			
Bundle Size	55%	70%	85%	100%
1	3359.1	3367.4	3388.3	3452.2
2	3349.9	3358.1	3373.8	3434.2
Improvement	0.003	0.003	0.004	0.005

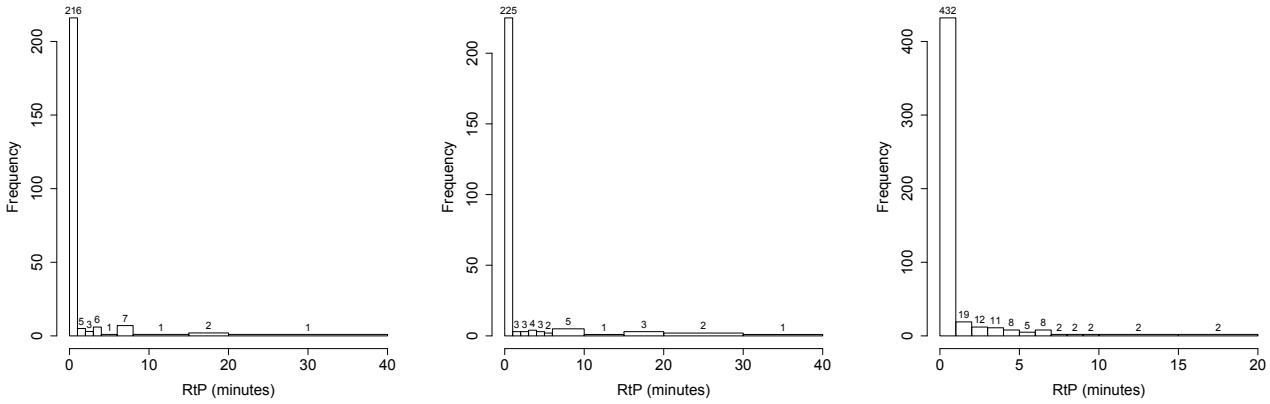
6.3.2 Click-to-door minimization

Detailed results of the experiments in which the average click-to-door time is minimized can be found in Table 9 in Appendix B, but closely related and more informative results (involving ready-to-pickup times) can be found in Figure 2.

Examining the solutions reveals that the observed CtD averages are close to best possible. The average CtD is around 30.6 minutes, of which 19.14 minutes can be attributed to (average) meal preparation time, 6.69 minutes can be attributed to (average) travel time from restaurant to drop-off location, and 4 minutes can be attributed to service time. That is, 98% of the observed average CtD time cannot be avoided. Focusing on average RtP time takes out these unavoidable factors. In Figure 2, we show the distributions of the RtP times in the solutions to three of the instances (the solutions to the other instances show similar patterns). We see that the solutions have small RtP values (less than one minute for a very large fraction of the orders). These results indicate that there is a little room for improvement in the average CtD time, i.e., increasing the delivery capacity or using the available capacity more efficiently will have little of no effect on the average CtD time.

The effect of delivery capacity planning on average CtD time can be analyzed by comparing the solutions to the S1 and S2 instances. In the S2 instances, the courier shifts have been determined using an optimization approach (see Reyes et al. (2018) for more details). In Table 4, we show the CtD time, RtD time, and RtP time, averaged over the S1 and S2 instances. We observe that, as expected, the difference in average CtD time is small, the difference in average RtP time is surprisingly large, optimized courier shifts reduce the average RtP from 1.07 minutes to 0.46 minutes, a reduction of the more than 50%. This clearly demonstrates the importance of having the right number of couriers at the right time (and having the dispatching technology to ensure that the couriers are at the right location).

Similar to what we have seen for cost minimization, the impact of bundling of orders is small when for CtD minimization; see Table 5. The reason for the small improvement is likely the fact that the delivery



(a) Instance 0o50t75s1p100 (b) Instance 0r50t75s1p100 (c) Instance 0o100t75s1p100

Figure 2: Ready-to-pickup distributions in the solutions.

Table 4: Effect of off-line courier schedule optimization

	CtD			RtD			RtP		
Courier Schedules	mean	min	max	mean	min	max	mean	min	max
s1	30.89	8.50	73.00	11.75	5.00	44.92	1.07	0.00	30.08
s2	30.29	8.25	71.00	11.15	5.00	27.75	0.46	0.00	16.42
Improvement	0.02	0.03	0.03	0.05	0.00	0.38	0.57	0.00	0.45

Table 5: Service Time Improvement Benefits of Order Bundling

	CtD		RtD		RtP	
Bundle Size	mean	max	mean	max	mean	max
1	30.59	72.00	11.45	36.33	0.76	23.25
2	30.54	71.54	11.40	34.63	0.69	21.08
Improvement	0.002	0.006	0.005	0.047	0.100	0.093

capacity is high, reflected by the very small RtP times in the solutions. When courier availability is not a limiting factor, order bundling offers no advantage in terms of service related objectives, e.g., CtD time and RtD time, as bundling of orders adds wait time at the restaurant and circuity to the delivery route.

