Please cite the paper as follows:

Kancharla, S. R., Woensel, T. V., Waller, S. T., and Ukkusuri, S. V. (2024). Meal Delivery Routing Problem with Stochastic Meal Preparation Times and Customer Locations. *Networks and Spatial Economics*, Accepted/In press.

Meal Delivery Routing Problem with Stochastic Meal Preparation Times and Customer Locations

Surendra Reddy Kancharla[a], Tom Van Woensel[b], S. Travis Waller[a,c,*], Satish V. Ukkusuri[d]

[a] *"FRIEDRICH LIST" Faculty of Transport and Traffic Sciences, Dresden University of Technology, Hettnerstraße 1-3, Dresden, 01069, Saxony, Germany*
[b] *School of Industrial Engineering and Innovation Sciences, Eindhoven University of Technology, 5600 MB, Eindhoven, P.O. Box 513, The Netherlands*
[c] *College of Engineering, Computing and Cybernetics, Australian National University , 108 North Rd, Canberra, 2601, Australia*
[d] *Lyles School of Civil Engineering, Purdue University, 550 W Stadium Ave., West Lafayette, 47907, Indiana, USA*

Abstract

We investigate the Meal Delivery Routing Problem (MDRP), managing courier assignments between restaurants and customers. Our proposed variant considers uncertainties in meal preparation times and future order numbers with their locations, mirroring real challenges meal delivery providers face. Employing a rolling-horizon framework integrating Sample Average Approximation (SAA) and the Adaptive Large Neighborhood Search (ALNS) algorithm, we analyze modified Grubhub MDRP instances. Considering route planning uncertainties, our approach identifies routes at least 25% more profitable than deterministic methods reliant on expected values. Our study underscores the pivotal role of efficient meal preparation time management, impacting order rejections, customer satisfaction, and operational efficiency.

Keywords: Meal delivery routing, Uncertainty, Sample Average Approximation, Adaptive Large Neighborhood Search.

*Corresponding author

1. Introduction

Integrating the gig economy into first and last-mile delivery services for freight and passenger sectors has significantly revolutionized urban transportation. This shift is in response to the escalating demands for efficient last-mile deliveries. Numerous startups specializing in restaurant meal delivery, such as Grubhub, Doordash, Deliveroo, Swiggy, and Ubereats, and on-demand transport services like Uber, Lyft, and Ola, have emerged to meet this growing need. Despite facing unique challenges, these services fundamentally address the same issue: they enable customers to conveniently request pickup and drop-off services through mobile apps or websites, typically ensuring delivery within a set timeframe. These companies charge a delivery fee, a portion of which is paid to the service provider. Success in this domain hinges on satisfying all parties involved: customers expect prompt, affordable, and reliable service; drivers aim for substantial earnings; and restaurants seek to expand their reach and customer base through these delivery services.

Our research addresses a vital issue in the freight industry: the intricacies of routing for restaurant meal deliveries. This is an expanded form of the classical Pickup and Delivery Problem (PDP), known for its computational complexity. Unlike traditional PDP, which often focuses on the limitations of vehicle numbers, our study concentrates on the variable elements that can significantly affect the efficiency of operations. In the context of the gig economy, the concept of a fixed fleet size is replaced by a potentially limitless number of independent couriers who work on a flexible schedule. This flexibility, however, brings unpredictability in several aspects - such as the couriers' working hours, their starting points for deliveries, and even their discretion to accept or decline orders.

Additionally, the variable nature of meal preparation times adds complexity to the food delivery sector. Minor variations in these times can lead to significant delays, adversely affecting customer satisfaction and the efficiency of delivery operations. The unpredictability of future orders intensifies this challenge, as the specific locations and quantities of these orders, crucial for route optimization, remain uncertain. Existing approaches in the meal delivery sector, such as those by Reyes et al. (2018) and Yildiz and Savelsbergh (2019), primarily rely on deterministic models with fixed meal preparation times and delivery windows, failing to account for the vari-

ability and unpredictability in these factors. In contrast, Ulmer et al. (2021) and Zheng et al. (2023) attempted to incorporate uncertainties in meal preparation and travel times, yet their models still lacked the dynamic adaptability required for real-life, uncertain scenarios. Relying on average estimates for meal preparation and future orders is insufficient, as this method overlooks the extensive range of variability and its influence on decision-making processes. These variables' complex and often unknown probability distributions render simple average-based approaches ineffective. This unpredictability dramatically expands the range of potential scenarios, making traditional deterministic solutions unfeasible. Such deterministic models, not accounting for this variability, tend to underestimate necessary resources, leading to operational inefficiencies and decreased profitability. Our goal is to develop a probabilistic model that accurately accounts for these uncertainties, allowing for the creation of more reliable and effective routing strategies that enhance the performance of the meal delivery sector.

Given the intricate challenges of uncertain meal preparation times and the unpredictable flow of orders, we've identified a collection of methods particularly adept at navigating the Meal Delivery Routing Problem (MDRP). To address the dynamic and uncertain aspects of MDRP, we utilize a rolling horizon framework that integrates the Sample Average Approximation (SAA) method with the Adaptive Large Neighborhood Search (ALNS), selected for its proven effectiveness in complex scenarios akin to MDRP. SAA, known for its strength in handling stochastic discrete optimization problems, uses a scenario-based approach to estimate the expected values of decision variables, considering various potential future states. This aspect is vital when relying on average estimates of uncertain factors, which could result in less optimal decisions. The SAA's incorporation facilitates the dynamic modification of routes in response to new information, aligning with the constantly changing delivery environment. This process involves repeatedly applying the SAA on selected scenarios over different time horizons, assessing the solutions against a broader range of scenarios to approximate the expected value function better, and updating routes based on actual developments. Such a strategy ensures that the solutions are always relevant and adaptable to the immediate operational demands.

The Adaptive Large Neighborhood Search (ALNS) algorithm, highly effective for large-scale routing problems (Li et al., 2016; Ghilas et al., 2016a,b; Zhu and Sheu, 2018a), is an integral

complement to the SAA method in our approach. ALNS (Ropke and Pisinger, 2006) is adept at exploring and optimizing within extensive solution spaces, making it particularly well-suited for addressing the Meal Delivery Routing Problem (MDRP). This algorithm is tailored to quickly adjust to changes in routing parameters, aligning with the unpredictable work schedules of couriers and the varying patterns of order acceptance typical in MDRP. Combining ALNS with the SAA method within a rolling horizon framework merges the SAA's ability to approximate stochastic elements with ALNS's capacity to efficiently navigate and fine-tune solutions in a broad and intricate solution environment. This synergistic approach enables us to develop resilient routing solutions in the face of uncertainties and adapt and respond to the dynamic nature of meal delivery operations.

The study primarily focuses on the challenges in the freight sector, particularly the restaurant meal delivery routing problem, a complex variant of the Pickup and Delivery Problem (PDP). This problem is characterized by unpredictable factors like varying courier availability, fluctuating working hours, and dynamic meal preparation times, significantly affecting operational efficiency. Traditional deterministic methods are inadequate due to the complexities and uncertainties involved, including the unpredictable nature of future orders. The study explores advanced methodologies like the Sample Average Approximation (SAA) and the Adaptive Large Neighborhood Search (ALNS) within a rolling horizon framework to address these challenges. These methods effectively handle the stochastic elements and complex solution spaces of the Meal Delivery Routing Problem (MDRP). The research includes developing tailored algorithms for the MDRP, conducting computational tests with real-world data, and performing sensitivity analysis to evaluate algorithm performance under varying levels of uncertainty.

The main contributions are as follows:

1. We developed innovative solution algorithms crafted explicitly for the meal delivery routing problem. These algorithms are strategically designed to adeptly handle uncertainties, such as the variability in meal preparation times, the fluctuating number of orders, and their locations. This tailored approach ensures that the routing solutions are efficient and highly responsive to the dynamic nature of meal delivery operations.

