The Dynamic Dispatch Waves Problem for Same-Day Delivery

Mathias A. Klapp[1,2], Alan L. Erera[1], Alejandro Toriello[1]

[1]H. Milton Stewart School of Industrial and Systems Engieering, Georgia Institute of Technology
Atlanta, Georgia 30332
maklapp *at* gatech *dot* edu, {aerera, atoriello} *at* isye *dot* gatech *dot* edu

[2]Engineering School, Pontificia Universidad Católica de Chile
Santiago, Chile
maklapp *at* ing.puc *dot* cl

October 7, 2016

Abstract

We study same-day delivery systems by formulating the Dynamic Dispatch Waves Problem (DDWP), which models a distribution center where geographically located delivery orders realize dynamically throughout the day. At each decision epoch (wave), the system's operator chooses whether or not to dispatch a vehicle route loaded with orders ready for service, to minimize vehicle travel and penalties for unserved requests. This work extends the one-dimensional variant in [23] to any network topology. We formulate an arc-based integer programming model and design local search heuristics to solve the deterministic DDWP and use this variant to design an *a priori* solution determined only with information disclosed at the start of the operation, and provide three approaches to obtain dynamic policies from it. We design two sets of computational experiments with different settings of geography, size, information dynamism, and order timing variability. Our results suggest that our best dynamic policies can cut the average cost of an *a priori* policy by 9.1%, achieving a gap of 12.1%. The marginal value of dynamic policies is concentrated in improving order coverage and increases for instances with greater variability and information dynamism. We also analyze the tradeoff between two common SDD objectives: total cost minimization versus maximizing order coverage. We find structural differences in the solution's structure for the two cases, and empirically show marginally increasing sacrifices to be made in vehicle routing efficiency to increase request coverage.

1 Introduction

Same-day delivery (SDD) is increasingly being offered by retailers and logistics service providers to expand e-commerce. The internet sector represented over 7% of the U.S. retail industry and was expected to grow 14% annually in 2015 [18, 26]. We define SDD as a distribution service that prepares, dispatches and delivers orders to the customer's location on the same day the order from the customer is placed. Amazon, a provider of SDD, has implemented the service in more than 25 U.S. metropolitan areas as of October 2016, and has also piloted “Prime Now,” an even faster one-hour delivery service. These services are designed to satisfy demand for “instant gratification” when ordering consumer products and at the same time to discourage physical store visits [23, 36]. SDD operations pose significant challenges: maximizing customer service while minimizing costs is particularly challenging given fewer opportunities for effective consolidation of small delivery orders.

A core logistics process within any delivery system is order distribution from stocking locations to delivery locations. Distribution decisions can be divided into vehicle dispatch decisions, which select the dispatch times and the orders to be delivered in each delivery vehicle trip, and routing decisions, which sequence the order of customer visits for each dispatched trip. Dispatch decisions are particularly challenging for same-day services because of the dynamic nature of the order arrival process. Only a fraction of all potential orders may be disclosed and picked for dispatch prior to the first possible dispatch epoch; the remaining orders are revealed throughout the operating period.

Dispatch operations are frequently organized such that a fixed and finite number of possible vehicle dispatch times from each stocking location (decision epochs) are used. This type of organization is often best due to many factors, including constraints associated with driver shifts and efficiencies gained by organizing warehouse activity using wave picking.

In [23], we defined the Dynamic Dispatch Waves Problem (DDWP) and solved it for a simplified SDD distribution system with delivery locations on the one-dimensional line. The DDWP is an order delivery problem with dynamic dispatch and routing decisions for a single vehicle operating over a fixed-duration operating period (*i.e.*, a day) partitioned in W dispatch *waves*. Each dispatch wave can be thought of as a point in time when picking and packing of a set of orders is completed, and a vehicle (or vehicles) can be loaded for dispatch. In this research, a dispatch wave represents a decision epoch where a single vehicle (if available at the depot) can wait for the next wave, or alternatively be loaded and dispatched from the depot to serve a subset of *open* customer delivery orders. Open orders are defined as those ready to be dispatched and not previously served. At each wave decision epoch, complete information is known for all open orders and probabilistic information is available describing potential future orders. After the vehicle completes a dispatch route, it returns to the depot and is ready to be dispatched again. The objective is to minimize vehicle operating costs and penalties for open orders that remain unserved at the end of the operating period. In this paper, we extend the model in [23] to a general network topology to make it compatible with same-day delivery operations using a typical road network.

The DDWP presents interesting challenges because of two fundamental tradeoffs discussed in [23]. First, there is a tradeoff between waiting and dispatching a vehicle. When a vehicle is dispatched, the set of open orders waiting to be served is reduced, but the opportunity to observe and serve future orders arriving geographically nearby to ones in the current route is lost. Conversely, when the vehicle is not dispatched, the time remaining to serve the set of open and future orders is reduced. Second, there is a tradeoff between dispatching longer routes serving many orders versus shorter ones with fewer visits. The former consume more time and keep the vehicle away from the depot longer, but require less time per customer visited due to density economies. Shorter routes require more time per customer, but enable the vehicle to be reused sooner.

The DDWP can be classified within the broad family of stochastic vehicle routing problems (VRP). These problems are extensions of the deterministic VRP [21, 34], where some parameters are unknown before planning is completed. The simplest models for this problem family are *a priori* optimization models [22], where an initial solution is designed before the operation starts and then pre-defined recourse rules are used to (potentially) modify it during operation as information is disclosed; see [14, 17, 20] for recent surveys. More complex models for dynamic-stochastic VRPs adapt to revealed information during the operating period, and allow for re-optimization of routing and scheduling decisions; see [25, 27, 32].

The problem that we consider in this paper is most similar to the VRP with probabilistic customer arrivals (PC-VRP) where customer orders realize over time with uncertainty; see [15, 19, 22, 24, 35] for examples of *a priori* models and [2, 3, 4, 5, 11, 37] for dynamic models for such problems. Our problem differs from those considered in this literature, which are designed primarily for pick-up operations. In pick-up operations, a vehicle route can be modified after dispatch, both by re-routing existing customers and adding new customers. In contrast, our order delivery model assumes customized orders to be delivered from a depot to the customer; it is not possible to add new customers to a route that has been previously dispatched without returning the vehicle to the depot.

Another closely-related model is the Dynamic Multiperiod Routing Problem (DMPRP) for delivery operations considered in [2], which models a depot with a fleet of capacitated vehicles transporting orders over a time horizon partitioned into days. Dynamic requests with service time windows are disclosed over time, and each day the decision maker chooses which customers to serve or postpone, and the corresponding vehicle routes. The problem is solved heuristically running a daily prize-collecting VRP over the set of open orders; the prize for each order is set according to an increasing function of proximity to the service deadline and a decreasing function of geographic proximity to probabilistic future orders. Note that this model assumes that a vehicle is available for dispatch at every dispatch epoch (wave). In SDD operations, it may be sensible to dispatch a vehicle on a route that spans multiple waves; in this case, if a vehicle is dispatched at wave t on a route of duration x to serve a set of orders, it cannot be dispatched again until a wave after its return to the depot at $t+x$.

A deterministic order delivery problem related to the DDWP is the VRP with release dates (VRP-rd) [7, 16] that considers a logistics system dispatching orders from a depot with known order release times. A feasible solution of the VRP-rd serves all orders. In [16], the authors study the deterministic VRP-rd with time windows and a homogeneous fleet of capacitated vehicles. The authors provide heuristic solutions focused on minimizing the vehicle distance travelled. In addition to ignoring the stochastic and dynamic nature of customer delivery orders, this work does not consider the problem of accepting or rejecting request which is important when a system operates with a fixed fleet of vehicles and a delivery deadline (or deadlines).

Perhaps the closest problem in the literature to the DDWP is the Same Day Delivery Problem for Online Sales [36], which considers a model for an order delivery system with dynamic order arrival and a fleet of vehicles, each performing multiple trips per day. However, there are fundamental differences in assumptions and solution methodologies between this setting and the DDWP. The model in [36] assumes narrow request service time windows, *i.e.*, one or two hours within a 10-hour day operating period; these tight time constraints correspond to rapid delivery settings, like Amazon's "Prime Now" service. Serving customers is the priority in this model, and the objective is to maximize the number of orders served, while ignoring vehicle travel costs. Alternately, the DDWP that we consider generalizes the objective to consider a weighted combination of costs for not serving customers and vehicle travel costs. An additional difference is the choice of solution approach; [36] finds solutions by adapting a scenario-based-planing heuristic (see [11]), which uses a consensus function to choose a best plan among those developed for a set of simulated scenarios at each decision epoch.

Another example of closely related work is found in [8, 10], where the authors develop a VRP model with time windows and multiple delivery routes per vehicle. Small and highly constrained problem instances are solved via an exact optimization approach, while larger instances are solved with an adaptive large neighborhood search (ALNS) heuristic. The optimization model objective is to maximize the number of served orders and travel costs in hierarchical order, while multiple trips per vehicle are induced by the inclusion of a route duration constraint; results show that this constraint leads to short routes, with an average of four customers served. This model is useful for settings in which the time between order placement and vehicle dispatch must be very short, *e.g.*, in the dispatch of perishable goods such as meals. A fundamental difference between this model and the DDWP is that we treat route duration as an unconstrained decision. In [9], the authors extend the model to a dynamic setting where online orders arrive throughout the operating period and the decision maker instantaneously accepts or rejects them for service. Acceptance decisions are made using a scenario-based planning approach that estimates the insertion profit via ALNS for multiple simulated future order scenarios.

In this paper, we model a canonical **prize-collecting** version of the DDWP that captures the fundamental tradeoffs in dynamic dispatch decision-making. Our problem extends the one-dimensional variant proposed in [23] to a general network topology. We consider the following to be our primary contributions.

1. We formulate a natural model for the deterministic variant of our problem, which we leverage to provide **lower bounds** for the stochastic-dynamic case via information relaxation and simulation.

2. We use the deterministic model to find an **optimal *a priori* solution** to the stochastic variant, by showing that the *a priori* optimization problem is equivalent to a deterministic instance with known customer order arrival times and adjusted penalties. We design construction and local search heuristics to complement commercial MIP solvers to speed up the identification of solutions to problem instances.

3. We then provide three approaches to obtain **dynamic policies** using the *a priori* model. The first uses a rollout scheme to dispatch according to the *a priori* solution and iteratively update it with new

information, and the latter approaches are based on fast heuristic modifications to the initial *a priori* solution.

4. We empirically show the benefits of dynamic policies with computational experiments that suggest that the marginal improvement provided by dynamic policies both in cost reduction and in optimality gap improvements of our dynamic policies over an *a priori* one in terms of cost reduction and duality gap are important for instances with greater order arrival variability and less information disclosed before the start of the operation. We also see that the quality of a dynamic solution has less variability over all instances tested.

5. Finally, we empirically analyze the tradeoff between two common objectives in SDD: minimizing total costs (including vehicle travel time) versus maximizing order coverage. One might think that these two objectives deliver similar results, since well-sequenced routes leave more vehicle time to cover additional orders. However, we find that there are fundamental differences in the solution's structure for both cases in terms of number of vehicles dispatched, route length and initial wait at the depot, that one should expect significant sacrifices in vehicle routing efficiency in order to maximize the order fill rate, and that the distance cost of an additional customer covered becomes more expensive as order coverage increases.

The remainder of the paper is organized in the following manner. Section 2 defines the notation and formulates the DDWP, Section 3 covers the deterministic problem, and Sections 4 and 5 respectively cover *a priori* and dynamic policies. Finally, Section 6 outlines the results of a computational study, and we conclude with Section 7.

2 DDWP problem formulation

We formally define the DDWP as a Markov Decision Process (MDP); see the text [29] for a reference on MDPs. We start by describing the notation and elements of our model:

1. **Operating period.** Let $\mathscr{W} := \{1, \dots, W\}$ be the set of *waves* (decision epochs), each with equal time duration ℓ. The number $w \in \mathscr{W}$ represents the "waves-to-go" before $w = 0$, the deadline for the vehicle to finish all deliveries and return to the depot.