6.3.3 Cost vs click-to-door time

Even though minimizing CtD time and minimizing cost may not be fully aligned, they are not necessarily in conflict either. Here, we analyze the interaction between these objectives. Specifically, we investigate whether a hierarchical approach, in which we first minimize cost and then minimize CtD time subject to the constraint that the cost must be close to the minimum possible cost (i.e., within α times the minimum cost) offers advantages. Detailed results for $\alpha \in \{1, 1.05\}$ can be found in Tables 10 and 11 in Appendix C, and are summarized (for $\alpha = 1$) in Figure 3.

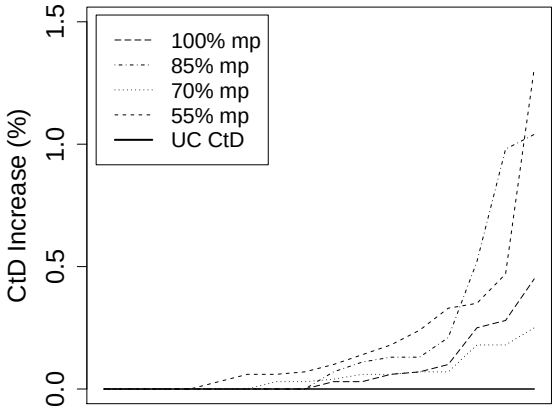


Figure 3: Unconstrained vs constrained CtD solutions ($\alpha = 1$)

We observe that in almost all cases, the difference between the CtD time in the solution to the unconstrained and constrained variants is less than one percent. This suggests that minimizing cost will, in most situations, also minimize CtD time. This (somewhat surprising) result is likely due to the balanced courier workloads and the structure of the compensation scheme. The implication is that if a meal delivery network employs a compensation scheme that ensures a large enough courier pool and that is aligned with the expected courier utilization, then high service quality can be achieved without an increase in operating costs.

6.3.4 Demand management

Even though capacity management, through well-designed courier compensation schemes and optimized courier shift schedules, is critical in an operation that relies on crowd-sourced delivery capacity, that does not mean that there is no place for demand management. And demand management in the context of meal delivery operations is especially interesting as it may be possible to redirect demand. Here, we investigate the potential benefits of such *demand redirection* strategies. To obtain an optimistic assessment of the potential benefits of demand redirection, we consider the situation in which the meal delivery company can select the restaurant from which to pick-up the order placed by a diner, i.e., we assume that the meal delivery company can convince any diner to place an order at the restaurant that is most favorable to the company's delivery operation. We refer to this setting as meal delivery with *perfect-demand-management*

(PDM). To assess the benefits of PDM, an instance is solved as before, i.e., with the same order placement and order ready times, but the optimizer is allowed to choose any restaurant as the pick-up location for the order. Note that such a change can be implemented in CR by simply replacing the delivery time function g^c with $g^{*c}(o,t,b) = \min_{r \in R} g^c(o,\ell_r,t,b)$. Detailed results of the PDM solutions when minimizing either cost or average CtD time can be found in Tables 13 and 12, respectively, in Appendix D.

Table 6 summarizes the results when minimizing CtD time. The results highlight the tremendous potential of demand redirection. Considering the fact that PDM does not change meal preparation time,

Table 6: PDM CtD Improvements

Solutions	CtD	RtD	RtP
without PDM	30.71	11.49	0.72
with PDM	25.60	6.42	0.26
Improvement	0.17	0.44	0.64

which accounts for more than 80% of CtD time, a 17% reduction in average CtD time is significant. A more in-depth look at the solutions reveals that the improvements are achieved by choosing restaurants that are closer to the drop-off location of a meal that is picked up (as opposed to choosing restaurants that are closer to the drop-off location of a meal that has been delivered). However, it is interesting to see that not only the average RtD time, but also the average RtP time has improved. More specifically, we see that in about 85% of the cases, the redirection is to a restaurant that is closer to the diner, and in about 15% of the cases, the redirection is to a restaurant that reduces the travel time for the courier to the courier's next pickup.

We also investigated the potential of PDM to reduce costs. As can be seen in Table7, the ability to use PDM to reduce costs is rather limited. In all the cases the reduction is less than three percent. This

Table 7: PDM Cost Improvements

Solution	55% mp	70% mp	%85 mp	%100 mp
without PDM	2482.1	2489.0	2499.9	2543.2
with PDM	2470.7	2472.6	2474.6	2514.6
Improvement	0.005	0.007	0.010	0.011

is not unexpected, since the margin for cost improvement is small as discussed in Section 6.3.1.