2. Furthermore, we undertake comprehensive computational testing using data derived from real-world scenarios. These tests were extensive, covering a wide range of scenarios that included various combinations of uncertainties. This approach allowed us to rigorously evaluate the algorithms in conditions that closely mimic actual operational environments, thereby ensuring the robustness and reliability of our solutions in practical settings.
3. We conduct a detailed sensitivity analysis to scrutinize how the algorithms performed under different levels of uncertainty. This systematic and thorough analysis provided deep insights into the behavior and performance nuances of the algorithms when confronted with varying degrees and types of uncertainties. Through this sensitivity analysis, we identified strengths and potential areas for improvement in our algorithms, ensuring they are effective and adaptable to the complex and ever-changing landscape of meal delivery services.

The paper's organization is as follows: Section 2 presents the literature review related to the MDRP and related problems. Section 3 formally introduces the MDRP and uncertainties considered. Section 4 describes the proposed solution algorithm for MDRP with uncertainties. Section 5 presents the test instances and discusses the computational results, followed by conclusions in Section 6.

2. Literature Review

We divide the literature into two parts. First, we review the work on Meal-delivery routing and related problems, followed by literature on stochastic Pickup and Delivery problems (PDPs).

Meal-delivery problems

Reyes et al. (2018) proposed a dynamic deterministic variant of a pickup and delivery problem called the Meal Delivery Routing Problem (MDRP), and they solved it using a rolling-horizon approach. Later, Yildiz and Savelsbergh (2019) proposed a mathematical formulation for the static variant and solved it using a simultaneous row and column generation-based algorithm. Both articles assumed unlimited capacity for couriers, a fixed delivery window of 90 minutes from the order's place, and fixed meal preparation times. Steever et al. (2019) relaxed the first two assumptions and allowed order placement from multiple restaurants. They proposed a heuristic

that accounts for future orders using equity and dispersion metrics. Ulmer et al. (2021) relaxed the latter two assumptions and proposed an anticipatory customer assignment policy that accounts for the random meal preparation times. However, the above models (except Steever et al. (2019)) only match couriers with orders and assume the couriers take the best routes. Moreover, they Yildiz and Savelsbergh (2019); Steever et al. (2019); Ulmer et al. (2021) also enforce the constraint of visiting all the orders. Liu (2019) proposes a MILP model for a problem similar to MDRP, where drones are used instead of regular couriers. Using drones helps remove uncertainty related to travel times and adds additional complexity, like charging requirements and limited capacity. Liao et al. (2020) proposes a two-stage solution algorithm for a static and deterministic variant of the problem to minimize carbon footprint. Recently, Zheng et al. (2023) proposed an iterative greedy algorithm to solve a meal delivery problem with uncertainties in meal preparation times and travel time and also proposed two time-saving strategies to improve the computational effort. However, their instances are not for dynamic scenarios closer to the real-life application.

Stochastic pickup and delivery problems

Stochastic variants of VRP have been extensively studied (Laporte et al., 2002; Verweij et al., 2003; Secomandi and Margot, 2009; Chu et al., 2015; Ghilas et al., 2016b; Zhu and Sheu, 2018b; Shi et al., 2018; Györgyi and Kis, 2019; Karoonsoontawong et al., 2020; Fachini et al., 2022). We can categorize the solution approaches for these variants into two groups. The first group uses stochastic programming with recourse, a well-known framework for modeling uncertainty optimization problems. In this method, some data is unknown at the moment of planning. First, a decision is made, and then the recourse costs of the consequences of the plan are minimized. The second group uses a multi-scenario approach, followed in this study. This method approximates expected costs by evaluating a solution based on generated scenarios. Metaheuristic algorithms are generally used in implementing the multi-scenario stochastic optimization approach. A good review of metaheuristic algorithms for stochastic combinatorial optimization can be found in Bianchi et al. (2009) and Gutjahr (2011).

Unlike the literature on stochastic VRP, literature on Pickup and Delivery Problems (PDP) with stochastic demands is limited. Powell et al. (1988) is one of the first studies that consid-

ered the dynamic PDP with stochastic demands. They also showed that considering uncertainty in the planning process results in substantial profits and increases the service level compared to the deterministic planning approach. In Ghilas et al. (2016b) integrated the PDP with the public transport system and considered stochastic demands. Zhu and Sheu (2018b) proposed a failure-specific cooperative recourse strategy for the simultaneous PDP with stochastic demand. Unlike the previous studies, Shi et al. (2018) considered uncertainty in travel and service times, Györgyi and Kis (2019) considered uncertainty in time windows. The typical result in all the above studies is that uncertainty in the planning process leads to significant improvements in objective over the deterministic case. In Zhang et al. (2023) introduced approximations based on the knapsack problem for estimating reward-to-go. These approximations serve as the foundation for creating efficient online scheduling policies and offline planning algorithms. In Wang et al. (2023), focused on meeting customer delivery punctuality expectations by estimating arrival times and success probabilities in uncertain scenarios. Their estimated success probabilities tended to be conservative lower bounds. They also introduced a solution approach based on a branch-price-and-cut framework.

3. Problem Description

Given a graph $G(N, A)$, where N denotes the set of nodes representing locations such as restaurants and customers, and A represents the arcs between these nodes. We have a set T, which represents tasks, divided into T_p for pickups and T_d for deliveries. The set V enumerates couriers involved in the delivery process. For each node $i \in T$, there are associated service times s_i and time windows defined by earliest e_i and latest l_i arrival times. The demand at each node is given by d_i, where positive values indicate pickups and negative values represent deliveries. Couriers $k \in V$ have specific on-times e_k, off-times l_k, and capacity constraints β_k. They expect a minimum payment m_k per unit time. The travel time between nodes i and j is denoted by t_{ij}. α, the cost per unit time, serves a pivotal role in our model by converting time metrics, like delay and waiting times, into cost metrics. Unlike c_{ij}, which directly represents the travel cost between nodes i and j, and p_i, which denotes the payment received for order i. Customers at node $i \in T_d$ have a maximum willingness to pay w_i. μ_i represents the deterministic time associated with waiting times and

delays at each node i. x_{ij}^k is a binary variable that is equal to 1 if courier k travels directly from node i to node j. a_{ij}^k is a continuous variable representing the arrival time of courier k at node j from node i. y_{ij}^k is a continuous variable representing the load carried by courier k when arriving at node j from node i. ξ is a random vector.

$$\max \sum_{i \in T_d} p_i - \sum_{i,j \in T} c_{ij} - \alpha \left(\sum_{i \in T_d} \mu_i - E\left[Q(x, \xi) \right] \right) \tag{1}$$

Subject to:

$$p_i \leq w_i \sum_{j \in N} \sum_{k \in V} x_{ji}^k \quad \forall i \in T_d \tag{2}$$

$$c_{ij} \geq \sum_{k \in V} m_k x_{ij}^k t_{ij} \quad \forall i, j \in T, i \neq j \tag{3}$$

$$\sum_{k \in V} \left(\sum_{j \in T \setminus \{i\}} x_{ji}^k \right) \leq 1 \quad \forall i \in T \tag{4}$$

$$\sum_{k \in V} \left(\sum_{j \in T \setminus \{i\}} x_{ij}^k \right) = \sum_{k \in V} \left(\sum_{j \in T \setminus \{i\}} x_{ji}^k \right) \quad \forall i \in T \tag{5}$$

$$\sum_{j \in T} \sum_{i \in T_p} x_{ij}^k = \sum_{j \in T} \sum_{i \in T_d} x_{ji}^k \quad \forall k \in V, i \neq j \tag{6}$$

$$\sum_{j \in N \setminus \{i\}} \sum_{k \in V} a_{ji}^k \leq \sum_{j \in N \setminus \{i\}} \sum_{k \in V} a_{ij}^k - (s_i + t_{ij}) x_{ij}^k \quad \forall i \in T \tag{7}$$

$$e_i x_{ij}^k \leq a_{ij}^k \quad \forall k \in V, i \in T, j \in T \setminus \{i\} \tag{8}$$

$$e_k x_{ij}^k \leq a_{ij}^k \leq l_k x_{ij}^k \quad \forall k \in V, i \in N, j \in N \setminus \{i\} \tag{9}$$

$$\sum_{j \in N \setminus \{i\}} a_{ji}^k \leq \sum_{j \in T \setminus \{i+n\}} a_{ji+n}^k \quad \forall k \in V, i \in P \tag{10}$$

$$\sum_{k \in V} \sum_{j \in N \setminus \{i\}} y_{ij}^k - d_i x_{ij}^k = \sum_{k \in V} \sum_{j \in N \setminus \{i\}} y_{ji}^k \quad \forall i \in T \tag{11}$$

$$y_{ij}^k \leq \beta_k x_{ij}^k \quad \forall k \in V, i \in N, j \in N \setminus \{i\} \tag{12}$$

$$\mu_i + l_i \geq \sum_{k \in V} \sum_{j \in N} a_{ji}^k + s_i x_{ij}^k \quad \forall i \in T_d \tag{13}$$

$$x_{ij}^k \in \{0,1\} \quad \forall k \in V, i \in N, j \in N \tag{14}$$

$$y_{ij}^k \geq 0, a_{ij}^k \geq 0, p_i \geq 0, c_{ij} \geq 0 \quad \forall k \in V, i \in N, j \in N \tag{15}$$

