

Delivery Deadlines in Same-Day Delivery

Marlin Ulmer¹

Received: 29 March 2017 / Accepted: 19 September 2017 / Published online: 9 November 2017
© The Author(s) 2017 This article is published with Open Access at www.bvl.de/lore

ABSTRACT

Delivery companies offer same-day delivery (SDD) in more and more cities. SDD defines a business model in which a customer order is fulfilled the same day it is issued. To gain a competitive advantage, companies start offering increasingly narrower delivery deadlines of a few hours length. Currently, many companies struggle to cost-efficiently maintain a network of depots and vehicles to deliver the orders within the promised delivery deadlines. In this paper, we quantify the impact of delivery deadlines on the delivery costs in a simulation study. We further analyze the impact of delivery deadlines to the design of the delivery network. To this end, we extend an existing dynamic routing method from the literature. In our computational case study of the Iowa City area, we show that the costs and the network-layout for SDD significantly depend on the delivery deadlines and that an extension of the deadline by only one hour may reduce delivery costs substantially.

Keywords Same-Day Delivery · Stochastic Orders · Dynamic Vehicle Routing · Deadlines

 Marlin Ulmer
m.ulmer@tu-braunschweig.de

¹ Assistant Professor, Technische Universität Braunschweig
Carl-Friedrich-Gauß-Fakultät,
Institut für Wirtschaftsinformatik
Mühlenpfordtstraße 23,
38106 Braunschweig, Germany

1. INTRODUCTION

Recent growth in same-day delivery (SDD) has led to significant challenges for logistics service providers and to an increasing stream of research. In SDD, customers order goods online and expect a delivery within the day at low or no costs (Wahba 2015). To reduce costs and delivery times, major e-commerce companies have started to acquire significant parts of the delivery supply-chain (Webb 2016, Rao 2015, Shao and Ruwitsch 2016). In the final step of these supply-chains, the goods are delivered from

warehouses in the city to the customers. This last-mile delivery within the city is highly expensive and comprises more than half of the entire delivery costs (Bernau et al. 2016). Thus, service providers struggle to perform SDD cost-efficiently (Howland 2016).

One major driver of the high last-mile delivery costs is that deliveries can usually not be combined to large batches and customers need to be served individually. This problem is amplified for SDD because of two reasons: First, customers order during the day with delivery vehicles already on the road. To serve these orders, a vehicle needs to return to the depot several times over the day (Voccia et al. 2017). Thus, on each trip, only a few customers are served. Another, even more challenging reason for high delivery costs in SDD is that delivery deadlines are promised to the customers (Ram 2015). Delivery deadlines are a powerful tool to bridge the gap of instant gratification compared to brick and mortar shopping (Anderson 2015). Thus, narrow delivery deadlines have become an important marketing instrument recently. In Berlin, for example, Amazon started in spring 2016 to offer 2 hours delivery deadlines (Birger 2016). One month later, a local service provider decreased delivery deadlines to 90 minutes (Grösch 2016). Another month later, Amazon reacted by offering 60 minutes delivery deadlines (Fuest et al. 2016).

Still, narrow deadlines come at a price. It has become common knowledge that temporal commitments in delivery routing can increase service costs significantly. Punakivi and Saranen (2001) and Ehmke and Campbell (2014) show how time-windows result in higher routing costs and/or a lower number of customers served. The purpose of this research is to quantify the impact of delivery deadlines on the costs of SDD and the layout of the SDD-network. To this end, we present a simulation study for the same-day delivery problem with soft deadlines (SDDPSD), a common SDD business model. In the SDDPSD, a fleet of vehicles and depots are given. The vehicles start and end their tours at one of the depots. During an order phase, customers order goods stochastically at arbitrary locations in the service area. Each order is picked up at one of the depots and dynamically delivered to the customer by a vehicle. Each delivery has a deadline defined by the

time of the order and the allowed time for delivery. In case the order is not delivered before this deadline, the customer experiences a delay. The objective of the dispatcher is to utilize the given resources to minimize the expected average delay per customer delivery.

For the SDDPSD, we analyze how deadlines impact the delay per customer. We further analyze how many orders can be served by a limited fleet given a specific deadline and determine the numbers of depots and/or vehicles as well as the amounts of working time required to fulfill a certain service level without significant delay. To this end, we extend a dynamic routing policy by Ulmer et al. (2016a) to the specification of the SDDPSD capturing assignment decisions, multiple depots, multiple vehicles, and deadlines. For a real-world case study of Iowa City and a variety of instance settings, we derive the following managerial implications:

1. Delivery deadlines significantly impact the number of required vehicles and working time to satisfy a certain service level. A delivery deadline of 60 minutes may require more than twice as many vehicles and working times compared to a delivery deadline of 3 or 4 hours.
2. Different delivery deadlines require a different layout of the delivery network. More specific, narrow deadlines demand for a higher density of depots in the city.
3. One key for more efficient SDD is the increase in the number of orders per day. This allows consolidation and reduces the delivery costs per customer significantly.

Our contribution is as follows. With the SDDPSD, we analyze an important stochastic dynamic vehicle routing problem of high practical relevance. We present a new and comprehensive route-based Markov decision process (MDP) model formulation capturing the dynamism and the stochasticity of the problem. For the SDDPSD, we determine a runtime-efficient dynamic assignment and routing directly operating on the MDP's decision space. We analyze our methods in a real-world case study. Our structured experimental design and analysis allow general statements about the impact of deadlines on network layout, routing costs, and fleet size. Finally, the SDDPSD is a promising starting point for a variety of future research directions.

This paper is outlined as follows. We present the related literature in §2. In §3, we define the SDDPSD and model it as Markov decision process. In §4, we describe the dynamic assignment and routing policy. We present the case study for Iowa City and analyze the results in §5. This paper concludes with a summary and an outlook in §6.

2. LITERATURE REVIEW

In this section, we present the related literature. Our work analyzes the impact of temporal commitment in a large-scale same-day delivery routing environment and has not yet been studied in the literature. The only work directly related to the SDDPSD is presented by Voccia et al. (2017). They consider a same-day delivery problem with hard deadlines and/or TWs. In contrast to the SDDPSD, they allow the rejection of customers. To determine assignment and routing decisions, they apply the multiple-scenario approach (MSA) by Bent and Van Hentenryck (2004). In each decision point, the MSA samples a set of future order requests and determines suitable plans based on these samples. Voccia et al. (2017) experience average runtimes per decision point of several minutes for instance settings with 3 vehicles and 96 expected orders. A transfer of the MSA to real-world size instances of the case study is therefore challenging because we often observe multiple orders per minute. In their work, Voccia et al. (2017) also analyze the impact of delivery deadlines and TWs on the algorithm's performance. They observe that evenly spread TWs allow more services while TWs late in the day allow more flexibility and a better anticipation. Other works either consider same-day delivery or the impact of temporal commitments such as time-windows (TWs) or delivery deadlines on routing costs and/or the number of served customers. In the following, we present relevant literature from these two domains.

2.1 Same-Day Delivery

Same-day delivery problems combine dynamic vehicle routing problems with stochastic order requests and depot returns. For a general overview on dynamic vehicle routing, the interested reader is referred to Ritzinger et al. (2015). The work on SDD-problems is limited. Work on SDD with time-windows is presented by Azi et al. (2012). In their problem, customers request during the day and are either accepted or rejected. They apply an MSA for instance settings up to 5 vehicles and 144 customer requests. The time-windows are uniformly distributed over the horizon and 60 minutes long. Other work on same-day delivery is presented by Klapp et al. (2016) and Ulmer et al. (2016a). Both papers present single-vehicle routing without deadlines. The problem presented in Klapp et al. (2016) considers customers and depot on a line. They apply an a-priori policy on a rolling horizon. This policy is determined by mixed-integer programming. Ulmer et al. (2016a) analyze the impact of preemptive depot returns on the potential to serve customers. To determine which customers to serve and when to return to the depot, they use value function approximation. To allow preemptive returns, they introduce a dynamic routing policy based on insertion methods. In this paper, we

extend this policy incorporating assignments, multiple depots, and delivery deadlines. Another paper related to the SDDPSD is presented in Ghiani et al. (2009). The authors consider a courier service problem, where a fleet of capacitated vehicles performs one-to-one pickup and deliveries within a city. Each request has a deadline of 45 minutes. They use an online sampling procedure to analyze the decision's impact on the near future requests.