7 Discussion

To study the MDRP, we have introduced a formulation based on work-packages that handles time in a novel way and requires both column and row generation for its solution. Traditional column generation formulations have been successful because they embed more information in the definition of the variables, thereby avoiding the need for complex constraints to model relations between "basic" variables (e.g., nonlinear constraints to represent the relation between a departure time and a departure direction). However, embedding too much information into the definition of variables can complicate the solution of the pricing problem and render the formulation computationally ineffective. Our work-package formulation seeks to properly balance the information embedded into the definition of the variables and

the complexity of the constraints required to model relationships between variables. We believe these ideas can be extended to many other problems, especially when the problem allows tasks/processes to be decomposed into independent pieces and the coordination/concatenation of the pieces can be modeled with relatively simple constraints. The MDRP has such a structure. The itinerary of a courier can be decomposed into independent pieces of work representing the pickup and delivery of an order (bundle). The temporal and spatial consistency between successive pieces of work can be modeled using relatively simple constraints. Note that an entire courier itinerary has a much more complex structure (e.g., it may contain cycles – when a courier picks up meals at the same restaurant several times), which makes the pricing problem harder to solve.

For the MDRP, we have shown that by exploiting properties of optimal solutions, we are even able to handle continuous time without producing intractable formulations. In transportation problems, the presence of time often leads to formulations based on time-expanded network models, which can quickly become prohibitively large. Recently, Boland et al. (2017) have proposed an approach that dynamically manages partially time-expanded networks to overcome this challenge. Work-package formulations may provide a viable alternative.

The MDRP is a member of a class of dynamic routing and scheduling problems that is expanding and attracting a lot of interest because it is closely aligned with new business models that are transforming urban logistics operations. With a retail industry that is rapidly changing and being shaped by e-commerce, the lessons learned from studying the MDRP may be of great value in other last-mile delivery contexts with stringent service level commitments. Our findings highlight the importance of sizing and scheduling delivery capacity (i.e., the set of couriers) and shows that well-managed delivery capacity reduces (almost eliminates) the need for order bundling. Another significant finding is that there appears to be great potential for demand management strategies focused on redirection (i.e., restaurant substitution).

The emergence of business models that rely on crowd-source delivery (as is typically the case in meal delivery) are changing the nature of the routing and scheduling problems of interest. Rather than being concerned, primarily, with finding optimal (or high-quality) routes to use available delivery capacity as effectively as possible, we have to be concerned with how to ensure the availability of reliable delivery capacity at the right time and at the right cost. This change in focus gives rise to many new and fascinating research opportunities.

Acknowledgment

This work was supported by the Scientific and Technological Research Council of Turkey (TUBITAK) under the grant number 2219. We acknowledge and appreciate the insightful discussions with the operations research and algorithm development team at Grubhub that have informed much of our thinking, and, thus, the analyses described above.

References

C. Archetti, D. Feillet, and M.G. Speranza. Complexity of routing problems with release dates. *European Journal of Operational Research*, 247(3):797–803, 2015.

Claudia Archetti, Martin Savelsbergh, and M. Grazia Speranza. The vehicle routing problem with occasional drivers. *European Journal of Operational Research*, 254(2):472 – 480, 2016. ISSN 0377-2217.

N. Azi, M. Gendreau, and J.-Y. Potvin. A dynamic vehicle routing problem with multiple delivery routes. *Annals of Operations Research*, 199(1):103–112, 2012.

Evan Bakker. *The on-demand meal delivery report: Sizing the market, outlining the business models, and determining the future market leaders*. `http://read.bi/2bU7EuD`, 2016. Accessed: 2017-11-02.

Cynthia Barnhart, Christopher A Hane, Ellis L Johnson, and Gabriele Sigismondi. A column generation and partitioning approach for multi-commodity flow problems. *Telecommunication Systems*, 3(3):239–258, 1994.

Cynthia Barnhart, Ellis L Johnson, George L Nemhauser, Martin WP Savelsbergh, and Pamela H Vance. Branch-and-price: Column generation for solving huge integer programs. *Operations research*, 46(3): 316–329, 1998.

Cynthia Barnhart, Christopher A Hane, and Pamela H Vance. Using branch-and-price-and-cut to solve origin-destination integer multicommodity flow problems. *Operations Research*, 48(2):318–326, 2000.

Russell W Bent and Pascal Van Hentenryck. Scenario-based planning for partially dynamic vehicle routing with stochastic customers. *Operations Research*, 52(6):977–987, 2004.

G. Berbeglia, J.F. Cordeau, and G. Laporte. Dynamic pickup and delivery problems. *European Journal of Operational Research*, 202(1):8–15, 2010.

Natashia Boland, Mike Hewitt, Luke Marshall, and Martin Savelsbergh. The continuous-time service network design problem. *Operations Research*, 65(5):1303–1321, 2017.