The objective function (1) is designed to maximize expected profit while accommodating the stochastic nature of demand and service times. It explicitly includes penalties for missed deliveries and service delays, addressed within the function's third term. This term employs a cost factor, scaled by α, that increases proportionally with the deviation from scheduled delivery times. By incorporating this penalty, the function effectively quantifies the financial and service quality impact of delays, ensuring that operational strategies seek to optimize profitability and uphold reliability and customer satisfaction. Constraints (2) limit the maximum payment expected from the customer. Constraints (3) ensure couriers receive at least the minimum expected payment. Constraints (4) ensure each pickup and delivery pair is visited at most once. Constraints (5)-(6) ensure flow conservation at each node. Constraints (7) track the arrival time at each node. Constraints (8) ensure the earliest arrival time at a node is respected. Constraints (9) ensure assignments to couriers are within their shift time. Constraints (10) ensure that the arrival time at the delivery node is later than the pickup node. Constraints (11) ensure demand satisfaction at each node. Constraints (12) ensure courier capacity is not violated. Constraints (13) ensure that the actual delay experienced by the customer is at least as large as the service time plus the travel time, ensuring that no penalty is applied for early or on-time deliveries. Constraints (14) and (15) define the decision variables' domains, ensuring that routes are binary decisions and all other variables are non-negative.

Unlike the traditional vehicle routing problems, we allow the dropping of orders. Constraints (4) make this dropping of orders possible. The x_{ij}^k terms in constraints (3), (4), (8) and (11) avoid considering the dropped order in the estimation of payment from the customers, payments made to couriers, arrival time tracking, and courier load tracking, respectively.

3.1. Recourse action

The meal preparation times are realized when the courier arrives at the restaurant, and future orders from a restaurant can be realized only when the order is placed. The longer waiting times due to delays in meal preparation can violate the time windows. Suppose the delay is within the given buffer. In that case, a delay penalty will be added to the objective, or when the delay is beyond the allowed buffer, the orders are dropped, and a penalty is applied to the objective. Unrealized orders also impact the availability of couriers for future orders because of the detours taken to meet the realized orders.

3.2. Modeling of meal preparation times

We model the meal preparation time at each restaurant node as a random variable given by $t + \delta_i$, where $\delta_i \geq 0$ is the duration of the stochastic disruption at node i and t is a deterministic meal preparation time. Specifically, δ_i follows a gamma distribution with a given shape parameter (k) and scale (θ) parameter that depends on the deterministic meal preparation time. The Gamma distribution is commonly used in the literature to describe stochastic times, as they follow convolution and non-negativity properties. The parameters k and θ allow for the generation of scenarios considering the degree by which preparation times vary, adjusted by the coefficient of variation ($\hat{c}_v$). For our analysis, $\hat{c}_v^2 = 0.25$ gave the best fit for the meal preparation times.

We derive parameters k and θ for a given value $\hat{c}_v$ as follows:

$$\hat{c}_v = \frac{\mathbb{E}(\delta_i)}{Var(\delta_i)} = \frac{k\theta}{k\theta_i^2} \Rightarrow k = \frac{1}{\hat{c}_v^2}; \theta_i = t\hat{c}_v^2 \tag{16}$$

3.3. Modeling of future orders and their locations

We assume that the number of orders within the upcoming interval follows a Poisson distribution with a mean arrival rate λ, represented by $O_{t+1} \sim \text{Poisson}(\lambda)$. The Poisson distribution is commonly used in the literature to represent random occurrences. After determining the number of orders, O_{t+1}, we identify their probable locations.

We divide the customer base into central and peripheral segments using the Isolation Forests method, as described by Liu et al. (2008). The proportion of customers within each segment

denoted as α_{central} for the central segment and $\alpha_{\text{peripheral}}$ for the peripheral segment, guides the determination of potential future order sites from these segments.

Once the number of future orders is predicted, it is divided into central and peripheral orders based on the α values. Specifically, the predicted number of orders, O_{t+1}, is multiplied by α_{central} to determine the number of central orders and by $\alpha_{\text{peripheral}}$ to determine the number of peripheral orders. These orders are then randomly assigned within their respective segments.

For example, if the mean arrival rate λ results in 10 predicted orders, with α_{central} at 0.6 and $\alpha_{\text{peripheral}}$ at 0.4, then six orders will be assigned to the central segment and four to the peripheral segment. Within each segment, orders are randomly selected based on the customer distribution in that area. This method ensures that the distribution of future orders adapts to the network's structure. We can accurately model future order locations within different network configurations using these proportions and random selection within those segments.

4. Solution methodology

We first divide the total time into η time buckets based on an interval length of ζ. All orders (pickup list) are grouped into these buckets based on their order times. The couriers available within these buckets have also been identified (couriers list). We re-optimize using the ALNS at the end of each time bucket. Algorithm 1 presents the pseudo-code for the algorithm used.

We relocate the unused couriers to their nearest restaurant. In the case of a tie, we relocate the courier with the highest difference between its nearest and second nearest restaurant. Couriers still performing deliveries from the previous buckets are not considered in the route planning of the present bucket.

To effectively handle the inherent uncertainties in the Meal Delivery Routing Problem (MDRP), we adopted the Sample Average Approximation (SAA) framework (Kleywegt et al., 2002), renowned for its efficacy in solving stochastic optimization challenges. This choice is motivated by SAA's ability to approximate complex expected value functions, a method first pioneered by Verweij et al. (2003) for two-stage stochastic routing problems. SAA simplifies the expected value function $\mathbb{E}[R(x, \omega)]$ into a more manageable sample average function $z(x) = \frac{1}{N} \sum_{i=1}^{N} R(x, \tilde{\omega}^i)$, focusing on a subset of realizations from the random vector ω. This approach allows us to solve a

Algorithm 1 Overview of the Rolling-Horizon framework implementation

Initialize data (list of pickups and couriers available at each time bucket of length λ)
for $pickuplist, courierslist \leftarrow timebucket$ **do**
    $courierslist = courierslist - removelist$
    calculate travel times
    Update *routes*, *arrival times* and *costs* based on Recourse actions (skip order)
    Use SAA framework on available pickups and couriers data along with stochastic variables
    relocate the *couriers* for next time bucket
    if *courier* finishes its route before the start of next time bucket **then**
        update *courier* location to nearest *pickup*.
        if two *couriers* has same nearest *pickup* **then**
            send the *courier* to nearest *pickup* that has the highest difference between second nearest to nearest *pickup*
        end if
    else
        update *couriers* location to the delivery location corresponding to last *pickup* before the next bucket start time.
    end if
    add *couriers* who will not be available in next time bucket to *removelist*
end for

more refined SAA problem $\min_{x \in X} \nu_\Omega(x), \nu_\Omega(x) = w'x + z(x)$, using a large set of realizations $\Omega' = \tilde{\omega}^{1'}, \ldots, \tilde{\omega}^{N'}, N' \gg N$, to approximate the true expected value. The process is iteratively refined until it meets predefined criteria, such as the maximum number of replications or a sufficiently small optimality gap.