2.2 Impact of Temporal Commitment

The impact of temporal commitments is analyzed in many publications considering routing problems with TWs. In the following, we present three examples from the literature. Punakivi and Saranen (2001) compare (static) delivery routing with and without TWs. They show that the use of 120 minutes TWs increases delivery costs by up to 50%. Campbell and Savelsbergh (2005) present a customer acceptance problem for attended home delivery. Customers request dynamically in a capture phase and can either be accepted or rejected. Each customer has a TW. Deliveries take place in a separate delivery phase. They show that 120 minutes TWs increase profits by more than 6% in comparison to 60 minutes TWs. Ehmke and Campbell (2014) analyze a similar customer acceptance problem but additionally with stochastic travel times. The objective is to determine customer acceptance decisions avoiding TW violations. Ehmke and Campbell (2014) vary the length of TWs and state that "longer service time windows enable the acceptance of a significantly larger number of requests." More specific, extending TWs from 30 minutes to 120 minutes leads to an increase in services of more than 15%.

3. PROBLEM DEFINITION

In the following, we define the same-day delivery problem with soft deadlines (SDDPSD). Notably, we assume the number and locations of depots as well as the vehicle fleet as externally given. We briefly describe the problem and then present the Markov decision process model for the SDDPSD.

3.1 Problem Statement

In this section, we describe the SDDPSD and introduce the according notation. The required notation is listed in Table 1. A fleet of m vehicles $V = \{v_1, \dots, v_m\}$ delivers goods from a set of depots $\mathbf{D} = \{D_1, \dots, D_n\}$ to a set of ordering customers \mathbf{C} in a service area A . The vehicles start and end their tours in the first depot D_1 . The orders occur during an order phase $T = [0, t_{max}]$. Each customer $C \in \mathbf{C}$ orders at a point of time $t(C) \in T$ and is unknown beforehand. The point of time and location of the customer's order follows a spatial-temporal probability distribution. The loading status of a customer $l(C)$ is initially not loaded with

$l(C) = 0$. The orders can be picked up at any depot D from the overall set of depots \mathbf{D} . In case the according good is picked up at a depot, the loading status changes to loaded, $l(C) = 1$. The travel durations between customers and/or the depot are determined by function $d(\cdot, \cdot)$. The loading time at the depot is t_D . The service time at a customer is t_s . The allowed time for delivery is τ . The delivery deadline $\delta(C)$ of customer C is therefore $\delta(C) = t(C) + \tau$. Over the day, the dispatcher maintains and updates a planned route $\theta(v)$ for each vehicle v . A planned route θ is a sequence of h customers C_i^θ and p depot visits $D_{d_j}^\theta$ with according arrival times $a(C_i^\theta)$ and $a(D_{d_j}^\theta)$ ending in the depot:

$$\theta = ((C_1^\theta, a(C_1^\theta)), \dots, (C_l^\theta, a(C_l^\theta)), (D_{d_1}^\theta, a(D_{d_1}^\theta)), (C_{l+1}^\theta, a(C_{l+1}^\theta)), \dots, (C_h^\theta, a(C_h^\theta)), (D_{d_p}^\theta, a(D_{d_p}^\theta))).$$

The first entry of a route θ represents the current location or the next location a vehicle will visit, in case it is currently on the road. A route may therefore also start in the depot. Because drivers and dispatcher usually communicate via mobile phone, diversion from the current destination is not permitted. Notation C_i^θ indicates the i^{th} customer in tour θ . The index d_j in D_{d_j} indicates the visit of Depot $D_{d_j} \in \mathbf{D}$. Index $j = 1$ represents the first planned visit of a depot. Index $j = p$ represents the last planned visit. The last depot visited is D_1 and $d_p = 1$. The routing θ is the overall set of routes, $\theta = (\theta(v_1), \dots, \theta(v_m))$. At time of a new order, the provider determines a new routing θ^x to integrate the new order. After the update, the vehicles proceed with the routing until the next customer orders. Each routing can be evaluated with respect to its planned delay $\Delta(\theta)$ as follows:

$$\Delta(\theta) = \sum_{v \in V} \sum_{C \in \theta(v)} \max(0, a(C) - \delta(C)).$$

The planned delay is the sum of the individual delays per customer C , calculated by the difference between arrival time and delivery deadline, $\max(0, a(C) - \delta(C))$. The objective of the SDDPSD is to minimize the expected sum of delay over all customers.

3.2 Route-Based Markov Decision Process

In the following, we model the SDDPSD as a Markov decision process (MDP). To incorporate the routing component, we draw on a route-based MDP formulation (Ulmer et al. 2016b). A route-based MDP extends a conventional MDP by adding planned routes in the state and decision space. The use of a route-based MDP enables our heuristic to operate directly on the routing-component of the MDP. A decision point $k = 0, \dots, K$ occurs when a customer orders delivery. The number of decision points K is therefore a random variable.

Table 1: Problem Notation

Description	Notation
Service area	\mathcal{A}
Depots	$\mathcal{D} = \{D_1, \dots, D_n\}$
Vehicles	$\mathcal{V} = \{v_1, \dots, v_m\}$
Customers	\mathcal{C}
Order phase	$T = [0, t_{\max}]$
Order time of C	$t(C) \in T$
Service time at customer	t_s
Loading time at depot	t_D
Loading status of C	$l(C) \in \{0, 1\}$
Travel time function	$d(C_1, C_2), C_1, C_2 \in \mathcal{C} \cup \{D\}$
Allowed delivery time	τ
Delivery deadline of C	$\delta(C)$
Current planned route of vehicle v	$\theta(v)$
Current arrival time at customer C	$a(C)$
Current planned routes	$\Theta = \{\theta(v_1), \dots, \theta(v_m)\}$
Planned delay of routing Θ	$\Delta(\Theta)$

A state S_k in a decision point k contains the point of time $t_k \in T$, the currently planned routes $\theta_k = (\theta_k(v_1), \dots, \theta_k(v_m))$, the customers including their delivery deadlines and their loading statuses summarized in

$$\mathbf{C}_k = ((C_k^1, l(C_k^1), \delta(C_k^1)), \dots, (C_k^n, l(C_k^n), \delta(C_k^n))).$$

A state further contains the new customer C_k . The overall state definition is therefore

$$S_k = (t_k, \theta_k, \mathbf{C}_k, C_k).$$

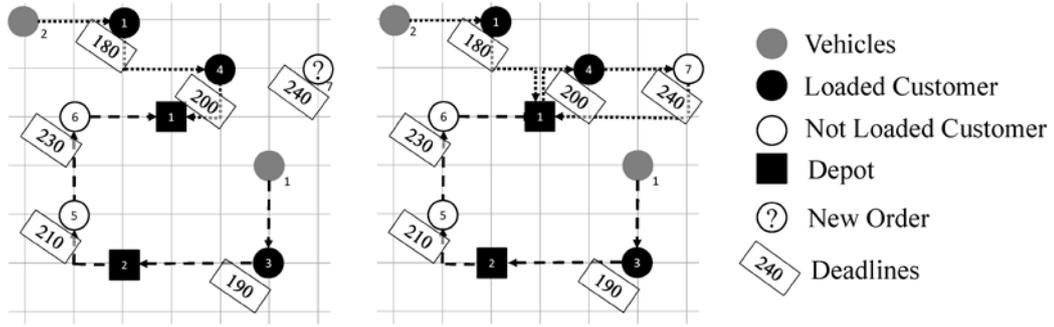
Decisions $x_k \in \mathbf{X}(S_k)$ are made about the routing update θ_k^x . An update is feasible if the following conditions hold: The first location and arrival time in the updated route are the same as in the original route for each vehicle meaning no diversion is allowed. It is feasible, if the set of routes contains each customer in C_k exactly once, the loaded customers are in the same route as before, and unloaded customers are only in sequences after a depot visit. Further, one route contains C_k . Finally, the arrival times reflect travel, service, and loading times. The costs R of a state S_k and a decision x_k are the marginal increase in delay, $R(S_k, x_k) = \Delta(\theta_k^x) - \Delta(\theta_k)$. After a decision is selected, a stochastic transition w_k leads to the next decision state. The transition consists of the proceeding with the routes θ_k^x until a new customer, C_{k+1} , orders at time t_{k+1} . The new state S_{k+1} contains an update of the routing θ_k^x and customers C_{k+1} as follows:

1. Services: All customers with arrival time $a(C) < t_{k+1}$ are removed.
2. Loading: If a route θ contains a depot visit with $a(D) < t_{k+1}$, this depot visit is removed and all customers in this route are set to loaded.
3. Idling: If a vehicle served all customers in the route, it idles at the depot. The updated route only contains the depot with arrival time $(D) = t_k, \theta = ((D, a(D)))$.

The initial state S_0 is in $t_0 = 0$ with $C_0 = \emptyset$, and $\theta_0(v) = (D_1, 0) \forall v \in V$. The first decision point $k = 1$ occurs in $t_1 = t(C_1)$ when the first customer C_1 orders. The termination state S_k is in $t_k = t_{\max}$. Due to the aforementioned restrictions of the plans, at that point of time, the routes only contain the depot, $\theta_k(v_i) = (D, t_{\max})$. A solution for the SDDPSD is a decision policy $\pi \in \Pi$, a sequence of decision rules $(X_0^\pi, \dots, X_K^\pi)$ mapping a state S_k to decision $x_k = X_k^\pi(S_k)$. The objective for the SDDPSD is to find a policy π^* minimizing the expected costs:

$$\pi^* = \operatorname{argmin}_{\pi \in \Pi} \mathbb{E} \left[\sum_{k=0}^K R(S_k, X_k^\pi(S_k)) | S_0 \right].$$

Figure 1: Example for the MDP Model



3.3 Example

In the following, we give an example of the MDP. Figure 1 shows a decision state on the left and a potential decision on the right. The current decision point is $k = 7$ where the 7th customer orders. The current point of time is $t_7 = 120$. For the purpose of presentation, we assume a Manhattan-style grid and travel times for each segment of 10 minutes. Loading times at the depot and service times at the customer are set to 10 minutes as well. The allowed time per delivery is $\tau = 120$ minutes.

The depots are represented by the black squares. The example depicts two depots $\mathbf{D} = \{D_1, D_2\}$. White and black circles indicate customers. White circles represent customers with goods not yet loaded to a vehicle. Black circles represent customers with goods already loaded to a vehicle. The deadline for each customer is depicted in the adjacent field. The set of customers \mathbf{C}_7 is therefore

$$\mathbf{C}_7 = ((C_1, 1, 180), (C_3, 1, 190), (C_4, 1, 200), (C_5, 1, 210), (C_6, 1, 230)).$$

Customer C_2 is not part of \mathbf{C}_7 . Thus, the customer was already served. The new order C_7 is represented by a white circle with a question mark. The deadline is $120 + 120 = 240$ minutes. For our example, we consider two vehicles indicated by the grey circles. Each vehicle has a planned route represented by the dashed and dotted lines. The first vehicle is on the way to serve Customer 3. The plan then determines a return to Depot 2 and service of Customers 5 and 6. Eventually, the vehicle returns to Depot 1. Mathematically, the plan is

$$\theta_7(v_1) = ((C_3, 140), (D_2, 180), (C_5, 210), (C_6, 240), (D_1, 270)).$$

The second vehicle is currently on its way to Customer 1. The current plan determines Vehicle 2 to then visit Customer 4 and to return to Depot 1 afterwards. The plan formulation for the second vehicle is therefore

$$\theta_7(v_2) = ((C_1, 140), (C_4, 180), (D_1, 210)).$$

The state is

$$S_7 = (t_7, (\theta_7(v_1), \theta_7(v_2)), \mathbf{C}_7, C_7).$$

The current delay of routing θ_7 is $\Delta(\theta_7) = 10$ minutes because the planned arrival $a(C_6) = 240$ at Customer 6 is 10 minutes after the deadline $\delta(C_6) = 230$.

Decisions are now made about the assignment of the customer and the according update of the routing. A potential decision x_7 is shown in the center panel of Figure 1. Decision x_7 assigns Customer 7 to Vehicle 2 and schedules a preemptive depot return before the visit of Customer 4. As later in our solution method, the decision maintains the sequence of existing customers and uses an insertion method to integrate C_7 and the depot return in the route of Vehicle 2, $\theta_7(v_2)$. Route $\theta_7(v_1)$ remains unaltered. The updated routing is

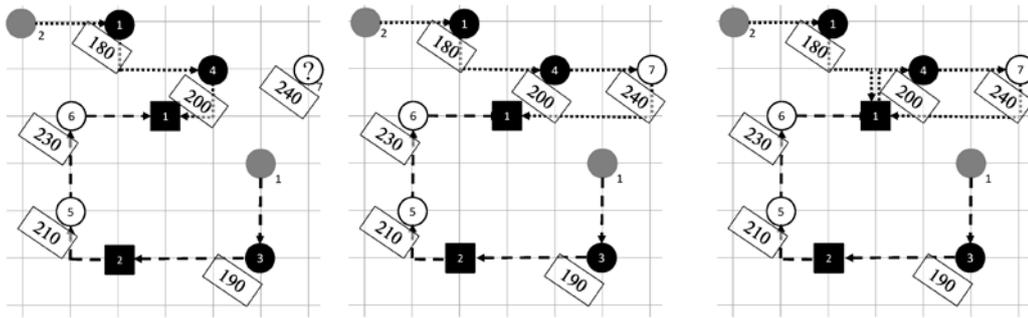
$$\theta_7^x = (\theta_7(v_1), ((C_1, 140), (D_1, 180), (C_4, 210), (C_7, 240), (D_1, 290))).$$

The delay of the routing update θ_7^x is $\Delta(\theta_7^x) = 20$ because Customer 4 is now visited 10 minutes after the deadline. The costs are ten: $R(S_7, x_7) = \Delta(\theta_7^x) - \Delta(\theta_7) = 20 - 10$. The vehicles now follow routing θ_7^x until a new customer orders delivery.

4. DYNAMIC ASSIGNMENT AND ROUTING POLICY

Decision making for the SDDPSD is complex because vehicles are required to return to the depot to load the good for a new order. Further, because vehicles are routed in real-time, the calculation time accessible for decision making is highly limited. We therefore apply an assignment and routing heuristic inspired by practice. The heuristic allows preemptive depot visits to pick up goods for new customers in case a depot is nearby. The original heuristic was introduced in Ulmer et al. (2016a). We extend the policy to integrate a fleet of vehicles, multiple depots, and delivery deadlines.

Figure 2: Example for the Dynamic Assignment and Routing Policy



4.1 Functionality

The main idea of the policy is to maintain the current routing and only change the routing for the vehicle the new order is assigned to. To this end, the policy iterates through the set of routes, hypothetically inserts the customer (and depot return), and measures the resulting costs, i.e., sum of marginal increase in delay for each customer in the route. The policy then selects the tour with minimal costs to insert the new customer. In case of several tours with identical costs, the algorithm selects the tour leading to the lowest marginal increase in tour duration.

