# The Driver-Aide Problem: Coordinated Logistics for Last-Mile Delivery

S. Raghavan
Robert H. Smith School of Business and Institute for Systems Research,
University of Maryland, College Park, Maryland 20742, USA, raghavan@umd.edu

Rui Zhang
Leeds School of Business, University of Colorado Boulder, Colorado 80309, USA, rui.zhang@colorado.edu

**Problem definition:** Last-mile delivery is a critical component of logistics networks, accounting for approximately 30-35% of costs. As delivery volumes have increased, truck route times have become unsustainably long. To address this issue, many logistics companies, including FedEx and UPS, have resorted to using a "Driver-Aide" to assist with deliveries. The aide can assist the driver in two ways. As a "Jumper", the aide works with the driver in preparing and delivering packages, thus reducing the service time at a given stop. As a "Helper", the aide can independently work at a location delivering packages, while the driver leaves to deliver packages at other locations and then returns. Given a set of delivery locations, travel times, service times, the jumper's savings and the helper's service times, the goal is to determine both the delivery route and the most effective way to use the aide (e.g., sometimes as a jumper and sometimes as a helper) to minimize the total delivery time. **Methodology/results:** We model this problem as an integer program with an exponential number of variables and an exponential number of constraints, and propose a branch-cut-and-price approach for solving it. Our computational experiments are based on simulated instances built on real-world data provided by an industrial partner and a dataset released by Amazon. More importantly, our results characterize the conditions under which this novel operation mode can lead to significant savings in terms of both the routing time and cost. **Managerial implications:** Our computational results show that the driver-aide with both jumper and helper modes is most effective when there are denser service regions and when the truck's speed is higher ($\geq$ 10 MPH). Coupled with an economic analysis, we come up with rules of thumb (that have close to 100% accuracy) to predict whether to use the aide, and in which mode. Empirically, we find that the service delivery routes with greater than 50% of the time devoted to delivery (as opposed to driving) are the ones that provide the greatest benefit. These routes are characterized by a high density of delivery locations.

*Key words*: Branch-Cut-and-Price, Last-Mile Delivery, Driver-Aide, Logistics, Service Delivery

## 1. Introduction

Over the last decade, consumer purchasing behavior has increasingly shifted online. According to U.S. Census Bureau data (U.S. Census Bureau 2022), e-commerce sales jumped by nearly 32 percent in 2020, compared to the prior year. The COVID-19 health crisis has accelerated this shift (United Nation 2021), triggering significant pressures on global supply chains. The result has been an unprecedented growth in last-mile package delivery (Younan 2020, Roberson 2020). Last-mile delivery is a critical component of logistics networks, accounting for approximately 30-35% of costs

(Rosenberger 2022). Thus, efficient management of last-mile delivery plays a key role in lowering overall supply chain costs.

As delivery volumes have increased, truck route times have become unsustainably long. In the short term, these can be addressed to some extent with the use of overtime. From a long-term perspective, however, this is undesirable due to several factors, including costs, pushing against labor/government rules on the workday length, and employee burnout, etc. One way to address the problem is to increase the number of trucks (and drivers) to ensure that the route times fall within acceptable norms. With the shortage of both delivery trucks (PYMNT 2020) and skilled labor (Ngo and Swanson 2021, Elite Extra 2022), as well as the fact that all logistics companies are attempting to hire qualified drivers, this is not a viable or cost-effective strategy. Alternatively, to address this issue, many logistics companies, including FedEx and UPS, have resorted to using a "Driver-Aide" to assist with deliveries. When a truck is equipped with both a "driver" and an "aide", the aide can assist the driver to help shorten the delivery time and reduce the overall route time. As the cost of a driver-aide is significantly lower than that of a driver, this enables the logistics company to shorten the overall duration of the route to bring it within acceptable norms without the need for additional trucks and drivers (we note that as the average package size has become smaller, truck capacity is no longer seen as a limiting constraint; thus, the limiting factor is the route duration).

In the industry, the driver-aide can be used to assist the driver in two ways. As a "Jumper", the aide works with the driver in preparing and delivering packages, thereby reducing the service time at a given stop. This is what is typically seen in practice. We note that a reduction in service time can vary from stop to stop. Depending on the location (e.g., a building with special entry requirements), the reduction may be limited or nonexistent. As a "Helper", the aide can independently work at a stop delivering packages, while the driver leaves to deliver packages at other stops and then returns. This is not as prevalent in practice, as it requires coordination between the driver and the aide. Moreover, it is not clear to logistics companies how best to leverage this capability. We should note that each delivery stop corresponds to multiple customer locations (e.g., all at the same building or in close geographic proximity). The aide can move between these customer locations within a stop on foot or by using a light vehicle (e.g., e-scooter or e-bike). Among many alternatives under consideration, a plausible one is an e-bike. E-bikes have been adopted for deliveries in many European countries (e.g., the U.K., Germany, and Estonia) and are actively being tested in the U.S. by the USPS (Toll 2022), Amazon (Sutton 2022), FedEx (FedEx 2020), and UPS (CBSNews 2022) as delivery options. Conveniently, an e-bike can be attached to the delivery truck when the aide is used in the jumper mode, while it can be detached from the delivery truck when the aide is in the helper mode.

In this paper, we consider the logistics company's problem of how best to utilize the driver-aide. Specifically, given a set of delivery locations, travel times, service times, the jumper's savings, and the helper's service time, the goal is to determine both the delivery route and the most effective way to use the aide (e.g., sometimes as a jumper and sometimes as a helper) to minimize the total delivery time. This problem is referred to as the "Driver-Aide (DA) Problem". We model it as an integer program (IP) with an exponential number of variables and an exponential number of constraints and propose a sophisticated branch-cut-and-price approach to solve it. We conduct computational experiments on simulated instances based on real-world data provided by an industrial partner and a dataset released by Amazon. These experiments demonstrate that the proposed branch-cut-and-price approach can find near-optimal solutions efficiently. More importantly, they allow us to conduct an economic analysis of the tradeoffs in using a driver-aide, and characterize the conditions under which this novel operation mode can lead to significant savings.

Our economic analysis shows that, across our instances, the driver-aide problem not only can reduce the total delivery time by 29% on average, but can also reduce the operating cost by 14% on average. Our economic analysis aims to come up with rules of thumb to use in practice. Roughly, delivery routes that have greater than 50% of the time devoted to delivery (as opposed to driving) are the routes that provide the greatest benefit. These routes are characterized by a high density of delivery locations. Our case study on the dataset released by Amazon (Merchán et al. 2022) shows that the driver-aide problem can potentially generate significant benefits (on average, a 35% reduction in the routing time and 22% in cost savings).

The driver-aide problem generalizes the celebrated Vehicle Routing Problem (Toth and Vigo 2002, Golden et al. 2008, Toth and Vigo 2014, Mor and Speranza 2022). However, the driver-aide problem requires the same truck to visit a node multiple times. While multiple visits of a customer have been previously considered in the literature (see Xu et al. 2017), these multiple visits are implemented by different trucks, and not by the same truck. The closest problem studied in the literature is the hybrid driver helper (HDH) model in Lu et al. (2022). The HDH model uses a completely different objective function that combines labor and fuel costs together; it also ignores the possibility of the HDH moving among the locations within a node. For the HDH model, Lu et al. (2022) present a mixed integer programming (MIP) formulation based on the big-M technique and report computational results without optimality gaps. This MIP formulation does not scale well, as demonstrated in our experiments. In the service industry, Fikar and Hirsch (2015) and Coindreau et al. (2019) consider a routing problem involving service workers on foot. Although this problem shares some similarities with the driver-aide problem (i.e., routing of independent service workers on foot), it is quite different in that there are no tradeoffs to consider. Specifically, in the driver-aide problem, there is a benefit to keeping the aide and driver together with regard

to service time, which is lost when the driver and aide work separately at nodes. Further, given the nature of the service industry, each route has many fewer nodes to visit, compared to package delivery, and service time dominates the cost of the route. Despite focusing on metaheuristics as solution approaches to solve the problem, these methods have limited success, with running times often over 10 hours.

The driver-aide problem shares some similarities with drone delivery problems, which have gained recent attention in the literature. Comprehensive reviews by Otto et al. (2018) and Chung et al. (2020) nicely summarize the most relevant work on this topic. There are several significant differences between the driver-aide and the drone delivery problems. First, the drone and the delivery truck move over different networks (typically the drone moves in Euclidean space, while the delivery truck moves over the street network), which is not the case in the driver-aide problem. Second, drones usually have two major capacity restrictions: i) the number of packages carried by a drone (which is often set as a small number or is even simply set as one due to the allowable weight that a drone can carry), and ii) the travel distance or speed allowed for a drone (due to battery capacity and energy consumption). The driver-aide problem also has some similarities to the capacitated autonomous vehicle assisted delivery problem studied by Reed et al. (2022a,b). In this problem, an autonomous vehicle with a single delivery person is used to make package deliveries. The autonomous vehicle does not make deliveries without the delivery person, whereas in the driver-aide problem, both the driver and driver-aide can independently and simultaneously make deliveries (this creates greater opportunities for a reduction in the overall service time in the driver-aide problem, which does not occur in their setting).

Most importantly, the driver-aide represents a cost-effective way to address the delivery of increased package volumes at scale today. Drone technology (or autonomous vehicle technology) is yet to be widely deployed and faces severe regulations from the Federal Aviation Administration (`https://www.faa.gov/uas/`) (or the National Highway Traffic Safety Administration) in the U.S. It is unclear when drones (or autonomous vehicles) will be approved for mass-market deliveries (at least in the current regulatory environment, this does not seem to be happening anytime soon), and whether (at least in the short term) it is a viable cost-effective technology for large-scale package delivery. Amazon Prime Air (Amazon 2022) is one of many examples that illustrate this situation. Amazon's CEO Jeff Bezos announced Amazon Prime Air in 2013, which was supposed to begin by 2018. However, the service has yet to launch and is still being tested in several locations across the world as of 2022.

Solution approaches based on column generation (branch-and-price, branch-cut-and-price, price-and-branch, and so on) are proven techniques for large and hard integer programming problems (Barnhart et al. 1998, Desaulniers et al. 2006). This is especially the case for Vehicle Routing

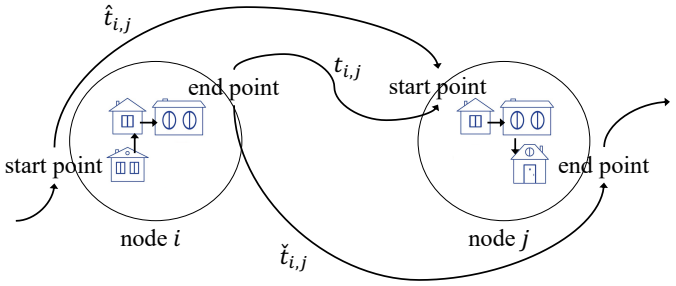



**Figure 1 Illustration of the Nodes and Travel Time Matrixes ($T$, $\hat{T}$, and $\check{T}$) in the Driver-Aide Problem.**

problems, where the current state of the art exact methods are all found in this category (Costa et al. 2019, Florio et al. 2020, Rostami et al. 2021). However, the typical set covering formulation in the literature cannot be applied to the driver-aide problem because it requires modeling the truck traveling with and without the aide. We propose a novel formulation to model the driver-aide problem. We also present valid inequalities to strengthen the formulation. Moreover, we use several algorithmic enhancements to improve the efficiency of the branch-cut-and-price approach. These enhancements improve the running time over 300 times, or by over two orders of magnitude, on average. All of these lead to a solution approach that can efficiently find near-optimal solutions for real-world-sized instances.

The rest of the paper is organized as follows. Section 2 provides a mathematical description of the problem, an illustrative example, and a compact MIP formulation for the problem. However, this MIP is not computationally viable. Consequently, Section 3 describes our branch-cut-and-price approach in solving the driver-aide problem. It provides a novel IP formulation with an exponential number of variables and constraints. Further, we describe several algorithmic enhancements that are key to speeding up and improving the computational tractability of the branch-cut-and-price approach. Section 4 describes our computational experience on simulated instances based on real-world data. Based on our solutions, we also present an economic analysis of the tradeoffs involved in using a driver-aide. This provides managers in the field with rules of thumb to determine which routes may benefit from a driver-aide, and how best to utilize them. Section 5 conducts a case study based on a dataset released by Amazon (Merchán et al. 2022) to further evaluate the impact of the driver-aide problem. Section 6 provides concluding remarks.

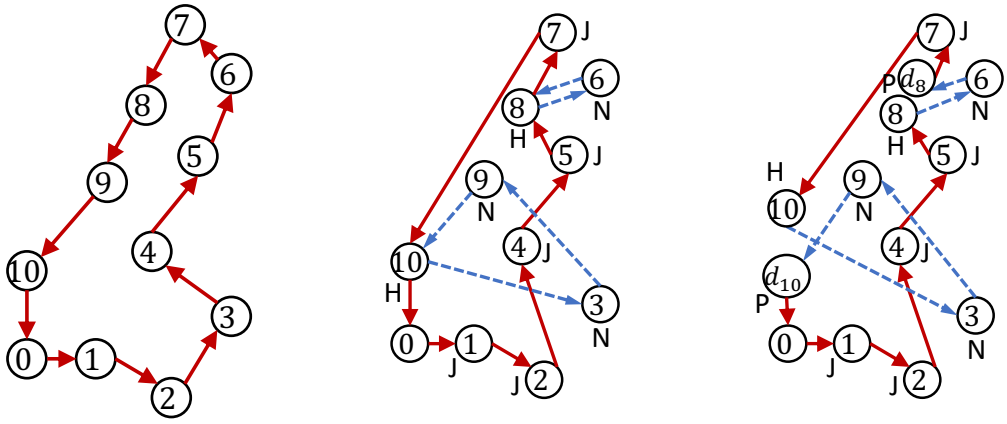
## 2. The Driver-Aide Problem: Definition and Formulation

We formally describe the driver-aide problem. In addition to the driver, a truck is equipped with an aide who can assist the driver in two ways. As a "Jumper", the aide works with the driver in preparing and delivering packages, thereby reducing the service time at a given stop. As a "Helper", the aide can independently work at a stop delivering packages, while the driver leaves to deliver packages at other stops and then returns. The key difference between the two modes is whether the

aide works with or without the driver at a stop. Given a set of delivery stops, travel times, service times, the jumper's savings, and the helper's service time, the goal is to determine both the delivery route and the most effective way to use the aide (e.g., sometimes as a jumper and sometimes as a helper) to minimize the total delivery time. As the application envisioned for driver-aide routes generally does not have time windows (time windows are typically associated with more expensive overnight deliveries—where a premium is paid—and are not common in package delivery), we do not consider them.

The formal definition of the problem is as follows. We have a graph $G = (V, A)$ with a set of nodes $V = \{0, 1, 2, \ldots, n\}$ and a set of arcs $A$. $V$ contains $n + 1$ nodes. Node 0 is the depot, and the other $n$ nodes, $C = V \setminus 0$, represent stops to visit. Arc set $A$ contains an arc from nodes $i$ to $j$ if the truck can travel from nodes $i$ to $j$. As we mentioned earlier, each node corresponds to multiple customers. These customers could be at the same location (i.e., same building) or in close geographical proximity. The order of visiting customer locations within a node is predetermined (the order is often provided by the service provider or can be obtained by solving a traveling salesman problem (TSP) of all customer locations in advance). Thus, each node has one start point and one end point within it (when the customers are all at the same location, this point is the same, whereas the start and end points are different when the customers are in close proximity). When the helper mode is used at a node, we drop off the aide at the start point and pick up the aide at the end point. Consequently, in the helper mode, the aide can move between customers within a node. A travel time matrix $T$ has entry $t_{i,j}$, showing the travel time from node $i$'s end point to node $j$'s start point. A travel time matrix $\hat{T}$ has entry $\hat{t}_{i,j}$, showing the travel time from node $i$'s start point to node $j$'s start point (the truck drops off the aide at the start point of node $i$ and moves to the start point of node $j$). A travel time matrix $\check{T}$ has entry $\check{t}_{i,j}$, showing the travel time from node $i$'s end point to node $j$'s end point (the truck moves from the end point of node $i$ to picks up the aide at the end point of node $j$). Without loss of generality, we assume that the travel time matrices satisfy the triangle inequality. Figure 1 illustrates the setting related to the nodes and the travel time matrixes ($T$, $\hat{T}$ and $\check{T}$) in detail. When customers are at the same location within a node, $T = \hat{T} = \check{T}$. Even when customers are not at the same geographical location within a node, the three travel times, $t_{i,j}$, $\hat{t}_{i,j}$, and $\check{t}_{i,j}$, are quite close to each other and frequently (more than 50% of the time) identical. Section 4 provides specific details with regard to our case study on an Amazon dataset.

For each node $i$ in $C$, there are three attributes: i) the service time, $s_i$, shows the time needed at node $i$, including the time to serve the customer locations within node $i$ and the time to travel from the start point to the end point of node $i$; ii) the jumper-saving factor, $f_i$, provides the percentage savings of the service time at node $i$ if an aide is on the truck and is used as a jumper; and iii) the




(a) Without the Helper Mode (b) With the Helper Mode (c) With Dummy Nodes in MIP1

**Figure 2 Illustration of Optimal Routes for the Driver-Aide Problem.**

helper-serving factor, $g_i$, provides the percentage change of the service time at node $i$ if an aide is used as a helper. Thus, with the helper, the delivery time needed is changed to $(1+g_i)s_i$. A positive $g_i$ means that the aide moves slower than the truck within the customer locations at node $i$, and a negative $g_i$ means that the aide moves faster. Notice that while $f_i$ is always nonnegative for savings, this is not necessarily the case for $g_i$ because utilizing an e-bike (or moving on foot or an e-scooter) may result in shorter travel times between some customer locations at a node, and thus, lead to savings. Consequently, we need to decide the most effective way to use the aide (e.g., sometimes as a jumper and sometimes as a helper). If the aide is dropped off at the start point of a node as a helper, the truck can move forward and make deliveries to other locations, while the truck must return to the end point of the node and pick up the aide at a later time point. The goal is to determine a route for the truck and how to use the aide while minimizing the total delivery time.

We use one small example to illustrate the driver-aide problem. In this example, there are 11 nodes: one depot and ten stops. The depot, node 0, is located at Euclidean coordinate $(3,9)$, and the ten stops are at the following Euclidean coordinates: $(5,9),(7,10),(8,8),(7,6),(8,3),(9,1),(9,0),(8,2),(7,4)$, and $(3,7)$. For ease of exposition, the travel time is set as the Euclidean distance between two points, with $T=\hat{T}=\check{T}$. The service times are 39, 91, 18, 30, 11, 29, 76, 29, 59, and 95 for nodes 1 to 10, respectively. The jumper-saving factors, $f_i$, are 0.5 for nodes 1, 2, 4, and 7, and are 0 for other nodes. For simplicity, the helper-serving factors, $g_i$, are 0 for all nodes. Figure 2 illustrates this instance. When the driver-aide is not available, the optimal route is shown in Figure 2(a). When there is no aide, the driver-aide problem is a TSP (Because the service time remains the same regardless of the route, minimizing the route duration is achieved by finding the shortest TSP tour). The optimal route time is 503.58 (travel and service time). When the aide can only be used as a jumper, the optimal route remains the same as the optimal TSP route. This can easily be proved by contradiction.

PROPOSITION 1. *When the aide can only be used as a jumper, the optimal route for the driver-aide problem is identical to the optimal TSP route.*

While the route is the same as the TSP route, the total route time with an aide is different. For the example, in Figure 2, if the aide can only be used as a jumper, the optimal routing time is 385.58 due to the savings on service time, which reflects a reduction of 118.00 units, compared to the situation without an aide. This is due to the reduction in service time because of the jumper mode. When the helper mode is available, Figure 2(b) shows the optimal route. The letters "J", "H", and "N" beside a node indicate the jumper mode, the helper mode, and no aide, respectively. Only the driver travels on the blue dashed loops (8-6-8 and 10-3-9-10). Thus, the truck drops off the aide as a helper at the start point of node 8 and visits node 6 before picking up the aide at the end point of node 8. Later, the truck drops off the aide as a helper at the start point of node 10 and visits nodes 3 and 9 before picking up the aide at the end point of node 10. The optimal route time is 276.67. The savings are 108.91 units, compared to the jumper-only mode. The total savings on time are 226.91 units, a 45% reduction, compared to the optimal TSP route with no aide.

We present an MIP formulation for the driver-aide problem below. A similar formulation is discussed in Lu et al. (2022). The key idea behind this formulation is to create a dummy node $d_i$ for each location $i$ in $C$ such that if the aide is used as a helper at node $i$, the truck must visit dummy node $d_i$ at a later time. Thus, the dummy nodes are used to capture the pick-up requirement in the helper mode. Figure 2(c) illustrates this idea by including the two dummy nodes, $d_8$ and $d_{10}$. The letters "P" beside them indicates that the aide is picked up. Each dummy node has a service time, a jumper-saving factor, and a helper-serving factor as 0. Let $D = \{d_1, d_2, \ldots, d_n\}$ denote the dummy nodes. Use $n(i)^-$ and $n(i)^+$ to denote the sets of node $i$'s incoming and outgoing neighbors, respectively. Then, for each dummy node $d_i$ in $D$, we add arcs such that $n(d_i)^- = n(i)^-$ and $n(d_i)^+ = n(i)^+$. For each newly added arc, $t_{j,d_i} = \hat{t}_{j,d_i} = \check{t}_{j,d_i} = \check{t}_{j,i}$ if $j$ in $n(d_i)^-$ and $t_{d_i,j} = \hat{t}_{d_i,j} = \check{t}_{d_i,j} = t_{i,j}$ if $j$ in $n(d_i)^+$. Let $\bar{A}$ denote the set of arcs after adding arcs for the dummy nodes. Thus, the transformed graph is denoted as $\bar{G} = (V \cup D, \bar{A})$.

Define a nonnegative variable $a_i$ for each node $i$ in $V \cup D$. For a node $i$ in $C$, $a_i$ represents the earliest time that the truck can arrive at node $i$. For a node $d_i$ in $D$, $a_{d_i}$ represents the earliest time that the truck can leave node $d_i$ for the next node after node $i$'s service is completed (assuming that node $i$ has been serviced in the helper mode). For the depot, $a_0$ represents the earliest route completion time (i.e., the time that the truck returns to the depot). Define a binary variable $h_i$ for each $i$ in $C$. If $h_i$ is 1, the aide is used as a helper at node $i$; otherwise, $h_i$ is 0. Define a binary variable $z_i$ for each node $i$ in $C \cup D$. If $z_i$ is 1, the aide is on the truck and is used as a jumper at

node $i$. Finally, for each arc $\{i,j\}$ in $\bar{A}$, a binary variable $x_{i,j}$ represents whether the truck travels from nodes $i$ to $j$ or not.

$$\text{(MIP1) Min } a_0 \tag{1}$$

$$\text{Subject to } \sum_{j\in n(i)^-} x_{j,i} = 1, \quad \forall i \in V, \tag{2}$$

$$\sum_{j\in n(i)^-} x_{j,i} \le 1, \quad \forall i \in D, \tag{3}$$

$$\sum_{j\in n(i)^-} x_{j,i} = \sum_{j\in n(i)^+} x_{i,j}, \quad \forall i \in V \cup D, \tag{4}$$

$$h_i + z_i \le 1, \quad \forall i \in C, \tag{5}$$

$$h_i + z_i \ge x_{0,i}, \quad \forall i \in C, \tag{6}$$

$$h_j + z_j \le z_i + (1 - x_{i,j}), \quad \forall i \in C \cup D, j \in n(i)^+ \cap C, \tag{7}$$

$$h_j + z_j \ge z_i - (1 - x_{i,j}), \quad \forall i \in C \cup D, j \in n(i)^+ \cap C, \tag{8}$$

$$\sum_{j\in n(d_i)^-} x_{j,d_i} = h_i, \quad \forall i \in C, \tag{9}$$

$$z_{d_i} = h_i, \quad \forall i \in C, \tag{10}$$

$$a_i \ge t_{0,i} x_{0,i}, \quad \forall i \in n(0)^+, \tag{11}$$

$$a_i + M(1 - x_{j,i}) \ge a_j + t_{j,i} x_{j,i} + s_j - f_j s_j z_j + (\hat{t}_{j,i} - t_{j,i} - s_j) h_j, \; \forall j \in C \cup D, i \in n(j)^+, \tag{12}$$

$$a_{d_i} + M(1 - h_i) \ge a_i + (1 + g_i) s_i, \quad \forall i \in C, \tag{13}$$

$$\mathbf{a} \ge 0, \mathbf{x}, \mathbf{z}, \mathbf{h} \in \{0,1\}. \tag{14}$$

Objective function (1) minimizes the total route time. Constraint (2) requires the truck to visit all nodes in $V$ exactly once. Constraint (3) requires the truck to visit a dummy node at most once. Constraint set (4) is the flow balance constraints. Constraint (5) ensures that each node in $C$ is served in one of three ways: by the jumper mode if $z_i = 1$, by the helper mode if $h_i = 1$, or by the driver alone if $h_i = z_i = 0$. Constraint (6) requires that the first node in $C$ visited by the truck can only be served either by the jumper mode or the helper mode, given that the aide must be on the truck as it just leaves the depot. Constraints (7) and (8) ensure that if the truck moves from nodes $i$ to $j$ and the jumper mode is used ($z_i = 1$) at node $i$, then node $j$ is either served by the jumper ($z_j = 1$) or helper ($h_j = 1$) mode. Constraint (9) requires the truck to visit dummy node $d_i$ if the aide is dropped off at node $i$ in the helper mode. Constraint (10) ensures that the jumper mode is enabled after the truck returns to pick up the aide at dummy node $d_i$. Constraints (11)–(13) track the arrival and departure times of each node in $C$ and $D$, respectively, by applying the big-$M$ approach, where $M$ is a very large number. In particular, constraint (11) applies the first node visited in the route. For each node $j$ in $C$ and $D$, constraint (12) calculates the travel and actual service times (depending on the mode used), if the truck travels from nodes $j$ to $i$, for the

arrival time of node $i$. Note that each dummy node has a service time as 0 and $\hat{t}_{j,i} = t_{j,i}$. Thus, the coefficient of the associated $h$ variable is 0. Constraint (13) ensures that the truck can only pick up the aide at node $d_i$ after the aide finishes his or her service at node $i$ (where the aide is used as a helper). Lastly, with a slight abuse of notations, constraint (14) enforces the nonnegativity and binary requirements on the respective variables.