Zhi-Long Chen and Hang Xu. Dynamic column generation for dynamic vehicle routing with time windows. *Transportation Science*, 40(1):74–88, 2006.

Amy Mainville Cohn and Cynthia Barnhart. Improving crew scheduling by incorporating key maintenance routing decisions. *Operations Research*, 51(3):387–396, 2003.

Iman Dayarian and Martin Savelsbergh. Crowdshipping and same-day delivery: Employing in-store customers to deliver online orders. *Optimization Online*, 2017.

Zeger Degraeve and Raf Jans. A new dantzig-wolfe reformulation and branch-and-price algorithm for the capacitated lot-sizing problem with setup times. *Operations research*, 55(5):909–920, 2007.

Guy Desaulniers. Branch-and-price-and-cut for the split-delivery vehicle routing problem with time windows. *Operations research*, 58(1):179–192, 2010.

Michel Gendreau, Francois Guertin, Jean-Yves Potvin, and Eric Taillard. Parallel tabu search for real-time vehicle routing and dispatching. *Transportation science*, 33(4):381–390, 1999.

Arthur M Geoffrion and Ramayya Krishnan. Prospects for operations research in the e-business era. *Interfaces*, 31(2):6–36, 2001.

Gregory A Godfrey and Warren B Powell. An adaptive dynamic programming algorithm for dynamic fleet management, i: Single period travel times. *Transportation Science*, 36(1):21–39, 2002.

Lars M Hvattum, Arne Løkketangen, and Gilbert Laporte. Solving a dynamic and stochastic vehicle routing problem with a sample scenario hedging heuristic. *Transportation Science*, 40(4):421–438, 2006.

Soumia Ichoua, Michel Gendreau, and Jean-Yves Potvin. Diversion issues in real-time vehicle dispatching. *Transportation Science*, 34(4):426–438, 2000.

Soumia Ichoua, Michel Gendreau, and Jean-Yves Potvin. Vehicle dispatching with time-dependent travel times. *European journal of operational research*, 144(2):379–396, 2003.

M. A. Klapp, A. L. Erera, and A. Toriello. The one-dimensional dynamic dispatch waves problem. *Transportation Science*, 2016. doi: 10.1287/trsc.2016.0682.

Mathias A. Klapp. *Dynamic optimization for same-day delivery operations*. PhD thesis, Georgia Institute of Technology, 2016.

Andreas Mladenow, Andreas Mladenow, Christine Bauer, Christine Bauer, Christine Strauss, and Christine Strauss. crowd logistics: the contribution of social crowds in logistics activities. *International Journal of Web Information Systems*, 12(3):379–396, 2016.

Roberto Montemanni, Luca Maria Gambardella, Andrea Emilio Rizzoli, and Alberto V Donati. Ant colony system for a dynamic vehicle routing problem. *Journal of Combinatorial Optimization*, 10(4): 327–343, 2005.

Morgan Stanley Research. Is online food delivery about to get 'amazoned'? `https://www.morganstanley.com/ideas/online-food-delivery-market-expands/`, sep 2017. Accessed: 2017-10-04.

Clara Novoa and Robert Storer. An approximate dynamic programming approach for the vehicle routing problem with stochastic demands. *European Journal of Operational Research*, 196(2):509–515, 2009.

Kyungchul Park, Seokhoon Kang, and Sungsoo Park. An integer programming approach to the bandwidth packing problem. *Management science*, 42(9):1277–1291, 1996.

Mark Parker and Jennifer Ryan. A column generation algorithm for bandwidth packing. *Telecommunication Systems*, 2(1):185–195, 1993.

Victor Pillac, Michel Gendreau, Christelle Guéret, and Andrés L Medaglia. A review of dynamic vehicle routing problems. *European Journal of Operational Research*, 225(1):1–11, 2013.

Harilaos N Psaraftis. A dynamic programming solution to the single vehicle many-to-many immediate request dial-a-ride problem. *Transportation Science*, 14(2):130–154, 1980.

H.N. Psaraftis, M. Wen, and C.A. Kontovas. Dynamic vehicle routing problems: Three decades and counting. *Networks*, 67(1):3–31, 2016.

Reuters. *How Amazon is making package delivery even cheaper*. `http://fortune.com/2016/02/18/amazon-flex-deliveries/`, 2016.

Damián Reyes, Alan L. Erera, and Martin W.P. Savelsbergh. Complexity of routing problems with release dates and deadlines. *European Journal of Operational Research*, 2017. ISSN 0377-2217.

Damian Reyes, Alan Erera, Martin Savelsbergh, Sagar Sahasrabudhe, and Ryan O'Neil. *The meal delivery routing problem*. Optimization Online, 2018.

Hugo P Simao, Jeff Day, Abraham P George, Ted Gifford, John Nienow, and Warren B Powell. An approximate dynamic programming algorithm for large-scale fleet management: A case application. *Transportation Science*, 43(2):178–197, 2009.