The procedure is shown for the aforementioned example in Figure 2. On the left, the decision state is shown. For the example, we only consider Vehicle 2, but the algorithm conducts the identical procedure for Vehicle 1. First, the algorithm removes any depot return between the next location of the vehicle, Customer 1, and the final depot return. In the example, no such depot return was previously scheduled. Thus, no depot return is removed. In the next step, the new order, Customer 7, is inserted in the resulting route by means of cheapest insertion. This is shown in the center of Figure 2. Customer 7 is inserted between Customer 4 and the final depot return.

The resulting route is not feasible because a depot return must be scheduled before Customer 7 can be served. To this end, the algorithm iterates through all depots, hypothetically inserts a depot by means of cheapest insertion, and inserts the depot with minimal extension of the route. This is shown on the right of Figure 2. Depot 1 is inserted between Customer 1 and Customer 4. Notably, Vehicle 2 performs a preemptive return to the depot and serves Customer 4 afterwards even though the good for Customer 4 is already loaded on the vehicle.

From a route-based MDP-perspective, the heuristic works on a restricted policy class due to the insertion methods and selects the routing update leading to the minimal immediate costs. Thus, the policy operates on the first term of the (route-based) Bellman Equation and is therefore myopic with respect to future developments.

4.2 Algorithm

In Algorithm 1, we present the algorithmic procedure of our policy. The algorithm starts with a current routing θ and the new order C and provides an updated routing θ^* . The algorithm iterates through all routes and compares the minimal costs “mincosts” and minimal increase in duration “mininsert”, both set to a large number initially. For each route θ , the algorithm generates a candidate route $\hat{\theta}$ by means of insertion procedures. To this end, the algorithm first removes any depot return except the last one. Due to the structure of this algorithm, there is at most one depot return to remove. In the candidate route without the depot, the new customer is inserted by means of cheapest insertion with function $CheapestInsertion(\hat{\theta}, C)$. This means that the customer is inserted in the position of the tour leading to the smallest increase in travel duration (Rosenkrantz et al. 1974).

To allow feasibility, a depot return needs to be inserted in $\hat{\theta}$. This return must be scheduled before the first not loaded customer in the route. To this end, the algorithm determines the position of this customer by using the variable “loaded”. With function $DepotInsertion(\hat{\theta}, loaded)$, a depot $D \in \mathbf{D}$ is inserted in the subroute between next/current destination and first unloaded customer. To this end, the algorithm iterates through all depots and hypothetically inserts the depot in the subroute by means of cheapest insertion. The depot with the smallest extension of the tour duration is inserted. The delay of the resulting candidate route is then determined by $Delay(\hat{\theta})$ and compared to the already existing delay of tour θ . Analogously, the increase in tour duration is calculated by function $Duration(\hat{\theta})$. In case the costs are smaller than the existing solution or equal but the increase in duration is smaller, the best found solution is updated. The update function $Update(\theta, \theta, \hat{\theta})$ removes θ from the routing and replaces it with $\hat{\theta}$. Eventually, the algorithm returns the routing θ^* with smallest costs.

We implement the algorithm in Java. We run the tests on Windows Server 2008 R2, 64 bit, with Intel-Xeon E7-4830@2.13GHz, 64 cores, and 128GB RAM. The runtimes per decision point are neglectable.

Algorithm 1: Dynamic Routing and Assignment Policy

```

Input : Routing  $\Theta$ , new Order  $C$ 
Output : Updated Routing  $\Theta^x$ 
1 mincosts  $\leftarrow$  bigM // Minimal Costs
2 mininsert  $\leftarrow$  bigM // Minimal Marginal Duration Increase
3  $\Theta^x \leftarrow$  NaN // Best Routing
4 for all  $\theta \in \Theta$  // For All Routes
5 do
6    $\hat{\theta} \leftarrow \theta$  // Generate Candidate Route
7   depot  $\leftarrow$  -1 // Find First Depot Location
8   for all  $\hat{\theta}^i \in \hat{\theta}$  // For All Stops in  $\hat{\theta}$ 
9     do
10    if  $\hat{\theta}^i == D \wedge \text{depot} == -1$  // If First Depot
11      then depot  $\leftarrow$  i
12    end
13  end
14  if depot  $< |\hat{\theta}|$  // Except Last Depot Visit
15    then
16     $\hat{\theta} \leftarrow \text{Remove}(\hat{\theta}, \text{depot})$  // Remove Depot
17     $\hat{\theta} \leftarrow \text{CheapestInsertion}(\hat{\theta}, C)$  // Insert Customer
18    loaded  $\leftarrow |\hat{\theta}|$  // Find First Unloaded Customer
19    for all  $\hat{\theta}^i \in \hat{\theta}$  // For All Stops in  $\hat{\theta}$ 
20      do
21      if  $l(\hat{\theta}^i) == 0 \wedge \text{loaded} == |\hat{\theta}|$  // If First Unloaded
22        then loaded  $\leftarrow$  i
23      end
24    end
25     $\hat{\theta} \leftarrow \text{DepotInsertion}(\hat{\theta}, \text{loaded})$  // Insert Depot before First Not Loaded
      Customer
26    costs  $\leftarrow \text{Delay}(\hat{\theta}) - \text{Delay}(\theta)$  // Calculate Costs
27    insert  $\leftarrow \text{Duration}(\hat{\theta}) - \text{Duration}(\theta)$  // Calculate Duration Increase
28    if (costs  $<$  mincosts)  $\vee$  (costs == mincosts  $\wedge$  insert  $<$  mininsert) // Update
29      then
30      mincosts  $\leftarrow$  costs
31      mininsert  $\leftarrow$  insert
32       $\Theta^x \leftarrow \text{Update}(\Theta, \theta, \hat{\theta})$ 
33    end
34  end
35 end
36 return  $\Theta^x$ 

```

5. CASE STUDY

In this section, we present the computational evaluation for the Iowa City case study. We first define the instances differing in number of depots, deadlines, and number of orders in §5.1. We then define the service level in §5.2 and briefly describe

how suitable depot locations are determined in §5.3. In our analysis, we examine the following questions:

§5.4: How does the average delay per customer develop for different deadlines and a fixed number of depots, orders, and vehicles?

§5.5: How many orders can be served for a specific deadline with a sufficient service level, meaning services without “significant” delays?

§5.6: How many vehicles are required to achieve a sufficient service level per instance setting given a fixed number of expected orders?

§5.7: How much working time is required to achieve a sufficient service level given a fixed number of expected orders?