While MIP1 is valid for the driver-aide problem, it is not a computationally viable one (one reason is that the big-$M$ approach is used in constraints (12)–(13)). Lu et al. (2022) uses a similar formulation. However, the optimality gaps are not reported. Thus, we have no idea regarding the quality of the solutions obtained by Lu et al. (2022). In our experiments, for an 11-node instance, we found that the optimality gap is over 98.6% after running MIP1 with a 12-hour time limit.

One reason the driver-aide problem is extremely challenging is that when the aide is left at a node as a helper, we cannot pick up the aide before the service is completed. Therefore, we need to track the service completion time at each node where the aide is used as a helper. While tracking the time is not new in the vehicle routing literature, the driver-aide problem uniquely requires the same truck to visit a node multiple times. Specifically, when the aide is used as a helper at a node, the driver leaves to deliver packages at other stops and then returns to the end point of the stop to pick up the aide. All of this motivates us to develop a more sophisticated and computationally viable approach for solving the driver-aide problem.

## 3. A Branch-Cut-and-Price Approach

To develop a branch-cut-and-price approach, we describe a formulation where we do not explicitly impose time tracking or the requirement of multiple visits. Observe that these two requirements are needed when the aide is dropped off at a node as a helper. On the one hand, the aide needs to finish the service at this node. On the other hand, the truck travels to other stops and returns to pick up the aide. We refer to this visiting sequence as a loop, as it starts and ends at the same stop, while the start and end points can be different. Comparing the aide's service completion time and the truck's return time, the larger value is the truck's departure time after picking up the aide. Therefore, in the following formulation, we include all such loops and determine which ones to include in the route.

Let $L$ be a set of all possible loops, $p$ be one such loop, and $Q_p$ be the set of nodes on loop $p$ that only the driver visits (i.e., $Q_p$ does not contain the starting and ending nodes). For each arc $\{i, j\}$ in $A$, a binary variable $x_{i,j}$ represents the route that the truck travels with the aide on it. It is 1 if the truck travels from nodes $i$ to $j$ with the aide on it; otherwise, it is 0. Let $L_i$ denote the loops starting and ending at node $i$ (i.e., the truck drops off and picks up the aide at node $i$). For each loop $p$ in $L$, a binary variable $y_p$ represents whether the loop is included in the route or not. If $y_p$

is 1, loop $p$ is included, and the driver visits the nodes in $Q_p$ by himself/herself. Otherwise, it is 0. Also, let $c_p$ denote loop $p$'s time duration, which is calculated as the larger of (i) the aide's service completion time at the starting node and (ii) the truck's return time. For each node $i$ in $C$, the binary variables $h_i$ and $z_i$ have the same meaning as in MIP1. If $h_i$ is 1, the aide is used as a helper at node $i$; otherwise, $h_i$ is 0. If $z_i$ is 1, the aide is on the truck and is used as a jumper at node $i$. Let $S$ be a set of nodes in $C$. We can write the following IP formulation for the driver-aide problem:

$$\text{(IP2)} \quad \text{Min} \quad \sum_{\{i,j\}\in A} t_{i,j}x_{i,j} + \sum_{i\in C}(1-f_i)s_i z_i + \sum_{p\in L} c_p y_p \tag{15}$$

$$\text{Subject to} \quad \sum_{j\in n(0)^-} x_{j,0} = 1, \ \sum_{j\in n(0)^+} x_{0,j} = 1, \tag{16}$$

$$\sum_{j\in n(i)^-} x_{j,i} + \sum_{p\in L: i\in Q_p} y_p = 1, \qquad \forall i \in C, \tag{17}$$

$$\sum_{j\in n(i)^-} x_{j,i} = \sum_{j\in n(i)^+} x_{i,j}, \qquad \forall i \in C, \tag{18}$$

$$h_i + z_i = \sum_{j\in n(i)^-} x_{j,i}, \qquad \forall i \in C, \tag{19}$$

$$\sum_{p\in L_i} y_p = h_i, \qquad \forall i \in C, \tag{20}$$

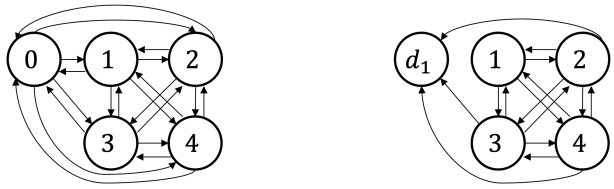
$$\sum_{i,j\in S:\{i,j\}\in A} x_{i,j} \le |S| - 1, \qquad \forall S \subseteq C, |S| \ge 2, \tag{21}$$

$$\mathbf{h}, \mathbf{x}, \mathbf{y}, \mathbf{z} \in \{0,1\}. \tag{22}$$

Objective function (15) minimizes the route's total routing time. Constraint (16) requires the truck to enter and leave the depot exactly once. Constraint (17) requires the truck to visit all nodes in $C$ exactly once, either with the aide ($x$ variables) or without the aide ($y$ variables). Constraint set (18) is the flow balance constraints for all nodes in $C$. Constraint (19) ensures that if the aide is on the truck when the truck visits node $i$ in $C$, the aide should be used in one of two modes: either as a jumper when $z_i = 1$ or as a helper when $h_i = 1$. Constraint (20) requires a loop $p$ in $L_i$ to be be selected if the helper mode is used at node $i$ in $C$. Constraint (21) is a subtour elimination constraint for the subset of the nodes in $C$ visited by the truck with the aide ($x$ variables). With a slight abuse of notation, constraint (22) enforces binary requirements on the variables.

### 3.1. Basic Components

We present the basic components for our branch-cut-and-price approach in this section. Before presenting the row- and column-generation procedure, let IP2R denote a restricted version of IP2 such that IP2R is identical to IP2, except that it only contains a subset of constraint (21) and a subset of the $y$ variables. Specifically, let $L'$ be a subset of $L$ and $S'$ be the set of subsets of the nodes in $C$ considered so far. We have




(a) The Original Graph $G=(V,A)$. (b) The Transformed Graph $G^i=(V^i,A^i)$ with $i=1$.

**Figure 3 Illustration of the Transformed Graph for the Basic Column-Generation Problem: Pricing$_i$.**

$$\text{(IP2R)} \quad \text{Min} \quad \sum_{\{i,j\}\in A} t_{i,j}x_{i,j} + \sum_{i\in C}(1-f_i)s_i z_i + \sum_{p\in L'} c_p y_p \tag{23}$$

$$\text{Subject to} \quad (16),(18),(19),(22),$$

$$\sum_{j\in n(i)^-} x_{j,i} + \sum_{p\in L':i\in Q_p} y_p = 1, \qquad \forall i\in C, \tag{24}$$

$$\sum_{p\in L_i'} y_p = h_i, \qquad \forall i\in C, \tag{25}$$

$$\sum_{\substack{i,j\in S:\\ \{i,j\}\in A}} x_{i,j} \le |S|-1, \qquad \forall S\in S'. \tag{26}$$

Then, let LP2R be the LP relaxation of IP2R.

*Basic Row Generation:* There are exponentially many constraints in IP2 because of constraint (21). We need a row-generation procedure for it. We want to eliminate subtours on the subset of the nodes in $C$. That means $\sum_{i,j\in S:\{i,j\}\in A} x_{i,j} \le |S|-1$ for all $S\subseteq C$ and $|S|\ge 2$. The following row-generation procedure is invoked when we have an optimal solution of LP2R, denoted by $(\mathbf{h}^*,\mathbf{x}^*,\mathbf{y}^*,\mathbf{z}^*)$. We can use a shortest path algorithm to find the directed cycles included by $\mathbf{x}^*$ (Grötschel et al. 1985). Specifically, eliminating the subtours means no directed cycles. Thus, for a given directed cycle $\mathcal{C}$, we have $\sum_{\{i,j\}\in\mathcal{C}} x_{i,j} \le |\mathcal{C}|-1$. Let $\mathbf{w}=1-\mathbf{x}^*$. We have $\sum_{\{i,j\}\in\mathcal{C}}(1-w_{i,j}) \le |\mathcal{C}|-1$ as $\sum_{\{i,j\}\in\mathcal{C}} x_{i,j} \le |\mathcal{C}|-1$. That leads to the inequalities $\sum_{\{i,j\}\in\mathcal{C}} w_{i,j} \ge 1$. Then, for each node $i$ in $C$, create a dummy node $d_i$ and add the incoming arcs of node $i$ to it such that $n(d_i)^- = n(i)^-$. If a shortest path based on $\mathbf{w}$ from nodes $i$ to $d_i$ has a distance smaller than 1, a violated subtour elimination constraint (a directed cycle starting and ending at node $i$) has been found. After that, we add the new constraint $\sum_{i,j\in S:\{i,j\}\in A} x_{i,j} \le |S|-1$ to IP2R, where $S$ contains the nodes in the shortest path, excluding dummy node $d_i$.

*Basic Column Generation:* IP2 has exponentially many $y$ variables because all loops are included in IP2. A pricing procedure is needed to dynamically add them to IP2R. The column-generation procedure for variable $\mathbf{y}$ is as follows. Let $\boldsymbol{\alpha}$ and $\boldsymbol{\beta}$ be the dual variables associated with constraints (24) and (25). We have a solution $(\boldsymbol{\alpha}^*,\boldsymbol{\beta}^*)$. For each node $i$, we can find a new loop by applying the following procedure. First, remove node 0 from graph $G$. Create a dummy node $d_i$ and connect all nodes in $n(i)^-$ to node $d_i$ by the incoming arcs, i.e., $n(d_i)^- = n(i)^-$. For each newly added arc, its travel times are $t_{j,d_i} = \hat{t}_{j,d_i} = \check{t}_{j,d_i} = \check{t}_{j,i}$ for $j$ in $n(d_i)^-$. Denote the resulting

graph as $G^i = (V^i, A^i)$. Thus, an elementary shortest path with side constraints (constraints (28), (29), and (30) below) from nodes $i$ to $d_i$ is equivalent to the loop we need. Figure 3 illustrates the transformation with a small example. Figure 3(a) is the original graph, and Figure 3(b) shows the transformed graph with $i = 1$. An elementary shortest path from node 1 to node $d_1$ is the loop we need.

For each arc $\{j, l\}$ in $A^i$, a binary variable $u_{j,l}$ represents the arc on which the truck travels. It is 1 if the truck travels from nodes $j$ to $l$, and 0 otherwise. For each node $l$ in $V^i \setminus \{i, d_i\}$, a binary variable $v_l$ is 1 if node $l$ is in the loop (which starts at node $i$ and ends at node $d_i$), and is 0 otherwise. Let a continuous variable $c_p$ denote the total time of this loop. We use $A(S)^+$ to denote the outgoing arcs of a set of nodes $S$. We have the following pricing problem.

$$\text{(Pricing}_i\text{)} \quad \text{Min} \quad c_p - \sum_{l \in V^i \setminus \{i,d_i\}} \alpha_l^* v_l - \beta_i^* \tag{27}$$

$$\text{Subject to} \quad \sum_{\{j,l\} \in A^i} u_{j,l} = v_l, \qquad \forall l \in V^i \setminus \{i, d_i\}, \tag{28}$$

$$c_p \geq (1 + g_i) s_i, \tag{29}$$

$$c_p \geq \sum_{\{i,k\} \in A^i} \hat{t}_{i,k} u_{i,k} + \sum_{\{k,d_i\} \in A^i} \check{t}_{k,d_i} u_{k,d_i} + \sum_{\substack{\{l,j\} \in A^i: \\ l \neq i, j \neq d_i}} t_{l,j} u_{l,j} + \sum_{l \in V^i \setminus \{i,d_i\}} s_l v_l, \tag{30}$$

$$\sum_{\{i,k\} \in A^i} u_{i,k} = 1, \tag{31}$$

$$\sum_{\{k,d_i\} \in A^i} u_{k,d_i} = 1, \tag{32}$$

$$\sum_{\{j,l\} \in A^i} u_{j,l} = \sum_{\{l,j\} \in A^i} u_{l,j}, \qquad \forall l \in V^i \setminus \{i, d_i\}, \tag{33}$$

$$\sum_{\{j,l\} \in A^i} u_{j,l} \leq 1, \qquad \forall l \in V^i \setminus \{i, d_i\}, \tag{34}$$

$$\sum_{\{m,j\} \in A(S)^+} u_{m,j} \geq \sum_{j \in n(l)^+} u_{l,j}, \qquad \forall S \subseteq V^i \setminus \{i, d_i\},\ l \in S, |S| \geq 2, \tag{35}$$

$$c_p \geq 0, \mathbf{u} \in \{0,1\}, \mathbf{v} \in \{0,1\}. \tag{36}$$

Objective function (27) minimizes the reduced cost of the loop. Constraint (28) links variables $u$ and $v$. Thus, if node $l$ is visited, $u_l$ is 1. Constraints (29) and (30) calculate the total time of loop $p$, which is the maximum of the service time at node $i$ by the helper and the total time for the truck to get back to node $i$. Constraint (31) ensures that a one-unit flow goes out from node $i$, while constraint (32) ensures that a one-unit flow goes into node $d_i$; they are the source and sink nodes. Constraint set (33) represents the flow balance constraints for all nodes in $V^i \setminus \{i, d_i\}$. Constraint (34) ensures that each node is visited at most once. Constraint (35) is a generalized cutset inequality (GCS), which ensures that there are no cycles in the solution. Constraint (36) enforces the nonnegative and binary requirements on the variables. If the optimal objective is negative, we add a $y$ variable to reflect the optimal solution of Pricing$_i$ to IP2R. The pricing

problem is NP-hard because it is the elementary shortest problem with negative cost cycles and side constraints (Garey and Johnson 1979).

*Basic Initial y Variables:* We initialize IP2R with some basic $y$ variables by creating loops in the following way: for each $i$ in $C$, we create a loop $\{i, i+1, i\}$, where $i+1=1$ if $i=n$. In other words, we create a loop starting and ending at node $i$. The loop also includes node $i+1$; however, when $i=n$, the loop includes node 1. Thus, in total, we include $n$ basic $y$ variables.

*Branching Rule:* In the branching procedure, rather than branching on the variables, we branch on constraint set (19) to create two branches; one with $h_i + z_i = 0$ and one with $h_i + z_i = 1$. In this way, we decide whether a node is on the driver-only loop ($h_i + z_i = 0$) or not ($h_i + z_i = 1$). Furthermore, we pick the most fractional constraint (19) on which to branch. Specifically, let $\hat{\mathbf{h}}$ and $\hat{\mathbf{z}}$ be the solution values of the $\mathbf{h}$ and $\mathbf{z}$ variables at the current branch node. We choose $\hat{i}$, where $\hat{i} = \arg\min\{i \in C : |\hat{h}_i + \hat{z}_i - 0.5|\}$. We note that this branching rule does not affect the pricing problem because it effectively branches on $\sum_{p \in L'_i} y_p$ and fixes it as 0 or 1.

### 3.2. Enhancements

In this section, we present several enhancements for the branch-cut-and-price approach. These enhancements significantly improve the computational tractability in our experiments.

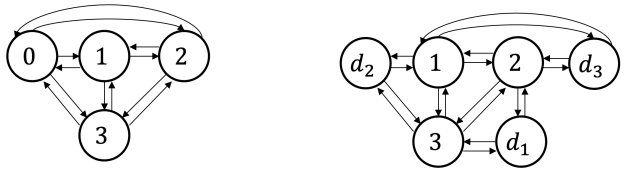
*Enhanced Row Generation:* We present a strengthened version of the subtour elimination constraints. While there exist similar results for vehicle routing problems in the literature, the existing results do not apply to the driver-aide problem. Our results are uniquely derived for the driver-aide problem. The set of strengthened valid inequalities are formally stated as follows:

PROPOSITION 2. *$\sum_{i,j \in S:\{i,j\} \in A} x_{i,j} - \sum_{i \in S \setminus \{k\}} (h_i + z_i) \leq 0, \forall S \subseteq C, |S| \geq 2, k \in S$ is valid for the driver-aide problem and dominates constraint set* (21)*, $\sum_{i,j \in S:\{i,j\} \in A} x_{i,j} \leq |S| - 1, \forall S \subseteq C, |S| \geq 2$.*

We refer to this set of valid inequalities as the DA-strengthened subtour elimination constraints. To illustrate the effectiveness of the DA-strengthened subtour elimination constraints, consider $S = \{1,2,3\}$ and $x^*_{1,2} = x^*_{2,3} = x^*_{3,1} = 2/3$. Thus, the corresponding constraint (21) is satisfied because $x^*_{1,2} + x^*_{2,3} + x^*_{3,1} = 2$. However, based on constraint (19), we have $h^*_1 + z^*_1 = h^*_2 + z^*_2 = h^*_3 + z^*_3 = 2/3$. Without loss of generality, let $k = 1$. Then, the corresponding DA-strengthened subtour elimination constraint is violated because $x^*_{1,2} + x^*_{2,3} + x^*_{3,1} - (h^*_2 + z^*_2) - (h^*_3 + z^*_3) = 2/3 > 0$. Thus, the corresponding DA-strengthened subtour elimination constraint cuts off this fractional solution.

Replacing constraint (21) with the DA-strengthened subtour elimination constraints, we obtain the following IP formulation, which is referred to as IP3.

$$\begin{aligned}
\text{(IP3)} \quad \text{Min} \quad & \sum_{\{i,j\} \in A} t_{i,j} x_{i,j} + \sum_{i \in C} (1 - f_i) s_i z_i + \sum_{p \in L} c_p y_p \\
\text{Subject to} \quad & (16), (17), (18), (19), (20), (22), \\
& \sum_{i,j \in S:\{i,j\} \in A} x_{i,j} - \sum_{i \in S \setminus \{k\}} (h_i + z_i) \leq 0, \qquad \forall S \subseteq C, |S| \geq 2, k \in S \qquad (37)
\end{aligned}$$




(a) The Original Graph $G = (V, A)$. (b) The Transformed Graph $G' = (C \cup D, A')$.

**Figure 4 Illustration of the Transformed Graph for the Strong Pricing Problem.**

Similar to IP2R and LP2R, we define IP3R and LP3R for IP3. Because the DA-strengthened subtour elimination constraints include constraint (21) (when all nodes in the subtour are either served in the helper or jumper mode), we can use the basic row generation as described earlier to separate this subset of violated inequalities (it also works to separate constraint (37) for the integral solutions of LP3R). To separate all of constraints (37), for a solution $(\mathbf{h}^*, \mathbf{x}^*, \mathbf{y}^*, \mathbf{z}^*)$ to (LP3R), we need to formulate the following problem. For each node $i$ in $V$, a binary variable $v_i$ is 1 if node $i$ is included in $S$; otherwise, it is 0 (this identifies the nodes in set $S$). For each node $k$ in $V$, a binary variable $w_k$ is 1 if node $k$ is the node from $S$ selected for exclusion from the summation of the second term of the left-hand side of constraint (37); otherwise, it is 0. For each arc $\{i, j\}$ in $A$, a binary variable $u_{i,j}$ is 1 if the arc $\{i, j\}$ is included in the the term $\sum_{i,j \in S:\{i,j\} \in A} x_{i,j}$. Otherwise, it is 0. We have the following separation problem.

$$\text{(SepEx)} \quad \text{Max} \quad \sum_{i \neq j \in C:\{i,j\} \in A} x^*_{i,j} u_{i,j} - \sum_{i \in C} (h^*_i + z^*_i) v_i + \sum_{k \in C} (h^*_k + z^*_k) w_k \tag{38}$$

$$\text{Subject to} \quad u_{i,j} \leq v_i,\ u_{i,j} \leq v_j, \qquad \forall i \neq j \in C : \{i, j\} \in A, \tag{39}$$

$$\sum_{i \in C} v_i \geq 2, \tag{40}$$

$$\sum_{k \in C} w_k = 1, \tag{41}$$

$$w_k \leq v_k, \qquad \forall k \in C, \tag{42}$$

$$\mathbf{u}, \mathbf{v}, \mathbf{w} \in \{0, 1\}. \tag{43}$$

Objective function (38) maximizes the left-hand side value of (37). If the objective value is strictly larger than zero, we have found a violated constraint. Constraint (39) ensures that an arc $\{i, j\}$ can be included only when both nodes $i$ and $j$ are selected in $S$. Constraint (40) states that we must select at least two nodes in $S$. Constraint (41) ensures that we exclude exactly one node. Constraint (42) states that only one of the selected nodes can be excluded. Constraint (43) enforces the binary requirements on the variables.

*Enhanced Column Generation:* The basic column generation looks for a loop starting and ending at node $i$ for each node $i$ in $C$. The elementary shortest path problem with negative cycles and side constraints is NP-Hard (Garey and Johnson 1979). Thus, it could be time consuming to exactly solve each Pricing$_i$. In our implementation, in the root node, we assign a time limit for each Pricing$_i$. After considering Pricing$_i$ at all nodes, an exact pricing procedure (described below) is invoked if no

promising loops are found. We refer to this exact pricing procedure as the strong pricing problem, which finds a promising loop (if it exists) across all starting nodes. In our initial computational experiments, we found that separately pricing at the root node (i.e., running Pricing$_i$ at each node with a time limit before strong pricing is invoked if necessary) yields multiple promising $y$ variables, which speeds up the running time initially. However, after adding a reasonable number of promising $y$ variables, separately running Pricing$_i$ increases the running time significantly. Further, it appears that after the root node, the model has a large enough pool of promising variables to provide near-optimal solutions. Thus, once Pricing$_i$ has been invoked five times for a given $i$ (which we found adequate in our initial experiments) or we leave the root node, only the strong pricing problem is invoked.

The strong pricing problem for $y$ variables can be modeled as follows. Let $\boldsymbol{\alpha}$ and $\boldsymbol{\beta}$ be the dual variables associated with constraints (24) and (25). Given a dual solution $(\boldsymbol{\alpha}^*, \boldsymbol{\beta}^*)$, we can find a promising loop by solving the MIP below. First, we start with the transformed graph $\bar{G}$ used for MIP1. Then, we remove node 0 and its associated arcs. Denote the resulting graph as $G' = (C \cup D, A')$. Thus, a shortest elementary path with side constraints between the best node $i$ and $d_i$ is equivalent to the loop we need. Figure 4 illustrates the transformation with a small example. Figure 4(a) is the original graph, and Figure 4(b) gives the transformed graph $G' = (C \cup D, A')$.