Barrett W Thomas. Dynamic vehicle routing. *Wiley Encyclopedia of Operations Research and Management Science*, 2010.

Marlin W Ulmer, Barrett W Thomas, and Dirk C Mattfeld. Preemptive depot returns for a dynamic same-day delivery problem. Technical report, working paper, 2016.

Rinde R.S. van Lon, Eliseo Ferrante, Ali E. Turgut, Tom Wenseleers, Greet Vanden Berghe, and Tom Holvoet. Measures of dynamism and urgency in logistics. *European Journal of Operational Research*, 253(3):614 – 624, 2016. ISSN 0377-2217. doi: https://doi.org/10.1016/j.ejor.2016.03.021.

Stacy A Voccia, Ann Melissa Campbell, and Barrett W Thomas. The same-day delivery problem for online purchases. *Transportation Science*, 2017.

Jian Yang, Patrick Jaillet, and Hani Mahmassani. Real-time multivehicle truckload pickup and delivery problems. *Transportation Science*, 38(2):135–148, 2004.

Barış Yıldız and Oya Ekin Karaşan. Regenerator location problem in flexible optical networks. *Operations Research*, 65(3):595–620, 2017.

Barış Yıldız, Okan Arslan, and Oya Ekin Karaşan. A branch and price approach for routing and refueling station location model. *European Journal of Operational Research*, 248(3):815–826, 2016.

Barıs Yıldız, Oya Ekin Karasan, and Hande Yaman. Network design problem with relays. Optimization Online, 2017.

Appendix A Cost Minimization Solutions

In Table 8, the columns labeled *gap* indicate the optimality gaps for the best IP solution versus the LP lover bound. The column labeled *cost* shows the base case optimal solution with $\gamma = 55\%$. The columns labeled *r.cost* indicate the optimal cost as a percentage of the base case cost. The average courier utilization levels, where the courier utilization level is defined as the number of courier busy hours divided by the total number of courier hours, are shown in the columns labeled *c.util*.

Table 8: Cost minimization

	55% minimum-pay			70% minimum-pay			80% minimum-pay			100% minimum-pay		
Instance	gap	cost	c.util	gap	r.cost	c.util	gap	r.cost	c.util	gap	r.cost	c.util
0r50t75s1p100	0.000	2420.0	0.56	0.004	1.004	0.57	0.006	1.006	0.57	0.028	1.032	0.58
0r50t75s1p125	0.002	2425.0	0.57	0.002	1.000	0.57	0.008	1.006	0.60	0.031	1.033	0.59
0r50t75s2p100	0.003	2427.5	0.56	0.005	1.002	0.58	0.010	1.007	0.59	0.025	1.023	0.56
0r50t75s2p125	0.000	2420.0	0.57	0.014	1.014	0.56	0.011	1.011	0.59	0.029	1.030	0.56
0r50t100s1p100	0.005	2432.5	0.67	0.017	1.012	0.67	0.015	1.010	0.67	0.036	1.035	0.67
0r50t100s1p125	0.006	2435.0	0.66	0.006	1.000	0.68	0.012	1.006	0.68	0.039	1.037	0.69
0r50t100s2p100	0.005	2431.3	0.65	0.013	1.008	0.65	0.017	1.012	0.67	0.036	1.032	0.67
0r50t100s2p125	0.005	2432.5	0.66	0.007	1.002	0.68	0.010	1.005	0.67	0.019	1.014	0.68
0o50t75s1p100	0.006	2535.0	0.58	0.008	1.002	0.57	0.014	1.008	0.58	0.026	1.021	0.59
0o50t75s1p125	0.004	2530.0	0.56	0.003	0.999	0.57	0.012	1.008	0.58	0.024	1.021	0.59
0o50t75s2p100	0.002	2526.3	0.55	0.000	0.998	0.55	0.007	1.004	0.56	0.017	1.015	0.56
0o50t75s2p125	0.007	2537.5	0.54	0.001	0.994	0.55	0.011	1.004	0.55	0.019	1.012	0.56
0o50t100s1p100	0.010	2545.5	0.68	0.010	1.000	0.69	0.018	1.008	0.69	0.031	1.022	0.69
0o50t100s1p125	0.015	2558.0	0.69	0.017	1.002	0.69	0.019	1.004	0.69	0.040	1.026	0.70
0o50t100s2p100	0.003	2527.5	0.60	0.010	1.007	0.64	0.012	1.009	0.62	0.021	1.018	0.65
0o50t100s2p125	0.004	2530.0	0.63	0.005	1.001	0.63	0.009	1.005	0.64	0.028	1.025	0.64
0o100t75s1p100	0.011	5105.8	0.54	0.026	1.015	0.55	0.019	1.008	0.54	0.064	1.063	0.54
0o100t75s1p125	0.011	5105.8	0.55	0.010	0.999	0.55	0.029	1.019	0.56	0.045	1.041	0.55
0o100t75s2p100	0.023	5170.0	0.51	0.010	0.987	0.52	0.023	1.000	0.53	0.027	1.004	0.52
0o100t75s2p125	0.006	5082.5	0.52	0.017	1.010	0.53	0.018	1.012	0.54	0.023	1.017	0.53
0o100t100s1p100	0.016	5131.3	0.63	0.024	1.009	0.64	0.030	1.014	0.63	0.050	1.042	0.64
0o100t100s1p125	0.019	5148.3	0.63	0.014	0.995	0.64	0.017	0.998	0.65	0.042	1.030	0.65
0o100t100s2p100	0.006	5080.0	0.58	0.012	1.006	0.59	0.024	1.019	0.60	0.035	1.031	0.61
0o100t100s2p125	0.006	5082.5	0.61	0.002	0.996	0.61	0.018	1.012	0.61	0.024	1.018	0.61
Average	0.007	3359.1	0.60	0.010	1.003	0.60	0.015	1.008	0.61	0.032	1.027	0.61