Algorithm 2 illustrates the main steps of our SAA implementation. Unlike the initial implementations of the SAA framework proposed by Kleywegt et al. (2002), which performs a fixed number of replications of SAA over a fixed sample size $|\Omega| = N$, we neither fix the number of replications nor samples used. Instead, we adjust the number of samples based on the gap (ϵ_m) in the current SAA replication, and the number of replications is solved until one of the two stopping criteria is met. Kleywegt et al. (2002) uses $\epsilon = \nu_{\Omega'}(x) - \hat{v}_\Omega$ as an estimator of the true optimality gap $\nu_{\Omega'}(x) - v^*$, where v^* is the optimal cost of the problem considering all possible realizations of ω. By solving each SAA replication to optimality, it can be shown that $v^* - \mathbb{E}[\hat{v}_\Omega]$ is monotonically decreasing in N Verweij et al. (2003). Since in our approach, individual SAA problems are solved heuristically, and the objective is a maximization function. The estimate $\hat{v}_\Omega$ is not neces-

sarily valid and tends to underestimate the true upper bound. However, we still compute and use the estimator $\hat{v}_\Omega$ to evaluate the performance of the SAA problems given the current sample size (N) and increase the sample size throughout the method, but only after achieving a certain level of convergence.

In SAA problems, choosing the sizes N and N' is a trade-off between solution quality and computational efficiency. As the N value increases, the runtime of each SAA problem also increases, but the estimated upper bound, $\hat{v}_\omega$, tends to be stronger, which results in a smaller SAA gap. In our implementation, we use a fixed $|\Omega'| = N'$ throughout the algorithm and start solving the SAA problems over small sample sets $|\Omega| = N$. An integer parameter Δ controls how many additional scenarios need to be considered (increase in N). We solve multiple replications of SAA with the same sample size N until they converge. Once they converge, we evaluate the gap ϵ_m, and if the gap is not within the tolerance, N is increased by Δ. This process is repeated until the gap is within the tolerance or the maximum value of N is reached. All solutions obtained by solving each SAA replication are evaluated over a new set Ω', and the best solution is returned. In Section 5, we conduct experiments to assess the implementation decisions taken in our SAA.

4.1. Solving the SAA Problem

To solve the SAA problem $\max_{x \in X} w'x + \frac{1}{N} \sum_{i=1}^{N} R(x, \tilde{\omega}')$, we use an Adaptive Large Neighborhood Search (ALNS) heuristic. We modified the objective in the heuristic by adding the sample average function. As a result, we also modified the local search operators such that the marginal cost of an insertion considers both the changes in travel time and recourse costs over N scenarios $\Omega = \tilde{\omega}^1, \ldots, \tilde{\omega}^N$. In particular, computing the change in the recourse costs before and after an operator is applied requires evaluating the modified solution (route) on the sample set Ω. Algorithm 3 overviews how second-stage recourse costs are computed for a given first-stage solution and recourse policy (R) considering a sample of K realizations.

Recourse decisions described in Section 3 are applied to each a priori route in the first-stage solution based on the realization of meal preparation time and future orders. A new route (second-stage route) is obtained in which customers are dropped. The penalty θ_c is incurred for every order $c \in U \subset C$ not visited in the second-stage route and the delay cost for ones visited beyond the time

Algorithm 2 Overview of the proposed SAA framework implementation

INPUT: Initial sample size N ; N' the large set of realizations $\Omega' = \{\omega^{1'}, \ldots, \omega^{N'}\}$; Number of replications l to check for convergence
OUTPUT: Solution $\boldsymbol{x}^*$

1: m $\leftarrow 1$
2: Generate $\Omega^m = \{\omega^1, \ldots, \omega^N\}$
3: Solve the SAA problem using ALNS over Ω^m, with objective v_{Ω^m} and solution $\boldsymbol{x^m}$
4: Compute lower bound $v_{\Omega'}(\boldsymbol{x^m}) = \boldsymbol{w'x^m} + \frac{1}{N'} \sum_{i=1}^{N'} R(\boldsymbol{x^m}, \Omega^{i'})$
5: Compute upper bound $\hat{v}_\Omega \leftarrow \frac{1}{m} \sum_{j=1}^{m} v_{\Omega^j}$
6: Compute the SAA gap estimate (ϵ_m) using upper and lower bounds
7: Compute $\sigma^2_{\hat{v}_\Omega}$ over the past l replications
8: **if** $\sigma_{\hat{v}_\Omega} \leq 0.01$ **then**
9: **if** $\epsilon_m \leq 0.05$ or $N >60$ **then**
10: Generate a new sample set Ω'
11: *return* $x^* = \max_{i=1,\ldots,m} v_{\Omega'}(x^*)$
12: **else** $N \leftarrow N + \Delta$
13: **end if**
14: **end if**
15: $m \leftarrow m + 1$
16: goto step 2

window (γ_c). Finally, Algorithm 3 returns the average recourse cost for solution x computed over all scenarios in Ω.

4.2. Adaptive Large Neighborhood Search

To solve the Meal Delivery Routing Problem (MDRP) effectively, we implement the Adaptive Large Neighborhood Search (ALNS) algorithm, recognized for its adeptness in large-scale routing and ability to adapt to dynamic conditions like variable courier schedules and order patterns. The process begins with an initial solution created using the random best insertion operator. This initial step is pivotal, as ALNS thrives on a starting point for optimization. In our approach, ALNS employs a dual strategy of high-impact and low-impact removal operators. High-impact operators, used every five iterations, remove both customer and restaurant nodes for substantial route adjustments, while low-impact operators, employed in other iterations, remove only customer nodes to

Algorithm 3 Evaluating stochastic costs for a ALNS solution $\boldsymbol{x}$

INPUT: ALNS solution $\boldsymbol{x}$; a sample $\Omega = \{\omega^1, \ldots, \omega^N\}$ of ready time realizations; a recourse action R

OUTPUT: Total cost $v + \dfrac{1}{N}\sum_{i=1}^{N} R(\boldsymbol{x^m}, \Omega^i)$

$R(x, \Omega) \leftarrow 0$
for each scenario $\omega^i \in \Omega$ **do**
    for each route (r), in $\boldsymbol{x}$ **do**
        Apply recourse action to r, considering new ready times and future orders, to obtain a new route r'
        for each order $c \in r'$ **do**
            if c is delayed within limits **then**
                $R(x, \Omega) \leftarrow R(x, \Omega) + \gamma_c$
            else if c is dropped **then**
                $R(x, \Omega) \leftarrow R(x, \Omega) + \theta_c$
            end if
        end for
    end for
end for
return $v + \dfrac{R(x, \Omega)}{N}$

refine the visit sequence within the same route.

These operators are chosen through a roulette wheel, guided by their previous performance. This method ensures that operators with a successful track record, indicated by higher weights, are more likely to be selected again. We utilize the simulated annealing (SA) criterion to accept new solutions. This criterion prefers solutions with higher profit but also, interestingly, accepts lower-profit solutions with a probability determined by $e^{-(f(S^*)-f(S)/kT)}$, where $f(S)$ represents the solution's objective value, T is the temperature, and k is the Boltzmann constant. The temperature T is methodically reduced by a factor α after each set of iterations at the current temperature. The ALNS process concludes either after reaching the maximum iteration limit or following β temperature reductions without any improvement in solution quality, with β set at 100 in our study.

This comprehensive method combines ALNS's dynamic route optimization with the SAA method's stochastic approximation. Integrating high-impact operators for broad changes and low-

impact operators for fine-tuning, coupled with the SA acceptance criterion and temperature-based iteration control, creates a versatile and efficient system. This synergy between ALNS and SAA forms a sophisticated, flexible framework tailored to meet meal delivery logistics' unique challenges and variabilities.

4.2.1. Removal operators

We use three removal operators: random removal and worst time removal introduced by Ropke and Pisinger (2006), and random route removal introduced by Hemmelmayr et al. (2012). Random and worst removal are modified to be used as high and low-impact operators, whereas route removal is used only as high-impact removal operators. In random removal, we randomly select q customers (restaurants for high-impact operators) and remove them from the solution. In case of worst-time removal, we calculate the gain in travel time after removing a customer and remove the first q customers (restaurants for high-impact operators) that will lead to the highest gain. We randomly remove r courier routes from the solution using random route removal.