For each arc $\{i, j\}$ in $A'$, $u_{i,j}$ represents the arc on which the truck travels. It is 1 if the truck travels from nodes $i$ to $j$; otherwise, it is 0. Let $c_p$ denote the total time in this loop. For each node $i$ in $C$, there are two binary variables associated with it. The binary variable $v_i$ is 1 if node $i$ is visited; otherwise, it is 0. The binary variable $w_i$ is 1 if the loop starts and ends at node $i$; otherwise, it is 0.

$$\text{(StrongPricing) Min} \quad c_p - \sum_{i \in C} \alpha_i^* v_i - \sum_{i \in C} \beta_i^* w_i \tag{44}$$

$$\text{Subject to} \quad \sum_{\{j,i\} \in A} u_{j,i} = v_i, \qquad \forall i \in C, \tag{45}$$

$$\sum_{i \in C} w_i = 1, \tag{46}$$

$$c_p \geq \sum_{i \in C} (1 + g_i) s_i w_i, \tag{47}$$

$$c_p \geq \sum_{\{i,j\} \in A} t_{i,j} u_{i,j} + \sum_{i \in C} s_i v_i + \sum_{i \in C} \sum_{\{i,j\} \in A} (\hat{t}_{i,j} - t_{i,j}) u_{i,j} w_i + \sum_{i \in C} \sum_{\{j,d_i\} \in A} (\check{t}_{j,d_i} - t_{j,d_i}) u_{j,d_i} w_i, \tag{48}$$

$$\sum_{\{j,i\} \in A} u_{j,i} \leq 1 - w_i, \qquad \forall i \in C, \tag{49}$$

$$\sum_{\{j,d_i\} \in A'} u_{j,d_i} = w_i, \qquad \forall i \in C, \tag{50}$$

$$w_i + \sum_{\{j,i\}\in A} u_{j,i} = \sum_{\{i,j\}\in A} u_{i,j}, \qquad \forall i \in C, \tag{51}$$

$$\sum_{\{k,j\}\in A(S)^+} u_{k,j} \geq \sum_{j\in n(i)^+} u_{i,j}, \qquad \forall S \in C, i \in S, |S| \geq 2, \tag{52}$$

$$c_p \geq 0, \mathbf{u} \in \{0,1\}, \mathbf{v} \in \{0,1\}, \mathbf{w} \in \{0,1\}. \tag{53}$$

Objective function (44) minimizes the reduced cost of the loop. Constraint (45) links the variables $u$ and $v$. Thus, if node $i$ is visited, $v_i$ is 1. Constraint (46) ensures that path $p$ starts at exactly one node in $C$. Constraints (47) and (48) calculate the total time of path $p$, which is the maximum of the service time at node $i$ by the helper and the total time for the truck to get back to node $i$ after visiting and serving other locations. Constraint (49) ensures that if a node $i$ in $C$ is the source node, there are no arcs going into it. Constraint (50) ensures that if a node $i$ in $C$ is the source node, then node $d_i$ is the sink node and has exactly one incoming arc. It also ensures that all other nodes in $D$ have no incoming arcs. Constraint (51) is a modified flow balance constraint for all nodes in $C$. Along with constraint (49), it ensures that if node $i$ is the source node, there is exactly one outgoing arc from it. Constraint set (52) are generalized cutset inequalities (GCS), which ensures that there are no cycles in the solutions. Constraint (53) enforces the nonnegativity on $c_p$ and the binary requirements on the remaining variables. While constraint (48) is nonlinear, it models the product of two binary variables, which can be easily linearized by a standard approach (see Williams 2013, Section 9.2). However, most off-the-shelf solvers handle them. Specifically, in our implementation, we let Gurobi handle (StrongPricing) directly rather than explicitly linearizing constraint (48).

*Enhanced Initial $y$ Variables:* Another important enhancement involves including promising loops ($y$ variables) initially. When the aide is used as a helper at a node, we hope the driver can return in about the same time as the aide finishes the service. In other words, for a loop $p$ starting at node $i$, the time spent in $Q_p$, $\hat{t}_{i,k:\{i,k\}\in p} + \sum_{\{l,j\}\in p: l,j\neq i} t_{l,j} + \check{t}_{k,i:\{k,i\}\in p} + \sum_{j\in Q_p} s_j$, is not too far away from the service time of node $i$, $s_i$. For each location, we find two loops initially. One loop tries to have the truck visit as many nodes as possible, while its time does not exceed the service time of the starting location. The other loop tries to do the opposite. It has the truck visit as few nodes as possible, while its time exceeds the service time of the starting location. For each node $i$ in $C$, the two loops can be found by solving the following two problems. The variables are defined in the same way as Pricing$_i$. Max_Initial$_i$ finds a loop not exceeding the service time by the helper of the starting location, node $i$.

$$\text{(Max\_Initial}_i\text{)}\ \text{Max} \sum_{\{i,k\}\in A^i} \hat{t}_{i,k} u_{i,k} + \sum_{\{k,d_i\}\in A^i} \check{t}_{k,d_i} u_{k,d_i} + \sum_{\{l,j\}\in A^i: l\neq i, j\neq d_i} t_{l,j} u_{l,j} + \sum_{l\in V^i\setminus\{i,d_i\}} s_l v_l$$

$$\text{Subject to } (28), (31), (32), (33), (34), (35),$$

$$\sum_{\{i,j\}\in A^i} \hat{t}_{i,j} u_{i,j} + \sum_{\{j,d_i\}\in A^i} \check{t}_{j,d_i} u_{j,d_i} + \sum_{\{l,j\}\in A^i: l\neq i, j\neq d_i} t_{l,j} u_{l,j} + \sum_{l\in V^i\setminus\{i,d_i\}} s_l v_l \leq (1+g_i) s_i.$$

Similarly, Min_Initial$_i$ finds a loop exceeding the service time by the helper of the starting location, node $i$.

$$\begin{aligned}(\text{Min\_Initial}_i)\ \text{Min}\ & \sum_{\{i,k\}\in A^i} \hat{t}_{i,k}u_{i,k} + \sum_{\{k,d_i\}\in A^i} \check{t}_{k,d_i}u_{k,d_i} + \sum_{\{l,j\}\in A^i: l\neq i, j\neq d_i} t_{l,j}u_{l,j} + \sum_{l\in V^i\setminus\{i,d_i\}} s_l v_l \\ \text{Subject to}\ & (28),(31),(32),(33),(34),(35), \\ & \sum_{\{i,j\}\in A^i} \hat{t}_{i,j}u_{i,j} + \sum_{\{j,d_i\}\in A^i} \check{t}_{j,d_i}u_{j,d_i} + \sum_{\{l,j\}\in A^i: l\neq i, j\neq d_i} t_{l,j}u_{l,j} + \sum_{l\in V^i\setminus\{i,d_i\}} s_l v_l \geq (1+g_i)s_i.\end{aligned}$$

Before concluding this section, we note that the pseudocode for the branch-cut-and-price procedure is described in Algorithms 1, 2, and 3 of EC.3.

## 4. Computational Experiments

In this section, we discuss our computational experience with the branch-cut-and-price approach on a set of simulated instances based on real-world data. Our computational experiments have two goals: *i*) to examine the computational efficacy of the branch-cut-and-price approach, and *ii*) to evaluate the benefit and provide managerial insights for the driver-aide problem (i.e., the impact of the jumper and helper modes). We make these evaluations on 120 simulated instances based on real-world data provided by our industrial partner. Our computational experiments were conducted on a machine with the following specifications: M1 Ultra processor, 128 GB RAM, and the macOS operating system. Further, we use Gurobi 9.5.2 with the Python API. In our implementation, we impose a time limit of 12 hours unless stated otherwise. Our best setting of the branch-and-cut approach takes on average 190.8 seconds over 120 test instances. The larger time limit facilitates comparison with alternative formulations and settings. In Section 5, we apply the proposed branch-cut-and-price approach to a case study based on Amazon data (Merchán et al. 2022).

### 4.1. Data and Instance Description

RouteSmart Technologies (`https://www.routesmart.com`) is a leading provider of route optimization solutions for postal/parcel delivery companies around the world. We created simulated test instances based on a sample of 32 real-world routes from seven of the largest metropolitan areas in the U.S. that they provided to us. While we cannot directly use this dataset due to a non-disclosure agreement (NDA), we describe the procedure for generating the instances below.

We found that over 90% of the real-world routes provided to us had 40 or fewer stops, which is also consistent with the findings in Allen et al. (2018). Note that, however, each stop is associated with multiple customers (e.g., multiple deliveries to a large office building). The service time distribution also varies by customer (due to package and/or location characteristics as well as service requirements). Consequently, in our experiments, we create test instances with 40 stops. We distribute the locations of the 40 stops in three regions with 1-mile $\times$ 1-mile, 3-miles $\times$ 3-miles,

and 5-miles × 5-miles areas, respectively, using a uniform distribution on the Euclidean plane. These sizes capture a range from relatively small dense urban areas to larger suburban areas. We use Euclidean distance to set the travel time $T$, and because the sample of 32 real-world routes only contained customers that were physically co-located (e.g., same building or office complex), $T = \hat{T} = \check{T}$ for these instances. We reiterate that even when customers within a node are not at the same location, in practice, the three travel times, $t_{i,j}$, $\hat{t}_{i,j}$, and $\check{t}_{i,j}$, are quite close. Based on the case study on Amazon's last-mile routing in Section 5, on average, the differences from $t_{i,j}$ to $\hat{t}_{i,j}$ $(t_{i,j} - \hat{t}_{i,j})$ and from $t_{i,j}$ to $\check{t}_{i,j}$ $(t_{i,j} - \check{t}_{i,j})$ are 1.4 and -1.9 seconds, respectively. Further, $t_{i,j}$ and $\hat{t}_{i,j}$ are the same for 50.7% of the arcs, and $t_{i,j}$ and $\check{t}_{i,j}$ are the same for 50.8% of the arcs.

For each stop, the number of customers follows the distribution of the number of customers at a stop in our real-world data. After that, the service time for each customer follows the distribution of our real-world data. Adding the service times up for the customers at a stop provides the service time of the stop. For travel speed, we consider 5, 10, 15, and 20 MPH (Miles Per Hour). These speeds also reflect the settings from relatively small dense urban areas to larger suburban areas. The jumper savings factor is difficult to quantify accurately. Some locations have strict entry requirements, and the use of a jumper provides minimal (or no) benefit. It can also be a function of the package sizes and layout of a building or location. After a discussion with our industry partner and considering the existing literature (see Lu et al. 2020, 2022), we settled on experimenting with instances where each of the 40 stops uniformly selects its jumper-saving factor from $\{0, 0.1, 0.2, 0.3, 0.4, 0.5\}$ and its helper-serving factor from $\{0, 0.05, 0.1, 0.15, 0.2\}$. For each combination of region and travel speed, 10 instances are generated. Overall, we have 120 instances in our experiments. While we focus on this set of instances and derive insights from it, a practitioner (e.g., our industrial partner) can conduct this analysis with various area-dependent instances.

### 4.2. Computational Performance of the Branch-Cut-and-Price Approach

Before embarking on the 120 instances, we first generate a set of 30 instances with a 10 MPH travel speed to fine-tune the branch-cut-and-price approach. These 30 instances can be viewed as a training set that we only use in this section to evaluate the impact of the enhancements in the branch-cut-and-price approach. We set the stopping tolerance as 3% such that we stop the search process when we prove that our best solution is within 3% of optimality. There are five settings in total. Each one is built on the previous one. We start with the one called "Base", which has the basic components described in Section 3.1 without the enhancements in Section 3.2. The second one solves IP3 by using the DA-strengthened subtour elimination constraint (37) instead of constraint set (21). We call it "SSE" (strengthened subtour elimination). The next setting is "SSE+Promising $y$", which includes the promising $y$ variables initially, as described in the advanced initial $y$ variables

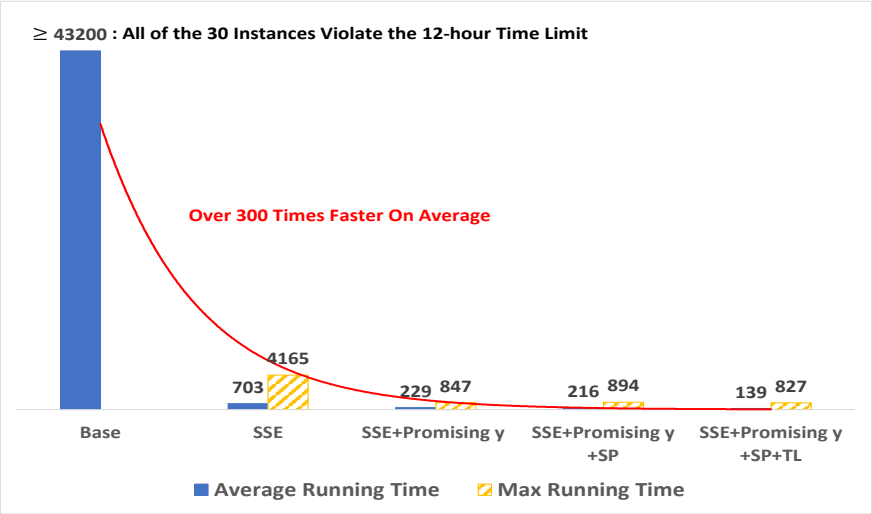



**Figure 5** **The Impact of the Enhancements on Running Time (Seconds) with a 3% Stopping Tolerance Over the 30 Instances with a 10 MPH Travel Speed.**

in Section 3.2. Further, in "SSE+Promising $y$+SP", the column-generation procedure uses Pricing$_i$ followed by the strong pricing procedure at the root node, but only the strong pricing procedure after the root node. Finally, a time limit (5 seconds) is enforced on each Pricing$_i$, and each Pricing$_i$ is invoked at most five times at the root node in the last setting, "SSE+Promising $y$+SP+TL".

Figure 5 plots the average and maximum running times of these five settings over the 30 instances. The trendline, based on the average running time, is displayed as well. The Base setting violates the time limit (12 hours) for all 30 instances. Thus, we plot the 12-hours time limit for the Base setting. This is an underestimation regarding the time needed for the Base setting. Clearly, only using the basic components cannot provide a viable solution. However, the DA-strengthened subtour elimination constraints are able to completely change the landscape. The SSE setting can find solutions with a 3% stopping tolerance in 703.11 seconds on average for the 30 instances. Including promising $y$ reduces the average running time to 229.73 seconds. Then, solely using the strong pricing procedure after the root node improves the average running time to 216.01 seconds. The final setting significantly speeds up the search process, with an average running time of 139.17 seconds.

Adding the DA-strengthened subtour elimination constraints to the base setting makes a tremendous improvement. While Proposition 2 demonstrates the superiority of the DA-strengthened subtour elimination constraints, we solve the LP relaxations of the Base and SSE settings to fully understand the impact of the DA-strengthened subtour elimination constraints. In this way, we can see the improvement achieved by the DA-strengthened subtour elimination constraints. This relative gap is calculated as $Z_{lp}/Z_{bfs}$, where $Z_{lp}$ is the objective value of the LP relaxation, and $Z_{bfs}$ is the objective value of the best feasible solution. The results are plotted in Figure 6. The red and blue bars are the relative gaps of the Base and SSE settings, respectively. We observe that the DA-strengthened subtour elimination constraints tighten the LP relaxation significantly. On

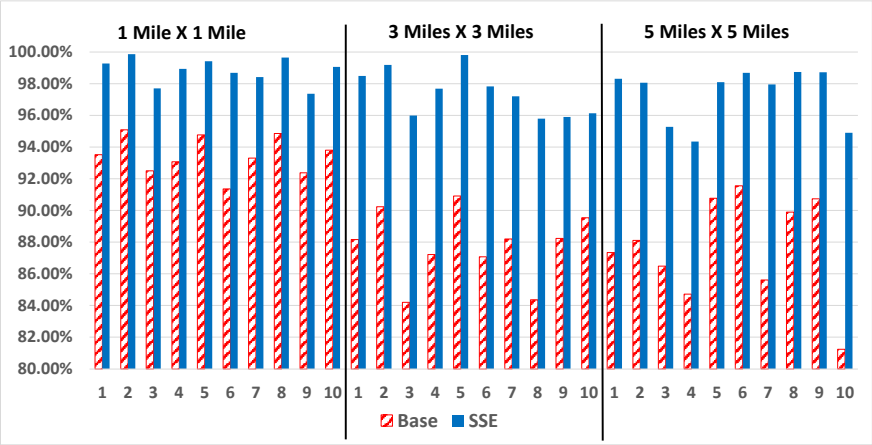



**Figure 6** **The Relative Gap With and Without the DA-strengthened Subtour Elimination Constraints.**

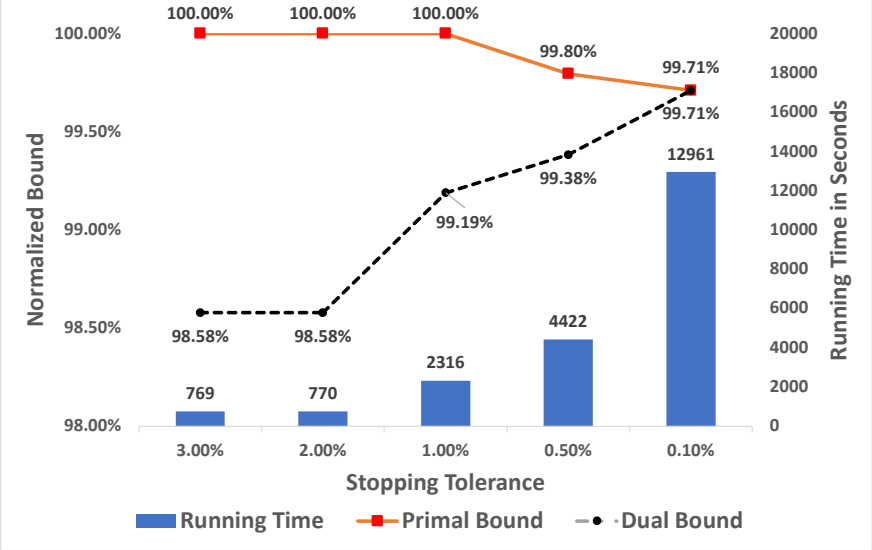



**Figure 7** **Illustration of the Behavior of the Branch-Cut-and-Price Approach with Various Stopping Tolerances.**

average, the relative gap is improved by over 8% because the relative gaps of the Base and SSE settings are 89.6% and 97.8%, respectively. The maximum improvement is about 14%, while the minimum improvement is about 5%. This shows that IP3 is a much stronger formulation than IP2. It also provides an explanation for the night-and-day contrast between the SSE and Base settings.

Compared to the Base setting, the enhanced branch-cut-and-price approach is over 300 times, or over two orders of magnitude, faster on average. Overall, the enhancements are crucial in improving the computational tractability of the solution procedure of the driver-aide problem. Hereafter, we only report the results for the driver-aide problem obtained by the branch-cut-and-price approach with all of the enhancements (i.e., "SSE+Promising $y$+SP+TL").

Before presenting the performance of the branch-cut-and-price approach on all 120 instances, we discuss the rationale for selecting the stopping tolerance. We tested a small subset of instances with various stopping tolerances. Figure 7 plots the behavior of a representative instance. Specifically, stopping tolerances are set as 3%, 2%, 1%, 0.5%, and 0.1%. The primal bound (best objective value) with 3% is used as the baseline to evaluate the primal and dual bounds obtained with various

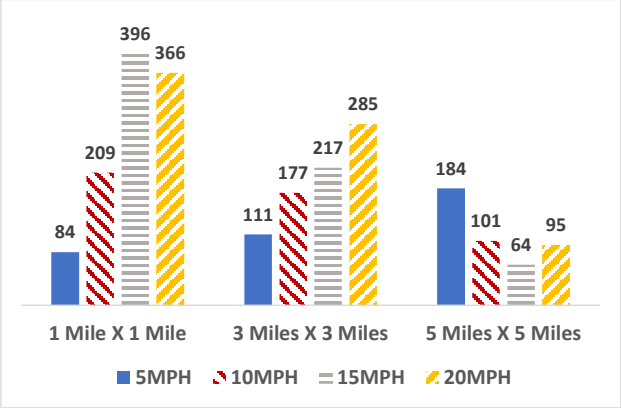



**Figure 8** **Average Running Time (Seconds) for Each Combination of the Region and Travel Speed over 120 Instances with a 2% Stopping Tolerance.**

stopping tolerances. The red solid line is for the primal bounds, and the black dashed line is for the dual bounds. The bars represent the running times. Generally, we observe that when the stopping tolerance decreases from 3% to 2%, it leads to a minimal or no increase in the running time. As a smaller stopping tolerance is used, the running time increases dramatically (the running time for 0.1% tolerance is almost 17 times that of 2% tolerance). On the other hand, a tighter stopping tolerance mainly helps produce a better dual bound and does not seem to significantly affect the primal bound. When the stopping tolerance is set to 0.1%, the dual bound is equal to the primal bound for this instance and proves optimality. Because the primal bound is the main factor for our later analysis on managerial insights, hereafter, we choose a 2% stopping tolerance, which hits the sweet spot between solution quality and computational time.

Table EC.1 contains the detailed computational performance of the branch-cut-and-price approach on the 120 instances with a 2% stopping tolerance. To better visualize the results, we plot the average running time in Figure 8. Generally, we observe that the driver-aide problem is more challenging when the region is smaller and the travel speed is higher. The intuition behind this is that when the region is smaller and the travel speed is higher, the branch-cut-and-price approach needs to consider more loops where the driver travels alone, leading to a higher computational burden in the search process. Overall, with a 2% stopping tolerance, the branch-cut-and-price approach finds solutions that are within a 1.2% optimality gap on average over the 120 instances. The average running time is 190.8 seconds.

### 4.3. Managerial Insights of the Jumper and Helper Modes

In this section, we study the benefit of the driver-aide problem by examining the impact of the jumper and helper modes. To achieve this, we solve two additional variants for each instance other than the driver-aide problem. Specifically, in one variant, we only have the driver in the truck. Thus, this becomes the celebrated traveling salesman problem (TSP). We call this the TSP variant. In the second variant, we include the driver-aide, but we only allow the driver-aide to be used

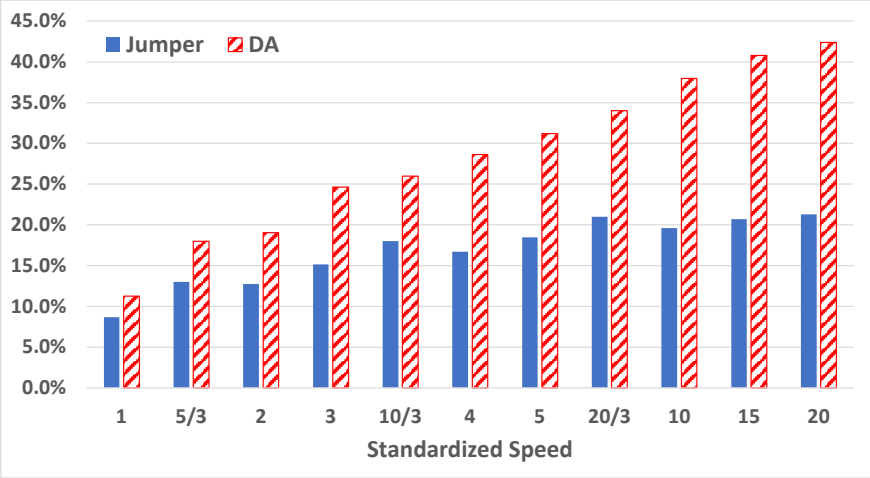



**Figure 9** **Average Percentage Reduction in the Routing Times for the Jumper Variant and the Driver-Aide Problem under Different Standardized Speeds.**

in the jumper mode, which is referred to as the jumper variant. For these two variants, we use the classical subtour elimination formulation for the TSP (see Lawler et al. 1985, Pataki 2003). Similar to Pferschy and Staněk (2017), we separate the violated subtour elimination constraints only for integral solutions (i.e., we relax the subtour elimination constraints in the branch-and-cut procedure for the TSP and only separate them when integer solutions are found at a branch-and-bound node). Notice that the jumper variant has the same optimal route as the one obtained by the corresponding TSP variant. While they are both NP-hard problems, practically, we are able to find the optimal solutions within a reasonable amount of time. Detailed results are presented in Tables EC.2 and EC.3.