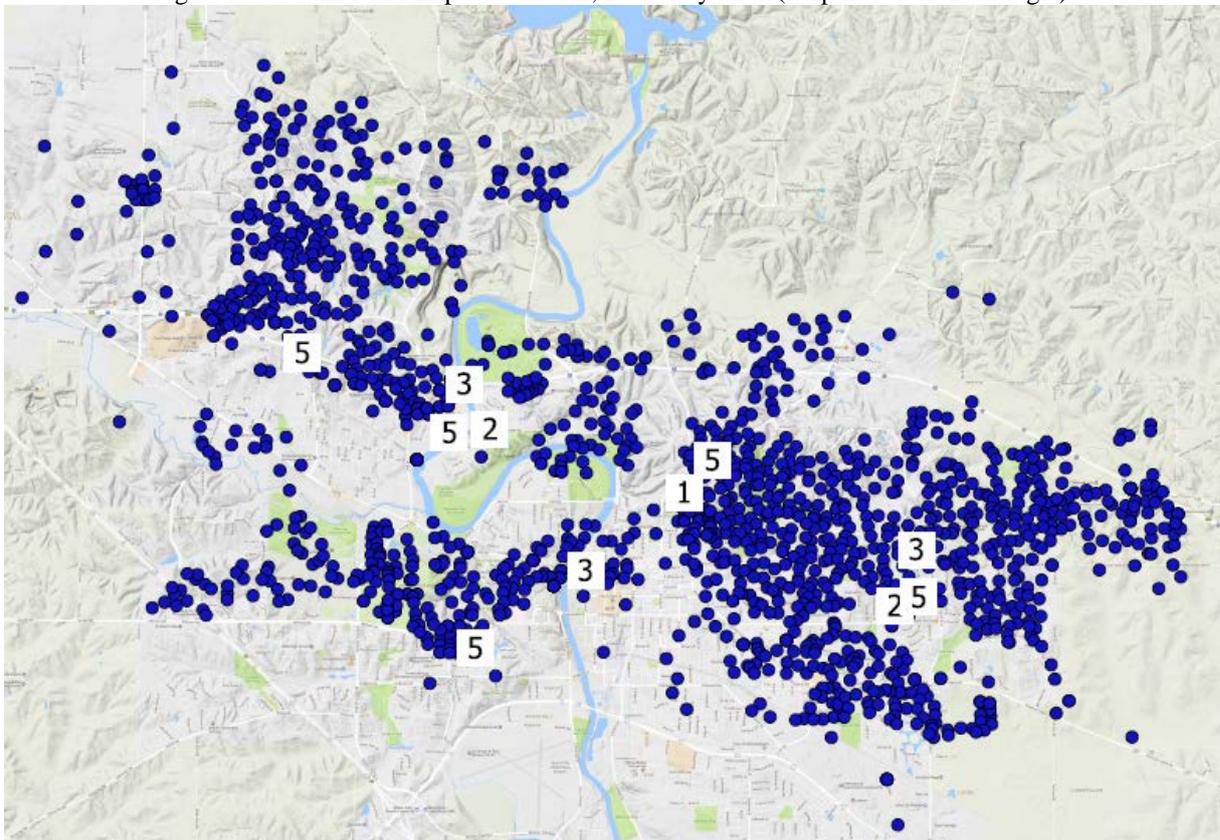
5.1 Instances

In the following, we describe the instance settings. We set the order time limit of the order phase to $t_{max} = 420$ minutes. All orders occur within this phase and are served. Thus, the working times per vehicle may exceed 420 minutes. We assume travel distances as Euclidean multiplied with factor 1.4 to represent a street network (Boscoe et al. 2012). The travel speed per vehicle is 25 km per hour. The service time at a customer is 2 minutes. The loading time at the depot is 10 minutes. We draw on 32840 potential customer locations as presented in Ulmer and Thomas (2017). The distribution of customer locations is depicted in Figure 3 showing the Iowa

City area. Each circle represents one customer. For the purpose of presentation, we only depict one out of 20 potential locations in Figure 3. The Iowa City area comprises a population of about 100,000 to 150,000 inhabitants and is therefore exemplary for a medium-sized US city. The area is split in three regions, Coralville in the northwest, University Heights in the southwest, and Downtown Iowa City in the east.

We generate instances with expected numbers of orders of 200, 300, 400, 500, and 800. For the customer orders, we assume uniformly order probabilities for each location and each point of time. To this end, we generate the overall number of orders with a Poisson-distribution based on the expected number of customers. For each customer, we then draw the order time in time interval $[0,420]$ and customer location with uniform probabilities. For each expected number of orders, we generate 1,000 realizations. We test different delivery deadlines of 30, 60, 90, 120, 180, and 240 minutes. We finally differentiate instances with a number of depots of 1, 2, 3, and 5. We describe the process of determining the depot locations in §5.3. Overall, this leads to 24 different settings. To allow a fair comparison, each setting draws on the identical 1,000 aforementioned realizations.

Figure 3: Customer and Depot Locations, Iowa City Area (Map data © 2017 Google)



5.2 Definition of a Sufficient Service Level

In the following, we describe the measure to quantify a sufficient service level. The measure indicates whether a solution avoids “significant” delays. Due to the stochastic and dynamic nature of this problem, a nearly infinite number of vehicles is required to avoid any delay. This is not suitable for practice. Thus, we determine a solution to be sufficient if the average delay per customer is less than 1 minute over the 1,000 realizations.

5.3 Determination of Depot Locations

To determine suitable depot locations, we consider 40 potential locations. These locations represent commercial houses in the Iowa City area. We determine the locations for each deadline by enumeration as described in the following:

For each number of depots (1,2,3,5), we enumerate the set of all potential subsets of depots. Each potential subset of depots is evaluated by the required number of vehicles as defined earlier. For evaluation, we generate additional 1,000 realizations. We select the subset of locations with the minimal number of vehicles V^{min} needed to reach the service level. To this end, we subsequently increase the number of vehicles and evaluate the delay until the threshold is met. In our computational study, we set the maximal number of vehicles to 100. Notably, for a deadline of 30 minutes and 1 or 2 depots, there were no feasible solutions with less than 100 vehicles.

Since the number of subsets scales fast, we only consider instance settings with 400 orders and use the same subset of depots for the other expected number of customers. The best depot locations for a deadline of 120 minutes are depicted in Figure 3. Still, the depot locations differ only slightly for different deadlines. In Figure 3, the numbers indicate to which subset the depot locations belong. As an example, the locations for instance settings with three depots are indicated with a “3”.

5.4 Average Delay

In this section, we analyze the delay for a specific instance setting. We choose the instance settings with an expected number of 400 orders, with 3 depots, and 10 vehicles. For this instance setting, we calculate the average delay per customer. The results are depicted in Figure 4. On the x-axis, the delivery deadline is shown. On the y-axis, the average delay per customer is depicted. We observe a constant increase in delay with narrower deadlines.

Notably, the increase is not linear but rather exponential. While the difference between 120 and 90 minutes is only 15 minutes, the difference between 60 and 30 minutes is 42. This difference is larger than the difference in deadlines. This indicates that the delay for 30 minutes deadlines may be smaller when the dispatcher follows the assignment and routing decision induced by the 60 minutes deadline. This can be explained by the myopic nature of the routing and assignment policy. This policy prioritizes the minimization of the current costs (or delay) and then, in case of similar delay, searches for the most efficient routing solution. In case of 30 minutes deadlines, the algorithm experiences delay in many routes and neglects potentially more efficient routing with higher immediate delay. This myopic decision then impedes the flexibility to integrate future orders efficiently. With larger deadlines, we observe more decision points, where an assignment to several vehicles is possible with zero delay and the algorithm is able to pick the most efficient routing. This provides more flexibility and results in more efficient future integrations of customers. In essence, this observation highlights that anticipation of future orders to maintain flexibility may be beneficial for the SDDPSD. Looking at the results in Figure 4, we observe that the delay for 240 minutes deadlines is below 1 minute.

Figure 4: Average Delay for 400 Orders and 10 Vehicles

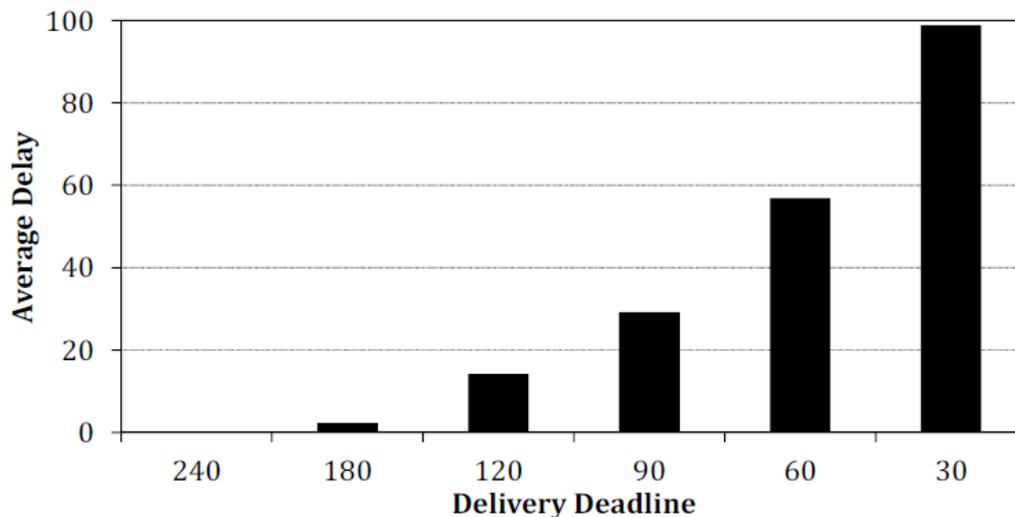


Table 2: Maximal Number of Orders Given a Limited Number of Vehicles

Vehicles	Deadlines				
	60	90	120	180	240
5	-	50	100	150	200
10	100	200	250	300	450
15	250	350	400	600	800

Thus, for this deadline, 10 vehicles are sufficient to satisfy the service level defined in §5.2. For business models with 30, 60, 90, 120, and 180 minutes delivery deadlines, the number of vehicles is not sufficient to meet the service level for 400 expected orders. For these deadlines, either the number of orders needs to be smaller or the number of vehicles needs to be larger. In the following two sections, we analyze both directions. First, we analyze how many orders can be served given a limited fleet. We then analyze how many vehicles are required given a specific number of expected orders.