2. **Customer orders and geography.** Let $N := \{1,\ldots,n\}$ be the set of all *potential* customer orders. Each $i \in N$ and the depot ($i = 0$) define known locations represented by a complete undirected graph $\mathscr{G} = (N \cup \{0\}, E)$, where E is the set of edges. We assume that the vehicle takes t_e time and spends c_e to traverse an edge $e \in E$; we assume for simplicity that time and cost values are proportional to each other ($c_e = \gamma t_e$), non-negative, and satisfy the triangle inequality.

3. **Order ready times.** Each order i is ready for dispatch at a random wave $\tau_i \in \mathscr{W} \cup \{-1\}$ drawn from an order dependent distribution; if $\tau_i = -1$ the orders i does not instantiate. We assume that these ready times are independent between different orders, and that the set $\{i \in N : \tau_i \geq w\}$ is known at wave w.

4. **Penalties.** Each $i \in N$ has a non-negative penalty β_i to be paid if order i realizes and is left unattended by $w = 0$.

We now are ready to formulate an MDP model for the DDWP. If the vehicle is available at the depot at wave w, the system state is (w,R,P), where $R \subseteq N$ is the set of open orders and $P \subseteq N$ is the set of remaining *potential* orders with an unknown arrival time ($\tau_i < w$). Orders in $N \setminus \{R \cup P\}$ are "closed", meaning they were ready and served before wave w. The pair (R,P) includes two disjoint subsets of N, *i.e.*, $(R,P) \in \Xi := \{(R,P) : R,P \subseteq N, R \cap P = \emptyset\}$. The maximum number of waves W and three possible states for each order (open-closed-potential) define an $\mathscr{O}(3^n W)$ bound on the cardinality of the state space.

An action at any non-terminal state $(w,R,P) : w \geq 1$ is defined as a vehicle dispatch that serves a subset of open orders $S \subseteq R$; $S = \emptyset$ represents waiting at the depot. The set S completely defines the action and takes $t(S)$ time, *i.e.*, the minimum time required by any tour to visit S and return to the depot (a Traveling Salesman Problem (TSP) tour over $S \cup \{0\}$). Once dispatched, the vehicle cannot serve any other order until it returns for reloading at wave $q_w(S) := w - \max\{1, \lceil t(S)/\ell \rceil\}$, and S is constrained such that the vehicle returns before the end of the day, $q_w(S) \geq 0$. The subsets $S \subset R$ imply $\mathscr{O}(2^n)$ possible actions. Selecting an action S in state (w,R,P) produces a transition to state $(q_w(S), (R \setminus S) \cup F_w(S), P \setminus F_w(S))$ where $F_w(S) := \{i \in N : w > \tau_i \geq q_w(S)\}$ is the random set of newly arriving orders in waves $w-1,\ldots,q_w(S)$, and $R \setminus S$ is the set of orders in R left open by action S.

Let $C_w(R,P)$ be a set function representing the minimum expected cost-to-go at state (w,R,P) and let $C^*(\hat{R}) := C_W\left(\hat{R}, N \setminus \hat{R}\right)$ be the minimum expected cost depending on $\hat{R}$, a given set of orders ready at $w = W$

(the order information known at the start of the operation). The dynamic program defined in (1) computes $C^*(\hat{R})$ recursively over w. Any terminal cost $C_0(R,P)$ is equal to the sum of penalties of unserved open orders R, and subsequently, for each $w \in \mathscr{W}$ the cost-to-go at state (w,R,P) is equal to the minimum cost between no dispatch and any feasible dispatch. Define an optimal action as a subset $S_w(R,P) \subseteq R$ that attains $C_w(R,P)$ at a given state (w,R,P) and an optimal policy as a vector of optimal actions for each possible state of the system.

$$C_0(R,P) = \textstyle\sum_{i\in R} \beta_i, \qquad \forall (R,P) \in \Xi \tag{1a}$$

$$C_w(R,P) = \min_{S\subseteq R: q_w(S)\geq 0} \left\{ t(S) + \mathbb{E}_{F_w(S)} \left[C_{q_w(S)} \left(R \setminus S \cup F_w(S), P \setminus F_w(S) \right) \right] \right\}, \qquad \forall w \in \mathscr{W}, \forall (R,P) \in \Xi. \tag{1b}$$

The model (1) shows the difficulty in finding an optimal policy; it has exponentially many states ($\mathscr{O}(3^n W)$), exponentially many actions per state ($\mathscr{O}(2^n)$), and exponentially many terms in the expectations ($\mathscr{O}(2^n)$). Moreover, it is NP-Hard to evaluate the cost-to-go at any state (w,R,P) given an action S, because we need to compute $t(S)$ which requires the solution of a TSP over $S \cup \{0\}$. Given all these levels of difficulty, we will focus on finding good heuristics policies for the DDWP.

3 The Deterministic DDWP

Suppose the wave τ_i at which order $i \in N$ is ready is known at the beginning of the operating horizon. Then, the set of orders ready at any wave w for each $w \in \mathscr{W}$ is known beforehand, but it remains infeasible to serve an order $i \in N$ with a vehicle dispatch prior to τ_i. Let $N_w := \{i \in N : a_i \leq w \leq \tau_i\}$ be the set of orders ready and feasible to serve at wave $w \in \mathscr{W}$; where a_i defines the latest wave to feasibly serve order i, *i.e.*, $a_i := \lceil 2t_{\{0,i\}}/\ell \rceil$. Problem 3.1 states the deterministic DDWP, which is NP-Hard since it generalizes the Prize-Collecting Traveling Salesman Problem (PC-TSP); just set all $\tau_i = 1$, $W = 1$, and $\ell = \sum_{i\in N} 2t_{0i}$.

Problem 3.1 (Deterministic DDWP). *Find a collection of mutually disjoint subsets $\{S_w \subset N_w : w \in \mathscr{W}\}$ that minimize $\sum_{w\in\mathscr{W}} \left\{ t(S_w) - \sum_{i\in S_w} \beta_i \right\}$ subject to:*

1. *$q_w(S_w) \geq 0$ for each $w \in \mathscr{W}$, and*

2. *if $S_w \neq \emptyset$, then $S_v = \emptyset$ for all $v \in \{w-1, \dots, q_w(S)\}$, for each $w \in \mathscr{W}$.*

Property 3.2 is taken from [23] and extended to a general network topology.

Property 3.2 (No wait after a dispatch)**.** *There exists an optimal solution to Problem 3.1 in which the vehicle does not wait after the first dispatch has occurred.*

In [23], we show for the one-dimensional case that an optimal solution contains consecutive vehicle dispatches with decreasing dispatch duration. This property does not hold for a general network topology. Consider a line segment with a centered depot and two orders located at each end of the line. Let $W = 6$, $\ell = 1$, $\alpha = 1$, $\beta_1 = \beta_2 = 7$, $\tau_1 = 4$, $\tau_2 = 6$, $t_{\{0,1\}} = 2$, $t_{\{0,2\}} = 1$, $t_{\{1,2\}} = 3$. Leaving any order unattended costs at least 7 and the unique solution serving both orders has cost 6. This solution dispatches a vehicle round-trip to serve order 2 at wave $w = 6$, and again at wave 4 to serve order 1.

We next formulate the deterministic DDWP as an Integer Program (IP). Define $E_w := \{e \in E, a_e \leq w \leq b_e\} \subset E$ for each $w \in \mathscr{W}$ as the set of feasible edges for a vehicle dispatch at wave w, where $a_{\{i,j\}} = \left\lceil \left(t_{\{0,i\}} + t_{\{i,j\}} + t_{\{0,j\}}\right)/\ell \right\rceil$ and $b_{\{i,j\}} = \min\{\tau_i, \tau_j\}$, for each $\{i, j\} \in E$. Also, define the cut set $E_w(S) = \{\{i, j\} \in E_w : i \in S, j \notin S\}$, for any subset $S \subseteq N_w$. Problem 3.1 is equivalent to the IP

$$C^*(\tau) = \min_{\{\mathbf{x},\mathbf{y},\mathbf{v},\mathbf{z}\}} \sum_{i \in N:\tau_i > 0} \beta_i \left(1 - \sum_{w=a_i}^{\tau_i} y_i^w\right) + \sum_{w \in \mathscr{W}} \sum_{e \in E_w} t_e x_e^w \tag{2a}$$

$$\text{s.t.} \sum_{w=a_i}^{\tau_i} y_i^w \leq 1, \qquad \forall i \in N \tag{2b}$$

$$\sum_{e \in E_w(0)} x_e^w \leq 2, \qquad \forall w \in \mathscr{W} \tag{2c}$$

$$\sum_{e \in E_w(S)} x_e^w \geq 2y_i^w, \qquad \forall w \in \mathscr{W}, \forall S \subseteq N_w, \forall i \in S \tag{2d}$$

$$\sum_{e \in E_w} t_e x_e^w \leq \ell \sum_{k<w} (w-k) v_k^w, \qquad \forall w \in \mathscr{W} \tag{2e}$$

$$\sum_{k<W} z_k + \sum_{k<W} v_k^W = 1 \tag{2f}$$

$$\sum_{k<w} v_k^w = \sum_{k>w} v_w^k + z_w, \qquad \forall w \in \mathscr{W} \setminus \{W\} \tag{2g}$$

$$v_k^w \in \{0,1\}, \qquad \forall w \in \mathscr{W}, \forall k \in \mathscr{W} \cup \{0\} : k < w \tag{2h}$$

$$z_k \in \{0,1\}, \qquad \forall k \in \mathscr{W} \cup \{0\} : k < W \tag{2i}$$

$$y_i^w \in \{0,1\}, \forall i \in N_w, \text{ and } x_e^w \in \{0,1,2\}, \forall e \in E_w. \qquad \forall w \in \mathscr{W} \qquad \text{(2j)}$$

Variable y_i^w is equal to 1 if a dispatch at wave w serves order i, and 0 otherwise; x_e^w is equal to $m \in \{0,1,2\}$ if the vehicle traverses edge e m times at a dispatch at wave w, and 0 otherwise; v_k^w is equal to 1 if a dispatch at w returns at wave k, and 0 otherwise; and z_k is equal to 1 if the vehicle waits at the depot until wave k, and 0 otherwise ($z_0 = 1$ implies no dispatch throughout the horizon).

Constraints (2b) guarantee serving each order i exactly once at wave w ($y_i^w = 1$) or, alternatively, leaving it unserved ($y_i^w = 0,\ w = a_i, \ldots, \tau_i$). Constraints (2c) - (2d) guarantee that vector $\mathbf{x}^w$ defines a feasible tour only visiting orders selected by the vector $\mathbf{y}^w$. Constraints (2e) force routes to satisfy durations limits determined by v_k^w. Finally, wave flow constraints (2f)-(2g) enforce vehicle conservation throughout time.

We can solve instances of (2) using a standard Branch & Cut approach with dynamic generation for subtour elimination cuts based on approaches for the TSP; see [6]. We start with singleton constraints (2d), *i.e.*, $S = \{i\}$. If we find an integer $\mathbf{x}^w$ for wave w at any given node in the Branch & Bound tree, we check if it has a subtour in $\mathscr{O}(n)$ running time and add the corresponding cut. Moreover, if $\mathbf{x}^w$ is fractional we can check if it violates 2-connectivity by solving a Minimum Cut problem efficiently. If the answer is yes, we add the corresponding cut from (2d) and repeat.

Consider again the stochastic DDWP defined in Section 2. We can estimate a lower bound on the optimal expected cost of (1) with a Perfect Information Relaxation (PIR) of cost C_{PIR} that disregards the "non-anticipative" dynamics of the problem and computes one solution for each possible realization of the random variables [13, 30]. To estimate the PIR, we simulate a set of $m \in \{1, \ldots, M\}$ realizations τ^m for the random vector of ready times τ, find the deterministic optimal cost $C^*(\tau^m)$ for each realization m, and take the sample average $C_{PIR} \approx \sum_{m=1}^{M} C^*(\tau^m)/M$.

4 *A priori* policies

In this section, we develop a procedure to compute an *a priori* policy in which a schedule specifying the waves at which to dispatch the vehicle and the subsets of orders to be covered at each dispatch is determined only with information disclosed at wave W.