4.2.2. Insertion operators

We use three new insertion operators: delivery deadline-based best insertion, best insertion with a limit on couriers, and lowest delay insertion, along with three insertion operators from the literature: random best insertion, random best with perturbation, and regret-2 insertion introduced by Ropke and Pisinger (2006). In all best insertion operators, we first calculate the insertion cost (increase in travel time) for the insertion of each node from the list of nodes removed using removal operators. If the node is a customer, we will only get the insertion cost for the courier containing its restaurant. In contrast, we will get the insertion cost for all the available couriers for a restaurant node. We randomly select a node, calculate its insertion cost, and insert it in the location, leading to the lowest cost increase in random best insertion. Random best with perturbation also works similarly to random best. The only difference is that we will introduce some noise to the insertion costs calculated and then insert it according to the revised insertion costs. We first rank the nodes based on the delivery deadline and use this order for insertion calculations in delivery deadline-based best insertion. In the best insertion with a limit on couriers, we first calculate the insertion costs and check if the courier with the lowest insertion cost does not already cater to more than

the limit. If the limit is not satisfied, we will insert it in the following best location. In the lowest delay insertion, we calculate the delay experienced by all the customers in the courier's route and insert the route that leads to the lowest increase in delay. Unlike the previous myopic operators, regret-2 insertion calculates the regret cost, which is the difference in the costs of inserting at the best location compared to its second-best location. We will then insert the nodes that have the highest regret costs.

4.2.3. Adaptive weights

We use a roulette wheel mechanism to select the insertion and removal operators. Initially, all operators have equal weights, and the score of all operators is set to zero. We update operators' scores in every iteration based on their performance. These scores are used to update the weights of the operators at the end of current temperature iterations, and then scores are reset to zero. Weights are updated as $W_o^i = W_o^{i-1} + s_o^i/\pi_i$, where W_o^i is the weight of the operator o after i^{th} temperature reduction and s_o^i is the score of the operator o during i^{th} temperature reduction, π_i is the sum of scores of all operators in the respective category (low impact removal, high impact removal, insertion) during the i^{th} temperature reduction. We maintain separate roulette lists for low and high-impact removal operators to make the learning of low and high-impact operators' performance independent.

We use ALNS for each time bucket of the instance, so instead of making the ALNS learn about the operators' performance from scratch, we pass the information about the operators' weights from one time bucket to another.

5. Results and Discussion

We use two instances from each set of set 0, set 1, and set 2 of MDRP instances created by Grubhub (https://github.com/grubhub/mdrplib) to create a total of 24 smaller instances with total orders of 126, 134, and 177. The original instances are based on real-world order arrival patterns and courier starting locations from different metropolitan areas. The instance's name gives complete information about the variations applied to it. For example, an instance name 0o50t100s2p100c1v2 suggests that it belongs to set 0 with 50% of original orders (o50), original

travel times (t100), optimized courier locations (s2), original order ready times (p100), considers first half of the orders (c1), and uses only second half of the couriers among the couriers available during the same time (v2).

We also include demand, time windows, maximum delay time allowed, and willingness to pay for each order. We use a service time of 4 minutes for pickup and delivery and a per-minute courier compensation of $ 0.4. Table 1 lists the instances and their characteristics. The columns orders, restaurants, and couriers represent their numbers, respectively. The column Time windows list the mean and standard deviation of time windows used. Similarly, the columns Willingness To Pay (WTP) and Demand list their mean and standard deviation, respectively.

Table 1: Instances Details

Instance	**Orders**	**Restaurants**	**Couriers**	**Time Window**		**WTP**		**Demand**	
				Mean	Std	Mean	Std	Mean	Std
0o50t100s1p100mc1v1	126	64	40	57.02	8.94	7.40	3.38	1.99	0.80
0o50t100s1p100mc1v2	126	64	40	57.02	8.94	7.40	3.38	1.99	0.80
0o50t100s1p100mc2v1	126	60	40	56.17	8.60	7.43	3.41	1.91	0.79
0o50t100s1p100mc2v2	126	60	40	56.17	8.60	7.43	3.41	1.91	0.79
0o50t100s2p100mc1v1	126	64	47	57.02	8.94	7.40	3.38	1.99	0.80
0o50t100s2p100mc1v2	126	64	47	57.02	8.94	7.40	3.38	1.99	0.80
0o50t100s2p100mc2v1	126	60	47	56.17	8.60	7.43	3.41	1.91	0.79
0o50t100s2p100mc2v2	126	60	47	56.17	8.60	7.43	3.41	1.91	0.79
1o50t100s1p100mc1v1	134	64	35	56.78	9.33	7.31	3.30	2.09	0.80
1o50t100s1p100mc1v2	134	64	35	56.78	9.33	7.31	3.30	2.09	0.80
1o50t100s1p100mc2v1	134	66	35	56.40	8.83	6.81	3.39	2.16	0.80
1o50t100s1p100mc2v2	134	66	35	56.40	8.83	6.81	3.39	2.16	0.80
1o50t100s2p100mc1v1	134	64	35	56.78	9.33	7.31	3.30	2.09	0.80
1o50t100s2p100mc1v2	134	64	35	56.78	9.33	7.31	3.30	2.09	0.80
1o50t100s2p100mc2v1	134	66	35	56.40	8.83	6.81	3.39	2.16	0.80
1o50t100s2p100mc2v2	134	66	35	56.40	8.83	6.81	3.39	2.16	0.80
2o50t100s1p100mc1v1	177	93	61	58.25	9.49	7.30	3.34	1.90	0.79
2o50t100s1p100mc1v2	177	93	61	58.25	9.49	7.30	3.34	1.90	0.79
2o50t100s1p100mc2v1	177	88	61	58.67	9.54	7.25	3.17	2.04	0.85
2o50t100s1p100mc2v2	177	88	61	58.67	9.54	7.25	3.17	2.04	0.85
2o50t100s2p100mc1v1	177	93	70	58.25	9.49	7.30	3.34	1.90	0.79
2o50t100s2p100mc1v2	177	93	70	58.25	9.49	7.30	3.34	1.90	0.79
2o50t100s2p100mc2v1	177	88	70	58.67	9.54	7.25	3.17	2.04	0.85
2o50t100s2p100mc2v2	177	88	70	58.67	9.54	7.25	3.17	2.04	0.85

The rolling horizon framework with SAA and ALNS is implemented in Python. All instances

are tested on an Intel Xeon Gold 5220 server running at 2.20GHz with 72 cores and 188 GB RAM running CentOS-7. We allow parallel processing for solving multiple SAA replications.

5.1. Parameter tuning

We conducted rigorous parameter tuning, exploring a wide range of values for each parameter. After extensive testing, we identified the optimal parameters for our study: a temperature setting of 50, a temperature reduction factor of 0.98, a maximum of 10 times the number of orders for temperature reduction iterations, and 20 times the number of orders for maximum iterations at a temperature. Additionally, we set the value of Ω' to 1000.

It's well-known that increasing the maximum number of replications (N) generally leads to improved results, albeit with a substantial increase in computational time. To strike a balance between solution quality and runtime efficiency, we tested various values of N, ranging from 20 to 100 with increments of 20, on three instances, each representing a different set (refer to Table 2). Our experiments revealed that increasing N beyond 60 resulted in only marginal improvements in the objective, accompanied by a significant rise in runtime. Consequently, we opted for N=60 in this study, finding it to be the optimal compromise between solution quality and computational efficiency.