5.5 Maximal Number of Orders given a Limited Fleet

In this section, we analyze how many orders we can serve with a specific number of vehicles. We select the instance settings with 3 depots and set the number of vehicles to 5, 10, and 15. We generate instance realizations for expected number of customers of 50, 100, 150, 200, etc. For each number of vehicles and deadline, we subsequently increase the number of orders by 50 until the average delay per customer exceeds one minute. For a delivery deadline of 30 minutes, not even 50 orders could be served without significant delay. Thus, we exclude this deadline from this analysis.

Table 2 shows the results for the number of vehicles and deadlines. We observe that the narrow deadlines decrease the number of services significantly. As an example, for 10 vehicles, a deadline of 4 hours allows

more than 4 times more services compared to a deadline of 60 minutes. Given 15 vehicles, a deadline of 240 allows service of twice as many customers as a deadline of 120 minutes. We also observe that longer deadlines require fewer vehicles to serve the same number of orders. As an example, for a deadline of 60 minutes, we require 15 vehicles to serve 250 customers while for a deadline of 120 minutes, we only require 10 vehicles to serve the same number of customers.

Based on these results, we finally analyze how the number of services per vehicle depends on the delivery deadlines. To this end, we calculate the average number of services per vehicle per deadline and number of vehicles. First, we analyze how many orders can be served given a limited fleet. We then analyze how many vehicles are required given a specific number of expected orders. The results are depicted in Figure 5. The x-axis depicts the number of vehicles, the y-axis the number of services per vehicle. Figure 5 indicates that with longer deadlines, significantly more customers can be served by a vehicle. As an example, for 10 vehicles, only 10 customers per vehicle can be served given deadlines of 60 minutes while 45 customers can be served by a vehicle given deadlines of 240 minutes. We further observe a scaling effect looking at the number of services with different fleet sizes. The more vehicles (and orders) we consider, the more services per vehicle are possible. We confirm both observations in the next section.

5.6 Required Number of Vehicles per Expected Number of Orders

In this section, we analyze how many vehicles are required for a specific number of orders, number of depots, and delivery deadline. To this end, we determine the minimal number of vehicles V^{\min} needed to satisfy the service level.

Figure 5: Average Number of Services per Vehicle

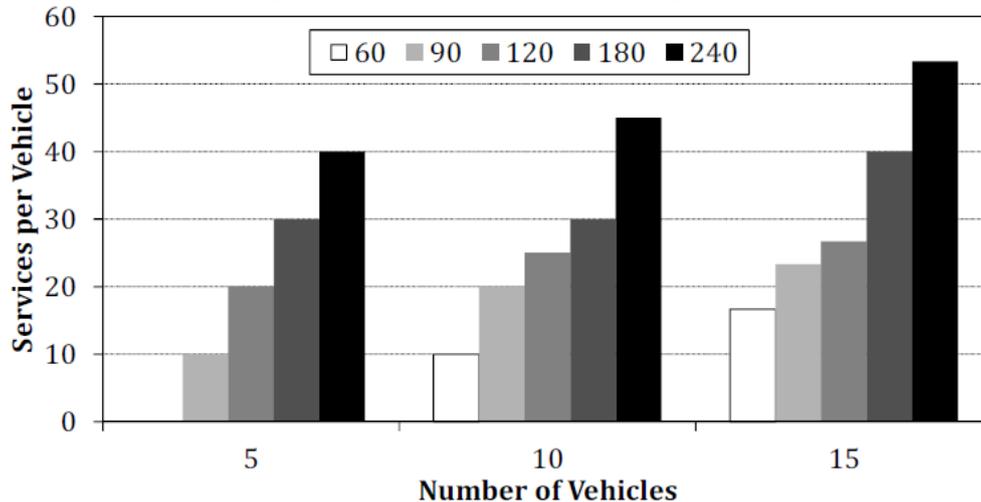


Figure 6: Required Number of Vehicles per Number of Orders, Number of Depots, and Delivery Deadline

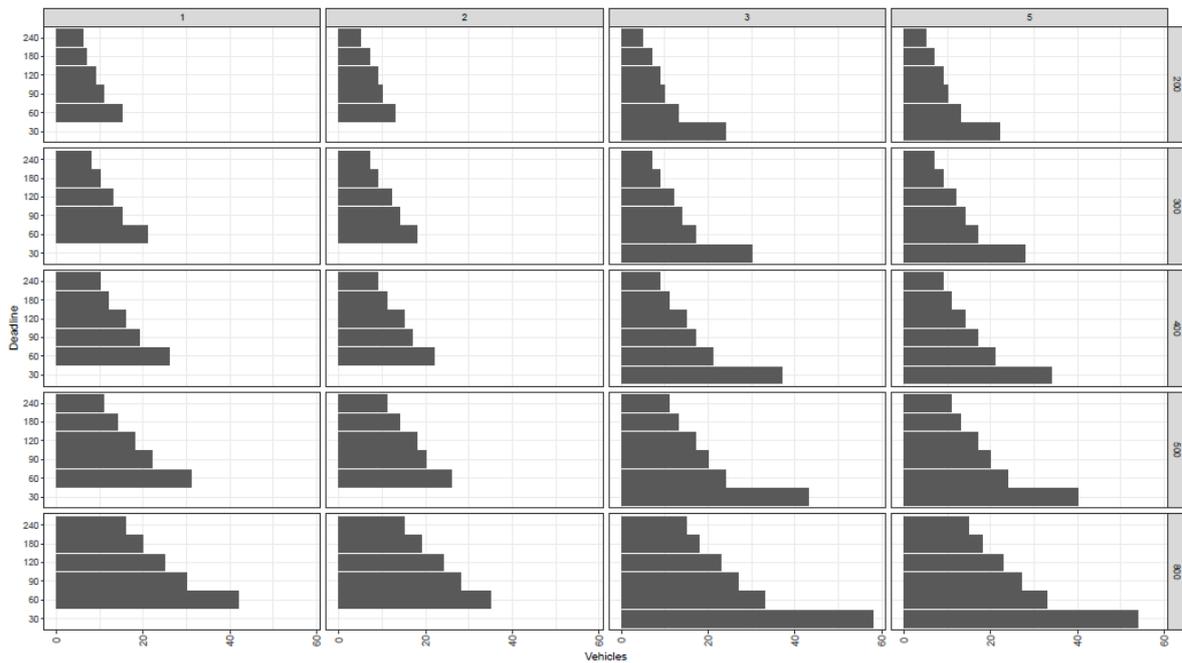


Figure 6 shows the results for the individual instance settings. On the global x-axis, the number of depots is depicted. The global y-axis shows the expected number of orders. The local y-axis in every subfigure depicts the deadlines. The local x-axis depicts the according number of required vehicles v^{\min} .