We start by planning an optimal *a priori* policy in which no recourse actions are allowed. The operating cost of such a policy is known beforehand, and the penalties paid for unserved orders depend on future order arrivals. In this case each order has a probability $\mathbb{P}(\tau_i = w)$ to be ready at wave w. To simplify notation, the probability values include all information regarding the initial set of orders $\hat{R}$, *i.e.*, $\mathbb{P}(\tau_i = W) = 1$ if $i \in \hat{R}$ and 0 otherwise. This *a priori* problem is equivalent to solving a deterministic DDWP instance in which each order $i \in N$ is duplicated at most W times and each copy is assumed to be ready at each wave $w \in \mathscr{W}$ in the support of the random variable τ_i, with an adjusted penalty for not serving the order equal to $\beta_i P(\tau_i = w)$.

Under this "extended" deterministic model, there exists an optimal solution visiting the customers within the same location at most once. If this is not the case, we could simply delete all but the latest visit and reduce the vehicle dispatch cost without a loss on customer coverage. This observation allows us to define an *a priori* IP problem where planning to serve order i at wave w indicates that order i is indeed serviced if it arrives during any wave $v \in \{w, w+1, \ldots, W\}$; this action thus reduces the expected penalty by $\beta_i \mathbb{P}(\tau_i \geq w)$. Define the expected penalty to be paid if no vehicle dispatches are planned as $\beta^0 := \sum_{i \in N} \beta_i \mathbb{P}(\tau_i \geq 1)$, the earliest wave that a vehicle can serve order i as $b_i := \max\{w : \mathbb{P}(\tau_i = w) > 0\}$, and redefine the sets N_w and E_w accordingly. Then, the *a priori* DDWP is defined by

$$C^*_{AP}(\hat{R}) = \min_{\{\mathbf{x},\mathbf{y},\mathbf{v},\mathbf{z}\}} \beta^0 - \sum_{i \in N} \sum_{w=a_i}^{b_i} \beta_i \mathbb{P}(\tau_i \geq w) y_i^w + \sum_{w \in \mathscr{W}} \sum_{e \in E_w} t_e x_e^w \tag{3a}$$

$$\text{s.t. (2c)} - \text{(2j)},$$

$$\sum_{w=a_i}^{b_i} y_i^w \leq 1, \qquad \forall i \in N, \tag{3b}$$

which shares its feasible region with (2), but now has a stochastic objective. In case of not serving an order, we pay a penalty discounted by the arrival probability $\mathbb{P}(\tau_i \geq 1)$, and in case of planning to serve i at wave w, we pay the penalty discounted by the probability of a later arrival, $\mathbb{P}(1 \leq \tau_i < w)$.

We can improve the performance of the *a priori* policy at execution by skipping planned orders that aren't ready on time from a route dispatched at wave w; the triangle inequality guarantees that the solution cannot become more costly by skipping. We define by $\mathscr{Q}_w(\mathbf{x}^\mathbf{w})$ the expected duration of a dispatch at wave w given by $\mathbf{x}^w$; [22] provides a closed form to compute this expected cost for a given route in $\mathscr{O}(n^2)$ time.

The expected cost of such a policy for a given *a priori* feasible solution is equal to

$$\beta^0 - \sum_{i\in N}\sum_{w=a_i}^{b_i} \beta_i \mathbb{P}(\tau_i \geq w) y_i^w + \sum_{w\in\mathscr{W}} \mathscr{Q}_w(\mathbf{x}^{\mathbf{w}}). \tag{4}$$

We could also design an *a priori* solution that considers the order-skipping recourse proactively when planning a solution with an extension of the L-shaped method described for the Probabilistic Traveling Salesman Problem (PTSP) in [24]. We implemented this approach for our problem and were only able to solve small instances to optimality ($n = 25$, $W = 3$). Moreover, the benefits in cost savings over the value of (3) were small (under 2%). The intuition is the following: If the arrival probabilities are high, the probabilistic routing cost in the objective collapses to a deterministic routing cost, just as the PTSP. However, this is a prize-collecting problem, which is fundamentally different from the PTSP, and if the arrival probabilities are small, the probabilistic routing cost reduces its comparative importance with respect to the penalty cost in the objective and it becomes more important to make good dispatch selections and less important to route. Nonetheless, it would be interesting for future research efforts to design efficient heuristic procedures over the a-priori cost with order-skipping recourse starting from an optimal solution of (3) to gain these marginal improvements over the *a priori* solution; see *e.g.*, [12, 31].

4.1 Practical considerations when solving the *a priori* model

To solve larger instances of the *a priori* model effectively, we propose a solution improvement local search heuristic (LS) that exploits problem structure and complements a MIP solver. Specifically, we use this heuristic in two phases when solving problem (3). First, we run the heuristic for any new feasible solution s identified during the branch-and-bound tree search, and update the incumbent if the local search produces a solution with lower cost. Second, we use the heuristic during a solution construction phase as described in Algorithm 8 to generate a good initial feasible solution for the MIP solver.

Let $\{r_w^s, w \in \mathscr{W}_s\}$ be the set of routes of a feasible solution s to (3) indexed over the wave subset $\mathscr{W}_s \subseteq \mathscr{W}$ where these dispatches occur; we refer to $\mathscr{W}_s$ as the dispatch profile of s. Each route r_w^s represents an elementary sequence of order visits starting and ending at the depot. The *a priori* cost c_s is defined by (3a), its first dispatch wave by $ini_s := \max\{w \in \mathscr{W}_s\}$, the duration of each route by d_w^s, and its unserved customer set by $N_s := N \setminus \bigcup_{w\in\mathscr{W}_s} r_w^s$. An example of a feasible solution is given in Table 1.

Table 1: Example of a feasible solution s for an instance with 25 probabilistic orders

$\mathscr{W}_s$	route (r^s_w)	duration (d^s_w)
1	$\{0,22,16,17,2,9,18,21,0\}$	1
3	$\{0,15,3,12,6,8,7,14,20,4,24,25,19,5,10,23,0\}$	2
4	$\{0,1,0\}$	1
N_s	$\{11,13\}$	

Our LS procedure uses three separate neighborhood searches given solution s: (1) intra-route local search (IntraLS), *i.e.*, single route node selection and re-sequencing; (2) inter-route local search (InterLS), *i.e.*, node exchanges between routes and re-sequencing; and (3) dispatch profile local search. Pseudocode for LS is given by Algorithm 1. We execute the three-level search until no improving solution is found; each separate search procedure is described below.

Algorithm 1 Local Search Procedure

```
1: procedure LOCALSEARCH(Solution $s$)
2:     loop
3:         if ($\neg$INTRALS($s$) and $\neg$INTERLS($s$) and $\neg$DPLS($s$)) then return $s$.
```

IntraLS, defined in Algorithm 2, exploits the relation between the DDWP and a prize-collecting TSP

$$PCTSP(m,Q,\beta) := \min_{\substack{S\subseteq Q:\\ t(S)\leq m\ell}} \left\{ \alpha t(S) - \sum_{i\in S} \beta_i \right\}, \tag{5}$$

with a set $Q \subseteq N$ of potential customer orders, prizes $\beta_i, i \in Q$, and a maximum route duration of m waves. IntraLS is a best move procedure, where a move is described by re-optimizing one route from s leaving all remaining routes unaltered. The procedure chooses a single route r^s_w from s, and solves a PCTSP over a set of nodes $Q := N_s \cup r^s_w$ defined by all orders left unattended if route r^s_w where removed, a maximum route duration $m = d^s_w$ predefined by the waves left available, and prizes β_i discounted by $\mathbb{P}(\tau_i \geq w)$, for $i \in Q$. Any solution s processed by IntraLS contains only routes w that are optimally sequenced and that cannot be improved by selecting a different subset of orders to service from $N_s \cup r^s_w$.

InterLS uses best move searches over pairs of routes using neighborhoods inspired by those in [33] for the CVRP: two-edge exchanges between routes, removal and reinsertion of a k-order sequence from one route to another, and order swaps between routes. To implement these ideas, we account for two differences between the CVRP and the DDWP. First, we model the prize-collecting component; a move changes penalty

Algorithm 2 Intra-route LS procedure

```
1: procedure IntraLS(Solution $s$)
2:     $improved \leftarrow false$ and $s^* \leftarrow s$
3:     repeat
4:         for $w \in W_s$ do
5:             Let $s'$ be a copy of $s$ without route $r^s_w$
6:             Solve PCTSP($d^s_w, N_{s'}, \{\beta_i \mathbb{P}(\tau_i \geq w)\}$) and add optimal route found to $s'$ at wave $w$
7:             if ($c_{s'} < c_{s^*}$) then $s^* \leftarrow s'$ and $improved \leftarrow true$
8:         if ($c_{s^*} < c_s$) then $s \leftarrow s^*$
9:     until $\neg improved$
```

savings (due to the different dispatch time). Second, we check the durations of the new routes to ensure that they remain compatible with the fixed dispatch times of the unchanged routes.

The third neighborhood search that we implement is a Dispatch Profile Local Search (DPLS), described in Algorithm 3. The DPLS search perturbs the structure of the dispatch profile W_s for solution s using five operators: Cut, Merge, Exchange, Start, and Reorder. The first four operators find potential new solutions by solving a PCTSP, while the fifth uses a job scheduling approach.

Algorithm 3 Dispatch Profile Local Search (DPLS)

```
1: procedure DPLS(Solution $s$)
2:     $improved \leftarrow true$
3:     repeat
4:         if ($\neg$Cut($s$) and $\neg$Merge($s$) and $\neg$Exchange($s$) and $\neg$Start($s$) and $\neg$Reorder($s$)) then
5:             $improved \leftarrow false$.
6:     until $\neg improved$
```

The Cut operator, described in Algorithm 4, searches over all possible dispatch profiles that result when splitting a single dispatch w with duration $d^s_w \geq 2$ into two dispatches with shorter duration; this operator always add an extra return trip to the depot, as depicted in Figure 1. The Merge operator, described in Algorithm 5, works in reverse and searches over all dispatch profiles that arise when merging two consecutive dispatches into a single longer duration dispatch, as shown in Figure 2. The Exchange operator, described in Algorithm 6, changes the dispatch durations of two consecutive dispatches (thus changing the dispatch wave w of the latter); see Figure 3. The Start operator, described in Algorithm 7, searches for a better solution among the (at most) two new dispatch profiles induced by moving the initial dispatch wave of solution s backward or forward one wave, when feasible without altering subsequent dispatches; when the move is

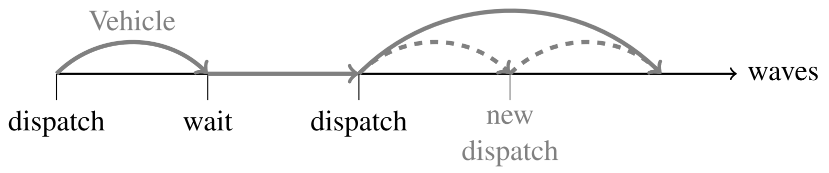


Figure 1: Example of a cut operation where a new dispatch profile is created (dashed flow) from an existing one (continuous flow) by cutting a route into two.

backward, the initial dispatch is extended by one wave and when the move is forward, it is reduced by one wave, as depicted in Figure 4.

The Reorder operator reassigns the routes in s to the best possible dispatch waves, without altering the customer visit sequences or the route durations. This assignment problem is equivalent to a single machine job scheduling problem of type $1||\sum_j f_j(w_j)$ (see *e.g.*, [28]): Consider a set of jobs, where each job j corresponds to a route with processing time p_j equal to the route duration (measured in waves). The cost of assigning job j to start wave w is $f_j(w) := \sum_{i \in r^j} \mathbb{P}(\tau_i < w)$. This job scheduling problem is *NP*-hard, since the single-machine scheduling problem minimizing the weighted sum of tardy jobs can be reduced to this problem. Small instances with fewer than 10 routes can be solved effectively with dynamic programming.

Algorithm 4 Cut operation over solution s

```
 1: procedure CUT(Solution $s$)
 2:     $improved \leftarrow false$ and $s^* \leftarrow s$
 3:     for $w \in W_s$ do
 4:         for $v : (w-1) \rightarrow (w - d^s_w + 1)$ do
 5:             Let $s'$ a copy of $s$ without route $r^s_w$
 6:             Solve PCTSP($w - v, N_{s'}, \{\beta_i \mathbb{P}(\tau_i \geq w)\}$) and add optimal route to $s'$ at wave $w$.
 7:             Solve PCTSP($v - w + d^s_w, N_{s'}, \{\beta_i \mathbb{P}(\tau_i \geq v)\}$) and add optimal route to $s'$ at wave $v$
 8:             if ($c_{s'} < c_{s^*}$) then $s^* \leftarrow s'$ and $improved \leftarrow true$
 9:     if ($c_s > c_{s^*}$) then $s \leftarrow s^*$
10:     return $improved$
```

We can in addition enhance the search by calling LS recursively, *i.e.*, we run LS within the Cut, Merge, Exchange, and Start operators on the temporary solution s', after the move gets implemented but before comparing to the best solution available s^*. In our experiments, we implemented a two-level local search to keep solution times manageable.