Appendix B CtD Minimization Solutions

In Table 9, the column *gap* indicates the optimality gap for the best IP solution. The average, minimum, and maximum values for the CtD, RtD and RtP metrics (in minutes) are shown in the columns labeled *mean*, *min* and *max*, respectively.

Table 9: CtD Minimization

		CtD			RtD			RtP		
Instance	gap	mean	min	max	mean	min	max	mean	min	max
0r50t75s1p100	0.0000	28.52	10	63	10.60	5	37	0.76	0	32
0r50t75s1p125	0.0000	32.46	8	76	10.43	5	32	0.60	0	16
0r50t75s2p100	0.0001	27.95	9	64	10.03	5	19	0.21	0	8
0r50t75s2p125	0.0000	32.00	8	77	9.98	5	22	0.16	0	11
0r50t100s1p100	0.0000	30.91	11	66	12.98	5	44	1.39	0	37
0r50t100s1p125	0.0000	34.76	8	78	12.74	5	41	1.14	0	21
0r50t100s2p100	0.0001	29.93	9	65	12.00	5	27	0.43	0	14
0r50t100s2p125	0.0000	33.86	8	78	11.83	5	27	0.26	0	14
0o50t75s1p100	0.0004	27.55	8	70	10.95	5	55	1.04	0	39
0o50t75s1p125	0.0001	31.15	8	76	10.81	5	34	0.89	0	23
0o50t75s2p100	0.0000	26.83	8	63	10.23	5	19	0.30	0	8
0o50t75s2p125	0.0000	30.52	8	76	10.17	5	19	0.24	0	12
0o50t100s1p100	0.0001	29.87	8	82	13.27	5	67	1.54	0	48
0o50t100s1p125	0.0002	33.51	8	85	13.16	5	67	1.43	0	48
0o50t100s2p100	0.0000	28.96	8	65	12.37	5	30	0.64	0	15
0o50t100s2p125	0.0001	32.60	8	78	12.25	5	25	0.53	0	17
0o100t75s1p100	0.0145	27.48	8	63	10.44	5	36	0.77	0	20
0o100t75s1p125	0.0221	31.54	8	76	10.64	5	39	0.98	0	30
0o100t75s2p100	0.0251	27.48	8	63	10.44	5	27	0.76	0	11
0o100t75s2p125	0.0211	31.31	9	76	10.41	5	24	0.72	0	17
0o100t100s1p100	0.0334	30.07	9	64	13.03	5	45	1.65	0	25
0o100t100s1p125	0.0017	32.88	8	77	11.98	5	42	0.60	0	22
0o100t100s2p100	0.0000	28.58	8	65	11.54	5	23	0.13	0	8
0o100t100s2p125	0.0315	33.46	8	82	12.56	5	71	1.15	0	62
Average	0.0063	30.59	8.4	72.0	11.45	5.0	36.3	0.76	0.0	23.3

Appendix C Cost Constrained CtD Solutions

Tables 10 and 11 facilitate comparing solutions obtained for $\alpha = 1$ and $\alpha = 1.05$ against the unconstrained CtD optimization results. The column labeled *CtD* restates the average CtD time values. Considering these as base values, the columns labeled *r.CtD* indicate the CtD time values as a percentage of the base values. Columns labeled *gap* indicate the optimality gap values.