We normalize the reduction in the routing time of each variant by the routing time of the TSP. Table EC.4 presents the detailed results. The relative reduction in the routing time of the jumper variant is calculated as $1 - z_{Jumper}/z_{TSP}$, where $z_{TSP}$ is the routing time of the TSP variant, and $z_{Jumper}$ denotes the routing time of the jumper variant. Similarly, the relative reduction in the routing time of the driver-aide problem is calculated as $1 - z_{DA}/z_{TSP}$, where $z_{DA}$ is the routing time of the best feasible solution (DA_UB) of the driver-aide problem. A higher value indicates a greater gain in the routing time, compared to the TSP variant.

To better visualize the pattern, we convert the travel speed to its equivalent one on a 1-mile $\times$ 1-mile service region. We refer to this as standardized speed. For example, 15 MPH on a 3-miles $\times$ 3-miles service region is equivalent to a 5 MPH standardized speed (i.e., 5MPH on a 1-mile $\times$ 1-mile service region). Overall, the 12 speed and region combinations correspond to the 11 standardized speeds: 1, 5/3, 2, 3, 10/3, 4, 5, 20/3, 10, 15, 20 MPHs.

Figure 9 plots the average percentage reduction in the routing time for both the jumper variant and the driver-aide problem under different standardized speeds. This shows us the impact of each of the jumper and helper modes. The differences between the TSP variant and the jumper variant

| | 5 MPH | | | 10 MPH | | | 15 MPH | | | 20 MPH | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Region | Jumper | Helper | Option | Jumper | Helper | Option | Jumper | Helper | Option | Jumper | Helper | Option |
| 1 X 1 | 100.9% | 83.3% | DA($\star$) | 97.6% | 75.3% | DA | 96.3% | 71.9% | DA | 95.6% | 70.0% | DA |
| 3 X 3 | 105.6% | 99.7% | DA($\star$) | 100.1% | 89.3% | DA($\star$) | 97.3% | 83.5% | DA | 95.6% | 80.0% | DA |
| 5 X 5 | 110.9% | 107.8% | TSP | 106.0% | 98.3% | DA($\star$) | 103.0% | 91.5% | DA($\star$) | 101.1% | 86.7% | DA($\star$) |

**Table 1 Average Relative Costs of the Jumper Variant and the Driver-Aide Problem Compared to the TSP Variant. Preferred Options between TSP and DA Based on the Average Relative Costs (DA: Driver-Aide with Both the Jumper and Helper Modes, TSP: Driver Only, ($\star$): Indicating that the Helper Mode Leads to DA).**

show the gain in the jumper mode, which are represented by the blue bars plotted in Figure 9. Also, the percentage reduction of the driver-aide problem (the red bars in Figure 9) shows the additional gain from the helper mode, which increases with standardized speed. Overall, the average routing time reduction of the jumper variant is 16.8%, and that of the driver-aide problem is 28.5%.

Clearly, the driver-aide problem (both the jumper and helper modes are allowed) has a much lower routing time, compared to the TSP and jumper variants. Figure 9 shows that the reduction obtained by optimally using the driver-aide is larger as the standardized speed increases. The reduction is 11.3% when the standardized speed is 1. It increases to 42.4% when the standardized speed is 20. This implies that the driver-aide with both the jumper and helper modes is most effective when we have *denser service regions* and *higher travel speeds*.

While the driver-aide problem achieves a significant reduction in the route completion time, an alternate analysis considers the impact from an overall cost perspective. According to the American Transportation Research Institute (Leslie and Murray 2022), the marginal cost per hour of the truck- and driver-based costs is \$74.65 in 2021 (which includes both the \$32.55 driver and \$42.10 truck costs). Based on data from the U.S. Bureau of Labor Statistics (2021) and data retrieved from Salary.com, Indeed.com, and Glassdoor.com on Oct-13-2022, we find that the average hourly salary for a driver-aide is \$16. Consequently, for the TSP variant, we use the hourly cost of \$74.65. For the jumper variant and the driver-aide problem, we use the hourly cost of \$90.65.

Notice that the normalized routing time can be interpreted as the time needed to complete the route finished by the TSP variant in one hour. We calculate the relative costs of the jumper variant and the driver-aide problem as $\$90.65 * z_{Jumper}/(\$74.65 * z_{TSP})$ and $\$90.65 * z_{DA}/(\$74.65 * z_{TSP})$, respectively. Table EC.5 shows the relative costs of the jumper variant and the driver-aide problem for each instance. A number higher (lower) than 100% indicates a higher (lower) cost than the TSP variant. Overall, the driver-aide problem has lower costs in 102 out of the 120 test instances. Compared to the TSP variant, the driver-aide problem has about a 13.6% lower cost on average. In the best-case scenario, the driver-aide problem can incur as much as 31.6% savings in cost.

To better understand this result, we use the average relative cost in Table 1 for comparison and provide the lowest cost option. In each cell, "TSP" means that the TSP variant has a lower cost, and "DA" means that the driver-aide problem has a lower one. It should be clear that the driver-aide problem is an option worthy of consideration in practice. It has lower costs and is chosen in

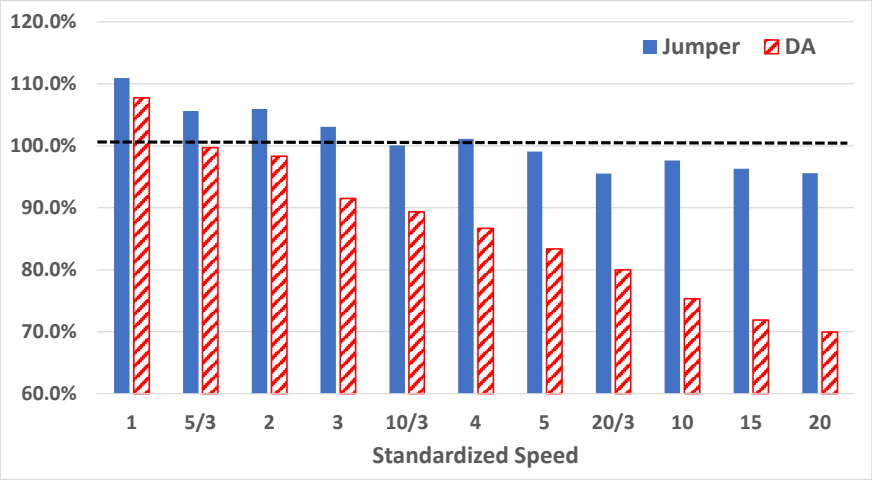



**Figure 10** **Average Relative Costs of the Jumper Variant and the Driver-Aide Problem Compared to the TSP Variant under Different Standardized Speeds.**

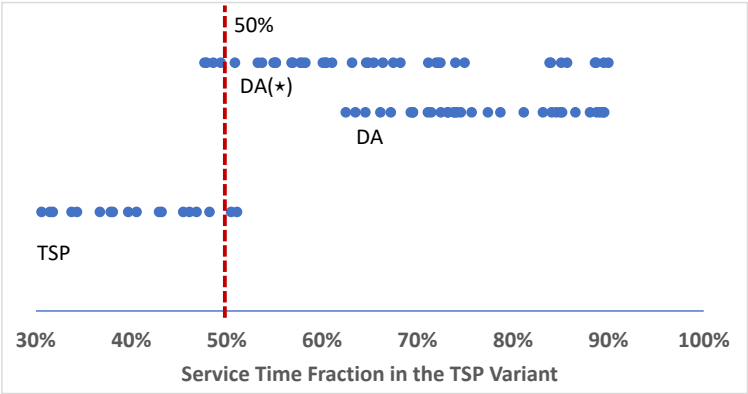



**Figure 11** **Threshold for Preferred Options Based on the Service Time Fraction in the TSP Variant.**

11 out of 12 situations. Furthermore, six out of the 11 "DA" cells have "⋆"s, which indicate that while the jumper variant does not have a lower cost than the TSP variant in the combination of the service region and travel speed, the helper mode is needed to make the driver-aide problem superior to the TSP variant. Over these 11 situations, on average, the driver-aide problem can incur 15.5% cost savings.

Figure 10 visualizes the comparison using the standardized speeds. We also add a reference line, which represents the cost of the TSP variant. Again, it reveals that the driver-aide problem leads to higher cost savings when the standardized speed increases. To summarize, the driver-aide problem provides cost savings in most situations, especially in situations where the service region is denser, and the travel speed is high. While the jumper-variant seems to require both situations to provide cost savings, the additional helper mode enables cost savings even with only one of the two situations. We note that we do not include the cost of the helper mode transport (e.g., e-scooter, e-bike). The main reason is that, comparatively, we expect the cost of this transport to be a negligible fraction of that of the truck, and thus would not affect our findings.

We want to further understand the situations leading to the preferred options for the 120 instances. We first calculate the travel and service times of the TSP variant for each instance. Then, we obtain the fraction of the service time of the TSP optimal route. Figure 11 plots our findings. The x-axis is the service time fraction of the TSP optimal tour in the TSP variant. We group all instances into two groups by their preferred options. Recall that "TSP" means that the TSP variant has a lower cost, and "DA" means that the driver-aide problem has a lower one. Within the "DA" group, we separate by "DA($\star$)" those instances where the helper mode was necessary to beat the TSP variant (i.e., the jumper variant has greater cost than the TSP variant). When the service time fraction is larger than 50%, the driver-aide problem with both the jumper and helper modes is generally preferred over the TSP variant. However, when the service time fraction is smaller than 50%, the driver-aide problem is not as attractive as before. The 50% threshold predicts 100 instances to be preferred by the driver-aide problem, and 98 of them are correct. Thus, this simple threshold's precision is 98%, and its recall is 96% (98/102). The overall accuracy is 95% (114/120). Interestingly, without the helper mode there is no clear-cut threshold separating the TSP and jumper variants. While the 50% threshold is based on our simulated test instances, a delivery company can conduct this analysis to obtain area-dependent thresholds in practice (which are likely to be close to this 50% threshold). We note that in a recent study, Allen et al. (2018) observed that the service time in London accounted for 62% of the total route time. Thus, it is likely that the 50% threshold will be fairly easy to satisfy in practice. Overall, this provides some simple rules of thumb for selecting the best option to minimize the cost.

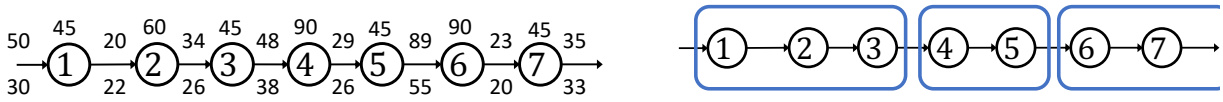
We can take this analysis further if we use average solution information to identify the thresholds that make the TSP variant, jumper variant, or driver-aide problem more cost-effective. Section EC.4 provides a detailed derivation. A delivery company usually knows its own average travel time, denoted by $\bar{t}$, between successive locations and the average service time, denoted by $\bar{s}$, at locations of the traditional TSP variant. We can approximate the cost of the TSP variant as $C_{TSP} = (n\bar{t} + n\bar{s})(\beta_T + \beta_D)$, where $\beta_T$ and $\beta_D$ are the costs for the truck and the driver, respectively. Similarly, the cost of the jumper variant is $C_{Jumper} = (n\bar{t} + n\bar{s}(1-\bar{f}))(\beta_T + \beta_D + \beta_A)$, where $\beta_A$ is the cost for the aide. Thus, for the jumper variant to have a lower cost than the TSP variant, we need $\bar{f} \geq \frac{\beta_A(\bar{t}+\bar{s})}{\bar{s}(\beta_T+\beta_D+\beta_A)} = \theta_J$. This provides a threshold $\theta_J$ that the jumper savings factor must be greater than for the jumper variant to be cheaper than the TSP variant. While this is a simple rule, applying it to the 120 instances (see Tables EC.6 and EC.7) correctly identifies all 52 instances that prefer the jumper variant over the TSP variant (i.e., its accuracy is 100% on the instances in this paper), indicating that it is fairly robust and reliable.

If $\bar{f} < \frac{\beta_A(\bar{t}+\bar{s})}{\bar{s}(\beta_T+\beta_D+\beta_A)}$, we may compare the cost of the driver-aide problem to that of the TSP variant directly. We can also approximate the cost of the driver-aide problem as $C_{DA} = ((J +$

$H)\bar{t}_{DA} + J\bar{s}_{DA}(1-\bar{f}_{DA}) + \Delta H)(\beta_T + \beta_D + \beta_A)$, where $J$, $H$, and $D$ are the number of nodes serviced in the jumper mode, helper mode and by the driver alone, respectively, $\Delta$ is the average time spent at a node with the helper mode, $\bar{t}_{DA}$ is the average travel time between successive nodes when the aide is on the truck in the driver-aide problem, and $\bar{s}_{DA}$ is the average service time among nodes when the aide is on the truck in the driver-aide problem. For the driver-aide problem to be more cost-effective than the TSP variant, we need $\Delta \leq \frac{(n\bar{t}+n\bar{s})(\beta_T+\beta_D)}{(\beta_T+\beta_D+\beta_A)H} - \frac{(J+H)\bar{t}_{DA}+J\bar{s}_{DA}(1-\bar{f}_{DA})}{H} = \rho_H$. The threshold $\rho_H$ provides the time limit under which the driver needs to travel to another node (or nodes), complete service, and return to the location where the driver-aide awaits. Smaller values of this threshold indicate fewer opportunities for cost savings (and instances where the TSP solution is of the least cost). This threshold has over 97% (117/120) accuracy based on the 120 instances (see Tables EC.6 and EC.7). Regarding the DA-preferred mode, it predicts 101 instances for the driver-aide problem, and 100 of them are correct. Thus, its precision and recall are over 99% (100/101) and 98% (100/102), respectively. After the driver-aide problem is implemented widely, a delivery company would have empirical estimates of these parameters based on the collected data. Thus, although this threshold depends on the average solution metrics of the driver-aide problem, they provide useful guidelines for a delivery company to follow when designing routes.

The cost analysis so far underestimates the benefit of the driver-aide problem with both the jumper and helper modes. First, we have not considered the restriction on the route time (i.e., the maximum work hours). In practice, a regular work shift is often between five to eight hours. The driver and the aide should receive an overtime pay rate if their work shift is longer than the regular work shift. In the U.S., the overtime pay rate is at least 1.5 times of the regular pay rate. Assume that the routing time of the driver-aide problem is normalized to be the maximum duration of a regular shift. Then, the overtime marginal cost per hour of the truck- and driver-based costs is \$90.93 in 2021 because the truck-based cost is \$42.10, and the driver-based cost is \$32.55 $\times$ 1.5 = \$48.83 for overtime hours. Table EC.8 shows the relative costs of the driver-aide problem, compared to the TSP variant, considering the overtime pay rate. They are calculated as \$90.65/(\$74.65 + \$90.93*($z_{TSP}/z_{DA} - 1$)). For example, for the cell with 5-Mile X 5-Mile and 5 MPH, the value is calculated as \$90.65/(\$74.65 + \$90.93*0.13) = 105.2%. In this case, the driver-aide problem with both the jumper and helper modes has a greater advantage and leads to an 18.4% cost savings, about 35% more savings than that of the regular pay rate.

A crucial point we would like to emphasize is that relying on overtime shifts is not sustainable for any business. There are many government regulations on the maximum number of overtime hours. Therefore, the only option for a company facing the TSP variant (i.e., which does not employ driver-aides) of the problem is to purchase more trucks and hire more drivers. We do not conduct this type of analysis, as its results would only skew the results further in favor of the driver-aide




(a) A Set of Customer Locations. (b) The Resulting Stops (Nodes)

**Figure 12 Illustration of the Clustering Procedure.**

problem. From a macro level, we can consider the normalized routing time in Table EC.4 as the relative fleet size between the TSP variant and the driver-aide problem with both the jumper and helper modes. The driver-aide problem provides an opportunity to reduce the fleet size, which can significantly lower a company's infrastructure and operating costs.

## 5. Case Study: Amazon's Last-Mile Routing

To further evaluate the benefit of optimizing the driver-aide routes, we conduct a case study utilizing real-world routes released by Amazon. In the 2021 "Last Mile Routing Research Challenge" organized by Amazon and MIT, Amazon released over 6,000 routes to the public. The details on this dataset are described in Merchán et al. (2022). We used 17 routes the winning team, Just Passing Through, selected and visualized on their website (Cook et al. 2021, 2022).

For each of the 17 routes, we cluster the customer locations into the stops (nodes) used in the driver-aide problem. We assume that an e-bike (Toll 2022, Sutton 2022, FedEx 2020, CBSNews 2022) is used by the aide in the helper mode. The dataset from Amazon contains truck travel time between all customer locations and the time needed in each location. We use Google Maps API to obtain the biking time between all customer locations. Then, following the visiting order of the customers given by the Amazon route (which is included in the dataset), if the biking time is more than 45 seconds, we stop including the next customer in the current stop and use that customer as the first one in the next stop. Figure 12 illustrates the clustering procedure. Figure 12(a) shows a set of customer locations. The numbers above and below the edges are the biking and trucking times, respectively. The numbers above the customer locations are the times needed to serve them. Figure 12(b) shows that we have formed two stops already, which are displayed as two blue rectangles. We use the visiting order of the Amazon route as the order for visiting customer locations within a stop, either by truck or by bike. Thus, we have the first and last customer locations as the start and end points at this stop, respectively. The service time of a stop is the sum of the total truck travel time and the time needed for the customer locations within that stop. Similarly, the time of the helper mode is the sum of the total bike travel time and the time needed for the customer locations. Specifically, for the stop consisting of customers 1, 2, and 3, its start point and end point are nodes 1 and 3, respectively. Further, its service time is 198 seconds, which increases to 204 seconds in the helper mode. The only thing left is the jumper saving factor, which is uniformly selected from $\{0, 0.1, 0.2, 0.3, 0.4, 0.5\}$ for each stop as before. For each of the 17 routes,

| | | | Average over Five Instances | | | |
|---|---|---|---|---|---|---|
| Route_ID | Customers | Stops | Gap | Run Time | Reduction in Routing Time | Savings in Cost |
| #0002 | 128 | 66 | 0.6% | 633.5 | 32.9% | 18.5% |
| #0003 | 142 | 41 | 0.6% | 546.1 | 37.3% | 23.9% |
| #0012 | 173 | 52 | 1.0% | 84.2 | 30.1% | 15.2% |
| #0019 | 157 | 46 | 0.9% | 78.3 | 35.1% | 21.2% |
| #0020 | 174 | 77 | 0.6% | 927.4 | 33.7% | 19.4% |
| #0023 | 139 | 66 | 0.3% | 108.8 | 29.2% | 14.0% |
| #0038 | 121 | 34 | 1.3% | 78.0 | 42.2% | 29.8% |
| #0062 | 167 | 66 | 1.2% | 145.6 | 31.4% | 16.7% |
| #0138 | 159 | 47 | 0.7% | 242.8 | 42.2% | 29.8% |
| #0145 | 144 | 68 | 0.8% | 415.5 | 35.8% | 22.0% |
| #0149 | 111 | 69 | 1.6% | 254.7 | 33.1% | 18.8% |
| #0180 | 151 | 71 | 0.7% | 353.6 | 39.6% | 26.7% |
| #0303 | 84 | 34 | 0.5% | 62.9 | 42.6% | 30.3% |
| #0307 | 151 | 69 | 0.4% | 301.6 | 34.3% | 20.2% |
| #0346 | 179 | 69 | 0.5% | 442.4 | 38.4% | 25.2% |
| #0431 | 125 | 46 | 1.6% | 174.5 | 31.4% | 16.7% |
| #0684 | 111 | 58 | 0.6% | 116.9 | 39.2% | 26.2% |

**Table 2 Summarized Results on the Amazon Instances.**

we generate five instances (by uniformly selecting the jumper savings factor). Thus, there are 85 instances in total, referred to as Amazon instances.

We present summarized results on the Amazon instances in Table 2. It contains the routes' ID in the column "Route_ID", the number of customer locations in "Customers", and the number of stops in the driver-aide problem after clustering in "Stops". The largest instance has 77 stops, which is much larger than the instances tested in Section 4. Then, it presents the average optimality gap and running time over the five instances generated for each route in the columns "Gap" and "Run Time", respectively. The proposed branch-cut-and-price approach can find near-optimal solutions efficiently. Overall, on average, the optimality gap is 0.8%, and the running time is 292.2 seconds. It also shows the average reduction in the routing time and cost savings (based on the regular pay rate), compared to the original Amazon routes. The driver-aide problem can reduce the routing time significantly, between 29.2% and 42.6% on average. Consequently, the average cost savings are between 14% and 29.8%. Overall, the Amazon instances show even larger benefits (a 35% reduction in the routing time and 22% in cost savings), compared to the simulated instances (a 29% reduction in the routing time and 14% in cost savings) in Section 4. Detailed results are in Tables EC.9 and EC.10. Finally, we comment that the comparative metrics also work well on the Amazon instances (see Tables EC.11 and EC.12). All of these reinforce the findings in Section 4.

## 6. Conclusions

Package volumes have grown exponentially over the past several years. Last-mile delivery comprises a significant portion of the cost in a logistic network. Moreover, logistics companies are under significant pressure to contain these costs while maintaining service and delivery commitments. To address the increased workload without additional investments of delivery trucks and drivers, last-mile delivery companies are hiring driver-aides to help speed up or shorten the service component

of delivery routes. In this paper, we introduced the "Driver-Aide Problem". We provide a novel IP formulation and a sophisticated branch-cut-and-price procedure that finds near-optimal solutions to instances of real-world size.

Our economic analysis considers the tradeoffs involved in using a driver-aide and provides a high-level understanding of when it is most beneficial to use one. Roughly, delivery routes that have greater than 50% of the time devoted to delivery (as opposed to driving) are the ones that provide the greatest benefit. Further, these routes are characterized by a high density of delivery locations. With regard to using the aide as a jumper or a helper, there seems to be greater use of the helper mode when the average vehicle speed is somewhat higher, as it allows the driver to simultaneously perform one or more deliveries and return to the driver-aide in about the same time it takes the aide to do his/her delivery. We also conduct a case study on the dataset released by Amazon to further demonstrate the impact of optimally using the driver-aide.

Looking ahead, we hope to study certain multiple delivery truck variants of the problem. It is envisioned that in the future, a driver-aide may be able to assist multiple trucks/drivers, with the idea that an aide dropped off by one truck may then be picked up by another truck (and then assist the driver in the new truck). This creates significant modeling and computational challenges that we see as an avenue for further research.

## References

Allen, J., Piecyk, M., Piotrowska, M., McLeod, F., Cherrett, T., Ghali, K., Nguyen, T., Bektas, T., Bates, O., Friday, A., et al. (2018). Understanding the impact of e-commerce on last-mile light goods vehicle activity in urban areas: The case of London. *Transportation Research Part D: Transport and Environment*, 61:325–338.

Amazon (2022). Amazon Prime Air. `https://www.amazon.com/Amazon-Prime-Air/b?ie=UTF8&node=8037720011`.

Barnhart, C., Johnson, E. L., Nemhauser, G. L., Savelsbergh, M. W., and Vance, P. H. (1998). Branch-and-price: Column generation for solving huge integer programs. *Operations Research*, 46(3):316–329.

CBSNews (2022). UPS tests electric cargo bicycles in congested cities. `https://www.cbsnews.com/news/ups-tests-tiny-battery-powered-bicycles-in-congested-cities/`.

Chung, S. H., Sah, B., and Lee, J. (2020). Optimization for drone and drone-truck combined operations: A review of the state of the art and future directions. *Computers & Operations Research*, 123:105004.

Coindreau, M.-A., Gallay, O., and Zufferey, N. (2019). Vehicle routing with transportable resources: Using carpooling and walking for on-site services. *European Journal of Operational Research*, 279(3):996–1010.

Cook, W., Held, S., and Helsgaun, K. (2021). Just passing through. `https://www.math.uwaterloo.ca/tsp/amz/maps.html`.

Cook, W., Held, S., and Helsgaun, K. (2022). Constrained local search for last-mile routing. *Transportation Science*. https://doi.org/10.1287/trsc.2022.1185.

Costa, L., Contardo, C., and Desaulniers, G. (2019). Exact branch-price-and-cut algorithms for vehicle routing. *Transportation Science*, 53(4):946–985.

Desaulniers, G., Desrosiers, J., and Solomon, M. M. (2006). *Column Generation*. Springer, New York, USA.