In addition to using the LS procedure during the branch-and-bound tree search, we also embed it in a

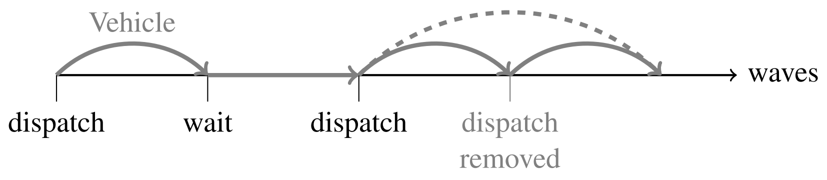


Figure 2: Example of a merge operation where a new dispatch profile is created (dashed flow) from an existing one (continuous flow) by merging two dispatches into one.

Algorithm 5 Merge operation over solution s

```
1: procedure MERGE(Solution $s$)
2:     $improved \leftarrow false$ and $s^* \leftarrow s$
3:     for $w \in W_s$ such that $w - d^s_w > 0$ do
4:         Let $s'$ a copy of $s$ without routes $r^s_w$ and $r^s_{w-d^s_w}$
5:         Solve PCTSP($d^s_w + d^s_{w-d^s_w}, N_{s'}, \{\beta_i \mathbb{P}(\tau_i \geq w)\}$) and add optimal route to $s'$ at wave $w$
6:         if ($c_{s'} < c_{s^*}$) then $s^* \leftarrow s'$ and $improved \leftarrow true$
7:     if ($c_s > c_{s^*}$) then $s \leftarrow s^*$
8:     return $improved$
```

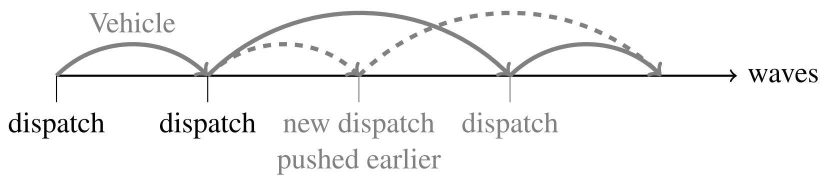


Figure 3: Example of an exchange operation where a dispatch gives one wave to its successor (dashed flow).

Algorithm 6 Exchange operation over solution s

```
1: procedure EXCHANGE(Solution $s$)
2:     $improved \leftarrow false$ and $s^* \leftarrow s$
3:     for pair $w_1, w_2 \in W_s$ such that $d^s_{w_1} > 1$ and ($w_1 = w_2 - d^s_{w_2}$ or $w_1 = w_2 + d^s_{w_1}$) do
4:         if ($w_1 > w_2$) then Let $w'_1 = w_1$, $w'_2 = w_2 + 1$, $d'_1 = d^s_{w_1} - 1$, and $d'_2 = d^s_{w_2} + 1$.
5:         else Let $w'_1 = w_2$, $w'_2 = w_1 - 1$, $d'_1 = d^s_{w_2} + 1$, and $d'_2 = d^s_{w_1} - 1$
6:         Let $s'$ a copy of solution $s$ without routes $r^s_{w_1}$ and $r^s_{w_2}$
7:         Solve PCTSP($d'_1, N_{s'}, \{\beta_i \mathbb{P}(\tau_i \geq w'_1)\}$) and add optimal route to $s'$ at wave $w'_1$
8:         Solve PCTSP($d'_2, N_{s'}, \{\beta_i \mathbb{P}(\tau_i \geq w'_2)\}$) and add optimal route to $s'$ at wave $w'_2$
9:         if ($c_{s'} < c_{s^*}$) then $s^* \leftarrow s'$ and $improved \leftarrow true$
10:    if ($c_s > c_{s^*}$) then $s \leftarrow s^*$
11:    return $improved$
```

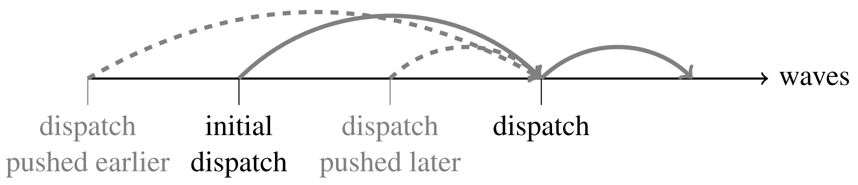


Figure 4: Example of a Start operation where the first dispatch is enlarged/reduced (dashed flow).

Algorithm 7 Start operation over solution s

```
1:  procedure START(Solution $s$)
2:      $improved \leftarrow false$ and $s^* \leftarrow s$
3:      Let $s_1$ and $s_2$ be two copies of solution $s$ without route $r^s_{ini_s}$
4:      if ($ini_s < W$) then
5:          Solve PCTSP($d^s_{ini_s}+1, N_{s_1}, \{\beta_i \mathbb{P}(\tau_i \geq ini_s+1)\}$) and add optimal route to $s_1$ at wave $ini_s+1$
6:          if ($c_{s_1} < c_{s^*}$) then $s^* \leftarrow s_1$ and $improved \leftarrow true$
7:      if ($d^s_{ini_s} > 1$) then
8:          Solve PCTSP($d^s_{ini_s}-1, N_{s_2}, \{\beta_i \mathbb{P}(\tau_i \geq ini_s-1)\}$) and add optimal route to $s_2$ at wave $ini_s-1$
9:          if ($c_{s_2} < c_{s^*}$) then $s^* \leftarrow s_2$ and $improved \leftarrow true$
10:     if ($c_s > c_{s^*}$) then $s \leftarrow s^*$
11:     return $improved$
```

heuristic for building a good initial feasible solution s_0 for the DDWP. This constructive heuristic is given a set of m dispatch profiles, and for each $W_k, k = 1, \ldots, m$ solves a series of sequential prize-collecting TSPs over time to build a solution which is then improved by the LS procedure. In our experiments, the set of dispatch profiles provided to the construction heuristic was $\{\{1,2,\ldots,q\}\}$ for all q in $1 \leq q \leq W$, *i.e.*, all possible profiles that include consecutive, single-wave dispatches up to the final wave.

5 Dynamic Policies

A priori policies, particularly when adjusted via recourse actions, may yield reasonable solutions to many problems. However, [23] provides an instance sequence for which an optimal *a priori* policy with recourse is arbitrarily worse than an optimal dynamic policy, *i.e.*, $C^*_{AP}(\hat{R})/C^*(\hat{R}) \to \infty$. Therefore, we next develop dynamic policies.

Algorithm 8 Constructive Heuristic

```
1: procedure CONSTRUCTIVE(Set of dispatch profiles $\{\mathscr{W}_1, \mathscr{W}_2, \ldots, \mathscr{W}_{s_m}\}$)
2:     Initialize $s^0$ as the empty solution
3:     for $k : 1 \to m$ do
4:         Build $\{d_w, w \in \mathscr{W}_k\}$, the dispatch durations of $\mathscr{W}_k$
5:         Let $s_k$ be a copy of $s_0$.
6:         for $w \in \mathscr{W}_k$ in decreasing order do
7:             Update $N_{s_k}$
8:             Solve a PC-TSP over $N_{s_k}$, with max duration $d_w$ and prizes $\beta_i \mathbb{P}(\tau_i \geq w)$
9:             Add optimal route to $s_k$.
10:        LOCALSEARCH($s_k$)
11:        if ($c_{s_k} < c_{s_0}$) then $s_0 \leftarrow s_k$
12:    return $s_0$
```

5.1 *A Priori*-Based Rollout Policy

A natural, but possibly computationally-expensive dynamic policy is to roll out the *a priori* policy. At each wave $w \in \mathscr{W}$ when the vehicle is available at the depot, we recompute an optimal *a priori* policy beginning at wave w using the current system state (w, R, P) to define a new, reduced problem over the remaining operating period. If the policy decides at wave w to dispatch a route serving customer set $S \subset R$, then this decision is implemented and a new *a priori* policy is computed again at wave $q_w(S)$; otherwise, a new *a priori* policy is computed at $w-1$ after waiting for one wave.

Computing such a rollout policy requires the solution of $\mathscr{O}(W)$ *a priori* problems and may be difficult to solve exactly for larger instances. However, there are multiple ways to improve the computational performance. First, one can warmstart the solution of the *a priori* problem at wave w using the most recently computed solution (from wave $q > w$), adjusted to skip all planned orders that are not ready yet and then improved via the LS procedure. It is also not necessary within this heuristic procedure to solve each *a priori* problem to optimality. A more substantial simplification is to begin the rollout process only at the first wave where a dispatch is planned in the initial *a priori* solution (computed at wave W); we call this approach the *restricted* rollout policy.

5.2 Greedy *a priori*-based prize-collecting TSP Heuristic

We also test simpler rollout strategies that do not rely on repeatedly solving the *a priori* problem. One such approach takes advantage of the relationship between the DDWP and the PC-TSP. To initiate the approach, we solve the *a priori* problem at initial wave W. We then use this solution to guide a rollout procedure that only solves deterministic PC-TSP problems as follows. At each wave $w < W$ for which the *a priori* solution dictates a vehicle dispatch, we determine the maximum duration of the route (by ensuring the vehicle is back at the depot for the next *a priori* dispatch wave). Then, we determine a vehicle route by solving a PC-TSP using only open orders that are not postponed in the *a priori* solution for dispatch at a later wave. Let $\mathscr{W}^{AP}$ be the set of waves where vehicle dispatches take place in the *a priori* solution, and let $Q_w \subset N_w$ be its set of planned orders to be dispatched at each $w \in \mathscr{W}^{AP}$. We implement the heuristic dynamic policy outlined in Algorithm 9.

Algorithm 9 Greedy *a priori*-based policy

```
1: Set $w \leftarrow \max\{v \in \mathscr{W}^{AP}\}$, $w^+ \leftarrow \max\{v \in \mathscr{W}^{AP} : v < w\}$
2: Wait at the depot until wave $w$
3: while $w > 0$ do
4:     Read system state $(w, R, P)$
5:     Compute $\bar{R} := R \setminus \bigcup_{v=1}^{w^+} Q_v$, the set of open orders not included in future dispatches of the a priori
   solution
6:     Solve PCTSP($w - w^+, \bar{R}, \beta$) and let $S \subset \bar{R}$ be the optimal set of sequenced orders selected
7:     if $q_w(S) = w^+$, then dispatch a vehicle to $S$, set $w \leftarrow w^+$, $w^+ \leftarrow \max\{v \in \mathscr{W}^{AP} : v < w\}$,
8:     else set $w \leftarrow w - 1$.
```

The greedy policy ignores information about orders dynamically realized throughout the operation, and only considers the available probabilistic information at the start of the horizon. However, it is dynamic enough to accommodate newly arriving orders, and to re-optimize routing decisions. This last feature makes it a better candidate than a simple *a priori* policy with order-skipping recourse.

6 Computational Experiments

We now present a set of computational experiments designed over a family of randomly generated instances with the objective of testing the quality of our heuristic policies and to get qualitative insights regarding the management of vehicle dispatches in a same-day delivery context. Table 2 summarizes the heuristic policies computed for each instance. All heuristics were programmed in Java and computed using one thread of a Xeon E5620 processor with up to 12Gb RAM, using CPLEX 12.6 when necessary as a MIP solver.

Table 2: Heuristic policies computed in our experiments

symbol	strategy	section
AP	*a priori* policy + order-skipping	4
GP	Greedy PCTSP-based policy	5.2
RP	Rollout of *a priori* policy	5.1
RRP	Restricted rollout of *a priori* policy	5.1

6.1 Design of data sets

We generated 240 data sets to evaluate our policies over different performance indicators. Each data set has a specific geography of 50 orders, a known subset $\hat{R} \subseteq \{1,\dots,50\}$ of orders ready at the start of the operating period, and a vector of ready wave probabilities for orders with unknown arrival wave.