Table 2: Effect of the value of N on objective and runtime

N	**Objective**	**Runtime (s)**	**% ↑ Obj**	**% ↑ runtime**
20	1624.02	3052	-	-
40	1642.56	3759	1.14	23.17
60	1773.14	5867	**7.95**	**56.05**
80	1777.79	7526	0.26	28.28
100	1780.33	11348	0.14	50.79

5.2. Stochasticity in both meal preparation times and future orders

This problem lacks established benchmark results. Consequently, we employ our Stochastic Approximation Algorithm (SAA) framework to generate stochastic and deterministic solutions where all random variables are substituted with their expected values. Subsequently, we compare

both solutions' routing costs (first stage) and recourse costs (second stage). Algorithm 2 is applied to compute the expected recourse cost, utilizing a scenario sample size of $|\Omega| = 1000$. This experiment illustrates the impact of incorporating uncertainty on a stochastic solution's first-stage decisions and costs.

In Figure 1, we present a comparison between the deterministic and stochastic solutions for the instances detailed in Table 1 using the recourse R. It is important to note that in R, no corrective actions are implemented; the second-stage costs solely account for penalties accrued due to missed orders and delays. This experiment generates scenarios with a coefficient of variance $\hat{c}_v = 0.25$. The results are averaged over three iterations of the SAA framework.

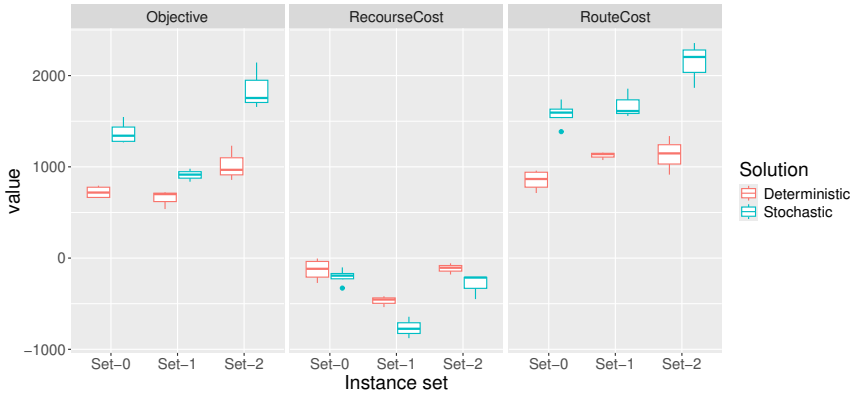

Figure 1: Comparison of stochastic and deterministic solutions

Table 3 indicates that employing probabilistic information, rather than relying solely on expected values, substantially boosts the objective value. This increase is primarily attributed to a significant rise in the total number of orders served, with an average increase of more than 13% across all datasets. However, it's important to note that the computational runtime experiences a substantial surge, exceeding 90% in all cases. This increase in runtime can be primarily attributed to the need for repeated solving for N replications of the SAA.

We ran two scenarios to see which stochastic variable is causing the significant increase in the number of orders served and, thereby, the objective value. In the first case, stochastic information about the meal preparation time alone is considered, and future orders are not considered. In the second case, stochastic information about the number of future orders alone is considered, and meal preparation times are assumed to be known.

Table 3: Percentage change in Objective value, Orders missed, and runtime by including probabilistic information of both meal preparation time and future orders

Instance set	Percentage change		
	↑ Objective value	↓ Orders missed	↑ Runtime
Set-0	36.79	15.67	90.22
Set-1	27.56	17.16	90.18
Set-2	34.43	13.56	90.24

5.3. Stochasticity only for meal preparation times

Like the previous scenario, we analyze the routing and recourse costs in stochastic and deterministic contexts. Figure 2 and Table 4 depict the outcomes. Interestingly, incorporating stochastic data solely for meal preparation times did not significantly enhance the solution compared to the deterministic approach. This was despite a considerable increase in runtime due to repeated problem-solving. Furthermore, not considering future orders led to a notable drop in the objective value, exceeding 20% compared to the scenario where both meal preparation time and future orders were considered stochastically. However, it is important to note that there was still an improvement over the purely deterministic solution.

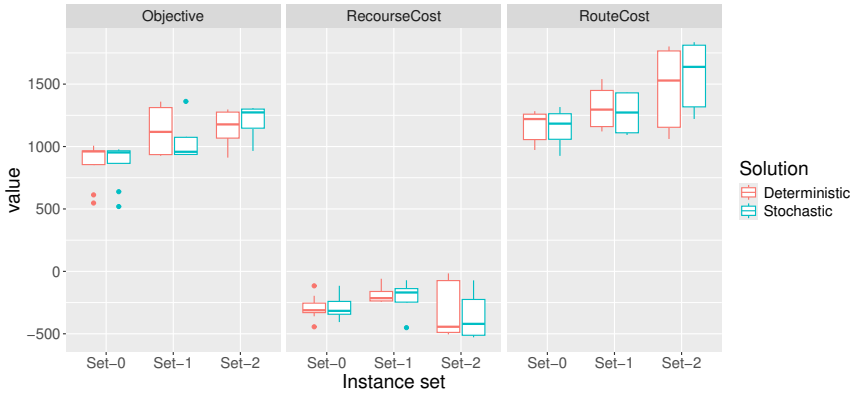

Figure 2: Comparison of stochastic and deterministic solutions with meal preparation time as the only stochastic variable

5.4. Stochasticity only for future orders

Here, we assume that meal preparation times are predetermined while the number of future orders follows a probability distribution. Figure 3 and Table 5 illustrate the impact of incorporating

Table 4: Percentage change in Objective value, Orders missed, and runtime by including probabilistic information only for meal preparation times

Instance set	**Percentage change**		
	↑ Objective	**↓ Orders missed**	**↑ Runtime**
Set-0	0.73	0.40	66.37
Set-1	3.60	0.93	62.45
Set-2	6.45	0.42	64.99

probabilistic information instead of relying solely on expected values. Notably, we observe a substantial improvement in the objective value, exceeding 32%, and an increase in the number of orders served by more than 13%. As expected, this enhancement comes at the cost of a significant runtime increase of over 90%. This approach yields superior solutions when considering meal preparation times and future orders as stochastic variables. The primary driver of this improvement is the precise knowledge of meal preparation times.

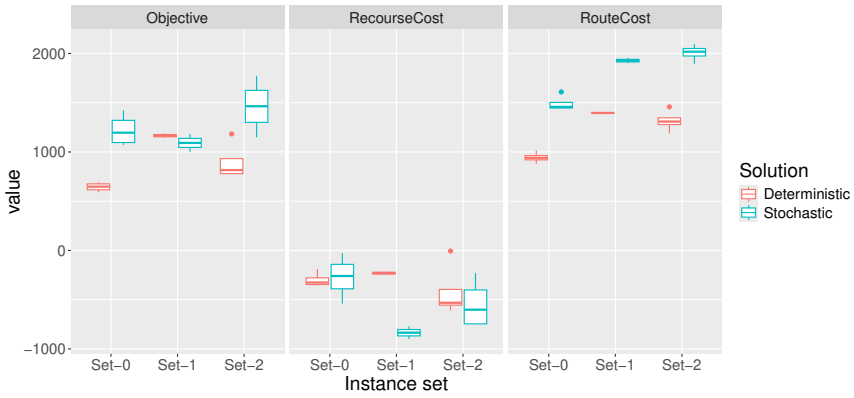

Figure 3: Comparison of stochastic and deterministic solutions with the number of future orders as the only stochastic variable

Table 5: Percentage change in Objective value, Orders missed, and runtime by including probabilistic information only for future orders

Instance set	**Percentage change**		
	↑ Objective	**↓ Orders missed**	**↑ Runtime**
Set-0	46.06	16.67	91.19
Set-1	32.96	13.18	90.38
Set-2	47.11	14.50	90.30

5.5. Variation in the level of stochasticity for both meal preparation times and future orders

Our study aimed to explore how varying degrees of randomness in the system affect its performance, specifically looking at meal preparation times and order frequencies. We employed two statistical distributions to achieve this: the gamma distribution for meal preparation times and the Poisson distribution for the number of orders. We adjusted a range of parameters within these distributions to simulate different levels of variability.

Firstly, we altered the gamma distribution's shape parameter, which controls the variability of meal preparation times. The idea was to mimic real-world scenarios where some meals might be prepared quickly while others take longer. By increasing the shape parameter, we introduced greater unpredictability in preparation times, reflecting a more realistic and challenging environment for the system.