Elite Extra (2022). The impact of last mile delivery driver shortage. `https://eliteextra.com/the-impact-of-last-mile-delivery-driver-shortage`.

FedEx (2020). FedEx steps and pedals into Europe's green delivery future. `https://www.fedex.com/en-us/sustainability/fedex-pilots-program-in-europe-to-improve-sustainable-deliveries.html`.

Fikar, C. and Hirsch, P. (2015). A matheuristic for routing real-world home service transport systems facilitating walking. *Journal of Cleaner Production*, 105:300–310.

Florio, A. M., Hartl, R. F., and Minner, S. (2020). New exact algorithm for the vehicle routing problem with stochastic demands. *Transportation Science*, 54(4):1073–1090.

Garey, M. R. and Johnson, D. S. (1979). *Computers and Intractability: A Guide to the Theory of NP-Completeness*. Freeman, San Francisco, USA.

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

Grötschel, M., Jünger, M., and Reinelt, G. (1985). On the acyclic subgraph polytope. *Mathematical Programming*, 33(1):28–42.

Lawler, E. L., Lenstra, J. K., Rinnooy Kan, A. H. G., and Shmoys, D. B. (1985). *The Traveling Salesman Problem: A Guided Tour of Combinatorial Optimization*. Wiley, Hoboken, New Jersey.

Leslie, A. and Murray, D. (2022). An analysis of the operational costs of trucking: 2022 update. *American Transportation Research Institute*. `https://truckingresearch.org/wp-content/uploads/2022/08/ATRI-Operational-Cost-of-Trucking-2022.pdf`.

Lu, S.-H., Suzuki, Y., and Clottey, T. (2020). The last mile: Managing driver helper dispatching for package delivery services. *Journal of Business Logistics*, 41(3):206–221.

Lu, S.-H., Suzuki, Y., and Clottey, T. (2022). Improving the efficiency of last-mile package deliveries using hybrid driver helpers. *Decision Sciences*. https://doi.org/10.1111/deci.12559.

Merchán, D., Arora, J., Pachon, J., Konduri, K., Winkenbach, M., Parks, S., and Noszek, J. (2022). 2021 Amazon last mile routing research challenge: Data set. *Transportation Science*.

Mor, A. and Speranza, M. G. (2022). Vehicle routing problems over time: A survey. *Annals of Operations Research*, 314:255–275.

Ngo, M. and Swanson, A. (2021). The biggest kink in America's supply chain: Not enough truckers. `https://www.nytimes.com/2021/11/09/us/politics/trucker-shortage-supply-chain.html`.

Otto, A., Agatz, N., Campbell, J., Golden, B., and Pesch, E. (2018). Optimization approaches for civil applications of unmanned aerial vehicles (UAVs) or aerial drones: A survey. *Networks*, 72(4):411–458.

Pataki, G. (2003). Teaching integer programming formulations using the traveling salesman problem. *SIAM Review*, 45(1):116–123.

Pferschy, U. and Staněk, R. (2017). Generating subtour elimination constraints for the TSP from pure integer solutions. *Central European Journal of Operations Research*, 25(1):231–260.

PYMNT (2020). Delivery van shortage shows ecommerce ripple effects along last mile. https://www.pymnts.com/news/delivery/2020/delivery-van-shortage-shows-ecommerce-ripple-effects-along-last-mile/.

Reed, S., Campbell, A. M., and Thomas, B. W. (2022a). Impact of autonomous vehicle assisted last-mile delivery in urban to rural settings. *Transportation Science*, 56(6):1530–1548.

Reed, S., Campbell, A. M., and Thomas, B. W. (2022b). The value of autonomous vehicles for last-mile deliveries in urban environments. *Management Science*, 68(1):280–299.

Roberson, C. M. (2020). Global parcel volumes expected to double by 2026 on e-commerce boom. https://www.joc.com/international-logistics/global-parcel-volumes-expected-double-2026-e-commerce-boom_20201012.html.

Rosenberger, S. (2022). Last mile delivery landscape in the transportation sector. https://www2.deloitte.com/global/en/pages/consumer-business/articles/last-mile-delivery-landscape-transportation-sector.html.

Rostami, B., Desaulniers, G., Errico, F., and Lodi, A. (2021). Branch-price-and-cut algorithms for the vehicle routing problem with stochastic and correlated travel times. *Operations Research*, 69(2):436–455.

Sutton, M. (2022). Amazon turns to electric cargo bikes for city delivery. https://cyclingindustry.news/amazon-turns-to-electric-cargo-bikes-for-city-delivery/.

Toll, M. (2022). USPS already testing mail delivery by electric bike with these neat little us-built mail bikes. https://electrek.co/2022/06/03/usps-mail-delivery-electric-mail-bike/#:~:text=As%20it%20turns%20out%2C%20the,by%20Montana%2Dbased%20Coaster%20Cycles.

Toth, P. and Vigo, D. (2002). *The Vehicle Routing Problem*. SIAM, Philadelphia, USA.

Toth, P. and Vigo, D. (2014). *Vehicle Routing: Problems, Methods, and Applications*. SIAM, Philadelphia, USA.

United Nation (2021). Global e-commerce jumps to $26.7 trillion, fuelled by COVID-19. https://news.un.org/en/story/2021/05/1091182.

U.S. Bureau of Labor Statistics (2021). May 2020 national occupational employment and wage estimates United States.

U.S. Census Bureau (2022). Latest quarterly e-commerce report. https://www.census.gov/retail/index.html.

Williams, H. P. (2013). *Model Building in Mathematical Programming.* John Wiley & Sons. New York, USA.

Xu, D., Li, K., Zou, X., and Liu, L. (2017). An unpaired pickup and delivery vehicle routing problem with multi-visit. *Transportation Research Part E: Logistics and Transportation Review*, 103:218–247.

Younan, E. (2020). Last-mile delivery volumes are higher than ever. Will it last? `https://routetitan.com/en/blog/last-mile-delivery-volumes-are-higher-than-ever-will-it-last`.

# Electronic Companion

## EC.1. Proof of Proposition 1

*Proof:* If the optimal TSP route (called route 1) is not optimal, there exists a strictly better route (called route 2) for the driver-aide problem with only the jumper mode. Route 2 must have a shorter travel time than route 1, as the total service times of both routes are the same. Therefore, route 2 is a strictly better route than route 1 for the TSP, which is a contradiction. □

## EC.2. Proof of Proposition 2

*Proof:* First, given $S \subseteq C$ and $k \in S$, we add $|S|-1$ to both sides of $\sum_{i,j\in S:\{i,j\}\in A} x_{i,j} - \sum_{i\in S\setminus\{k\}}(h_i+z_i) \le 0$ to obtain $\sum_{i,j\in S:\{i,j\}\in A} x_{i,j} + \sum_{i\in S\setminus\{k\}}(1-h_i-z_i) \le |S|-1$. For a feasible solution of the driver-aide problem, if a node $i$ is served by the jumper or helper mode, we have $h_i+z_i=1$ and $\sum_{j\in n(i)^-} x_{j,i} = \sum_{j\in n(i)^+} x_{i,j} = 1$ because of constraints (18) and (19). Thus, we include node $i$'s adjacent arcs on the left-hand side. If a node $i$ is served by the driver alone, we have this node included in a loop by the corresponding $y$ variable. Thus, we have $h_i+z_i=0$, and $\sum_{j\in n(i)^-} x_{j,i} = \sum_{j\in n(i)^+} x_{i,j} = 0$ because of constraints (18) and (19). Then, we can reduce the right-hand side by 1 as $\sum_{j\in n(i)^-} x_{j,i} = \sum_{j\in n(i)^+} x_{i,j} = 0$. Given that node $k$ is removed from $S$, the largest value of $\sum_{i\in S\setminus\{k\}}(1-h_i-z_i)$ is $|S|-1$, which does not exceed the right-hand side value. Therefore, it ensures that for a subset of nodes in $C$, the graph induced by the $x$ variables does not have any cycles. Any feasible solution of the driver-aide problem satisfies $\sum_{i,j\in S:\{i,j\}\in A} x_{i,j} - \sum_{i\in S\setminus\{k\}}(h_i+z_i) \le 0, \forall S \subseteq C, |S| \ge 2, k \in S$.

Next, notice that combining constraints (17) and (19) gives $h_i + z_i = 1 - \sum_{p\in L: i\in Q_p} y_p \le 1$, as node $i$ is included in at most one loop, $\sum_{p\in L:i\in Q_p} y_p \le 1$. Thus, we have $\sum_{i,j\in S:\{i,j\}\in A} x_{i,j} \le \sum_{i,j\in S:\{i,j\}\in A} x_{i,j} + \sum_{i\in S\setminus\{k\}}(1-h_i-z_i) \le |S|-1$, as $1-h_i-z_i \ge 0$. It shows that $\sum_{i,j\in S:\{i,j\}\in A} x_{i,j} - \sum_{i\in S\setminus\{k\}}(h_i+z_i) \le 0, \forall S \subseteq C, |S| \ge 2, k \in S$ dominates constraint (21), $\sum_{i,j\in S:\{i,j\}\in A} x_{i,j} \le |S| - 1, \forall S \subseteq C, |S| \ge 2$. □

## EC.3. The Pseudocode for the Branch-Cut-and-Price Approach

**Algorithm 1** The Branch-Cut-and-Price Approach for the Driver-Aide Problem

Initialize UB (upper bound) as $\infty$ and LB (lower bound) as $-\infty$
Run (Min_Initial$_i$) and (Max_Initial$_i$) for all $i$ in $C$ for enhanced initial loops ($y$ variables)
Initialize IP3R and its relaxation LP3R with both of the basic and enhanced initial loops ($y$ variables)
Execute `RootNodeFunction()` described in Algorithm 2
Let RunningTime track the running time, and TimeLimit and ST (stopping tolerance) be user specified input parameters
**while** (UB - LB)/UB $>$ ST and RunningTime $\leq$ TimeLimit **do**
  Execute `BranchingPhaseFunction()` described in Algorithm 3
**end while**

**Algorithm 2** The Procedure at the Root Node (`RootNodeFunction`)

**function** RootNodeFunction( )
  Set stopFlag as 0 and countFlag as 0
  **while** stopFlag is 0 **do**
    Solve LP3R to obtain a solution $(\mathbf{h}^*, \mathbf{x}^*, \mathbf{y}^*, \mathbf{z}^*)$ with objective value $z_{lb}$
    Run *Enhanced Row Generation* for violated constraint (37). Specifically, for each $i$ in $C$, run a shortest path algorithm. If no violated constraints are found, run (SepEX).
    **if** the solution is integral and no violated constraints are found **then**
      A feasible solution has been found. Let its objective value be $z$
      If z $<$ UB, set UB $\leftarrow z$. Then, if (UB - LB)/UB $\leq$ ST, set stopFlag as 1.
    **else if** Violated constraints are found **then**
      Include the newly found constraint (37) to LP3R
    **end if**
    Run *Enhanced Column Generation* for promising $y$ variables. Specifically,
    **procedure** Enhanced Column Generation
      **if** countFlag $<$ 5 **then**
        for each $i$ in $C$, run (Pricing$_i$) with a time limit of 5 seconds.
      **end if**
      **if** no promising $y$ variables are found **then**
        run (StrongPricing).
      **end if**
      **if** Some promising $y$ variables are found **then**
        Include the newly found $y$ variables to LP3R
      **end if**
    **end procedure**
    **if** no violated constraints or promising $y$ variables are found **then**
      set stopFlag as 1.
    **end if**
    **if** stopFlag is 1 **then**
      Apply *Branching Rule* to create two nodes and include the two nodes in the list *currentLevel*
    **else**
      countFlag $+= 1$
    **end if**
  **end while**
  Set LB $\leftarrow z_{lb}$. Then, if (UB - LB)/UB $\leq$ ST, set stopFlag as 1.
**end function**

**Algorithm 3** The Procedure for the Branching Phase (`BranchingPhaseFunction`)

**function** BRANCHINGPHASEFUNCTION( )
  Initialize an empty list *nextLevel* and set stopFlag as 0
  **while** stopFlag is 0 **do**
    **for** each node in *currentLevel* **do**
      Solve LP3R at the node to obtain a solution $(\mathbf{h}^*, \mathbf{x}^*, \mathbf{y}^*, \mathbf{z}^*)$
      Run *Enhanced Row Generation* for violated constraint (37). Specifically, for each $i$ in $C$,
      run a shortest path algorithm. If no violated constraints are found, run (SepEX).
      **if** the solution is integral and no violated constraints are found **then**
        A feasible solution has been found. Let its objective value be $z$
        If z < UB, set UB $\leftarrow z$. Then, if (UB - LB)/UB $\leq$ ST, set stopFlag as 1 and execute **Break**
      **else if** Violated constraints are found **then**
        Include the newly found constraint (37) to LP3R
      **end if**
      Run (StrongPricing) for a promising $y$ variable
      **if** a promising $y$ variable is found **then**
        Include the newly found $y$ variables to LP3R
      **end if**
      If no violated constraints or any promising $y$ variable is found,
      apply *Branching Rule* to create two nodes and include the two nodes in the list *nextLevel*.
    **end for**
    Let $z_{lb}$ be the smallest objective value for nodes in *currentLevel*. If $z_{lb}$ > LB, set LB $\leftarrow z_{lb}$.
    Then, if (UB - LB)/UB $\leq$ ST, set stopFlag as 1.
    Set *currentLevel* $\leftarrow$ *nextLevel*
  **end while**
**end function**

## EC.4. Comparative Statics Calculation

We first introduce some notation. Let $\bar{t}$ be the average travel time between nodes in the TSP/Jumper variant, and $\bar{s}$ be the average service time among nodes in the TSP/Jumper variant. Similarly, let $\bar{t}_{DA}$ be the average travel time between nodes when the aide is on the truck in the driver-aide problem, and $\bar{s}_{DA}$ be the average service time among nodes when the aide is on the truck in the driver-aide problem. Further, let $\beta_T$, $\beta_D$, and $\beta_A$ denote the cost (per unit time) for the truck, driver, and aide, respectively. Let $\Delta$ be the average time between the truck dropping off and picking up the aide at a node served by the helper mode. Let $J$, $H$, and $D$ be the number of nodes served by the jumper mode, helper mode, and driver alone, respectively, and $\gamma$ be the average number of nodes served by the driver alone after the aide is dropped off as a helper. Thus, we can approximate the costs for the TSP, jumper variants, and the driver-aide problem as follows:

$$C_{TSP} = (n\bar{t} + n\bar{s})(\beta_T + \beta_D) \tag{EC.1}$$

$$C_{Jumper} = (n\bar{t} + n\bar{s}(1-\bar{f}))(\beta_T + \beta_D + \beta_A) \tag{EC.2}$$

$$C_{DA} = ((J+H)\bar{t}_{DA} + J\bar{s}_{DA}(1-\bar{f}_{DA}) + \Delta H)(\beta_T + \beta_D + \beta_A) \tag{EC.3}$$

For the jumper variant to be more cost-effective than the TSP variant, we need to satisfy

$$C_{TSP} \geq C_{Jumper} \tag{EC.4}$$

$$\implies (n\bar{t} + n\bar{s})(\beta_T + \beta_D) \geq (n\bar{t} + n\bar{s}(1-\bar{f}))(\beta_T + \beta_D + \beta_A) \tag{EC.5}$$

$$\implies \bar{t}\beta_T + \bar{s}\beta_T + \bar{t}\beta_D + \bar{s}\beta_D \geq \bar{t}\beta_T + \bar{t}\beta_D + \bar{t}\beta_A + \bar{s}\beta_T + \bar{s}\beta_D + \bar{s}\beta_A - \bar{s}\bar{f}\beta_T - \bar{s}\bar{f}\beta_D - \bar{s}\bar{f}\beta_A \tag{EC.6}$$

$$\implies \bar{s}\bar{f}(\beta_T + \beta_D + \beta_A) \geq \bar{t}\beta_A + \bar{s}\beta_A \tag{EC.7}$$

$$\implies \bar{f} \geq \frac{\beta_A(\bar{t} + \bar{s})}{\bar{s}(\beta_T + \beta_D + \beta_A)} \tag{EC.8}$$

We refer to the right-hand side of (EC.8) as $\theta_J$.

Moreover, we can derive the following condition that shows when the driver-aide route is more cost-effective than the TSP variant:

$$C_{TSP} \geq C_{DA} \tag{EC.9}$$

$$\implies (n\bar{t} + n\bar{s})(\beta_T + \beta_D) \geq ((J+H)\bar{t}_{DA} + J\bar{s}_{DA}(1-\bar{f}_{DA}) + \Delta H)(\beta_T + \beta_D + \beta_A) \tag{EC.10}$$

$$\implies \Delta H \leq \frac{(n\bar{t} + n\bar{s})(\beta_T + \beta_D)}{(\beta_T + \beta_D + \beta_A)} - (J+H)\bar{t}_{DA} - J\bar{s}_{DA}(1-\bar{f}_{DA}) \tag{EC.11}$$

$$\implies \Delta \leq \frac{(n\bar{t} + n\bar{s})(\beta_T + \beta_D)}{(\beta_T + \beta_D + \beta_A)H} - \frac{(J+H)\bar{t}_{DA} + J\bar{s}_{DA}(1-\bar{f}_{DA})}{H} \tag{EC.12}$$

The left-hand side of (EC.11) shows the time spent in helper mode. On the right-hand side of (EC.11), the first term shows the maximum time possible for a driver-aide instance, considering the cost difference, the second term is the travel time when the aide is on the truck, and the third term is the service time when the aide is in jumper mode. Let $\rho_H$ denote the right-hand side of (EC.12).

## EC.5. Detailed Computational Results

| 1-Mile X 1-Mile | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| | 5 MPH | | 10 MPH | | 15 MPH | | 20 MPH | |
| | Gap | Time | Gap | Time | Gap | Time | Gap | Time |
| 1 | 1.1% | 150.2 | 1.4% | 237.6 | 1.2% | 299.3 | 0.8% | 348.9 |
| 2 | 0.8% | 159.2 | 1.6% | 531.7 | 1.0% | 1088.3 | 0.6% | 626.5 |
| 3 | 1.9% | 110.8 | 1.6% | 132.9 | 1.2% | 196.0 | 1.2% | 241.8 |
| 4 | 1.7% | 71.5 | 0.8% | 225.3 | 1.1% | 1320.5 | 0.7% | 243.8 |
| 5 | 1.0% | 75.8 | 0.2% | 222.6 | 0.0% | 314.0 | 0.6% | 334.8 |
| 6 | 1.4% | 69.9 | 0.9% | 364.2 | 0.5% | 256.3 | 0.6% | 1350.9 |
| 7 | 1.9% | 42.9 | 0.9% | 96.5 | 0.9% | 114.2 | 0.5% | 118.4 |
| 8 | 0.0% | 41.9 | 0.1% | 68.2 | 0.9% | 117.2 | 0.0% | 135.5 |
| 9 | 1.1% | 65.8 | 0.8% | 109.3 | 0.6% | 158.3 | 1.1% | 160.9 |
| 10 | 2.0% | 49.2 | 1.0% | 100.4 | 0.7% | 97.1 | 0.9% | 98.6 |
| Avg | 1.3% | 83.7 | 0.9% | 208.9 | 0.8% | 396.1 | 0.7% | 366.0 |
| 3-Miles X 3-Miles | | | | | | | | |
| | 5 MPH | | 10 MPH | | 15 MPH | | 20 MPH | |
| | Gap | Time | Gap | Time | Gap | Time | Gap | Time |
| 1 | 1.8% | 37.2 | 0.7% | 60.9 | 1.7% | 92.8 | 2.0% | 112.3 |
| 2 | 1.2% | 267.9 | 0.6% | 255.8 | 1.5% | 606.1 | 1.4% | 765.0 |
| 3 | 1.7% | 20.8 | 1.8% | 34.2 | 1.9% | 54.4 | 1.9% | 37.3 |
| 4 | 1.8% | 188.3 | 1.3% | 761.6 | 1.1% | 712.8 | 1.1% | 1128.9 |
| 5 | 0.4% | 76.9 | 1.1% | 177.1 | 0.8% | 264.3 | 1.6% | 344.9 |
| 6 | 1.3% | 43.1 | 0.9% | 65.6 | 1.2% | 82.6 | 1.6% | 110.2 |
| 7 | 1.1% | 196.7 | 1.3% | 61.1 | 0.8% | 65.5 | 1.8% | 94.5 |
| 8 | 1.9% | 206.4 | 1.5% | 233.8 | 1.9% | 179.7 | 2.0% | 102.2 |
| 9 | 2.0% | 42.7 | 1.9% | 65.5 | 1.8% | 35.3 | 1.8% | 45.0 |
| 10 | 0.7% | 32.4 | 1.6% | 57.8 | 1.2% | 72.3 | 0.7% | 112.9 |
| Avg | 1.4% | 111.2 | 1.3% | 177.3 | 1.4% | 216.6 | 1.6% | 285.3 |
| 5-Miles X 5-Miles | | | | | | | | |
| | 5 MPH | | 10 MPH | | 15 MPH | | 20 MPH | |
| | Gap | Time | Gap | Time | Gap | Time | Gap | Time |
| 1 | 1.7% | 446.4 | 1.9% | 134.4 | 1.9% | 49.0 | 1.0% | 81.6 |
| 2 | 1.0% | 75.8 | 1.2% | 48.4 | 1.9% | 61.7 | 0.7% | 127.5 |
| 3 | 1.5% | 159.8 | 1.6% | 44.0 | 1.8% | 37.9 | 1.7% | 45.4 |
| 4 | 1.3% | 52.0 | 2.0% | 47.9 | 1.7% | 62.3 | 2.0% | 99.8 |
| 5 | 0.3% | 100.1 | 0.3% | 129.6 | 1.2% | 76.4 | 1.6% | 119.3 |
| 6 | 0.7% | 44.8 | 1.4% | 63.1 | 1.6% | 125.8 | 1.2% | 220.1 |
| 7 | 1.9% | 266.1 | 1.8% | 126.8 | 1.9% | 89.8 | 1.2% | 52.6 |
| 8 | 1.2% | 219.6 | 1.5% | 103.8 | 1.9% | 41.0 | 1.7% | 62.3 |
| 9 | 0.1% | 136.2 | 0.1% | 225.1 | 0.1% | 54.5 | 0.4% | 89.5 |
| 10 | 2.0% | 337.8 | 1.9% | 89.7 | 2.0% | 41.6 | 1.6% | 52.3 |
| Avg | 1.2% | 183.9 | 1.4% | 101.3 | 1.6% | 64.0 | 1.3% | 95.0 |

**Table EC.1 Computational Performance (Optimality Gap and Time in Seconds) of the Branch-Cut-and-Price Approach with a 2% Stopping Tolerance.**

In Table EC.1, the first column shows the instance ID. For each region and travel speed, we report the optimality gap in "Gap" and the running time in "Time". In addition, we report the average values in the row "Avg".