The geography for each data set is defined by a random seed $g \in \{0,\dots,4\}$ used to assign 50 different locations over a 51×51 square subset of $\mathbb{R}^2$ following a discrete uniform distribution $U(0,50)$ for each component of the location's coordinate and with the depot located at coordinate $(25,25)$. We ruled out repeated coordinates to have a more interesting geography. Travel times between locations are given by the ℓ_1-norm (Manhattan distance) between two locations, chosen to model urban travel times. All data sets share a common horizon with $W = 6$ possible dispatch waves, each with duration $\ell = 100$ time units. The duration of a round-trip visiting any single order is less than or equal to 100 time units, and thus can be completed in a single dispatch wave.

For each order $i \in \{1,..,50\}$, its ready wave τ_i is a discrete random variable, independently distributed with probability $\mathbb{P}(\tau_i = W) = p_{start}$ of being ready at the start of the operating period (the order's degree of dynamism); a conditional probability $\mathbb{P}(\tau_i = -1 | \tau_i < W) = p_{out}$ of not arriving at all during the operation period; and a conditional discrete uniform distribution with probability $\mathbb{P}(\tau_i = w | 1 \leq \tau_i < W) = (\min(W-1, \mu_i + \sigma) - \max(1, \mu_i - \sigma) + 1)^{-1}$ of arriving during the operation at any wave $w \in \{\max(1, \mu_i - \sigma), \dots, \min(W-1, \mu_i + \sigma)\}$. The parameter μ_i represents the mean ready wave and is drawn for each i with equal probability between 1 and $W - 1$, and σ represents variability. Each data set uses a triple $(\sigma, p_{start}, p_{out}) : \sigma \in \{\text{Lo} = 1, \text{Hi} = 6\}$, $p_{start} \in \{10\%, 15\%, 25\%, 50\%\}$, $p_{out} \in \{20\%, 40\%\}$, and a setting of the seed $h \in \{0,1,2\}$ to draw the set $\hat{R}$ of orders ready at start.

We created $M = 50$ realizations of the ready time vector τ for each data set using the probabilistic model above. These scenarios are used as a common sample to estimate lower bounds based on a perfect information relaxation and the expected cost of all policies. Each data set has a unique value of $(g, \sigma, p_{start}, p_{out}, h)$, and all sets, including their simulated realizations, are published online at `sites.google.com/site/maklappor/ddwp-data-sets`.

6.2 Set 1: Base experiments

For our first set of experiments, we build 3 instances by considering the first 25, 35 and 50 orders for each one of the 240 data sets. This makes a set of 720 instances with 3 different problem sizes. The penalty for leaving order i unattended if it appears is set as the duration of a round-trip to order i from the depot, $\beta_i := 2t_{\{0,i\}}$; this setting models a system that serves all realized orders by outsourcing the service of each order that remains open at the end of the day to a direct delivery service. Under these penalties, there is always a potential service profitability; in particular, there is an economic incentive to dispatch the vehicle in the last wave if any order is open.

We computed the following metrics for each policy over each realization from each instance:

- Total cost ($cost$), travel time ($duration$) and penalties paid ($penalty$),
- gap: the percentage increase of the policy's cost over the perfect information bound,
- fill rate (fr): the percentage of orders served by the vehicle over all realized orders,
- $duration/order$: the routes' duration over the number of served orders,
- $nRoutes$: number of vehicle dispatches,
- $nWaves$: average dispatch length in waves used by each route,
- $iWait$ and $pWait$: number of waves spent waiting at the depot before/after the initial dispatch,
- $time_{off}$: average "off-line" solution time, *i.e.*, before the solution is implemented,
- $totaltime_{on}$: total "on-line" solution time over the operating period,

- $time_{on}$: total "on-line" solution time divided by the number of active decision epochs, *i.e.*, the number of waves in which the policy makes a dispatch decision (which excludes all predetermined waits established by the initial *a priori* policy and dispatch waves jumped by the vehicle routes).

These metrics are averaged for each instance over all $M = 50$ realizations. Table 3 presents average results for each heuristic policy over all instances.

Table 3: Average results of heuristic policies

metric policy	**AP**	**GrAP**	**RRP**	**RP**
cost (decrease from AP %)	555	523 (5.8%)	505 (9.1%)	505 (9.1%)
duration	298	305 (-2.5%)	305 (-2.4%)	311 (-4.4%)
penalty	258	218 (15.2%)	200 (22.3%)	194 (24.7%)
gap	23.1%	16.1%	12.1%	12.1%
fr	81.6%	85.0%	86.2%	86.6%
$duration/order$	11.0	11.2	11.2	11.4
nRoutes	2.5	2.5	2.6	2.7
nWaves (std)	1.4	1.4	1.4	1.4
iWait	2.6	2.6	2.6	2.5
pWait	0.003	0.005	0.002	0.003
$time_{off}$	836.2s.			
$totaltime_{on}$	0.0001s.	0.08s.	252.0s.	607.5s.
$time_{on}$	0.0002s.	0.18s.	85.8s.	120.0s.

On average, the AP policy's cost is 23.1% over the perfect information bound and, as expected based on each heuristic's recourse possibility, rolling it out improves upon AP and cuts the average gap by 47.7%, with an average cost reduction of 9.1%. The dynamic heuristic GP achieves 63% of that cost reduction, suggesting that a simple but dynamic policy can capture a significant amount of the benefits of dynamism. The RRP policy achieves similar average cost reduction and gap as RP, indicating that the AP policy is adequately choosing an initial dispatch. We find this to be a useful insight for managers: Assuming average behavior in the future appears to be sufficient when making an initial dispatch decision. Conversely, once the AP policy recommends an initial dispatch, RP and RRP perform better by incorporating newly arrived information.

Also, RP increases the order fill rate over AP by 6.2%, by redesigning the solution at each decision moment, which is crucial for logistics service providers interested in providing a better customer service; GP and RRP respectively achieve 68.1% and 90.6% of this fill rate increase. The dynamic policies' benefits mostly stem from reducing penalty costs, while on average they produce a slight increase in the routes' duration. This is a tradeoff wherein dynamic policies significantly improve order service, while incurring small increases in duration per order. We also see that dynamic policies slightly increase the average number of dispatches and reduce the initial waiting time. In terms of waiting at the depot after the first dispatch (*pWave*), our results suggest it does not occur often, and it may be better to keep the vehicle busy serving more orders.

All policies share the $time_{off}$ value, but radically differ in on-line solution time: AP and GP are almost instantaneous online policies, while each rollout policy requires more computational power. RRP's total online time is significantly less than RP, even though the former closely approximates the latter in terms of cost and the other metrics.

In Figure 5, we observe the distribution of the gap over all instances for each heuristic policy. We observe that RP not only outperforms AP on average, but is also less variable in solution quality, with 3.7% deviation versus 7.4%; RRP has a similar distribution to RP's, while GP has an intermediate 4.9% deviation.

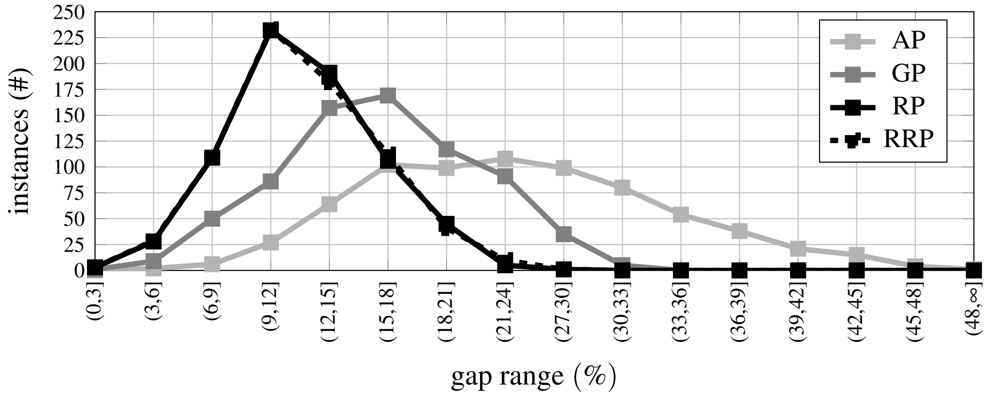


Figure 5: Distribution of gap over all instances

In Figure 6 we compare the average gap of our heuristic policies between instances sharing parameters of size n, degree of dynamism p_{start}, probability of not showing up p_{out}, and arrival variability σ. In the first graph on the left we see how the average gap increases with the number of potential orders n; this increase may be related to an increase in the problem's size and complexity, but also to the lower bound's tightness. Moreover we see that RPs gap difference over AP increases as n grows, with GP in between and RRP indistinguishable from RP. We also observe a significant gap reduction for the *a priori* policy (AP) as p_{start} increases. As expected, the more information available at the initial wave, the closer we can get to a deterministic problem and the better we can optimize exactly. For dynamic policies the gap is significantly smaller and tends to be stable over different values of p_{start}, showing the benefit and importance of complex recourse actions when dealing with higher degrees of dynamism. Regarding the orders' conditional probability of not showing up and the variability of the ready wave, Figure 6 suggests that it may be harder to optimize instances with higher value of p_{out} and σ due to an increase in the problem's uncertainty and/or possibly due to a deterioration of our lower bound. Again, we see that the average gap reduction of dynamic policies over the *a priori* policy is particularly valuable for instances with higher ready time variability.

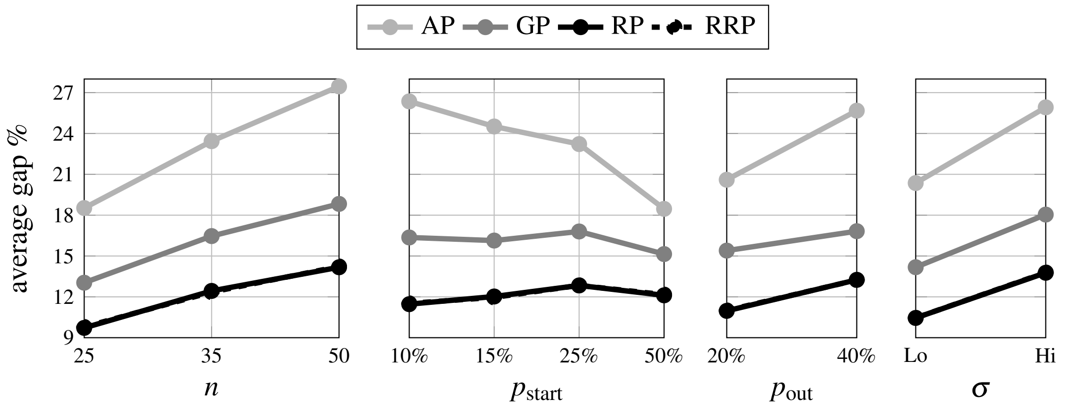


Figure 6: Average gap versus n, p_{start}, p_{out}, and σ

Figure 7 presents the average fill rate as a function of the instance parameters; for reference, maximizing fill rate is the objective function of the model presented in [36]. We see that fr decreases as n increases over all policies, which may indicate congestion related to available vehicle time. Interestingly, this fill rate reduction is marginally decreasing with n, presumably through order consolidation; *i.e.*, at $N = 25$, an increase of 10 customers results in a larger fill rate decrease than an increase of 15 customers at $N = 35$. Also, RP's fill rate is consistently over AP by more than 4.6% regardless of n. The second graph from the left shows how the average fill rate increases by a 16.1% amount for AP and 10.6% for RP when p_{start} increases from 10% to 50%, meaning that more information available at the start can significantly increase the number of orders served, especially for the AP policy. The relative difference between AP and RP is higher for instances with higher dynamism showing again the additional value of dynamic policies. The third graph from the left shows how RP (and dynamic policies more generally) can be useful when opportunities to increase fill rate are presented. While the *a priori* policy's rate remains stable over the probability of not showing up (p_{out}), all dynamic policies increase order fill rate, *e.g.*, RP's increases by 1.6%. This may be explained by the fact that RP uses the time gained when initially planned orders do not realize to serve unexpected and initially unplanned orders that do show up. This effect is also observed when the average fill rate is compared versus σ. The fill rate of AP decreases with σ, but RP's fill rate is more stable and the difference between RP and AP increases as the variability parameter increases.