Table 10: Cost constrained CtD minimization ($\alpha = 1$)

			Cost Constrained CtD Minimization							
	CtD Minimization		%55 minimum-pay		%70 minimum-pay		%85 minimum-pay		%100 minimum-pay	
instance	gap	CtD	gap	r.CtD	gap	r.CtD	gap	r.CtD	gap	r.CtD
0o50t100s1p100	0.0001	29.87	0.0001	1.0000	0.0001	1.0000	0.0001	1.0000	0.0001	1.0000
0o50t100s1p125	0.0002	33.51	0.0001	1.0000	0.0002	1.0000	0.0002	1.0000	0.0002	1.0000
0o50t100s2p100	0.0000	28.96	0.0011	1.0014	0.0001	1.0003	0.0035	1.0052	0.0019	1.0028
0o50t100s2p125	0.0001	32.60	0.0006	1.0006	0.0004	1.0006	0.0087	1.0098	0.0004	1.0003
0o50t75s1p100	0.0004	27.55	0.0004	1.0000	0.0004	1.0000	0.0004	1.0000	0.0004	1.0000
0o50t75s1p125	0.0001	31.15	0.0009	1.0010	0.0006	1.0006	0.0001	1.0000	0.0001	1.0000
0o50t75s2p100	0.0000	26.83	0.0003	1.0007	0.0005	1.0007	0.0077	1.0104	0.0036	1.0045
0o50t75s2p125	0.0000	30.52	0.0001	1.0003	0.0004	1.0007	0.0005	1.0013	0.0006	1.0010
0r50t100s1p100	0.0000	30.91	0.0004	1.0006	0.0000	1.0000	0.0000	1.0000	0.0001	1.0000
0r50t100s1p125	0.0000	34.76	0.0000	1.0000	0.0000	1.0000	0.0000	1.0000	0.0000	1.0000
0r50t100s2p100	0.0001	29.93	0.0022	1.0033	0.0001	1.0000	0.0008	1.0007	0.0006	1.0007
0r50t100s2p125	0.0000	33.86	0.0014	1.0024	0.0014	1.0018	0.0016	1.0021	0.0005	1.0006
0r50t75s1p100	0.0000	28.52	0.0012	1.0035	0.0002	1.0004	0.0000	1.0000	0.0000	1.0000
0r50t75s1p125	0.0000	32.46	0.0017	1.0018	0.0017	1.0018	0.0000	1.0000	0.0000	1.0000
0r50t75s2p100	0.0001	27.95	0.0024	1.0047	0.0013	1.0025	0.0011	1.0011	0.0025	1.0025
0r50t75s2p125	0.0000	32.00	0.0104	1.0131	0.0001	1.0003	0.0011	1.0013	0.0001	1.0003
Average	0.0001	30.7113	0.0015	1.0021	0.0005	1.0006	0.0016	1.0020	0.0007	1.0008

Table 11: Cost constrained CtD minimization ($\alpha = 1.05$)

			Cost Constrained CtD Minimization							
	CtD Minimization		%55 minimum-pay		%70 minimum-pay		%85 minimum-pay		%100 minimum-pay	
instance	gap	CtD	gap	r.CtD	gap	r.CtD	gap	r.CtD	gap	r.CtD
0o50t100s1p100	0.0001	29.87	0.0001	1.0000	0.0001	1.0000	0.0001	1.0000	0.0001	1.0000
0o50t100s1p125	0.0002	33.51	0.0001	1.0000	0.0002	1.0000	0.0002	1.0000	0.0002	1.0000
0o50t100s2p100	0.0000	28.96	0.0007	1.0007	0.0001	1.0003	0.0042	1.0045	0.0026	1.0028
0o50t100s2p125	0.0001	32.60	0.0007	1.0006	0.0005	1.0006	0.0065	1.0064	0.0004	1.0003
0o50t75s1p100	0.0004	27.55	0.0004	1.0000	0.0004	1.0000	0.0004	1.0000	0.0004	1.0000
0o50t75s1p125	0.0001	31.15	0.0009	1.0010	0.0006	1.0006	0.0001	1.0000	0.0001	1.0000
0o50t75s2p100	0.0000	26.83	0.0004	1.0007	0.0007	1.0007	0.0086	1.0089	0.0044	1.0045
0o50t75s2p125	0.0000	30.52	0.0001	1.0003	0.0004	1.0007	0.0010	1.0013	0.0006	1.0007
0r50t100s1p100	0.0000	30.91	0.0007	1.0006	0.0000	1.0000	0.0000	1.0000	0.0001	1.0000
0r50t100s1p125	0.0000	34.76	0.0000	1.0000	0.0000	1.0000	0.0000	1.0000	0.0000	1.0000
0r50t100s2p100	0.0001	29.93	0.0014	1.0013	0.0001	1.0000	0.0008	1.0007	0.0006	1.0007
0r50t100s2p125	0.0000	33.86	0.0007	1.0009	0.0002	1.0003	0.0009	1.0009	0.0005	1.0006
0r50t75s1p100	0.0000	28.52	0.0006	1.0007	0.0003	1.0004	0.0000	1.0000	0.0000	1.0000
0r50t75s1p125	0.0000	32.46	0.0008	1.0006	0.0005	1.0006	0.0000	1.0000	0.0000	1.0000
0r50t75s2p100	0.0001	27.95	0.0023	1.0025	0.0005	1.0007	0.0010	1.0011	0.0025	1.0025
0r50t75s2p125	0.0000	32.00	0.0015	1.0016	0.0001	1.0003	0.0012	1.0013	0.0001	1.0003
Average	0.0001	30.7113	0.0007	1.0007	0.0003	1.0003	0.0016	1.0016	0.0008	1.0008