| | 5 MPH | | | | 10 MPH | | | |
|---|---|---|---|---|---|---|---|---|
| | 1-Mile X 1-Mile | | | | 1-Mile X 1-Mile | | | |
| | TSP | Jumper | DA_UB | DA_LB | TSP | Jumper | DA_UB | DA_LB |
| 1 | 15955 | 13666 | 11022 | 10901 | 13876 | 11589 | 8620 | 8496 |
| 2 | 16027 | 12511 | 10727 | 10638 | 13922 | 10410 | 8644 | 8509 |
| 3 | 14089 | 11029 | 9648 | 9460 | 12199 | 9140 | 7575 | 7455 |
| 4 | 14707 | 11546 | 9940 | 9770 | 12677 | 9520 | 7823 | 7759 |
| 5 | 16081 | 14027 | 10593 | 10487 | 14066 | 12012 | 8337 | 8319 |
| 6 | 13771 | 12060 | 9637 | 9504 | 11853 | 10143 | 7431 | 7363 |
| 7 | 14085 | 12293 | 9632 | 9450 | 12140 | 10348 | 7513 | 7445 |
| 8 | 13467 | 11217 | 9437 | 9437 | 11521 | 9272 | 7433 | 7423 |
| 9 | 15660 | 12805 | 10581 | 10466 | 13631 | 10777 | 8213 | 8147 |
| 10 | 13424 | 11156 | 9670 | 9480 | 11286 | 9020 | 7190 | 7121 |
| | 3-Miles X 3-Miles | | | | 3-Miles X 3-Miles | | | |
| | TSP | Jumper | DA_UB | DA_LB | TSP | Jumper | DA_UB | DA_LB |
| 1 | 24533 | 21168 | 19236 | 18886 | 18799 | 15433 | 12893 | 12798 |
| 2 | 23894 | 20948 | 19551 | 19313 | 17747 | 14803 | 12721 | 12639 |
| 3 | 20428 | 17780 | 17203 | 16911 | 14856 | 12208 | 11414 | 11214 |
| 4 | 22436 | 18996 | 18161 | 17835 | 16563 | 13123 | 11920 | 11768 |
| 5 | 26083 | 21991 | 20131 | 20052 | 20232 | 16140 | 14031 | 13879 |
| 6 | 23753 | 20525 | 19194 | 18939 | 17735 | 14509 | 12705 | 12595 |
| 7 | 20807 | 18772 | 17319 | 17122 | 15197 | 13164 | 11434 | 11286 |
| 8 | 23907 | 21047 | 19656 | 19291 | 18033 | 15174 | 13391 | 13184 |
| 9 | 20127 | 17604 | 17482 | 17137 | 14380 | 11859 | 11280 | 11071 |
| 10 | 21201 | 18534 | 18110 | 17988 | 15177 | 12510 | 11776 | 11590 |
| | 5-Miles X 5-Miles | | | | 5-Miles X 5-Miles | | | |
| | TSP | Jumper | DA_UB | DA_LB | TSP | Jumper | DA_UB | DA_LB |
| 1 | 33305 | 30530 | 29772 | 29254 | 22952 | 20179 | 18655 | 18294 |
| 2 | 33187 | 29524 | 28454 | 28166 | 23324 | 19659 | 17657 | 17454 |
| 3 | 28987 | 26854 | 25930 | 25530 | 19382 | 17251 | 15963 | 15713 |
| 4 | 30796 | 28780 | 27120 | 26759 | 21262 | 19242 | 16495 | 16172 |
| 5 | 30858 | 28337 | 27954 | 27881 | 20719 | 18199 | 17473 | 17413 |
| 6 | 34248 | 30835 | 29567 | 29371 | 23921 | 20503 | 18823 | 18556 |
| 7 | 27137 | 24787 | 24573 | 24107 | 17877 | 15526 | 14851 | 14590 |
| 8 | 28532 | 26749 | 26099 | 25790 | 18625 | 16844 | 15976 | 15734 |
| 9 | 29763 | 26210 | 25609 | 25580 | 20339 | 16789 | 16072 | 16050 |
| 10 | 29621 | 27179 | 26584 | 26054 | 19474 | 17029 | 15902 | 15597 |

**Table EC.2 Routing Times in Seconds of the TSP, Jumper Variants, and the Driver-Aide Problem with 5 MPH and 10 MPH Speeds.**

.

| | 15 MPH | | | | 20 MPH | | | |
|---|---|---|---|---|---|---|---|---|
| | 1-Mile X 1-Mile | | | | 1-Mile X 1-Mile | | | |
| | TSP | Jumper | DA_UB | DA_LB | TSP | Jumper | DA_UB | DA_LB |
| 1 | 13185 | 10896 | 7784 | 7695 | 12839 | 10550 | 7386 | 7331 |
| 2 | 13226 | 9710 | 7779 | 7697 | 12876 | 9359 | 7332 | 7285 |
| 3 | 11570 | 8510 | 6866 | 6781 | 11255 | 8195 | 6464 | 6385 |
| 4 | 12005 | 8845 | 7198 | 7120 | 11667 | 8507 | 6798 | 6750 |
| 5 | 13394 | 11341 | 7602 | 7602 | 13058 | 11005 | 7321 | 7281 |
| 6 | 11215 | 9504 | 6601 | 6571 | 10895 | 9185 | 6279 | 6242 |
| 7 | 11492 | 9699 | 6839 | 6779 | 11168 | 9375 | 6392 | 6362 |
| 8 | 10874 | 8624 | 6690 | 6632 | 10549 | 8300 | 6248 | 6247 |
| 9 | 12958 | 10102 | 7403 | 7362 | 12621 | 9764 | 7112 | 7033 |
| 10 | 10577 | 8309 | 6485 | 6439 | 10221 | 7953 | 6071 | 6016 |
| | 3-Miles X 3-Miles | | | | 3-Miles X 3-Miles | | | |
| | TSP | Jumper | DA_UB | DA_LB | TSP | Jumper | DA_UB | DA_LB |
| 1 | 16886 | 13521 | 10947 | 10758 | 15931 | 12565 | 9968 | 9769 |
| 2 | 15700 | 12755 | 10743 | 10584 | 14676 | 11731 | 9565 | 9430 |
| 3 | 12997 | 10350 | 9290 | 9117 | 12068 | 9421 | 8189 | 8035 |
| 4 | 14605 | 11165 | 9953 | 9843 | 13627 | 10186 | 8846 | 8749 |
| 5 | 18282 | 14190 | 11927 | 11830 | 17307 | 13215 | 10987 | 10814 |
| 6 | 15732 | 12504 | 10565 | 10443 | 14729 | 11501 | 9518 | 9361 |
| 7 | 13329 | 11294 | 9240 | 9162 | 12394 | 10360 | 8365 | 8217 |
| 8 | 16076 | 13216 | 11269 | 11050 | 15097 | 12238 | 10063 | 9862 |
| 9 | 12467 | 9944 | 8933 | 8769 | 11510 | 8987 | 7860 | 7718 |
| 10 | 13170 | 10502 | 9338 | 9222 | 12166 | 9498 | 8218 | 8161 |
| | 5-Miles X 5-Miles | | | | 5-Miles X 5-Miles | | | |
| | TSP | Jumper | DA_UB | DA_LB | TSP | Jumper | DA_UB | DA_LB |
| 1 | 19503 | 16728 | 14509 | 14232 | 17778 | 15003 | 12490 | 12365 |
| 2 | 20034 | 16370 | 14148 | 13884 | 18390 | 14726 | 12301 | 12214 |
| 3 | 16183 | 14050 | 12433 | 12212 | 14582 | 12450 | 10724 | 10543 |
| 4 | 18080 | 16063 | 13157 | 12939 | 16491 | 14474 | 11468 | 11243 |
| 5 | 17341 | 14820 | 13675 | 13506 | 15652 | 13131 | 11761 | 11579 |
| 6 | 20471 | 17058 | 14807 | 14577 | 18749 | 15336 | 12699 | 12544 |
| 7 | 14789 | 12439 | 11527 | 11305 | 13245 | 10896 | 9694 | 9577 |
| 8 | 15326 | 13543 | 12449 | 12213 | 13675 | 11892 | 10649 | 10463 |
| 9 | 17202 | 13649 | 12487 | 12478 | 15632 | 12079 | 10659 | 10622 |
| 10 | 16088 | 13645 | 12227 | 11983 | 14396 | 11954 | 10278 | 10114 |

**Table EC.3 Routing Times in Seconds of the TSP, Jumper Variants, and the Driver-Aide Problem with 15 MPH and 20 MPH Speeds.**

In Tables EC.2 and EC.3, we present the optimal routing time of the TSP variant in "TSP" and that of the jumper variant in "Jumper". Columns "DA_UB" and "DA_LB" contain the best upper and lower bounds of the driver-aide problem, respectively.

.

| 1-Mile X 1-Mile | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| | 5 MPH | | 10 MPH | | 15 MPH | | 20 MPH | |
| | Jumper | DA | Jumper | DA | Jumper | DA | Jumper | DA |
| 1 | 14.3% | 30.9% | 16.5% | 37.9% | 17.4% | 41.0% | 17.8% | 42.5% |
| 2 | 21.9% | 33.1% | 25.2% | 37.9% | 26.6% | 41.2% | 27.3% | 43.1% |
| 3 | 21.7% | 31.5% | 25.1% | 37.9% | 26.4% | 40.7% | 27.2% | 42.6% |
| 4 | 21.5% | 32.4% | 24.9% | 38.3% | 26.3% | 40.0% | 27.1% | 41.7% |
| 5 | 12.8% | 34.1% | 14.6% | 40.7% | 15.3% | 43.2% | 15.7% | 43.9% |
| 6 | 12.4% | 30.0% | 14.4% | 37.3% | 15.3% | 41.1% | 15.7% | 42.4% |
| 7 | 12.7% | 31.6% | 14.8% | 38.1% | 15.6% | 40.5% | 16.0% | 42.8% |
| 8 | 16.7% | 29.9% | 19.5% | 35.5% | 20.7% | 38.5% | 21.3% | 40.8% |
| 9 | 18.2% | 32.4% | 20.9% | 39.8% | 22.0% | 42.9% | 22.6% | 43.7% |
| 10 | 16.9% | 28.0% | 20.1% | 36.3% | 21.4% | 38.7% | 22.2% | 40.6% |
| Avg | 16.9% | 31.4% | 19.6% | 38.0% | 20.7% | 40.8% | 21.3% | 42.4% |
| 3-Miles X 3-Miles | | | | | | | | |
| | 5 MPH | | 10 MPH | | 15 MPH | | 20 MPH | |
| | Jumper | DA | Jumper | DA | Jumper | DA | Jumper | DA |
| 1 | 13.7% | 21.6% | 17.9% | 31.4% | 19.9% | 35.2% | 21.1% | 37.4% |
| 2 | 12.3% | 18.2% | 16.6% | 28.3% | 18.8% | 31.6% | 20.1% | 34.8% |
| 3 | 13.0% | 15.8% | 17.8% | 23.2% | 20.4% | 28.5% | 21.9% | 32.1% |
| 4 | 15.3% | 19.1% | 20.8% | 28.0% | 23.6% | 31.9% | 25.2% | 35.1% |
| 5 | 15.7% | 22.8% | 20.2% | 30.6% | 22.4% | 34.8% | 23.6% | 36.5% |
| 6 | 13.6% | 19.2% | 18.2% | 28.4% | 20.5% | 32.8% | 21.9% | 35.4% |
| 7 | 9.8% | 16.8% | 13.4% | 24.8% | 15.3% | 30.7% | 16.4% | 32.5% |
| 8 | 12.0% | 17.8% | 15.9% | 25.7% | 17.8% | 29.9% | 18.9% | 33.3% |
| 9 | 12.5% | 13.1% | 17.5% | 21.6% | 20.2% | 28.3% | 21.9% | 31.7% |
| 10 | 12.6% | 14.6% | 17.6% | 22.4% | 20.3% | 29.1% | 21.9% | 32.5% |
| Avg | 13.0% | 17.9% | 17.6% | 26.4% | 19.9% | 31.3% | 21.3% | 34.1% |
| 5-Miles X 5-Miles | | | | | | | | |
| | 5 MPH | | 10 MPH | | 15 MPH | | 20 MPH | |
| | Jumper | DA | Jumper | DA | Jumper | DA | Jumper | DA |
| 1 | 8.3% | 10.6% | 12.1% | 18.7% | 14.2% | 25.6% | 15.6% | 29.7% |
| 2 | 11.0% | 14.3% | 15.7% | 24.3% | 18.3% | 29.4% | 19.9% | 33.1% |
| 3 | 7.4% | 10.5% | 11.0% | 17.6% | 13.2% | 23.2% | 14.6% | 26.5% |
| 4 | 6.5% | 11.9% | 9.5% | 22.4% | 11.2% | 27.2% | 12.2% | 30.5% |
| 5 | 8.2% | 9.4% | 12.2% | 15.7% | 14.5% | 21.1% | 16.1% | 24.9% |
| 6 | 10.0% | 13.7% | 14.3% | 21.3% | 16.7% | 27.7% | 18.2% | 32.3% |
| 7 | 8.7% | 9.4% | 13.2% | 16.9% | 15.9% | 22.1% | 17.7% | 26.8% |
| 8 | 6.3% | 8.5% | 9.6% | 14.2% | 11.6% | 18.8% | 13.0% | 22.1% |
| 9 | 11.9% | 14.0% | 17.5% | 21.0% | 20.7% | 27.4% | 22.7% | 31.8% |
| 10 | 8.2% | 10.3% | 12.6% | 18.3% | 15.2% | 24.0% | 17.0% | 28.6% |
| Avg | 8.7% | 11.3% | 12.7% | 19.1% | 15.1% | 24.6% | 16.7% | 28.6% |

**Table EC.4 Relative Reductions in the Routing Times of the Driver-Aide Problem with the Jumper Mode Only and the Driver-Aide Problem.**

Table EC.4 presents the detailed results with a similar format as Table EC.1. The relative reduction in the routing time of the jumper variant is shown in the column "Jumper". It is calculated as $1 - z_{Jumper}/z_{TSP}$, where $z_{TSP}$ is the routing time of the TSP variant, and $z_{Jumper}$ denotes the routing time of the jumper variant. Similarly, the relative reduction in the routing time of the driver-aide problem is shown in the column "DA".

| 1-Mile X 1-Mile | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| | 5 MPH | | 10 MPH | | 15 MPH | | 20 MPH | |
| | Jumper | DA | Jumper | DA | Jumper | DA | Jumper | DA |
| 1 | 104.0% | 83.9% | 101.4% | 75.4% | 100.4% | 71.7% | 99.8% | 69.9% |
| 2 | 94.8% | 81.3% | 90.8% | 75.4% | 89.1% | 71.4% | 88.3% | 69.1% |
| 3 | 95.1% | 83.2% | 91.0% | 75.4% | 89.3% | 72.1% | 88.4% | 69.7% |
| 4 | 95.3% | 82.1% | 91.2% | 74.9% | 89.5% | 72.8% | 88.5% | 70.8% |
| 5 | 105.9% | 80.0% | 103.7% | 72.0% | 102.8% | 68.9% | 102.3% | 68.1% |
| 6 | 106.3% | 85.0% | 103.9% | 76.1% | 102.9% | 71.5% | 102.4% | 70.0% |
| 7 | 106.0% | 83.0% | 103.5% | 75.1% | 102.5% | 72.3% | 101.9% | 69.5% |
| 8 | 101.1% | 85.1% | 97.7% | 78.3% | 96.3% | 74.7% | 95.5% | 71.9% |
| 9 | 99.3% | 82.0% | 96.0% | 73.2% | 94.7% | 69.4% | 93.9% | 68.4% |
| 10 | 100.9% | 87.5% | 97.1% | 77.4% | 95.4% | 74.5% | 94.5% | 72.1% |
| 3-Miles X 3-Miles | | | | | | | | |
| | 5 MPH | | 10 MPH | | 15 MPH | | 20 MPH | |
| | Jumper | DA | Jumper | DA | Jumper | DA | Jumper | DA |
| 1 | 104.8% | 95.2% | 99.7% | 83.3% | 97.2% | 78.7% | 95.8% | 76.0% |
| 2 | 106.5% | 99.4% | 101.3% | 87.0% | 98.7% | 83.1% | 97.1% | 79.1% |
| 3 | 105.7% | 102.3% | 99.8% | 93.3% | 96.7% | 86.8% | 94.8% | 82.4% |
| 4 | 102.8% | 98.3% | 96.2% | 87.4% | 92.8% | 82.8% | 90.8% | 78.8% |
| 5 | 102.4% | 93.7% | 96.9% | 84.2% | 94.3% | 79.2% | 92.7% | 77.1% |
| 6 | 104.9% | 98.1% | 99.3% | 87.0% | 96.5% | 81.6% | 94.8% | 78.5% |
| 7 | 109.6% | 101.1% | 105.2% | 91.4% | 102.9% | 84.2% | 101.5% | 82.0% |
| 8 | 106.9% | 99.8% | 102.2% | 90.2% | 99.8% | 85.1% | 98.4% | 80.9% |
| 9 | 106.2% | 105.5% | 100.1% | 95.3% | 96.9% | 87.0% | 94.8% | 82.9% |
| 10 | 106.2% | 103.7% | 100.1% | 94.2% | 96.8% | 86.1% | 94.8% | 82.0% |
| 5-Miles X 5-Miles | | | | | | | | |
| | 5 MPH | | 10 MPH | | 15 MPH | | 20 MPH | |
| | Jumper | DA | Jumper | DA | Jumper | DA | Jumper | DA |
| 1 | 111.3% | 108.6% | 106.8% | 98.7% | 104.2% | 90.3% | 102.5% | 85.3% |
| 2 | 108.0% | 104.1% | 102.4% | 91.9% | 99.2% | 85.8% | 97.2% | 81.2% |
| 3 | 112.5% | 108.6% | 108.1% | 100.0% | 105.4% | 93.3% | 103.7% | 89.3% |
| 4 | 113.5% | 106.9% | 109.9% | 94.2% | 107.9% | 88.4% | 106.6% | 84.4% |
| 5 | 111.5% | 110.0% | 106.7% | 102.4% | 103.8% | 95.8% | 101.9% | 91.2% |
| 6 | 109.3% | 104.8% | 104.1% | 95.6% | 101.2% | 87.8% | 99.3% | 82.2% |
| 7 | 110.9% | 110.0% | 105.5% | 100.9% | 102.1% | 94.6% | 99.9% | 88.9% |
| 8 | 113.8% | 111.1% | 109.8% | 104.2% | 107.3% | 98.6% | 105.6% | 94.6% |
| 9 | 106.9% | 104.5% | 100.2% | 96.0% | 96.3% | 88.1% | 93.8% | 82.8% |
| 10 | 111.4% | 109.0% | 106.2% | 99.2% | 103.0% | 92.3% | 100.8% | 86.7% |

**Table EC.5 Relative Costs of the Jumper Variant and the Driver-Aide Problem Compared to the TSP Variant.**

We calculate the relative costs of the jumper variant and the driver-aide problem as $\$90.65 * z_{Jumper}/(\$74.65 * z_{TSP})$ and $\$90.65 * z_{DA}/(\$74.65 * z_{TSP})$, respectively. Table EC.5 shows the relative costs of the jumper variant and the driver-aide problem for each instance. A number higher (lower) than 100% indicates a higher (lower) cost than the TSP variant.

| | | ID | $\bar{t}$ | $\bar{s}$ | $\bar{f}$ | $\theta_J$ | H | D | J | $\bar{t}_{DA}$ | $\bar{s}_{DA}$ | $\bar{f}_{DA}$ | $\gamma$ | $\Delta$ | $\rho_H$ | Mode |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 5 MPH | 1-Mile X 1-Mile | 1 | 101.33 | 287.80 | 0.19 | 0.24 | 10 | 22 | 9 | 154.4 | 129.0 | 0.32 | 2.2 | 734.0 | 941.8 | DA |
| | | 2 | 102.47 | 288.44 | 0.30 | 0.24 | 9 | 14 | 18 | 108.4 | 169.1 | 0.37 | 1.6 | 668.1 | 929.1 | DA |
| | | 3 | 92.18 | 251.46 | 0.30 | 0.24 | 10 | 16 | 15 | 120.9 | 195.1 | 0.36 | 1.6 | 490.2 | 670.8 | DA |
| | | 4 | 98.85 | 259.85 | 0.30 | 0.24 | 11 | 13 | 17 | 111.0 | 269.4 | 0.33 | 1.2 | 379.4 | 539.3 | DA |
| | | 5 | 98.29 | 293.93 | 0.17 | 0.24 | 11 | 20 | 10 | 107.3 | 125.0 | 0.27 | 1.8 | 675.3 | 916.1 | DA |
| | | 6 | 93.51 | 242.37 | 0.17 | 0.24 | 10 | 18 | 13 | 134.0 | 149.5 | 0.38 | 1.8 | 542.9 | 706.2 | DA |
| | | 7 | 94.88 | 248.66 | 0.18 | 0.24 | 13 | 18 | 10 | 126.1 | 186.8 | 0.24 | 1.4 | 415.3 | 559.9 | DA |
| | | 8 | 94.87 | 233.59 | 0.23 | 0.25 | 9 | 13 | 19 | 103.0 | 172.5 | 0.27 | 1.4 | 477.4 | 645.3 | DA |
| | | 9 | 98.86 | 283.10 | 0.25 | 0.24 | 11 | 19 | 11 | 120.5 | 266.1 | 0.35 | 1.7 | 571.4 | 757.3 | DA |
| | | 10 | 104.16 | 223.24 | 0.25 | 0.26 | 10 | 13 | 18 | 129.3 | 178.9 | 0.34 | 1.3 | 401.6 | 532.2 | DA |
| | 3-Miles X 3-Miles | 1 | 279.8 | 318.6 | 0.26 | 0.33 | 8 | 8 | 25 | 302.6 | 278.5 | 0.33 | 1.0 | 633.5 | 692.1 | DA |
| | | 2 | 299.8 | 283.0 | 0.25 | 0.36 | 8 | 11 | 22 | 352.8 | 219.9 | 0.28 | 1.4 | 722.8 | 702.2 | DA |
| | | 3 | 271.8 | 226.4 | 0.29 | 0.39 | 6 | 6 | 29 | 292.0 | 190.7 | 0.30 | 1.0 | 539.8 | 455.1 | TSP |
| | | 4 | 286.5 | 260.7 | 0.32 | 0.37 | 8 | 9 | 24 | 314.5 | 181.1 | 0.37 | 1.1 | 678.2 | 707.5 | DA |
| | | 5 | 285.4 | 350.8 | 0.28 | 0.32 | 7 | 18 | 16 | 348.2 | 169.1 | 0.28 | 2.6 | 1468.2 | 1646.6 | DA |
| | | 6 | 293.5 | 285.9 | 0.28 | 0.36 | 6 | 10 | 25 | 338.0 | 193.5 | 0.30 | 1.7 | 901.0 | 946.3 | DA |
| | | 7 | 273.6 | 233.9 | 0.21 | 0.38 | 10 | 11 | 20 | 326.0 | 183.2 | 0.24 | 1.1 | 480.7 | 455.2 | TSP |
| | | 8 | 286.5 | 296.6 | 0.24 | 0.35 | 7 | 10 | 24 | 330.6 | 246.1 | 0.26 | 1.4 | 713.9 | 722.3 | DA |
| | | 9 | 280.2 | 210.7 | 0.29 | 0.41 | 4 | 4 | 33 | 301.8 | 211.4 | 0.28 | 1.0 | 364.9 | 104.7 | TSP |
| | | 10 | 293.8 | 223.3 | 0.29 | 0.41 | 5 | 6 | 30 | 325.3 | 194.2 | 0.24 | 1.2 | 531.6 | 328.9 | TSP |
| | 5-Miles X 5-Miles | 1 | 505.0 | 307.4 | 0.22 | 0.47 | 4 | 5 | 32 | 571.9 | 256.1 | 0.27 | 1.3 | 768.1 | 218.0 | TSP |
| | | 2 | 481.2 | 328.2 | 0.27 | 0.44 | 5 | 6 | 30 | 505.2 | 259.0 | 0.28 | 1.2 | 1115.1 | 815.7 | TSP |
| | | 3 | 468.4 | 238.6 | 0.22 | 0.52 | 4 | 4 | 33 | 503.9 | 175.3 | 0.28 | 1.0 | 781.2 | 272.7 | TSP |
| | | 4 | 465.2 | 285.9 | 0.17 | 0.46 | 4 | 4 | 33 | 495.1 | 250.5 | 0.26 | 1.0 | 614.8 | 232.0 | TSP |
| | | 5 | 494.5 | 258.1 | 0.24 | 0.51 | 4 | 4 | 33 | 539.7 | 223.4 | 0.27 | 1.0 | 676.3 | 20.3 | TSP |
| | | 6 | 504.0 | 331.3 | 0.25 | 0.45 | 4 | 4 | 33 | 526.5 | 288.6 | 0.28 | 1.0 | 838.8 | 456.2 | TSP |
| | | 7 | 451.8 | 210.1 | 0.27 | 0.56 | 2 | 2 | 37 | 462.2 | 197.6 | 0.26 | 1.0 | 550.2 | -547.7 | TSP |
| | | 8 | 483.2 | 212.8 | 0.20 | 0.58 | 3 | 3 | 35 | 506.7 | 191.1 | 0.25 | 1.0 | 591.6 | -267.4 | TSP |
| | | 9 | 459.5 | 266.4 | 0.33 | 0.48 | 6 | 6 | 29 | 458.8 | 250.9 | 0.29 | 1.0 | 827.5 | 543.3 | TSP |
| | | 10 | 495.1 | 227.3 | 0.26 | 0.56 | 4 | 4 | 33 | 532.6 | 216.1 | 0.30 | 1.0 | 529.6 | -71.2 | TSP |
| 10 MPH | 1-Mile X 1-Mile | 1 | 50.6 | 287.8 | 0.19 | 0.21 | 9 | 22 | 10 | 77.0 | 232.2 | 0.33 | 2.4 | 634.7 | 934.4 | DA |
| | | 2 | 51.1 | 288.4 | 0.30 | 0.21 | 7 | 21 | 13 | 65.9 | 241.2 | 0.42 | 3.0 | 810.5 | 1188.3 | DA |
| | | 3 | 46.1 | 251.5 | 0.30 | 0.21 | 8 | 20 | 13 | 65.2 | 230.8 | 0.38 | 2.5 | 564.5 | 853.7 | DA |
| | | 4 | 49.3 | 259.9 | 0.30 | 0.21 | 10 | 23 | 8 | 72.9 | 212.9 | 0.41 | 2.3 | 557.9 | 813.0 | DA |
| | | 5 | 49.1 | 293.9 | 0.17 | 0.21 | 10 | 25 | 6 | 69.1 | 118.0 | 0.42 | 2.5 | 683.4 | 1006.5 | DA |
| | | 6 | 46.7 | 242.4 | 0.17 | 0.21 | 9 | 23 | 9 | 77.8 | 173.0 | 0.42 | 2.6 | 576.4 | 829.1 | DA |
| | | 7 | 47.4 | 248.7 | 0.18 | 0.21 | 12 | 23 | 6 | 73.6 | 156.8 | 0.38 | 1.9 | 470.0 | 674.4 | DA |
| | | 8 | 47.4 | 233.6 | 0.23 | 0.21 | 12 | 21 | 8 | 67.2 | 180.6 | 0.33 | 1.8 | 431.9 | 597.4 | DA |
| | | 9 | 49.4 | 283.1 | 0.25 | 0.21 | 11 | 20 | 10 | 61.9 | 358.4 | 0.46 | 1.8 | 464.3 | 726.5 | DA |
| | | 10 | 52.0 | 223.2 | 0.25 | 0.22 | 13 | 19 | 9 | 67.9 | 200.2 | 0.42 | 1.5 | 363.2 | 520.1 | DA |
| | 3-Miles X 3-Miles | 1 | 139.9 | 318.6 | 0.26 | 0.25 | 10 | 15 | 16 | 169.0 | 338.4 | 0.41 | 1.5 | 550.4 | 787.0 | DA |
| | | 2 | 149.9 | 283.0 | 0.25 | 0.27 | 10 | 16 | 15 | 185.8 | 227.1 | 0.31 | 1.6 | 597.5 | 763.1 | DA |
| | | 3 | 136.0 | 226.4 | 0.29 | 0.28 | 7 | 9 | 25 | 159.2 | 180.5 | 0.30 | 1.3 | 466.3 | 568.5 | DA |
| | | 4 | 143.2 | 260.7 | 0.32 | 0.27 | 8 | 14 | 19 | 182.1 | 186.1 | 0.36 | 1.8 | 609.9 | 809.1 | DA |
| | | 5 | 142.7 | 350.8 | 0.28 | 0.25 | 5 | 20 | 16 | 180.9 | 169.9 | 0.31 | 4.0 | 1685.6 | 2195.3 | DA |
| | | 6 | 146.7 | 285.9 | 0.28 | 0.27 | 10 | 15 | 16 | 166.3 | 170.1 | 0.37 | 1.5 | 673.2 | 856.4 | DA |
| | | 7 | 136.8 | 233.9 | 0.21 | 0.28 | 10 | 17 | 14 | 177.0 | 204.6 | 0.25 | 1.7 | 524.5 | 612.0 | DA |
| | | 8 | 143.2 | 296.6 | 0.24 | 0.26 | 11 | 16 | 14 | 195.6 | 180.6 | 0.29 | 1.5 | 610.5 | 743.0 | DA |
| | | 9 | 140.1 | 210.7 | 0.29 | 0.29 | 8 | 9 | 24 | 153.7 | 154.8 | 0.32 | 1.1 | 483.6 | 550.5 | DA |
| | | 10 | 146.9 | 223.3 | 0.29 | 0.29 | 7 | 8 | 26 | 157.8 | 192.4 | 0.24 | 1.1 | 462.9 | 500.3 | DA |
| | 5-Miles X 5-Miles | 1 | 252.4 | 307.4 | 0.22 | 0.32 | 8 | 12 | 21 | 324.4 | 243.8 | 0.35 | 1.5 | 743.1 | 769.4 | DA |
| | | 2 | 240.7 | 328.2 | 0.27 | 0.31 | 9 | 15 | 17 | 291.6 | 213.1 | 0.32 | 1.7 | 862.1 | 1019.3 | DA |
| | | 3 | 234.2 | 238.6 | 0.22 | 0.35 | 6 | 6 | 29 | 258.1 | 181.2 | 0.29 | 1.0 | 527.1 | 532.7 | TSP |
| | | 4 | 232.7 | 285.9 | 0.17 | 0.32 | 13 | 14 | 14 | 312.1 | 179.6 | 0.33 | 1.1 | 489.8 | 568.7 | DA |
| | | 5 | 247.2 | 258.1 | 0.24 | 0.35 | 4 | 4 | 33 | 260.8 | 211.3 | 0.28 | 1.0 | 735.0 | 606.9 | TSP |
| | | 6 | 252.1 | 331.3 | 0.25 | 0.31 | 6 | 15 | 20 | 299.8 | 163.8 | 0.31 | 2.5 | 1441.2 | 1606.6 | DA |
| | | 7 | 225.9 | 210.1 | 0.27 | 0.37 | 9 | 9 | 23 | 270.2 | 169.3 | 0.30 | 1.0 | 376.6 | 370.3 | TSP |
| | | 8 | 241.5 | 212.8 | 0.20 | 0.38 | 5 | 6 | 30 | 267.2 | 192.4 | 0.25 | 1.2 | 435.7 | 327.7 | TSP |
| | | 9 | 229.7 | 266.4 | 0.33 | 0.33 | 7 | 11 | 23 | 224.0 | 259.1 | 0.31 | 1.6 | 801.6 | 848.1 | DA |
| | | 10 | 247.6 | 227.3 | 0.26 | 0.37 | 7 | 8 | 26 | 282.7 | 212.8 | 0.30 | 1.1 | 426.3 | 404.6 | DA |