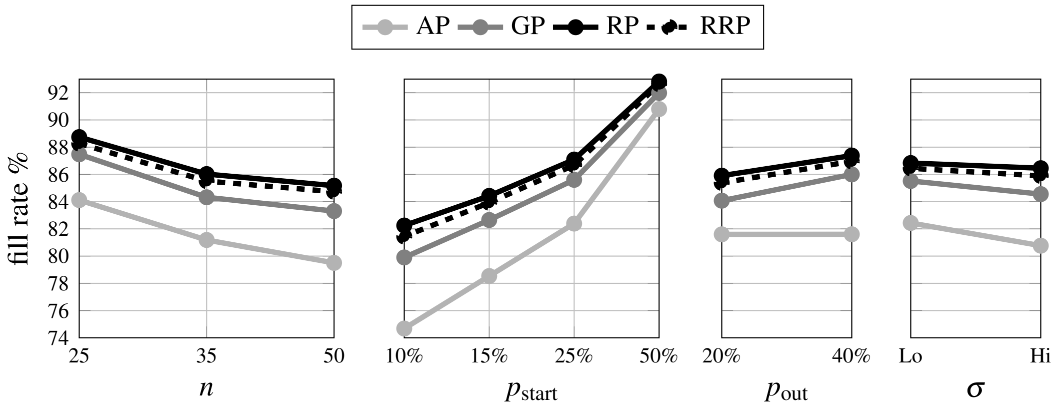


Figure 7: Average fill rate versus n, p_{start}, p_{out}, and σ

The graphs presented in Figures 8,9, and 10 show for each policy how the average number of dispatches, duration in waves per dispatch, and initial number of waves spent waiting at the depot evolve over the different instance parameters. From the first graphs on the left, we see that all policies dispatch more vehicles as n grows and wait a smaller amount of initial waves at the depot, meaning that a relatively dense geography justifies a higher number of dispatches during the day and more "active" dispatch waves during the operating period. We also see that the difference in routes dispatched between RP and AP increases with n, and that dynamic policies slightly reduce the waves used per dispatch as n grows. This shows empirically how dynamic policies increase recourse opportunities, *i.e.*, more returns to the depot, as n increases. The second set of graphs from left to right show how our policies have fewer and longer vehicle dispatches with shorter initial waiting periods as off-line information increases (higher p_{start}). This means that as deterministic information increases, fewer recourse opportunities are needed and routing efficiency becomes the focus.

The shorter initial wait periods can be explained because there may be relatively less need to "wait and see" versus instances where less information is given at start. We also see how the RP policy comparatively increases the number of dispatches and reduces route duration (gaining recourse opportunities) as p_{start} gets smaller. Similar effects can also be observed from the graphs on the right. It is interesting to note that RP slightly increases the average number of dispatches and the route length over AP as p_{out} and *sigma* increase. Again, it shows how this policy can recover from uncertainty, *e.g.*, orders realizing later than expected or not showing up, by dynamically inserting unplanned orders as substitutes.

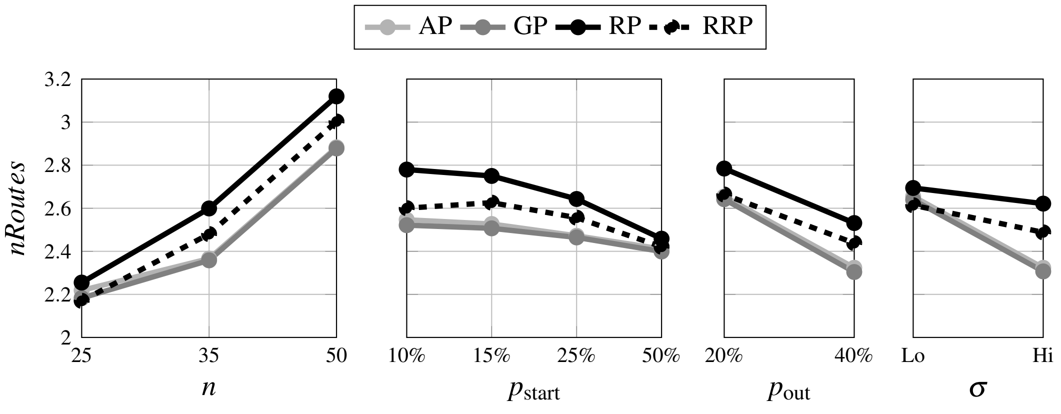


Figure 8: Average number of vehicles dispatched versus n, p_{start}, p_{out}, and σ

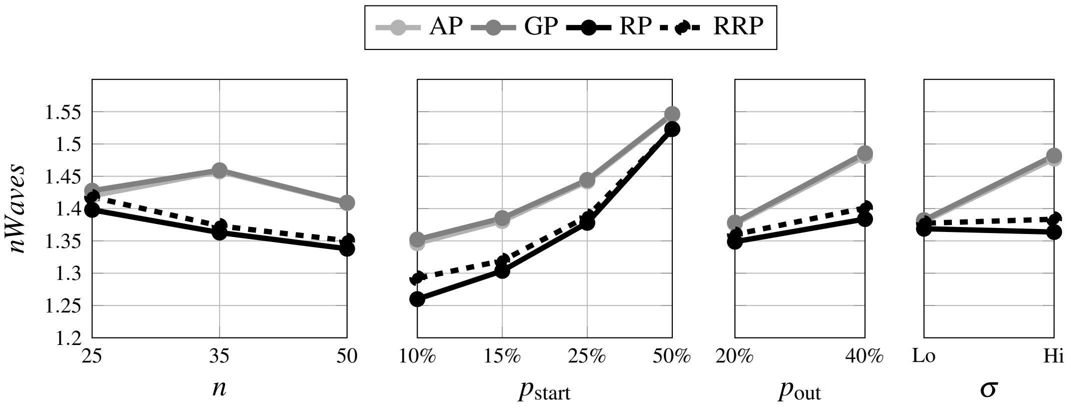


Figure 9: Average number of waves per vehicles dispatch versus n, p_{start}, p_{out}, and σ

The average solution times over all instances disaggregated by number of customers n and the probability p_{start} are presented Table 4. The results on the left table show the off-line solution times shared by all

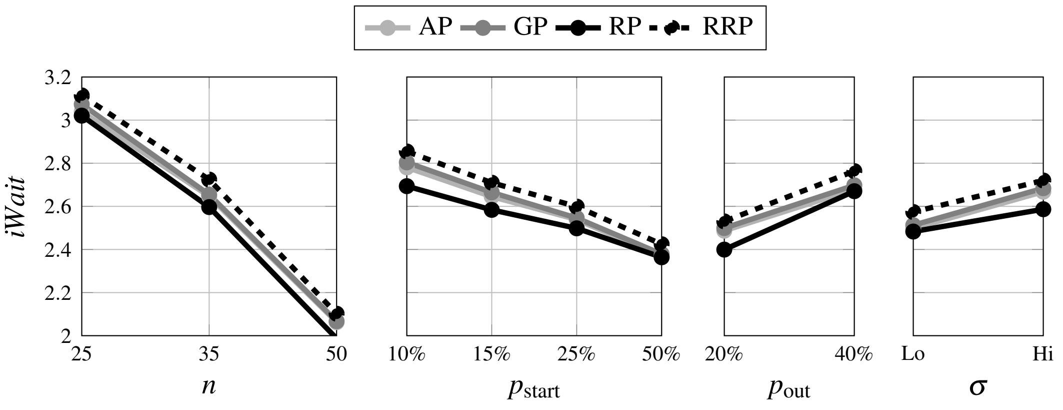


Figure 10: Average number waves waited before initial dispatch versus n, p_{start}, p_{out}, and σ

policies. As expected due to the nature of exact MIP models, this time increases exponentially with the number of orders, but should not be problematic since this procedure is intended to run before the start of the operating horizon. In addition, average times increase with p_{start}, probably because the initial model has a bigger feasible region. The right-hand table shows the average solution times per dispatch of GP, RRP and RP policies. We observer a similar increasing effect over the number of customers, which depending on available computing resources could be an issue for the full rollout policy if n is large. Our results suggest substituting RRP for RP, since both policies showed almost equivalent cost reductions over AP. For even larger n, the GP policy is a simpler alternative to consider if RRP becomes inefficient; GP still takes less than 1 second per decision and generates better solutions than AP. Another possibility would be to further exploit local search and meta-heuristic procedures; we leave this question for future work.

Table 4: Average $time_{on}$ and $time_{off}$ versus p_{start} and n

Average time offline in secs.

$p_{start}\backslash n$	25	35	50
10%	121	338	1689
15%	130	375	1870
25%	224	494	1886
50%	245	579	2084

Average time per decision in secs.

	GP			**RRP**			**RP**		
$p_{start}\backslash n$	25	35	50	25	35	50	25	35	50
0.10	0.06	0.10	0.16	3.4	16.5	147	5.5	44.0	273
0.15	0.05	0.12	0.14	4.1	16.1	186	6.0	41.4	284
0.25	0.04	0.05	0.07	4.3	32.8	256	7.1	57.8	324
0.50	0.03	0.03	0.06	4.7	51.9	305	7.8	62.8	326

6.3 Set 2: Second set of experiments

We now present a second set of computational experiments to study the tradeoff between two plausible objective functions in same-day delivery: minimizing total cost and maximizing (weighted) order coverage; in the DDWP setting this is equivalent to minimizing penalty costs. This coverage goal also matches the objective in the models in [36]. We look for basic tradeoffs, performance of our heuristic policies, and structural differences in our solutions to provide qualitative insights.

We build a new set of 720 instances by making 3 instances for each one of the 240 data sets (with 35 orders each). Each instance has a different value of $\alpha \in \{1, 2, 100\}$ for the penalty setting $\beta_i = 2\alpha d_{0i}$. While the first set ($\alpha = 1$) balances both vehicle traveling costs and penalty costs, the last one ($\alpha = 100$) hierarchically focuses on covering orders first, travel time minimization as a secondary objective; the second set ($\alpha = 2$) is an intermediate case.

Table 5: Average cost and reduction percentage over AP for each policy under different settings of α

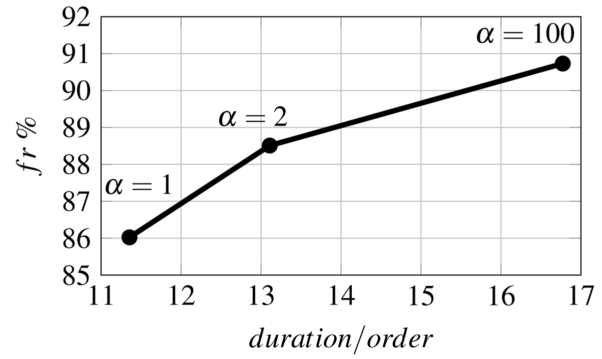
policy\objective	$\alpha = 1$	$\alpha = 2$	$\alpha = 100$
LB	441	546	9395
AP	541	760	18958
GP	511 (5.5%)	711(6.4%)	16402 (13.5%)
RRP	493 (8.8%)	670 (11.8%)	13625 (28.1%)
RP	494 (8.7%)	669 (11.9%)	13614 (28.2%)

Table 5 presents average costs of all heuristic policies and perfect information bound over all instances under different settings of α. It also shows for each dynamic policy the average cost reduction percentage over AP. We first observe that cost reduction percentages of dynamic policies substantially increase with α; this is explained by the order coverage improvement that dynamic policies enjoy, discussed in the first set of experiments. It may also occur because the cost savings from delivering an additional order become more significant as the weight on penalty costs is bigger. For example, the cost reduction from serving one more order when there are two left unattended is 50%. Second, we observe that RP becomes more attractive over GP as the relative importance of penalty costs increases, suggesting that sophisticated dynamic policies that get better fill rate improvements become more valuable as the focus shifts towards coverage.