Secondly, we modified the lambda parameter in the Poisson distribution, which dictates the average frequency of orders. This allowed us to simulate high and low-order periods, examining how the system copes with fluctuating demand.

Through these adjustments, we sought to understand the system's resilience to uncertainty comprehensively. As illustrated in Figure 4, the results revealed a direct correlation: higher variability in both meal preparation times and order frequencies led to an increase in missing orders. This, in turn, had a ripple effect, causing a rise in both route and recourse costs as the system struggled to adapt to the heightened unpredictability.

This finding highlights the challenge of dealing with increased uncertainty in the system. As the orders and meal preparation times became more unpredictable, the system had difficulty managing resources and planning routes effectively. Despite efforts to adapt, the system faced more missed orders, driving up costs. These observations, as illustrated in Figure 4, emphasize the delicate balance needed to handle the complexities of heightened uncertainty.

5.6. Variation in the level of stochasticity only for meal preparation times

To study the impact of stochasticity in meal preparation times, we maintained a consistent order distribution while varying the levels of uncertainty in meal preparation. As depicted in Figure 5, there was a substantial rise in the percentage of missed orders, exceeding 40% in certain instances,

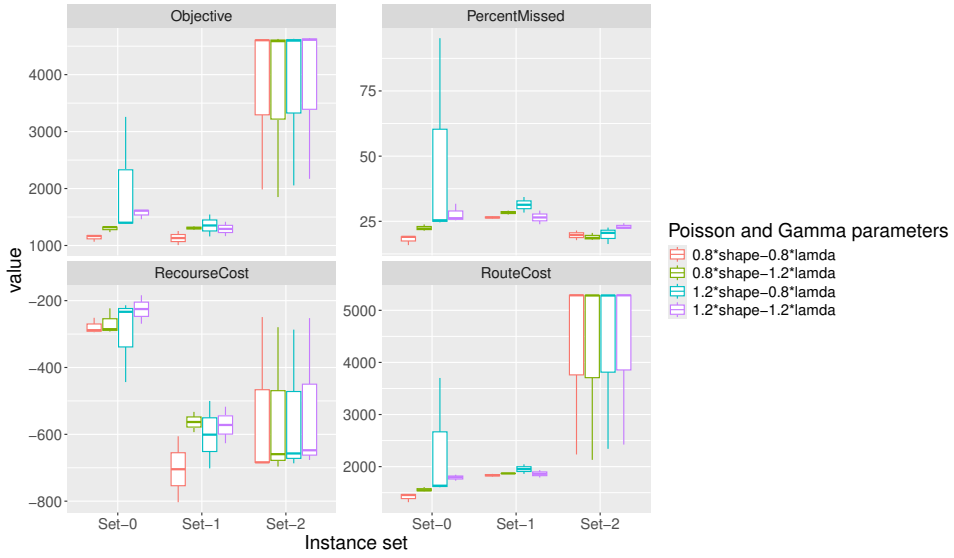

Figure 4: Comparison of solutions with variation in the level of stochasticity for both meal preparation times and future orders

especially when the shape parameter was increased by 50%. This trend was consistently observed across all three sets of instances.

As uncertainty increased and the distribution spread wider, meal preparation times grew longer. This extended waiting period affected delivery drivers, causing them to experience delays. Consequently, many orders had to be rejected due to the shortage of available drivers. This situation leads to delayed deliveries and results in lost sales opportunities and operational inefficiencies. Managing meal preparation times effectively is vital to optimizing resources, minimizing order rejections, and enhancing operational efficiency and customer satisfaction.

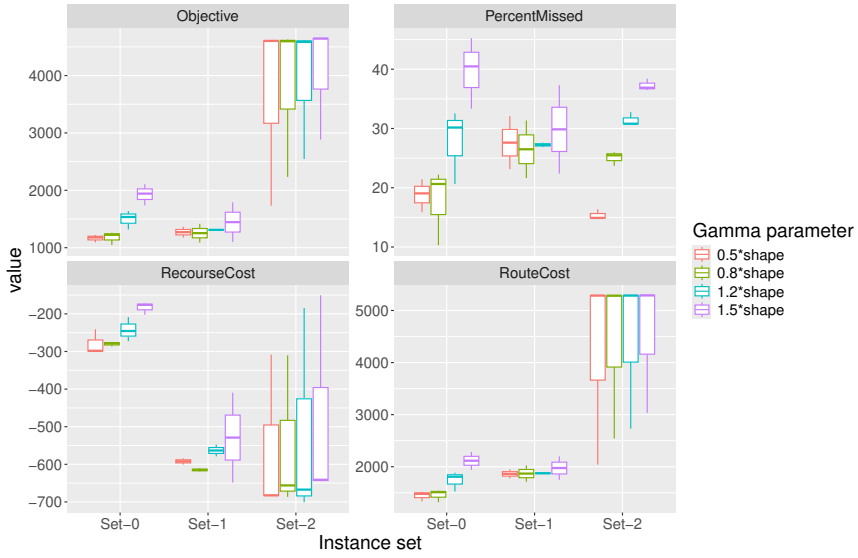

Figure 5: Comparison of solutions with variation in the level of stochasticity only for meal preparation times

5.7. Variation in the level of stochasticity only for future orders

To study the impact of stochasticity on the number of orders, we maintained consistent meal preparation times while introducing different levels of uncertainty in future order numbers. As illustrated in Figure 6, Surprisingly, the variation in future orders did not significantly influence the number of missed orders.

A closer analysis reveals an interesting phenomenon: a slightly higher influx of orders within shorter intervals might have resulted in the occasional missing of a few orders during peak rush periods. However, because the total number of orders remained constant, this brief rush was followed by relatively quieter periods. During quieter periods, all orders were successfully delivered because more time was available to handle these orders, given that they were spread out over a longer time frame.

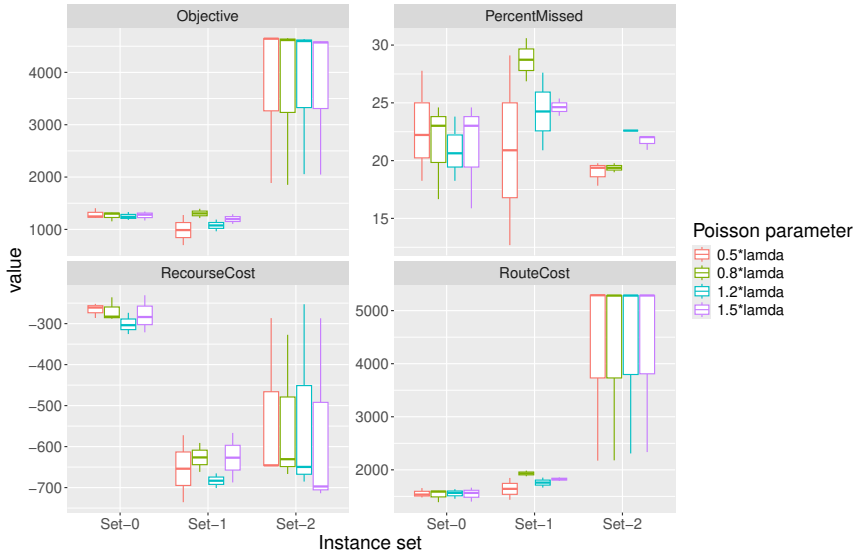

Figure 6: Comparison of solutions with variation in the level of stochasticity only for future orders

6. Conclusions

We introduced uncertainties in meal preparation times and future order locations as stochastic variables, emphasizing the crucial need to incorporate these uncertainties in route planning. Our approach involved employing a Sample Average Approximation method within a rolling horizon framework, utilizing the Adaptive Large Neighborhood Search algorithm for the first-stage problem, and implementing a recourse action in the second stage.

We made significant observations Through extensive experiments on instances derived from Grubhub MDRP instances. The utilization of variables, rather than expected values, resulted in notably profitable routes, primarily due to accommodating a larger number of orders. We explored scenarios where individual uncertainties were relaxed. Notably, the performance did not significantly improve when uncertainty was considered, only in meal preparation times and disregarding future orders. It was, in fact, 20% worse due to the absence of future order considerations. Conversely, when only future orders were regarded as uncertain and meal preparation was assumed known, performance significantly improved, displaying an average enhancement of over 30%.