**Table EC.6 Detailed Calculations of the Comparative Metrics.**

| | | ID | $\bar{t}$ | $\bar{s}$ | $\bar{f}$ | $\theta_J$ | H | D | J | $\bar{t}_{DA}$ | $\bar{s}_{DA}$ | $\bar{f}_{DA}$ | $\gamma$ | $\Delta$ | $\rho_H$ | Mode |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 15 MPH | 1-Mile X 1-Mile | 1 | 33.8 | 287.8 | 0.19 | 0.20 | 9 | 28 | 4 | 70.2 | 96.8 | 0.33 | 3.1 | 737.7 | 1076.0 | DA |
| | | 2 | 34.2 | 288.4 | 0.30 | 0.20 | 10 | 22 | 9 | 45.3 | 216.4 | 0.42 | 2.2 | 591.8 | 890.5 | DA |
| | | 3 | 30.7 | 251.5 | 0.30 | 0.20 | 10 | 24 | 7 | 47.1 | 275.9 | 0.40 | 2.4 | 507.5 | 756.9 | DA |
| | | 4 | 32.9 | 259.9 | 0.30 | 0.20 | 11 | 24 | 6 | 49.2 | 176.3 | 0.38 | 2.2 | 526.4 | 763.4 | DA |
| | | 5 | 32.8 | 293.9 | 0.17 | 0.20 | 8 | 31 | 2 | 58.1 | 89.5 | 0.35 | 3.9 | 864.2 | 1291.6 | DA |
| | | 6 | 31.2 | 242.4 | 0.17 | 0.20 | 11 | 26 | 4 | 59.9 | 192.5 | 0.43 | 2.4 | 483.4 | 717.6 | DA |
| | | 7 | 31.6 | 248.7 | 0.18 | 0.20 | 12 | 24 | 5 | 47.0 | 162.4 | 0.46 | 2.0 | 468.5 | 685.5 | DA |
| | | 8 | 31.6 | 233.6 | 0.23 | 0.20 | 12 | 23 | 6 | 48.5 | 176.3 | 0.33 | 1.9 | 430.8 | 614.7 | DA |
| | | 9 | 33.0 | 283.1 | 0.25 | 0.20 | 11 | 20 | 10 | 41.9 | 359.7 | 0.46 | 1.8 | 428.3 | 713.5 | DA |
| | 3-Miles X 3-Miles | 10 | 34.7 | 223.2 | 0.25 | 0.20 | 11 | 18 | 12 | 41.4 | 193.2 | 0.38 | 1.6 | 380.2 | 573.4 | DA |
| | | 1 | 93.3 | 318.6 | 0.26 | 0.23 | 9 | 15 | 17 | 116.8 | 318.5 | 0.41 | 1.7 | 549.1 | 850.2 | DA |
| | | 2 | 99.9 | 283.0 | 0.25 | 0.24 | 10 | 19 | 12 | 129.8 | 276.2 | 0.36 | 1.9 | 600.5 | 794.6 | DA |
| | | 3 | 90.6 | 226.4 | 0.29 | 0.25 | 11 | 15 | 15 | 119.1 | 170.9 | 0.36 | 1.4 | 419.6 | 542.2 | DA |
| | | 4 | 95.5 | 260.7 | 0.32 | 0.24 | 9 | 16 | 16 | 107.9 | 230.5 | 0.43 | 1.8 | 582.4 | 803.7 | DA |
| | | 5 | 95.1 | 350.8 | 0.28 | 0.22 | 5 | 25 | 11 | 148.7 | 186.2 | 0.31 | 5.0 | 1642.3 | 2252.3 | DA |
| | | 6 | 97.8 | 285.9 | 0.28 | 0.24 | 11 | 17 | 13 | 106.9 | 223.2 | 0.35 | 1.5 | 569.6 | 772.0 | DA |
| | | 7 | 91.2 | 233.9 | 0.21 | 0.25 | 10 | 19 | 12 | 126.8 | 205.3 | 0.29 | 1.9 | 489.5 | 644.1 | DA |
| | | 8 | 95.5 | 296.6 | 0.24 | 0.23 | 12 | 18 | 11 | 144.3 | 146.6 | 0.35 | 1.5 | 578.6 | 740.0 | DA |
| | | 9 | 93.4 | 210.7 | 0.29 | 0.25 | 13 | 17 | 11 | 118.3 | 144.2 | 0.39 | 1.3 | 396.6 | 497.1 | DA |
| | 5-Miles X 5-Miles | 10 | 97.9 | 223.3 | 0.29 | 0.25 | 11 | 14 | 16 | 113.4 | 226.6 | 0.32 | 1.3 | 375.3 | 483.1 | DA |
| | | 1 | 168.3 | 307.4 | 0.22 | 0.27 | 11 | 13 | 17 | 206.1 | 238.9 | 0.39 | 1.2 | 573.5 | 711.7 | DA |
| | | 2 | 160.4 | 328.2 | 0.27 | 0.26 | 8 | 13 | 20 | 183.4 | 261.7 | 0.35 | 1.6 | 715.1 | 992.0 | DA |
| | | 3 | 156.1 | 238.6 | 0.22 | 0.29 | 11 | 12 | 18 | 194.4 | 144.4 | 0.33 | 1.1 | 463.9 | 541.5 | DA |
| | | 4 | 155.1 | 285.9 | 0.17 | 0.27 | 12 | 14 | 15 | 198.3 | 168.4 | 0.33 | 1.2 | 512.0 | 654.1 | DA |
| | | 5 | 164.8 | 258.1 | 0.24 | 0.29 | 8 | 13 | 20 | 198.9 | 180.5 | 0.29 | 1.6 | 707.8 | 766.2 | DA |
| | | 6 | 168.0 | 331.3 | 0.25 | 0.27 | 8 | 23 | 10 | 229.6 | 119.9 | 0.28 | 2.9 | 1223.5 | 1482.7 | DA |
| | | 7 | 150.6 | 210.1 | 0.27 | 0.30 | 11 | 13 | 17 | 190.7 | 149.5 | 0.29 | 1.2 | 394.4 | 457.4 | DA |
| | | 8 | 161.1 | 212.8 | 0.20 | 0.31 | 8 | 10 | 23 | 198.4 | 205.6 | 0.28 | 1.3 | 356.7 | 385.0 | DA |
| | | 9 | 153.2 | 266.4 | 0.33 | 0.28 | 8 | 14 | 19 | 164.3 | 279.1 | 0.32 | 1.8 | 596.4 | 762.8 | DA |
| 20 MPH | 1-Mile X 1-Mile | 10 | 165.0 | 227.3 | 0.26 | 0.30 | 10 | 13 | 18 | 206.3 | 224.3 | 0.35 | 1.3 | 404.3 | 484.8 | DA |
| | | 1 | 25.3 | 287.8 | 0.19 | 0.19 | 7 | 29 | 5 | 50.6 | 103.2 | 0.32 | 4.1 | 922.8 | 1373.5 | DA |
| | | 2 | 25.6 | 288.4 | 0.30 | 0.19 | 10 | 23 | 8 | 31.5 | 275.8 | 0.43 | 2.3 | 565.0 | 876.9 | DA |
| | | 3 | 23.0 | 251.5 | 0.30 | 0.19 | 10 | 24 | 7 | 34.8 | 231.9 | 0.39 | 2.4 | 502.3 | 768.0 | DA |
| | | 4 | 24.7 | 259.9 | 0.30 | 0.19 | 13 | 21 | 7 | 32.9 | 446.0 | 0.43 | 1.6 | 350.2 | 551.3 | DA |
| | | 5 | 24.6 | 293.9 | 0.17 | 0.19 | 9 | 29 | 3 | 40.4 | 107.0 | 0.40 | 3.2 | 739.9 | 1119.6 | DA |
| | | 6 | 23.4 | 242.4 | 0.17 | 0.19 | 10 | 27 | 4 | 48.9 | 141.5 | 0.37 | 2.7 | 528.6 | 793.4 | DA |
| | | 7 | 23.7 | 248.7 | 0.18 | 0.19 | 12 | 24 | 5 | 32.3 | 162.4 | 0.46 | 2.0 | 452.0 | 684.1 | DA |
| | | 8 | 23.7 | 233.6 | 0.23 | 0.19 | 11 | 26 | 4 | 41.5 | 164.5 | 0.35 | 2.4 | 476.7 | 694.3 | DA |
| | | 9 | 24.7 | 283.1 | 0.25 | 0.19 | 12 | 20 | 9 | 31.6 | 368.1 | 0.47 | 1.7 | 399.2 | 663.5 | DA |
| | 3-Miles X 3-Miles | 10 | 26.0 | 223.2 | 0.25 | 0.20 | 12 | 22 | 7 | 37.2 | 213.4 | 0.43 | 1.8 | 381.7 | 571.4 | DA |
| | | 1 | 69.9 | 318.6 | 0.26 | 0.22 | 9 | 17 | 15 | 87.8 | 267.3 | 0.41 | 1.9 | 635.3 | 962.2 | DA |
| | | 2 | 74.9 | 283.0 | 0.25 | 0.22 | 11 | 20 | 10 | 95.8 | 319.8 | 0.39 | 1.8 | 526.6 | 738.4 | DA |
| | | 3 | 68.0 | 226.4 | 0.29 | 0.23 | 13 | 15 | 13 | 90.1 | 212.2 | 0.36 | 1.2 | 320.5 | 448.8 | DA |
| | | 4 | 71.6 | 260.7 | 0.32 | 0.22 | 10 | 18 | 13 | 88.1 | 220.3 | 0.43 | 1.8 | 527.1 | 756.5 | DA |
| | | 5 | 71.3 | 350.8 | 0.28 | 0.21 | 6 | 26 | 9 | 115.3 | 306.2 | 0.38 | 4.3 | 1261.2 | 1801.3 | DA |
| | | 6 | 73.4 | 285.9 | 0.28 | 0.22 | 11 | 18 | 12 | 84.8 | 262.2 | 0.39 | 1.6 | 522.2 | 751.4 | DA |
| | | 7 | 68.4 | 233.9 | 0.21 | 0.23 | 11 | 20 | 10 | 100.6 | 220.6 | 0.31 | 1.8 | 445.8 | 597.5 | DA |
| | | 8 | 71.6 | 296.6 | 0.24 | 0.22 | 11 | 21 | 9 | 112.2 | 192.1 | 0.33 | 1.9 | 608.3 | 821.4 | DA |
| | | 9 | 70.1 | 210.7 | 0.29 | 0.24 | 13 | 17 | 11 | 91.3 | 132.5 | 0.38 | 1.3 | 366.6 | 491.2 | DA |
| | 5-Miles X 5-Miles | 10 | 73.5 | 223.3 | 0.29 | 0.23 | 11 | 16 | 14 | 89.7 | 232.2 | 0.34 | 1.5 | 371.4 | 510.5 | DA |
| | | 1 | 126.2 | 307.4 | 0.22 | 0.25 | 13 | 18 | 10 | 162.0 | 264.3 | 0.34 | 1.4 | 545.9 | 705.4 | DA |
| | | 2 | 120.3 | 328.2 | 0.27 | 0.24 | 10 | 19 | 12 | 152.7 | 197.4 | 0.36 | 1.9 | 748.2 | 1026.4 | DA |
| | | 3 | 117.1 | 238.6 | 0.22 | 0.26 | 12 | 14 | 15 | 155.4 | 134.8 | 0.31 | 1.2 | 431.3 | 535.3 | DA |
| | | 4 | 116.3 | 285.9 | 0.17 | 0.25 | 12 | 14 | 15 | 148.3 | 146.2 | 0.33 | 1.2 | 504.7 | 674.9 | DA |
| | | 5 | 123.6 | 258.1 | 0.24 | 0.26 | 9 | 11 | 21 | 138.3 | 179.2 | 0.34 | 1.2 | 582.8 | 694.3 | DA |
| | | 6 | 126.0 | 331.3 | 0.25 | 0.24 | 7 | 26 | 8 | 194.3 | 112.9 | 0.32 | 3.7 | 1309.4 | 1702.3 | DA |
| | | 7 | 112.9 | 210.1 | 0.27 | 0.27 | 11 | 14 | 16 | 146.8 | 150.8 | 0.31 | 1.3 | 364.6 | 479.3 | DA |
| | | 8 | 120.8 | 212.8 | 0.20 | 0.28 | 10 | 15 | 16 | 168.6 | 161.8 | 0.29 | 1.5 | 433.3 | 503.3 | DA |
| | | 9 | 114.9 | 266.4 | 0.33 | 0.25 | 9 | 18 | 14 | 135.5 | 293.0 | 0.34 | 2.0 | 576.4 | 784.5 | DA |
| | | 10 | 123.8 | 227.3 | 0.26 | 0.27 | 11 | 14 | 16 | 152.0 | 230.5 | 0.33 | 1.3 | 356.7 | 478.4 | DA |

**Table EC.7 Detailed Calculations of the Comparative Metrics.**

.

| 1-Mile X 1-Mile | | | | |
|---|---|---|---|---|
| | 5 MPH | 10 MPH | 15 MPH | 20 MPH |
| 1 | 78.6% | 69.7% | 65.8% | 63.9% |
| 2 | 75.8% | 69.6% | 65.5% | 63.2% |
| 3 | 77.8% | 69.6% | 66.2% | 63.8% |
| 4 | 76.7% | 69.2% | 67.0% | 64.9% |
| 5 | 74.5% | 66.1% | 63.0% | 62.1% |
| 6 | 79.8% | 70.4% | 65.6% | 64.1% |
| 7 | 77.7% | 69.4% | 66.4% | 63.6% |
| 8 | 79.9% | 72.7% | 68.9% | 66.0% |
| 9 | 76.6% | 67.3% | 63.4% | 62.5% |
| 10 | 82.4% | 71.7% | 68.7% | 66.3% |
| Avg | 78.0% | 69.6% | 66.1% | 64.0% |
| 3-Miles X 3-Miles | | | | |
| | 5 MPH | 10 MPH | 15 MPH | 20 MPH |
| 1 | 90.9% | 77.9% | 73.1% | 70.2% |
| 2 | 95.6% | 82.0% | 77.7% | 73.6% |
| 3 | 98.9% | 88.8% | 81.7% | 77.0% |
| 4 | 94.4% | 82.4% | 77.4% | 73.2% |
| 5 | 89.3% | 78.9% | 73.6% | 71.4% |
| 6 | 94.2% | 81.9% | 76.1% | 72.9% |
| 7 | 97.5% | 86.7% | 78.9% | 76.5% |
| 8 | 96.1% | 85.4% | 79.9% | 75.5% |
| 9 | 102.5% | 91.0% | 82.0% | 77.6% |
| 10 | 100.5% | 89.8% | 81.0% | 76.6% |
| Avg | 96.0% | 84.5% | 78.1% | 74.4% |
| 5-Miles X 5-Miles | | | | |
| | 5 MPH | 10 MPH | 15 MPH | 20 MPH |
| 1 | 106.1% | 94.8% | 85.6% | 80.1% |
| 2 | 101.0% | 87.3% | 80.6% | 75.8% |
| 3 | 106.2% | 96.3% | 88.8% | 84.4% |
| 4 | 104.2% | 89.8% | 83.4% | 79.2% |
| 5 | 107.8% | 99.0% | 91.5% | 86.6% |
| 6 | 101.8% | 91.3% | 82.8% | 76.8% |
| 7 | 107.7% | 97.3% | 90.3% | 84.0% |
| 8 | 109.0% | 101.0% | 94.8% | 90.2% |
| 9 | 101.4% | 91.8% | 83.2% | 77.4% |
| 10 | 106.6% | 95.3% | 87.7% | 81.6% |
| Avg | 105.2% | 94.4% | 86.9% | 81.6% |

**Table EC.8 Relative Costs of the Driver-Aide Problem Compared to the TSP Variant Considering the Overtime Pay Rate.**

Table EC.8 shows the relative costs of the driver-aide problem, compared to the TSP variant, considering the overtime pay rate. They are calculated as \$90.65/(\$74.65 + \$90.93*($z_{TSP}/z_{DA}-1$)).