The left table in Figure 11 presents results for RP averaged over all instances sharing the same settings of α. The first two rows present the tradeoff between coverage cost (*penalty*) and vehicle traveled time

metric	$\alpha = 1$	$\alpha = 2$	$\alpha = 100$
duration	305	352	448
penalty/α	189	159	132
duration/order	11.4	13.1	16.8
fr	86.0%	88.5%	90.7%
nRoutes	2.60	3.40	4.80
nWaves	1.36	1.22	1.13
iWait	2.60	1.97	0.58
pWait	0.003	0.013	0.047

(a) Metrics for RP under different settings of α



(b) Pareto chart for RP with *fr* versus *duration/order*

Figure 11: Average results for RP under different settings of α

(*duration*) over the three cases of α. The third and fourth rows present these results in comparable metrics: *duration/order* representing routing efficiency and *fr* representing order satisfaction. These results are plotted in the graph on the right as a Pareto chart. As expected, we observe that *fr* increases as order coverage becomes more relevant in the objective, but this fill rate improvement requires a sacrifice in *distance/order*, *i.e.*, the higher the order fill rates, the more inefficient the routes. Moreover, the marginal rate of substitution between *fr* and *distance/order* decreases with α. From $\alpha = 1$ to $\alpha = 2$ the average gain in *fr* per *distance/order* sacrificed is 1.4%. The same number from $\alpha = 2$ to $\alpha = 100$ is 0.61%, a 57% reduction. For decision makers, this suggests that even in the efficient frontier, the distance cost of an additional customer covered becomes increasingly more expensive; a cost-focused manager may be willing to sacrifice coverage for routing efficiency at a sufficiently high fill rate.

We further explain the decreasing marginal substitution rate between coverage and routing efficiency by looking at the last four rows of the table in Figure 11. RP has to create more recourse opportunities (more returns to the depot), increases by 85% the average number of dispatches and reduces the average dispatch length in waves by 17% to improve coverage; by the triangle inequality, this reduces routing efficiency. Moreover, it drastically reduces the initial wait time by 78%, implying a longer usage of the vehicle throughout the day. This is also an interesting insight for managers: When coverage is the objective, vehicles operate longer periods of time with higher maintenance costs and longer workdays for drivers (a higher cost in human resources). Conversely, order consolidation increases when total cost is the objective, routes

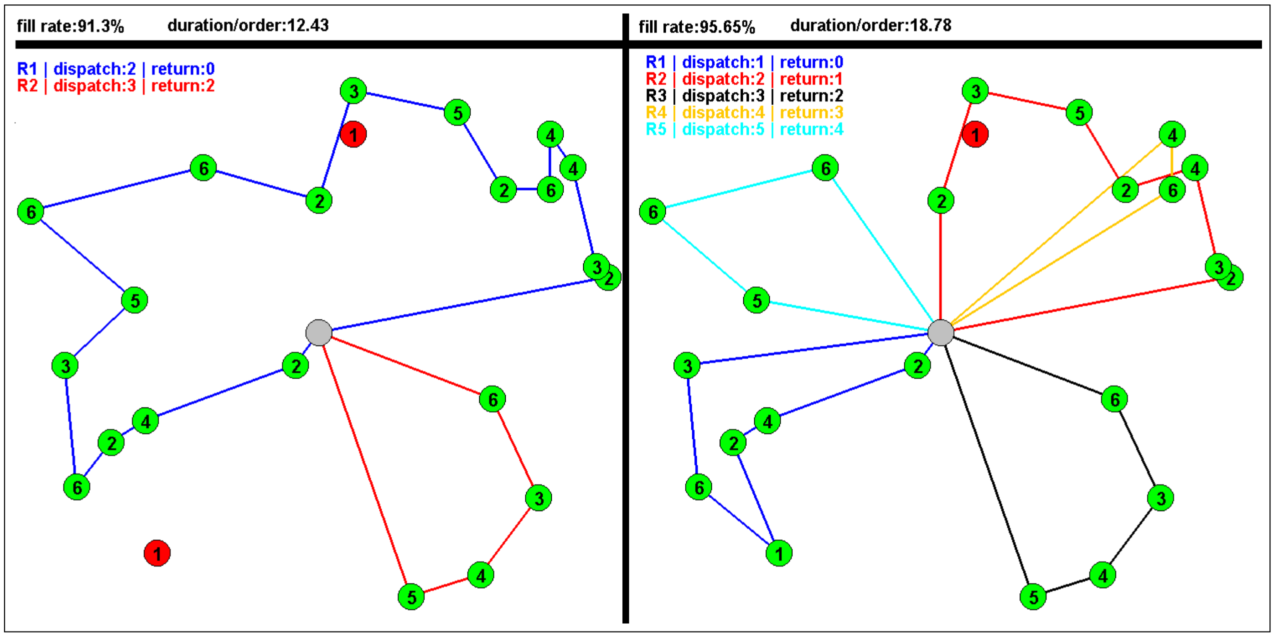

fill rate:91.3%
duration/order:12.43
R1 | dispatch:2 | return:0
R2 | dispatch:3 | return:2
fill rate:95.65%
duration/order:18.78
R1 | dispatch:1 | return:0
R2 | dispatch:2 | return:1
R3 | dispatch:3 | return:2
R4 | dispatch:4 | return:3
R5 | dispatch:5 | return:4

Figure 12: Example of two different solutions to the same instance realization. The left one minimizes total cost and the right one minimizes lost penalties first. Green orders are served and red ones are left unattended. Each order's ready wave is labeled at its node and route dispatch/return waves are shown in the upper-left corner.

become longer and efficient, and dispatches are pushed forward in time. This reduces the operation's length, but increases the time between an order's arrival and its dispatch; logistics providers may want to minimize this value to guarantee good service. In Figure 12 we show this tradeoff graphically, presenting two solutions for the same instance realization. The left solution minimizes total cost while the right one minimizes lost penalties. We clearly see how the right solution has to sacrifice roughly 50% in routing efficiency to increase its order coverage by one order. It also makes 3 more vehicle dispatches and actively uses the vehicle for 2 more waves.

We next analyze the performance of all heuristics under different values of α and the data set parameters p_{start}, p_{out} and σ. In Figure 13 we present the previously shown Pareto charts for all heuristic policies under different settings of p_{start}. We observe that RP can approximately cut the gap between the *a priori* heuristic and the perfect information bound in half. Approximately two thirds of that improvement can be achieved by GP, while RPP appears equivalent to RP; this is consistent with the first round of experiments. When p_{start} is small (10%) the value of dynamic policies is higher and the marginal rate of substitution between fr and $duration/order$ is smaller. When p_{start} in higher (50%) there is less additional value in implementing

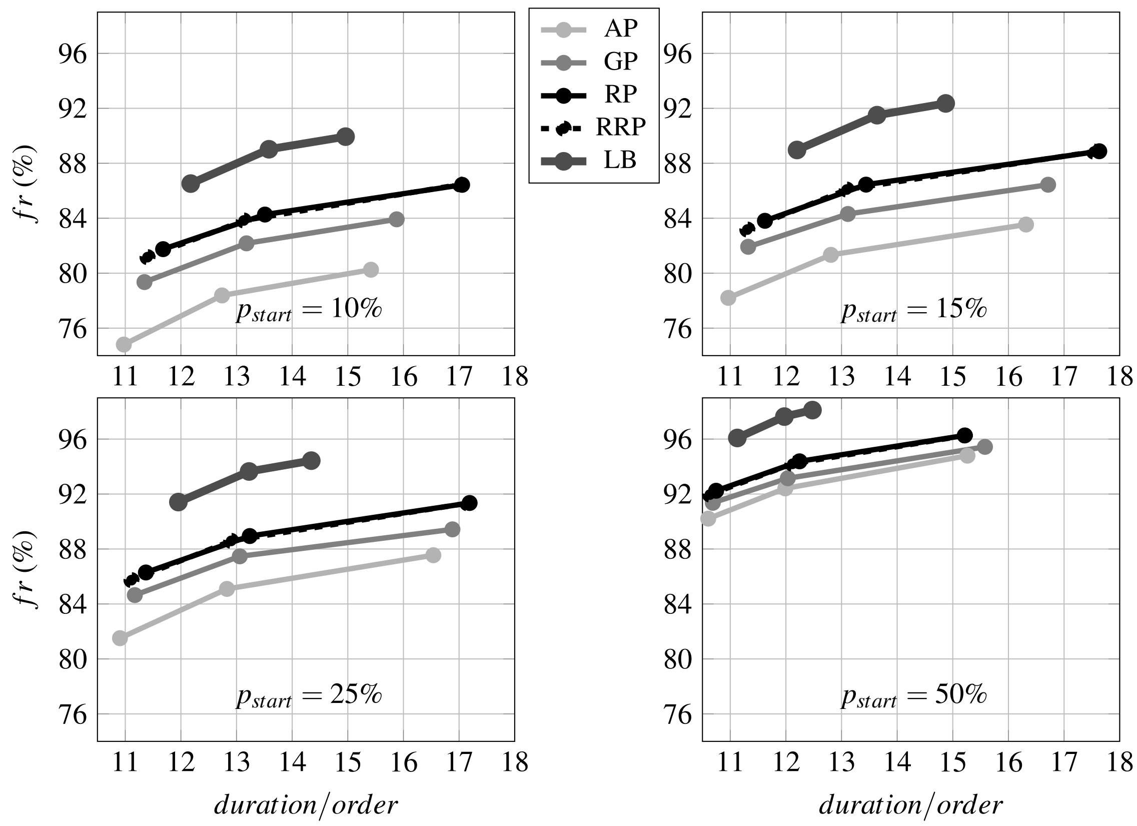


Figure 13: Pareto charts showing each policy's average fr and $duration/order$ for different settings of p_{start}

dynamic policies and the marginal rate of substitution becomes higher, implying that one has to sacrifice less routing efficiency for a marginal increase in order coverage. We also see that as p_{start} grows the system moves to the upper left, better for both fr and $duration/order$.

In Figure 14 we present the Pareto charts of all heuristics for each value of the conditional probability of a order not showing up, p_{out}. We observe that as p_{out} increases all heuristics move to the right, a loss in routing efficiency explained by the increased amount of uncertainty in the order arrival process that makes *a priori* plans less reliable; this is especially true for problems maximizing fill rate, making the marginal rate of substitution smaller. Dynamic policies provide more value as p_{out} increases and the curves move up in relation to AP, because they can better recover from unexpected order late arrivals and corresponding changes in the route plans.

In Figure 15 we analyze the Pareto charts of all heuristics under the two settings of order ready wave

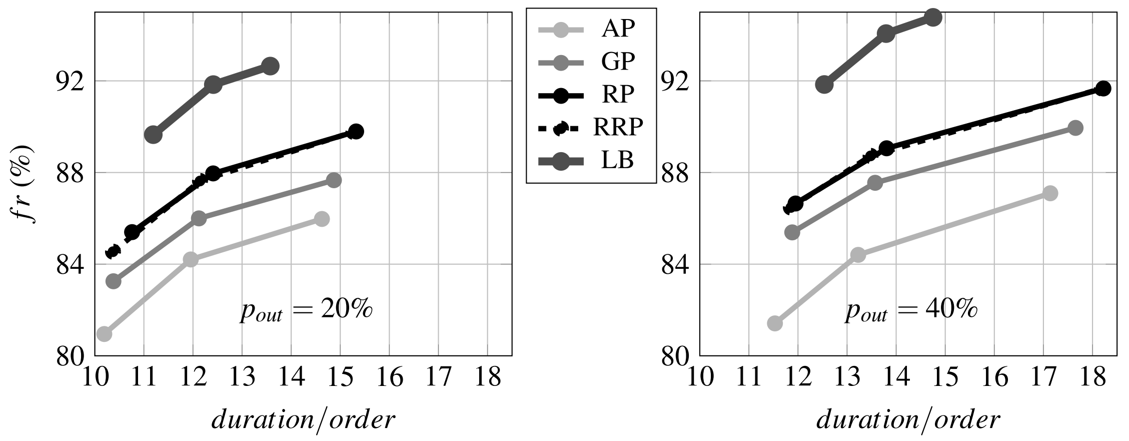


Figure 14: Pareto charts showing each policy's average fr and $duration/order$ for different settings of p_{out}

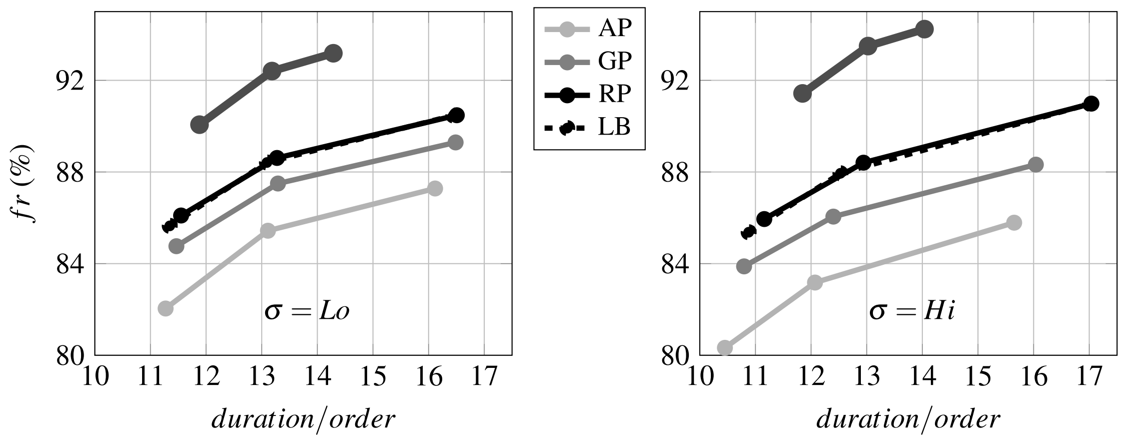


Figure 15: Pareto charts showing each policy's average fr and $duration/order$ for different settings of σ

variability (σ). We observe two effects as variability increases. First, all curves get wider, with a smaller marginal rate of substitution. This implies that as arrival variability increases it is more expensive to gain coverage and more sacrifices in efficiency have to be made. Also, we observe that AP and GP move down as σ increases, meaning that our static policy and simple dynamic policy lose order coverage. This is not the case of RP and RRP; again we see the importance of sophisticated dynamic policies as variability increases.