Generally, we observe tendencies with an increasing number of depots and an increasing number of orders. The more orders are expected, the higher is the required number of vehicles. Notably, we experience a consolidation effect meaning that the number of vehicles does not increase linearly with the number of orders. As an example, for 1 depot, a deadline of 120 minutes, and 200 orders, 9 vehicles are required while for the same setting and 800 orders, 25 vehicles are sufficient. Even though we experience 4 times more orders, less than 3 times the number of vehicles is required. This indicates that SDD becomes more efficient, the more customers order during the day. Another observation is that the number of vehicles decreases with an increasing number of depots. Still, for larger delivery deadlines, the difference is relatively small.

As an example, for a deadline of 120 minutes and 800 orders, the difference in vehicles is only $25-23=2$ considering 1 or 5 depots. The number of depots is more important for narrower delivery deadlines such as 30 or 60 minutes. In these settings, the addition of one depot replaces up to 7 vehicles. Furthermore, for a delivery deadline of 30 minutes, a single-depot layout may not even be feasible.

Comparing the delivery deadlines, we experience significant differences in the number of required vehicles. For the instance setting with 1 depot and

800 orders, a delivery deadline of 90 minutes requires $42-30=12$ (or about 30%) vehicles less than a delivery deadline of 60 minutes. A difference of only 30 minutes as discussed in the introduction may therefore result in a significant difference in delivery cost. Furthermore, for this instance setting, setting the delivery deadline to 3 hours requires less than half of the fleet size compared to a 60 minutes deadline. This tendency is even more distinct for delivery deadlines of 30 minutes. In essence, we observe that offering narrow delivery deadlines comes at a high price with respect to infrastructure and fleet size.

5.7 Required Working Times per Expected Number of Orders

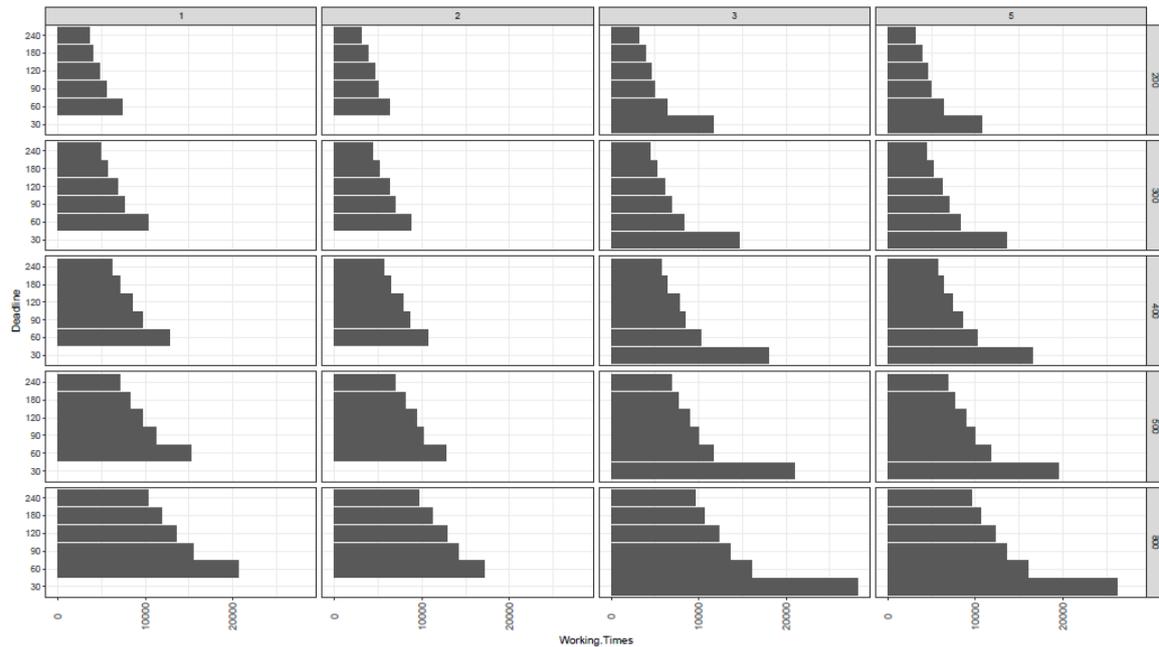
Finally, we analyze the according working times required to serve the customers. The working times reflect the sum of shift durations over the fleet. A vehicle's shift ends in case the following conditions hold:

1. The capture phase ended ($t \geq 420$).
2. No more orders are assigned to the vehicle.
3. The vehicle is located at one of the depots.

For deadlines of 4 hours, the last delivery may be conducted around 240 minutes after the end of the capture phase while for 60 minute deadlines, the last delivery may be conducted up to 60 minutes after the end of the capture phase. Longer deadlines may therefore lead to longer shifts compared to short deadlines.

The working times for the different deadlines are depicted in Figure 7. The individual working times per instance settings are shown in Table A2 in the Appendix. As depicted in Figure 7, we observe

Figure 7: Required Working Times per Number of Orders, Number of Depots, and Delivery Deadline



similar proportions as with the required number of vehicles but the differences are slightly smaller. For the aforementioned example of 1 depot and 800 orders, the delivery deadline of 90 minutes requires about 25% less in working times than deadlines of 60 minutes. A deadline of 3 hours requires only 57% and deadlines of 4 hours around 50% of the working time compared to 60 minutes deadlines. In essence, shorter deadlines require both a larger fleet and, in sum, longer working times.

6. CONCLUSION

Same-day delivery is a major trend in e-commerce. Recently, companies offer customers delivery within narrow delivery deadlines. In this paper, we have quantified the impact of delivery deadlines on the resulting routing costs for the stochastic and dynamic same-day delivery problem with soft deadlines. To this end, we modeled the SDDPSD as a Markov decision process and extended an existing dynamic assignment and routing policy. Our case study on the Iowa City area shows that narrow delivery deadlines lead to substantially higher costs. We have further shown that narrow delivery deadlines may require a different delivery network, mainly a higher density of depots within the city. Finally, we have observed that the efficiency of SDD increases with a larger number of orders and vehicles.

Future research may focus on developing solution methods for the SDDPSD or on extensions of the problem. As our analysis indicates, anticipation of future orders in routing and assignment decisions may be beneficial. Solution methods may incorporate

future orders by online simulations as discussed in Ghiani et al. (2009) or Voccia et al. (2017). Still, as aforementioned, the accessible runtime for the SDDPSD is short. Suitable methods may focus on transferring the majority of simulation effort in an offline learning phase by means of approximate dynamic programming. Developing these anticipatory policies may be challenging because of large fleet size, the complex state and action space, and the limited online runtimes. Future research may further focus on problems related to the SDDPSD. For example, suitable policies may be developed to control the depots' stocks and/or the transport flows between the depots.

Further, research may analyze the impact of pickup points where customers pick up their orders (Savelsbergh and Van Woensel 2016) as well as crowdsourced delivery, where a subset of orders is delivered by private drivers (Archetti et al. 2016). Another interesting research direction is to offer different delivery deadlines to different customers, e.g., by dynamic slotting or pricing mechanisms. Central customers, which may be integrated in the tour efficiently, may be offered narrower delivery deadlines than customers located in rural areas. Finally, the fleet may be complemented by drones to allow efficient service of isolated customers (Agatz et al. 2015). Here, the challenge may be to determine which customers are served by delivery vehicles and which are served by drones.

ACKNOWLEDGEMENT

The author likes to thank Ninja Söffker and Barrett Thomas as well as the Associate Editor and the two anonymous Referees for their valuable advice.