Appendix D PDM Solutions

Table 12 compares the solutions obtained when minimizing CtD with those obtained with perfect demand management. Columns labeled CtD, RtD and RtP report the mean CtD, RtD, and RtP metrics for the CtD minimization solutions (base values) and columns labeled $r.CtD$, $r.RtD$, and $r.RtP$ indicate the values of these metrics for the PDM solutions as a percentages of the base values, respectively.

Table 12: PDM with CtD minimization

	P_{CTD} Solution			PDM Improvements (%)		
Instance	CtD	RtD	RtP	r.CtD	r.RtD	r.RtP
0r50t75s1p100	28.52	10.60	0.76	84.99	59.62	30.26
0r50t75s1p125	32.46	10.43	0.6	87.06	59.73	30.00
0r50t75s2p100	27.95	10.03	0.21	85.94	60.82	42.86
0r50t75s2p125	32.00	9.98	0.16	87.81	60.82	43.75
0r50t100s1p100	30.91	12.98	1.39	81.11	55.01	30.94
0r50t100s1p125	34.76	12.74	1.14	83.40	54.63	29.82
0r50t100s2p100	29.93	12.00	0.43	82.36	56.00	34.88
0r50t100s2p125	33.86	11.83	0.26	84.58	55.87	42.31
0o50t75s1p100	27.55	10.95	1.04	82.79	56.71	40.38
0o50t75s1p125	31.15	10.81	0.89	84.82	56.15	31.46
0o50t75s2p100	26.83	10.23	0.3	83.56	56.89	40.00
0o50t75s2p125	30.52	10.17	0.24	85.62	56.93	41.67
0o50t100s1p100	29.87	13.27	1.54	78.54	51.77	42.21
0o50t100s1p125	33.51	13.16	1.43	80.69	50.84	33.57
0o50t100s2p100	28.96	12.37	0.64	79.14	51.17	34.38
0o50t100s2p125	32.60	12.25	0.53	81.53	50.86	24.53
Average	30.71	11.49	0.72	83.37	55.86	35.81

Table 13 compares the solutions obtained when minimizing cost with those obtained with perfect demand management. The section labeled P_{COST} presents the minimum cost solution values (base values) and the section labeled PDM Solution presents the percentage improvements when perfect demand management is in place.

Table 13: PDM with cost minimization

	P_{COST} Solution				PDM Solution			
instance	55% mp	70% mp	%85 mp	%100 mp	55% mp	70% mp	%85 mp	%100 mp
0o50t100s1p100	2545.50	2545.50	2565.50	2600.50	0.991	0.992	0.985	0.986
0o50t100s1p125	2558.00	2563.00	2568.00	2625.50	0.985	0.983	0.983	0.974
0o50t100s2p100	2527.50	2545.00	2551.25	2573.75	0.997	0.990	0.992	0.987
0o50t100s2p125	2530.00	2532.50	2542.50	2592.50	0.996	0.995	0.991	0.978
0o50t75s1p100	2535.00	2540.00	2556.00	2588.00	0.994	0.992	0.988	0.987
0o50t75s1p125	2530.00	2528.00	2550.00	2583.00	0.996	0.997	0.993	0.993
0o50t75s2p100	2526.25	2521.25	2537.50	2563.75	0.998	1.000	0.993	0.991
0o50t75s2p125	2537.50	2522.50	2548.75	2567.50	0.993	1.000	0.991	0.989
0r50t100s1p100	2432.50	2462.50	2457.50	2517.75	0.997	0.987	0.986	0.993
0r50t100s1p125	2435.00	2435.00	2450.00	2525.00	0.994	0.994	0.988	1.000
0r50t100s2p100	2431.25	2451.25	2461.25	2510.00	0.996	0.996	0.986	0.982
0r50t100s2p125	2432.50	2436.25	2443.75	2466.25	0.995	0.994	0.994	0.995
0r50t75s1p100	2420.00	2430.00	2435.50	2497.75	1.000	0.996	0.995	0.995
0r50t75s1p125	2425.00	2425.00	2440.00	2505.00	0.998	0.998	0.993	0.998
0r50t75s2p100	2427.50	2432.50	2443.75	2482.50	0.997	0.995	0.991	0.990
0r50t75s2p125	2420.00	2453.75	2447.50	2492.50	1.000	0.986	0.990	0.982
Average	2482.09	2489.00	2499.92	2543.20	0.995	0.993	0.990	0.989