7 Conclusions

We have formulated the Dynamic Dispatch Waves Problem (DDWP) on general network topologies to investigate the fundamental tradeoffs in same-day delivery distribution systems. We have formulated an integer program to solve the deterministic version of the problem, and used this model to derive an optimal solution for the stochastic *a priori* problem by converting it into a deterministic equivalent.

From the *a priori* solution we derive four heuristic policies that differ in their dynamism, from an *a priori* policy with simple recourse to a fully dynamic rollout policy. The first policy, AP, is an *a priori* solution with a order-skipping recourse; the second policy is a direct rollout of the *a priori* solution, RP; the third, RRP, is a restricted version of RP that waits until the first dispatch wave given by AP to roll out; and the fourth one is a computationally cheaper alternative to RP that rolls out a prize-collecting TSP guided by the initial *a priori* solution.

We designed a first set of computational instances to test these policies under different settings of geography, problem size, level of information disclosed before the operation starts (degree of dynamism), and variability of the arrival process. Our computational experiments indicate that the performance of the *a priori* policy has costs that are 23.1% over our perfect information bound, even if we include heuristic order-skipping improvements. Its order fill rate is 81.6%. The benefit of a fully dynamic policy over an *a priori* one can be significant; in our experiments, RP is able to cut AP's cost by 9.1% on average, yielding an average gap of 12.1% over the lower bound. It also improves the order fill rate to 86.6%, which is highly desirable for SDD services. Part of this benefit is achieved by increasing recourse opportunities to catch newly arrived orders, and by sacrificing a small amount of routing efficiency. This also shows that the marginal benefit of RP is concentrated in order coverage and not in vehicle routing costs. A more surprising benefit is that dynamic solutions also reduce the gap standard deviation by 3.6%, giving consistently good solutions compared to AP. The cost reduction and fill rate increase are especially significant when the instance dynamism is high (smaller p_{start}), and the order arrival variability is high (bigger p_{out} and σ); this can be commonplace in SDD systems. We also found little benefit and few occurrences of the vehicle waiting at the depot during the planning horizon once dispatches begin.

RP's computational effort may prevent its deployment (at least using exact optimization) for bigger problem sizes, *e.g.*, $n \geq 50$. In this case, we provide a simpler and computationally efficient dynamic policy

(GP) that attains on average two thirds of the cost reduction benefits and fill rate increase over AP. We also show that a practically equivalent result can be achieved by the restricted rollout RRP; there appears to be almost no value in changing the initial dispatch wave established by AP. In general, the DDWP proved quite challenging to solve and future work may consider improvements in heuristic solution of the *a priori* problem using local search and meta-heuristics. This could allow us to solve bigger problems and to make the RP policy practical for on-line deployment in larger instances. Another interesting question is whether we can exploit probabilistic routing methods to solve the *a priori* problem with recourse.

We also empirically studied the tradeoff between two possible SDD objectives, minimizing total cost and maximizing the weighted order fill rate. One might think that these two objectives deliver similar results, since well-sequenced routes leave more vehicle time available to cover more orders. However, we found that one should expect significant sacrifices in vehicle routing efficiency in order to maximize fill rate, and that the distance cost of an additional customer covered becomes more expensive as order coverage increases. In our results, an RP solution that maximizes order coverage and one that minimizes costs have a 50% difference in traveled distance per order and a 5% difference in number of orders covered; also, a solution focusing on coverage makes more dispatches (85% more), has 17% shorter routes on average, and starts dispatching earlier, having on average a 50% longer vehicle utilization throughout the day. This has direct implications on driver salaries and vehicle maintenance costs. We also observed that, as fill rate becomes more important, the average improvement of dynamic policies over *a priori* policies increases, and sophisticated dynamic policies such as RP improve in relative terms over simpler dynamic policies like GP.

Possible extensions of the DDWP include incorporating vehicle service times at each location or including customer service time windows instead of a deadline at the end of the day. The extension of this model to multiple vehicles also seems natural; having a fleet of vehicles could pool the risk associated with leaving orders unattended and therefore reduce costs, *e.g.*, [1]. In general, same-day delivery offers many new challenges to the logistics research community.

References

[1] A. Ak and A. Erera, *A paired-vehicle recourse strategy for the vehicle-routing problem with stochastic demands*, Transportation science **41** (2007), no. 2, 222–237.

[2] M. Albareda-Sambola, E. Fernández, and G. Laporte, *The dynamic multiperiod vehicle routing problem with probabilistic information*, Computers & Operations Research **48** (2014), no. 0, 31–39.

[3] E. Angelelli, N. Bianchessi, R. Mansini, and M. G. Speranza, *Short term strategies for a dynamic multi-period routing problem*, Transportation Research Part C: Emerging Technologies **17** (2009), no. 2, 106–119.

[4] E. Angelelli, M. Savelsbergh, and M.G. Speranza, *Competitive analysis of a dispatch policy for a dynamic multi-period routing problem*, Operations Research Letters **35** (2007), no. 6, 713–721.

[5] E. Angelelli, M.G. Speranza, and M.W.P. Savelsbergh, *Competitive analysis for dynamic multiperiod uncapacitated routing problems*, Networks **49** (2007), no. 4, 308–317.

[6] D.L. Applegate, R.E. Bixby, V. Chvátal, and W.J. Cook, *The traveling salesman problem: A computational study*, Princeton University Press, Princeton, New Jersey, 2006.

[7] C. Archetti, D. Feillet, and M.G. Speranza, *Complexity of routing problems with release dates*, (2015).

[8] N. Azi, M. Gendreau, and J.-Y. Potvin, *An exact algorithm for a vehicle routing problem with time windows and multiple use of vehicles*, European Journal of Operational Research **202** (2010), no. 3, 756 – 763.

[9] ______, *A dynamic vehicle routing problem with multiple delivery routes*, Annals of Operations Research **199** (2012), no. 1, 103–112.

[10] ______, *An adaptive large neighborhood search for a vehicle routing problem with multiple routes*, Computers & Operations Research **41** (2014), 167 – 173.

[11] R. Bent and P. van Hentenryck, *Scenario-based planning for partially dynamic vehicle routing with stochastic customers*, Operations Research **52** (2004), no. 6, 977–987.

[12] L. Bianchi and A.M. Campbell, *Extension of the 2-p-opt and 1-shift algorithms to the heterogeneous probabilistic traveling salesman problem*, European Journal of Operational Research **176** (2007), no. 1, 131 – 144.

[13] D. Brown, J. Smith, and P. Sun, *Information relaxations and duality in stochastic dynamic programs*, Operations research **58** (2010), 785–801.

[14] A.M. Campbell and B. Thomas, *Challenges and advances in a priori routing*, The Vehicle Routing Problem: Latest Advances and New Challenges, Springer, 2008, pp. 123–142.

[15] ______, *Probabilistic traveling salesman problem with deadlines*, Transportation Science **42** (2008), no. 1, 1–21.

[16] D. Cattaruzza, N. Absi, and D Feillet, *The multi-trip vehicle routing problem with time windows and release dates*, Transportation Science **50** (2015), no. 2, 676–693.

[17] J.F. Cordeau, G. Laporte, M. Savelsbergh, and D. Vigo, *Vehicle routing*, Transportation, handbooks in operations research and management science **14** (2006), 367–428.

[18] R. DeNale, X. Liu, and D. Weidenhamer, *U.S. Census Bureau News: quaterly retail E-commerce sales - 3rd quarter 2015*, (2015), 1–3.

[19] A. Erera, M. Savelsbergh, and E. Uyar, *Fixed routes with backup vehicles for stochastic vehicle routing problems with time constraints*, Networks **54** (2009), no. 4, 270–283.

[20] M. Gendreau, G. Laporte, and R. Séguin, *Stochastic vehicle routing*, European Journal of Operational Research **88** (1996), no. 1, 3–12.

[21] B.L. Golden, S. Raghavan, and E.A. Wasil (eds.), *The Vehicle Routing Problem: Latest Advances and New Challenges*, Springer, 2008.

[22] P. Jaillet, *A priori solution of a traveling salesman problem in which a random subset of the customers are visited*, Operations Research **36** (1988), no. 6, 929–936.

[23] M. Klapp, A. Erera, and A Toriello, *The one-dimensional dynamic dispatch waves problem*, published online on Transportation Science (2016).

[24] G. Laporte, F. Louveaux, and H. Mercure, *A priori optimization of the probabilistic traveling salesman problem*, Operations Research **42** (1994), no. 3, 543–549.

[25] A. Larsen, O.B.G. Madsen, and M.M. Solomon, *Recent developments in dynamic vehicle routing systems*, The Vehicle Routing Problem: Latest Advances and New Challenges, Springer, 2008, pp. 199–218.

[26] Matt Lindner, *Global e-commerce sales set to grow 25% in 2015*, https://www.internetretailer.com (2015).

[27] V. Pillac, M. Gendreau, C. Guéret, and A. Medaglia, *A review of dynamic vehicle routing problems*, European Journal of Operational Research **225** (2013), no. 1, 1–11.

[28] M.L. Pinedo, *Scheduling: theory, algorithms, and systems*, Springer Science & Business Media, 2012.

[29] M. Puterman, *Markov decision processes: discrete stochastic dynamic programming*, John Wiley & Sons, 2009.

[30] N. Secomandi and F. Margot, *Reoptimization approaches for the vehicle-routing problem with stochastic demands*, Operations Research **57** (2009), no. 1, 214–230.

[31] H. Tang and E. Miller-Hooks, *Approximate procedures for probabilistic traveling salesperson problem*, Transportation Research Record: Journal of the Transportation Research Board **1882** (2004), 27–36.

[32] B. Thomas, *Dynamic vehicle routing*, Wiley Encyclopedia of Operations Research and Management Science, John Wiley and Sons, Inc., 2010, pp. 1–11.

[33] P. Toth and D. Vigo, *The granular tabu search and its application to the vehicle-routing problem*, Informs Journal on computing **15** (2003), no. 4, 333–346.

[34] ______, *Vehicle routing: Problems, methods, and applications*, vol. 18, SIAM, 2014.

[35] S. Voccia, A.M. Campbell, and B. Thomas, *The probabilistic traveling salesman problem with time windows*, EURO Journal on Transportation and Logistics (2012), 1–19.

[36] S. Voccia, A.M. Campbell, and B.W. Thomas, *The same-day delivery problem for online purchases*, Tippie College of Business, University of Iowa (2015), 1–40.

[37] M. Wen, J.-F. Cordeau, G. Laporte, and J. Larsen, *The dynamic multi-period vehicle routing problem*, Computers & Operations Research **37** (2010), no. 9, 1615–1623.