Furthermore, our study delved into the intricate dynamics of operational uncertainties. We discovered that increased stochasticity could lead to more missed orders and escalated operational costs. Interestingly, occasional order misses were observed during peak demand periods, but these

were compensated for during quieter times. The effective management of meal preparation times emerged as a pivotal factor influencing order rejections, customer satisfaction, and overall operational efficiency. These findings underscore the necessity of adaptive strategies in balancing these trade-offs effectively, offering valuable insights for decision-makers in the operational management domain.

Funding

This research received no specific grant from funding agencies in the public, commercial, or not-for-profit sectors.

References

Bianchi, L., Dorigo, M., Gambardella, L.M., Gutjahr, W.J., 2009. A survey on metaheuristics for stochastic combinatorial optimization. Natural Computing 8, 239–287. doi:10.1007/s11047-008-9098-4.

Chu, J.C., Yan, S., Huang, H.J., 2015. A multi-trip split-delivery vehicle routing problem with time windows for inventory replenishment under stochastic travel times. Networks and Spatial Economics 17, 41–68. URL: http://dx.doi.org/10.1007/s11067-015-9317-3, doi:10.1007/s11067-015-9317-3.

Fachini, R.F., Armentano, V.A., Toledo, F.M.B., 2022. A granular local search matheuristic for a heterogeneous fleet vehicle routing problem with stochastic travel times. Networks and Spatial Economics 22, 33–64. URL: http://dx.doi.org/10.1007/s11067-021-09553-6, doi:10.1007/s11067-021-09553-6.

Ghilas, V., Demir, E., Van Woensel, T., 2016a. An adaptive large neighborhood search heuristic for the pickup and delivery problem with time windows and scheduled lines. Computers & Operations Research 72, 12–30. doi:https://doi.org/10.1016/j.cor.2016.01.018.

Ghilas, V., Demir, E., Woensel, T.V., 2016b. A scenario-based planning for the pickup and delivery problem with time windows, scheduled lines and stochastic demands. Transportation Research Part B: Methodological 91, 34–51. doi:10.1016/j.trb.2016.04.015.

Gutjahr, W.J., 2011. Recent trends in metaheuristics for stochastic combinatorial optimization. Central European Journal of Computer Science 1, 58–66. doi:10.2478/s13537-011-0003-3.

Györgyi, P., Kis, T., 2019. A probabilistic approach to pickup and delivery problems with time window uncertainty. European Journal of Operational Research 274, 909–923. doi:10.1016/j.ejor.2018.10.031.

Hemmelmayr, V.C., Cordeau, J.F., Crainic, T.G., 2012. An adaptive large neighborhood search heuristic for two-echelon vehicle routing problems arising in city logistics. Computers & Operations Research 39, 3215–3228. doi:10.1016/j.cor.2012.04.007.

Karoonsoontawong, A., Punyim, P., Nueangnitnaraporn, W., Ratanavaraha, V., 2020. Multi-trip time-dependent vehicle routing problem with soft time windows and overtime constraints. Networks and Spatial Economics 20, 549–598. URL: http://dx.doi.org/10.1007/s11067-019-09492-3, doi:10.1007/s11067-019-09492-3.

Kleywegt, A.J., Shapiro, A., Homem-de Mello, T., 2002. The sample average approximation method for stochastic discrete optimization. SIAM Journal on Optimization 12, 479–502. doi:10.1137/S1052623499363220.

Laporte, G., Louveaux, F.V., van Hamme, L., 2002. An integer l-shaped algorithm for the capacitated vehicle routing problem with stochastic demands. Operations Research 50, 415–423.

Li, Y., Chen, H., Prins, C., 2016. Adaptive large neighborhood search for the pickup and delivery problem with time windows, profits, and reserved requests. European Journal of Operational Research 252, 27–38. doi:10.1016/j.ejor.2015.12.032.

Liao, W., Zhang, L., Wei, Z., 2020. Multi-objective green meal delivery routing problem based on a two-stage solution strategy. Journal of Cleaner Production 258, 120627. doi:10.1016/j.jclepro.2020.120627.

Liu, F.T., Ting, K.M., Zhou, Z.H., 2008. Isolation forest, in: 2008 Eighth IEEE International Conference on Data Mining, pp. 413–422. doi:10.1109/ICDM.2008.17.

Liu, Y., 2019. An optimization-driven dynamic vehicle routing algorithm for on-demand meal delivery using drones. Computers & Operations Research 111, 1–20. doi:10.1016/j.cor.2019.05.024.

Powell, W.B., Sheffi, Y., Nickerson, K.S., Butterbaugh, K., Atherton, S., 1988. Maximizing profits for north american van lines' truckload division: A new framework for pricing and operations. Interfaces 18, 21–41. doi:10.1287/inte.18.1.21.

Reyes, D., Erera, A.L., Savelsbergh, M.W.P., Sahasrabudhe, S., O 'Neil, R.J., 2018. The Meal Delivery Routing Problem. Optimization Online , 1–70.

Ropke, S., Pisinger, D., 2006. An adaptive large neighborhood search heuristic for the pickup and delivery problem with time windows. Transportation science 40, 455–472.

Secomandi, N., Margot, F., 2009. Reoptimization approaches for the vehicle-routing problem with stochastic demands. Operations Research 57, 214–230. doi:10.1287/opre.1080.0520.

Shi, Y., Boudouh, T., Grunder, O., Wang, D., 2018. Modeling and solving simultaneous delivery and pick-up problem with stochastic travel and service times in home health care. Expert Systems with Applications 102, 218–233. doi:10.1016/j.eswa.2018.02.025.

Steever, Z., Karwan, M., Murray, C., 2019. Dynamic courier routing for a food delivery service. Computers & Operations Research 107, 173–188. doi:10.1016/j.cor.2019.03.008.

Ulmer, M.W., Thomas, B.W., Campbell, A.M., Woyak, N., 2021. The restaurant meal delivery problem: Dynamic pickup and delivery with deadlines and random ready times. Transportation Science 55, 75–100. doi:10.1287/TRSC.2020.1000.

Verweij, B., Ahmed, S., Kleywegt, A.J., Nemhauser, G., Shapiro, A., 2003. The sample average approximation method applied to stochastic routing problems: A computational study. Computational Optimization and Applications 24, 289–333. doi:10.1023/A:1021814225969.

Wang, Z., Dessouky, M., Van Woensel, T., Ioannou, P., 2023. Pickup and delivery problem with hard time windows considering stochastic and time-dependent travel times. EURO Journal on Transportation and Logistics 12, 100099. doi:10.1016/j.ejtl.2022.100099.

Yildiz, B., Savelsbergh, M., 2019. Provably High-quality solutions for the meal delivery routing problem. Transportation Science 53, 1372–1388. doi:10.1287/trsc.2018.0887.

Zhang, J., Luo, K., Florio, A.M., Van Woensel, T., 2023. Solving large-scale dynamic vehicle routing problems with stochastic requests. European Journal of Operational Research 306, 596–614. doi:10.1016/j.ejor.2022.07.015.

Zheng, J., Wang, L., Wang, L., Wang, S., Chen, J.F., Wang, X., 2023. Solving stochastic online food delivery problem via iterated greedy algorithm with decomposition-based strategy. IEEE Transactions on Systems, Man, and Cybernetics: Systems 53, 957–969. doi:10.1109/TSMC.2022.3189771.

Zhu, L., Sheu, J.B., 2018a. Failure-specific cooperative recourse strategy for simultaneous pickup and delivery problem with stochastic demands. European Journal of Operational Research 271, 896–912. doi:https://doi.org/10.1016/j.ejor.2018.05.049.

Zhu, L., Sheu, J.B., 2018b. Failure-specific cooperative recourse strategy for simultaneous pickup and delivery problem with stochastic demands. European Journal of Operational Research 271, 896–912. doi:10.1016/j.ejor.2018.05.049.