REFERENCES

- Agatz, Niels, Paul Bouman, Marie Schmidt. 2015. Optimization approaches for the traveling salesman problem with drone. Tech. rep.
- Anderson, Mae. 2015. Amazon is offering a key benefit of traditional brick-and-mortar stores. *Business Insider* URL <http://www.businessinsider.com/ap-everyone-loves-same-day-delivery-until-they-have-to-pay-2015-11?IR=T>. [Online; accessed 03-January-2017].
- Archetti, Claudia, Martin Savelsbergh, M Grazia Speranza. 2016. The vehicle routing problem with occasional drivers. *European Journal of Operational Research* 254(2) 472–480.
- Azi, Nabila, Michel Gendreau, Jean-Yves Potvin. 2012. A dynamic vehicle routing problem with multiple delivery routes. *Annals of Operations Research* 199(1) 103–112.
- Bent, Russell W, Pascal Van Hentenryck. 2004. Scenario-based planning for partially dynamic vehicle routing with stochastic customers. *Operations Research* 52(6) 977–987.
- Bernau, Varinia, Lea Hampel, Nils Wischmeyer. 2016. Her mit den Päckchen, aber bitte schnell und kostenlos. *Süddeutsche Zeitung* URL <http://www.sueddeutsche.de/wirtschaft/liefergesellschaft-alles-immer-bitte-sofort-1.3154867>. [Online; accessed 20-September-2016].
- Birger, Nicolai. 2016. Amazon liefert Pakete innerhalb von zwei Stunden. *Welt* URL <https://www.welt.de/wirtschaft/article153690106/Amazon-liefert-Pakete-innerhalb-von-zwei-Stunden.html>. [Online; accessed 06-December-2016].
- Boscoe, Francis P, Kevin A. Henry, Michael S Zdeb. 2012. Nationwide comparison of driving distance versus straight-line distance to hospitals. *The Professional Geographer* 64(2) 188–196.
- Campbell, Ann M, Martin WP Savelsbergh. 2005. Decision support for consumer direct grocery initiatives. *Transportation Science* 39(3) 313–327.
- Ehmke, Jan F, Ann M Campbell. 2014. Customer acceptance mechanisms for home deliveries in metropolitan areas. *European Journal of Operational Research* 233(1) 193–207.
- Fuest, Benedikt, Stephan Maaß, Max Zimmermann. 2016. So funktioniert das neue Ein-Stunden-Amazon. *Welt* URL <https://www.welt.de/wirtschaft/webwelt/article155266101/So-funktioniert-das-neue-Ein-Stunden-Amazon.html>. [Online; accessed 06-December-2016].
- Ghiani, Gianpaolo, Emanuele Manni, Antonella Quaranta, Chefi Triki. 2009. Anticipatory algorithms for same-day courier dispatching. *Transportation Research Part E: Logistics and Transportation Review* 45(1) 96–106.
- Grösch, Thilo. 2016. Lieferung in 90 Minuten: LocaFox launcht Same-Day-Delivery in Berlin. *LocaFox* URL <https://www.locafox.de/locafox-news/same-day-delivery-berlin/>. [Online; accessed 06-December-2016].
- Howland, Daphne. 2016. Costco, Nordstrom, UPS join researchers to tackle last mile delivery. *Retail DIVE* URL <http://www.retaildive.com/news/costco-nordstrom-ups-join-researchers-to-tackle-last-mile-delivery/428214/>. [Online; accessed 07-November-2016].
- Klapp, Mathias A, Alan L Erera, Alejandro Toriello. 2016. The one-dimensional dynamic dispatch waves problem. *Transportation Science*.
- Punakivi, Mikko, Juha Saranen. 2001. Identifying the success factors in grocery home delivery. *International Journal of Retail & Distribution Management* 29(4) 156–163.
- Ram, Aliya. 2015. Uk retailers face high cost of online deliveries. URL <http://www.ft.com/cms/s/0/516aa75a-a04c-11e5-beba-5e33e2b79e46.html#axzz4CPxX2fjJ>. [Online; accessed 23-June-2016].
- Rao, Leena. 2015. Amazon to roll out a fleet of branded trailer trucks. *Fortune* URL <http://fortune.com/2015/12/04/amazon-trucks/>. [Online; accessed 06-June-2016].
- Ritzinger, Ulrike, Jakob Puchinger, Richard F Hartl. 2015. A survey on dynamic and stochastic vehicle routing problems. *International Journal of Production Research* 1–17.
- Rosenkrantz, Daniel J, Richard E Stearns, Philip M Lewis. 1974. Approximate algorithms for the traveling salesperson problem. *Switching and Automata Theory, 1974., IEEE Conference Record of 15th Annual Symposium on. IEEE*, 33–42.
- Savelsbergh, Martin, Tom Van Woensel. 2016. 50th anniversary invited article—city logistics: Challenges and opportunities. *Transportation Science* 50(2) 579–590.

Shao, Xiaoyi, John Ruwitch. 2016. Alibaba’s logistics arm raises funds from Singapore investors. Reuters URL <http://www.reuters.com/article/us-alibaba-cainiao-idUSKCN0WG0AD>. [Online; accessed 01-June-2016].

Ulmer, Marlin W, Barrett W Thomas, Dirk C Mattfeld. 2016a. Preemptive depot returns for a dynamic same-day delivery problem. Submitted.

Ulmer, Marlin W, Justin C Goodson, Dirk C Mattfeld, Barrett W Thomas. 2016b. Route-based Markov decision processes for dynamic vehicle routing problems. Submitted.

Ulmer, Marlin W, Barrett W Thomas. 2017. Meso-Parametric Value Function Approximation for Dynamic Customer Acceptances in Delivery Routing. Submitted.

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

Wahba, Phil. 2015. Amazon waives same-day delivery fees for prime members. Fortune URL <http://fortune.com/2015/05/28/amazon-same-day-delivery-prime/>. [Online; accessed 27-August-2015].

Webb, Jonathan. 2016. Amazon And Alibaba Bet The Future On Supply Chain Management: eRetailers Invest Big In Logistics. Forbes URL <http://www.forbes.com/sites/jwebb/2016/05/31/amazon-and-alibaba-bet-the-future-on-supply-chain-management-eretailers-invest-big-in-logistics/>. [Online; accessed 01-June-2016].

APPENDIX

In the Appendix, we present the individual results. Table A1 shows the required number of vehicles per instance setting. Table A2 shows the average working time per instance setting in minutes.

Table A1: Required Number of Vehicles per Instance Setting

Orders	Depots	Deadlines					
		30	60	90	120	180	240
200	1	-	15	11	9	7	6
	2	-	13	10	9	7	5
	3	24	13	10	9	7	5
	5	22	13	10	9	7	5
300	1	-	21	15	13	10	8
	2	-	18	14	12	9	7
	3	30	17	14	12	9	7
	5	28	17	14	12	9	7
400	1	-	26	19	16	12	10
	2	-	22	17	15	11	9
	3	37	21	17	15	11	9
	5	34	21	17	14	11	9
500	1	-	31	22	18	14	11
	2	-	26	20	18	14	11
	3	43	24	20	17	13	11
	5	40	24	20	17	13	11
800	1	-	42	30	25	20	16
	2	-	35	28	24	19	15
	3	58	33	27	23	18	15
	5	54	33	27	23	18	15

Table A2: Average Working Time per Instance Setting in Minutes

Orders	Depots	Deadlines					
		30	60	90	120	180	240
200	1	-	7342	5538	4753	4006	3585
	2	-	6332	4996	4611	3870	3117
	3	11666	6324	4973	4577	3864	3135
	5	10693	6323	4968	4575	3841	3102
300	1	-	10275	7586	6794	5694	4912
	2	-	8762	6972	6236	5173	4404
	3	14581	8273	6941	6175	5163	4420
	5	13608	8273	6932	6176	5128	4380
400	1	-	12743	9618	8451	7031	6233
	2	-	10718	8547	7845	6445	5686
	3	17984	10221	8490	7764	6431	5695
	5	16526	10220	8482	7414	6385	5651
500	1	-	15200	11233	9717	8301	7115
	2	-	12669	10099	9402	8054	6933
	3	20899	11701	10022	8981	7659	6936
	5	19441	11696	10007	8975	7609	6888
800	1	-	20658	15428	13555	11847	10307
	2	-	17076	14129	12823	11146	9668
	3	28190	16084	13637	12304	10687	9640
	5	26245	16079	13613	12298	10630	9585