| #0002 | Amz_routing | TSP | Jumper | DA_UB | DA_LB | Gap | Time | Savings in Cost |
|---|---|---|---|---|---|---|---|---|
| 1 | 31382 | 31008 | 26052 | 21015 | 20937 | 0.4% | 644.0 | 33.0% |
| 2 | 31382 | 31008 | 26026 | 21068 | 20913 | 0.7% | 586.2 | 32.9% |
| 3 | 31382 | 31008 | 24880 | 20974 | 20894 | 0.4% | 680.7 | 33.2% |
| 4 | 31382 | 31008 | 26964 | 21116 | 20893 | 1.1% | 538.1 | 32.7% |
| 5 | 31382 | 31008 | 27352 | 21119 | 21012 | 0.5% | 718.6 | 32.7% |
| #0003 | Amz_routing | TSP | Jumper | DA_UB | DA_LB | Gap | Time | Savings in Cost |
| 1 | 27076 | 26639 | 21794 | 17064 | 16913 | 0.9% | 471.9 | 37.0% |
| 2 | 27076 | 26639 | 23127 | 16992 | 16841 | 0.9% | 564.0 | 37.2% |
| 3 | 27076 | 26639 | 22729 | 16899 | 16809 | 0.5% | 731.7 | 37.6% |
| 4 | 27076 | 26639 | 21017 | 16994 | 16876 | 0.7% | 573.8 | 37.2% |
| 5 | 27076 | 26639 | 21561 | 16879 | 16856 | 0.1% | 389.3 | 37.7% |
| #0012 | Amz_routing | TSP | Jumper | DA_UB | DA_LB | Gap | Time | Savings in Cost |
| 1 | 24987 | 24341 | 19391 | 17447 | 17278 | 1.0% | 75.5 | 30.2% |
| 2 | 24987 | 24341 | 20810 | 17448 | 17226 | 1.3% | 88.1 | 30.2% |
| 3 | 24987 | 24341 | 21985 | 17434 | 17306 | 0.7% | 88.6 | 30.2% |
| 4 | 24987 | 24341 | 20665 | 17488 | 17254 | 1.3% | 80.9 | 30.0% |
| 5 | 24987 | 24341 | 20623 | 17455 | 17341 | 0.6% | 87.8 | 30.1% |
| #0019 | Amz_routing | TSP | Jumper | DA_UB | DA_LB | Gap | Time | Savings in Cost |
| 1 | 20973 | 20690 | 16724 | 13428 | 13330 | 0.7% | 83.5 | 36.0% |
| 2 | 20973 | 20690 | 16012 | 13631 | 13509 | 0.9% | 77.8 | 35.0% |
| 3 | 20973 | 20690 | 16692 | 13555 | 13445 | 0.8% | 81.8 | 35.4% |
| 4 | 20973 | 20690 | 17881 | 13680 | 13562 | 0.9% | 74.9 | 34.8% |
| 5 | 20973 | 20690 | 16272 | 13754 | 13564 | 1.4% | 73.6 | 34.4% |
| #0020 | Amz_routing | TSP | Jumper | DA_UB | DA_LB | Gap | Time | Savings in Cost |
| 1 | 22866 | 21897 | 18052 | 15126 | 15033 | 0.6% | 1034.8 | 33.8% |
| 2 | 22866 | 21897 | 18706 | 15125 | 15041 | 0.6% | 964.9 | 33.9% |
| 3 | 22866 | 21897 | 18060 | 15329 | 15216 | 0.7% | 960.6 | 33.0% |
| 4 | 22866 | 21897 | 17715 | 15107 | 14981 | 0.8% | 819.6 | 33.9% |
| 5 | 22866 | 21897 | 18329 | 15150 | 15099 | 0.3% | 857.0 | 33.7% |
| #0023 | Amz_routing | TSP | Jumper | DA_UB | DA_LB | Gap | Time | Savings in Cost |
| 1 | 24511 | 23593 | 20056 | 17515 | 17404 | 0.6% | 116.0 | 28.5% |
| 2 | 24511 | 23593 | 20740 | 17359 | 17350 | 0.1% | 104.0 | 29.2% |
| 3 | 24511 | 23593 | 19497 | 17237 | 17199 | 0.2% | 98.0 | 29.7% |
| 4 | 24511 | 23593 | 19946 | 17295 | 17255 | 0.2% | 109.0 | 29.4% |
| 5 | 24511 | 23593 | 20834 | 17353 | 17277 | 0.4% | 117.0 | 29.2% |
| #0038 | Amz_routing | TSP | Jumper | DA_UB | DA_LB | Gap | Time | Savings in Cost |
| 1 | 25595 | 25114 | 20033 | 14796 | 14606 | 1.3% | 71.8 | 42.2% |
| 2 | 25595 | 25114 | 19964 | 14741 | 14570 | 1.2% | 91.2 | 42.4% |
| 3 | 25595 | 25114 | 20713 | 14803 | 14588 | 1.4% | 75.2 | 42.2% |
| 4 | 25595 | 25114 | 19762 | 14785 | 14615 | 1.2% | 80.6 | 42.2% |
| 5 | 25595 | 25114 | 20026 | 14811 | 14614 | 1.3% | 71.2 | 42.1% |
| #0062 | Amz_routing | TSP | Jumper | DA_UB | DA_LB | Gap | Time | Savings in Cost |
| 1 | 29044 | 28130 | 22932 | 19901 | 19653 | 1.2% | 130.5 | 31.5% |
| 2 | 29044 | 28130 | 24116 | 19961 | 19685 | 1.4% | 140.6 | 31.3% |
| 3 | 29044 | 28130 | 22871 | 19975 | 19738 | 1.2% | 119.4 | 31.2% |
| 4 | 29044 | 28130 | 23629 | 19876 | 19686 | 1.0% | 145.6 | 31.6% |
| 5 | 29044 | 28130 | 23789 | 19948 | 19676 | 1.4% | 192.1 | 31.3% |
| #0138 | Amz_routing | TSP | Jumper | DA_UB | DA_LB | Gap | Time | Savings in Cost |
| 1 | 28056 | 26741 | 22085 | 16236 | 16124 | 0.7% | 266.5 | 42.1% |
| 2 | 28056 | 26741 | 21535 | 16302 | 16104 | 1.2% | 229.4 | 41.9% |
| 3 | 28056 | 26741 | 20899 | 16132 | 16080 | 0.3% | 213.8 | 42.5% |
| 4 | 28056 | 26741 | 21226 | 16235 | 16141 | 0.6% | 219.7 | 42.1% |
| 5 | 28056 | 26741 | 22285 | 16239 | 16153 | 0.5% | 284.8 | 42.1% |

**Table EC.9 Detailed Results of the Amazon Instances.**

| #0145 | Amz_routing | TSP | Jumper | DA_UB | DA_LB | Gap | Time | Savings in Cost |
|---|---|---|---|---|---|---|---|---|
| 1 | 27201 | 25679 | 21728 | 17464 | 17349 | 0.7% | 396.1 | 35.8% |
| 2 | 27201 | 25679 | 22280 | 17514 | 17321 | 1.1% | 409.4 | 35.6% |
| 3 | 27201 | 25679 | 20568 | 17451 | 17339 | 0.6% | 308.3 | 35.8% |
| 4 | 27201 | 25679 | 21710 | 17466 | 17343 | 0.7% | 437.3 | 35.8% |
| 5 | 27201 | 25679 | 22219 | 17433 | 17280 | 0.9% | 526.1 | 35.9% |
| #0149 | Amz_routing | TSP | Jumper | DA_UB | DA_LB | Gap | Time | Savings in Cost |
| 1 | 29683 | 27854 | 23106 | 19710 | 19395 | 1.6% | 290.7 | 33.6% |
| 2 | 29683 | 27854 | 23834 | 19931 | 19565 | 1.8% | 247.7 | 32.9% |
| 3 | 29683 | 27854 | 23164 | 19997 | 19615 | 1.9% | 244.0 | 32.6% |
| 4 | 29683 | 27854 | 23065 | 19890 | 19586 | 1.5% | 246.0 | 33.0% |
| 5 | 29683 | 27854 | 23525 | 19700 | 19488 | 1.1% | 245.2 | 33.6% |
| #0180 | Amz_routing | TSP | Jumper | DA_UB | DA_LB | Gap | Time | Savings in Cost |
| 1 | 25074 | 23784 | 19700 | 15219 | 15087 | 0.9% | 333.9 | 39.3% |
| 2 | 25074 | 23784 | 19136 | 15022 | 14946 | 0.5% | 319.5 | 40.1% |
| 3 | 25074 | 23784 | 19411 | 15186 | 15062 | 0.8% | 322.2 | 39.4% |
| 4 | 25074 | 23784 | 19639 | 15194 | 15074 | 0.8% | 397.6 | 39.4% |
| 5 | 25074 | 23784 | 20708 | 15075 | 14978 | 0.6% | 395.0 | 39.9% |
| #0303 | Amz_routing | TSP | Jumper | DA_UB | DA_LB | Gap | Time | Savings in Cost |
| 1 | 29854 | 29115 | 23345 | 17173 | 17066 | 0.6% | 71.1 | 42.5% |
| 2 | 29854 | 29115 | 23336 | 17098 | 17023 | 0.4% | 55.9 | 42.7% |
| 3 | 29854 | 29115 | 22409 | 17160 | 17078 | 0.5% | 56.2 | 42.5% |
| 4 | 29854 | 29115 | 23097 | 17168 | 17060 | 0.6% | 73.5 | 42.5% |
| 5 | 29854 | 29115 | 24288 | 17106 | 17072 | 0.2% | 57.9 | 42.7% |
| #0307 | Amz_routing | TSP | Jumper | DA_UB | DA_LB | Gap | Time | Savings in Cost |
| 1 | 27092 | 25774 | 22083 | 17799 | 17778 | 0.1% | 238.5 | 34.3% |
| 2 | 27092 | 25774 | 21546 | 17774 | 17695 | 0.4% | 360.0 | 34.4% |
| 3 | 27092 | 25774 | 21557 | 17860 | 17750 | 0.6% | 232.6 | 34.1% |
| 4 | 27092 | 25774 | 21349 | 17834 | 17744 | 0.5% | 284.6 | 34.2% |
| 5 | 27092 | 25774 | 21967 | 17727 | 17688 | 0.2% | 392.2 | 34.6% |
| #0346 | Amz_routing | TSP | Jumper | DA_UB | DA_LB | Gap | Time | Savings in Cost |
| 1 | 33785 | 32571 | 28172 | 20696 | 20601 | 0.5% | 558.9 | 38.7% |
| 2 | 33785 | 32571 | 28396 | 20925 | 20777 | 0.7% | 415.7 | 38.1% |
| 3 | 33785 | 32571 | 25938 | 20734 | 20681 | 0.3% | 498.8 | 38.6% |
| 4 | 33785 | 32571 | 26840 | 20878 | 20796 | 0.4% | 320.0 | 38.2% |
| 5 | 33785 | 32571 | 27388 | 20777 | 20665 | 0.5% | 418.3 | 38.5% |
| #0431 | Amz_routing | TSP | Jumper | DA_UB | DA_LB | Gap | Time | Savings in Cost |
| 1 | 20584 | 20150 | 16283 | 13999 | 13787 | 1.5% | 211.6 | 32.0% |
| 2 | 20584 | 20150 | 16687 | 14077 | 13910 | 1.2% | 114.1 | 31.6% |
| 3 | 20584 | 20150 | 17370 | 14174 | 13923 | 1.8% | 240.7 | 31.1% |
| 4 | 20584 | 20150 | 17777 | 14150 | 13899 | 1.8% | 154.7 | 31.3% |
| 5 | 20584 | 20150 | 17290 | 14191 | 13926 | 1.9% | 151.4 | 31.1% |
| #0684 | Amz_routing | TSP | Jumper | DA_UB | DA_LB | Gap | Time | Savings in Cost |
| 1 | 28414 | 27313 | 22065 | 17266 | 17186 | 0.5% | 111.1 | 39.2% |
| 2 | 28414 | 27313 | 22732 | 17269 | 17185 | 0.5% | 122.2 | 39.2% |
| 3 | 28414 | 27313 | 22453 | 17233 | 17130 | 0.6% | 116.6 | 39.4% |
| 4 | 28414 | 27313 | 21866 | 17309 | 17224 | 0.5% | 141.1 | 39.1% |
| 5 | 28414 | 27313 | 22458 | 17287 | 17163 | 0.7% | 93.7 | 39.2% |

**Table EC.10 Detailed Results of the Amazon Instances.**

| Route_ID | | $\bar{t}$ | $\bar{s}$ | $\bar{f}$ | $\theta_J$ | H | D | J | $\bar{t}_{DA}$ | $\bar{s}_{DA}$ | $\bar{f}_{DA}$ | $\gamma$ | $\Delta$ | $\rho_H$ | Mode |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| #0002 | 1 | 173.5 | 296.3 | 0.25 | 0.28 | 18 | 33 | 15 | 236.5 | 217.7 | 0.38 | 1.8 | 632.2 | 872.6 | DA |
| | 2 | 173.5 | 296.3 | 0.25 | 0.28 | 18 | 35 | 13 | 254.9 | 179.6 | 0.42 | 1.9 | 660.3 | 903.8 | DA |
| | 3 | 173.5 | 296.3 | 0.31 | 0.28 | 18 | 40 | 8 | 303.8 | 290.1 | 0.43 | 2.2 | 659.9 | 905.6 | DA |
| | 4 | 173.5 | 296.3 | 0.21 | 0.28 | 19 | 38 | 9 | 284.3 | 182.3 | 0.40 | 2.0 | 646.1 | 873.2 | DA |
| | 5 | 173.5 | 296.3 | 0.19 | 0.28 | 19 | 38 | 9 | 277.8 | 192.6 | 0.31 | 2.0 | 648.8 | 871.7 | DA |
| #0003 | 1 | 222.3 | 427.5 | 0.28 | 0.27 | 11 | 27 | 3 | 409.3 | 90.0 | 0.20 | 2.5 | 1008.6 | 1453.8 | DA |
| | 2 | 222.3 | 427.5 | 0.20 | 0.27 | 11 | 27 | 3 | 409.3 | 90.0 | 0.33 | 2.5 | 1008.6 | 1457.0 | DA |
| | 3 | 222.3 | 427.5 | 0.22 | 0.27 | 11 | 25 | 5 | 365.4 | 222.5 | 0.38 | 2.3 | 951.8 | 1400.1 | DA |
| | 4 | 222.3 | 427.5 | 0.32 | 0.27 | 10 | 24 | 7 | 351.0 | 276.0 | 0.34 | 2.4 | 1005.1 | 1470.2 | DA |
| | 5 | 222.3 | 427.5 | 0.29 | 0.27 | 10 | 24 | 7 | 345.0 | 263.1 | 0.43 | 2.4 | 1001.0 | 1502.0 | DA |
| #0012 | 1 | 207.7 | 260.4 | 0.37 | 0.32 | 14 | 19 | 19 | 244.6 | 259.1 | 0.36 | 1.4 | 469.0 | 631.1 | DA |
| | 2 | 207.7 | 260.4 | 0.26 | 0.32 | 14 | 25 | 13 | 292.0 | 240.2 | 0.35 | 1.8 | 548.5 | 724.6 | DA |
| | 3 | 207.7 | 260.4 | 0.17 | 0.32 | 16 | 27 | 9 | 306.4 | 240.3 | 0.41 | 1.7 | 531.5 | 694.4 | DA |
| | 4 | 207.7 | 260.4 | 0.27 | 0.32 | 15 | 26 | 11 | 298.5 | 212.4 | 0.35 | 1.7 | 559.7 | 718.3 | DA |
| | 5 | 207.7 | 260.4 | 0.27 | 0.32 | 16 | 25 | 11 | 292.0 | 157.5 | 0.35 | 1.6 | 531.8 | 689.1 | DA |
| #0019 | 1 | 148.7 | 301.1 | 0.29 | 0.26 | 12 | 23 | 11 | 167.2 | 379.2 | 0.37 | 1.9 | 611.7 | 881.3 | DA |
| | 2 | 148.7 | 301.1 | 0.34 | 0.26 | 14 | 21 | 11 | 153.9 | 242.5 | 0.40 | 1.5 | 589.8 | 827.8 | DA |
| | 3 | 148.7 | 301.1 | 0.29 | 0.26 | 14 | 23 | 9 | 166.0 | 339.7 | 0.42 | 1.6 | 581.2 | 818.1 | DA |
| | 4 | 148.7 | 301.1 | 0.20 | 0.26 | 16 | 26 | 4 | 186.0 | 181.6 | 0.25 | 1.6 | 595.9 | 798.3 | DA |
| | 5 | 148.7 | 301.1 | 0.32 | 0.26 | 14 | 24 | 8 | 165.4 | 262.9 | 0.36 | 1.7 | 635.6 | 861.4 | DA |
| #0020 | 1 | 108.5 | 175.8 | 0.29 | 0.29 | 18 | 35 | 24 | 129.3 | 107.0 | 0.33 | 1.9 | 450.6 | 603.8 | DA |
| | 2 | 108.5 | 175.8 | 0.24 | 0.29 | 19 | 36 | 22 | 131.3 | 121.4 | 0.33 | 1.9 | 428.3 | 571.8 | DA |
| | 3 | 108.5 | 175.8 | 0.28 | 0.29 | 18 | 36 | 23 | 135.2 | 94.9 | 0.37 | 2.0 | 468.4 | 617.4 | DA |
| | 4 | 108.5 | 175.8 | 0.31 | 0.29 | 18 | 33 | 26 | 120.7 | 125.3 | 0.32 | 1.8 | 433.4 | 582.9 | DA |
| | 5 | 108.5 | 175.8 | 0.26 | 0.29 | 18 | 37 | 22 | 132.0 | 107.9 | 0.32 | 2.1 | 465.1 | 619.1 | DA |
| #0023 | 1 | 158.8 | 198.6 | 0.27 | 0.32 | 23 | 32 | 11 | 217.3 | 107.8 | 0.25 | 1.4 | 407.4 | 485.1 | DA |
| | 2 | 158.8 | 198.6 | 0.22 | 0.32 | 20 | 28 | 18 | 203.7 | 105.5 | 0.29 | 1.4 | 419.3 | 516.9 | DA |
| | 3 | 158.8 | 198.6 | 0.31 | 0.32 | 17 | 24 | 25 | 190.7 | 144.3 | 0.37 | 1.4 | 414.4 | 537.6 | DA |
| | 4 | 158.8 | 198.6 | 0.28 | 0.32 | 20 | 28 | 18 | 202.8 | 115.2 | 0.34 | 1.4 | 413.2 | 518.1 | DA |
| | 5 | 158.8 | 198.6 | 0.21 | 0.32 | 22 | 30 | 14 | 214.4 | 92.4 | 0.29 | 1.4 | 402.0 | 490.7 | DA |
| #0038 | 1 | 174.1 | 564.5 | 0.26 | 0.23 | 7 | 21 | 6 | 245.8 | 213.9 | 0.32 | 3.0 | 1547.7 | 2372.8 | DA |
| | 2 | 174.1 | 564.5 | 0.27 | 0.23 | 7 | 22 | 5 | 251.6 | 135.0 | 0.34 | 3.1 | 1616.6 | 2459.4 | DA |
| | 3 | 174.1 | 564.5 | 0.23 | 0.23 | 8 | 23 | 3 | 272.4 | 62.0 | 0.17 | 2.9 | 1455.7 | 2191.3 | DA |
| | 4 | 174.1 | 564.5 | 0.28 | 0.23 | 8 | 24 | 2 | 303.8 | 62.0 | 0.15 | 3.0 | 1454.4 | 2192.2 | DA |
| | 5 | 174.1 | 564.5 | 0.27 | 0.23 | 8 | 24 | 2 | 306.4 | 62.0 | 0.20 | 3.0 | 1452.9 | 2189.8 | DA |
| #0062 | 1 | 176.4 | 249.8 | 0.32 | 0.30 | 25 | 32 | 9 | 236.9 | 200.4 | 0.33 | 1.3 | 430.8 | 556.3 | DA |
| | 2 | 176.4 | 249.8 | 0.24 | 0.30 | 23 | 29 | 14 | 227.2 | 236.6 | 0.35 | 1.3 | 423.7 | 548.1 | DA |
| | 3 | 176.4 | 249.8 | 0.32 | 0.30 | 21 | 27 | 18 | 220.4 | 179.7 | 0.37 | 1.3 | 453.5 | 596.2 | DA |
| | 4 | 176.4 | 249.8 | 0.27 | 0.30 | 23 | 30 | 13 | 227.3 | 158.6 | 0.35 | 1.3 | 458.4 | 593.5 | DA |
| | 5 | 176.4 | 249.8 | 0.26 | 0.30 | 23 | 30 | 13 | 223.4 | 155.1 | 0.37 | 1.3 | 465.2 | 602.3 | DA |
| #0138 | 1 | 169.5 | 399.5 | 0.20 | 0.25 | 10 | 30 | 7 | 256.0 | 273.5 | 0.40 | 3.0 | 1091.2 | 1652.0 | DA |
| | 2 | 169.5 | 399.5 | 0.23 | 0.25 | 10 | 34 | 3 | 327.0 | 415.3 | 0.33 | 3.4 | 1130.4 | 1693.9 | DA |
| | 3 | 169.5 | 399.5 | 0.26 | 0.25 | 10 | 32 | 5 | 279.2 | 233.2 | 0.40 | 3.2 | 1135.1 | 1713.4 | DA |
| | 4 | 169.5 | 399.5 | 0.24 | 0.25 | 11 | 32 | 4 | 284.5 | 337.6 | 0.38 | 2.9 | 1026.0 | 1537.2 | DA |
| | 5 | 169.5 | 399.5 | 0.20 | 0.25 | 10 | 35 | 2 | 346.7 | 177.1 | 0.50 | 3.5 | 1190.2 | 1768.3 | DA |

**Table EC.11 Detailed Calculations of the Comparative Metrics for the Amazon Instances.**

| Route_ID | | $\bar{t}$ | $\bar{s}$ | $\bar{f}$ | $\theta_J$ | H | D | J | $\bar{t}_{DA}$ | $\bar{s}_{DA}$ | $\bar{f}_{DA}$ | $\gamma$ | $\Delta$ | $\rho_H$ | Mode |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| #0145 | 1 | 148.0 | 229.7 | 0.25 | 0.29 | 20 | 35 | 13 | 207.3 | 177.6 | 0.41 | 1.8 | 466.1 | 646.9 | DA |
| | 2 | 148.0 | 229.7 | 0.22 | 0.29 | 21 | 34 | 13 | 194.2 | 134.2 | 0.32 | 1.6 | 466.2 | 635.7 | DA |
| | 3 | 148.0 | 229.7 | 0.33 | 0.29 | 21 | 32 | 15 | 185.0 | 138.1 | 0.38 | 1.5 | 457.9 | 628.7 | DA |
| | 4 | 148.0 | 229.7 | 0.25 | 0.29 | 21 | 34 | 13 | 192.5 | 122.4 | 0.32 | 1.6 | 472.7 | 644.1 | DA |
| | 5 | 148.0 | 229.7 | 0.22 | 0.29 | 19 | 34 | 15 | 200.2 | 174.2 | 0.35 | 1.8 | 478.7 | 664.8 | DA |
| #0149 | 1 | 173.4 | 230.3 | 0.30 | 0.31 | 19 | 33 | 17 | 215.5 | 202.6 | 0.41 | 1.7 | 530.7 | 691.2 | DA |
| | 2 | 173.4 | 230.3 | 0.25 | 0.31 | 22 | 35 | 12 | 220.0 | 179.4 | 0.37 | 1.6 | 510.2 | 640.6 | DA |
| | 3 | 173.4 | 230.3 | 0.30 | 0.31 | 20 | 33 | 16 | 213.3 | 209.3 | 0.31 | 1.7 | 512.3 | 647.9 | DA |
| | 4 | 173.4 | 230.3 | 0.30 | 0.31 | 19 | 30 | 20 | 209.6 | 176.0 | 0.31 | 1.6 | 502.3 | 650.1 | DA |
| | 5 | 173.4 | 230.3 | 0.27 | 0.31 | 18 | 31 | 20 | 207.8 | 137.1 | 0.35 | 1.7 | 562.7 | 735.9 | DA |
| #0180 | 1 | 96.0 | 239.0 | 0.24 | 0.25 | 26 | 42 | 3 | 119.9 | 114.3 | 0.43 | 1.6 | 443.2 | 612.1 | DA |
| | 2 | 96.0 | 239.0 | 0.27 | 0.25 | 22 | 37 | 12 | 106.7 | 227.2 | 0.43 | 1.7 | 452.4 | 655.2 | DA |
| | 3 | 96.0 | 239.0 | 0.26 | 0.25 | 25 | 40 | 6 | 114.0 | 103.0 | 0.38 | 1.6 | 450.9 | 626.9 | DA |
| | 4 | 96.0 | 239.0 | 0.24 | 0.25 | 26 | 42 | 3 | 118.7 | 154.1 | 0.33 | 1.6 | 441.3 | 609.0 | DA |
| | 5 | 96.0 | 239.0 | 0.18 | 0.25 | 24 | 41 | 6 | 115.7 | 188.9 | 0.38 | 1.7 | 458.8 | 642.4 | DA |
| #0303 | 1 | 202.9 | 653.4 | 0.26 | 0.23 | 8 | 25 | 1 | 412.4 | 0.0 | 0.20 | 3.1 | 1682.6 | 2533.0 | DA |
| | 2 | 202.9 | 653.4 | 0.26 | 0.23 | 7 | 23 | 4 | 345.4 | 394.9 | 0.50 | 3.3 | 1786.9 | 2769.5 | DA |
| | 3 | 202.9 | 653.4 | 0.30 | 0.23 | 8 | 24 | 2 | 369.9 | 134.7 | 0.40 | 3.0 | 1665.8 | 2514.4 | DA |
| | 4 | 202.9 | 653.4 | 0.27 | 0.23 | 8 | 23 | 3 | 345.4 | 229.0 | 0.33 | 2.9 | 1617.4 | 2464.8 | DA |
| | 5 | 202.9 | 653.4 | 0.22 | 0.23 | 7 | 23 | 4 | 343.9 | 353.1 | 0.35 | 3.3 | 1798.1 | 2753.5 | DA |
| #0307 | 1 | 176.2 | 197.4 | 0.27 | 0.33 | 22 | 37 | 10 | 245.2 | 166.7 | 0.35 | 1.7 | 409.7 | 558.9 | DA |
| | 2 | 176.2 | 197.4 | 0.31 | 0.33 | 22 | 37 | 10 | 248.3 | 166.9 | 0.32 | 1.7 | 405.7 | 552.1 | DA |
| | 3 | 176.2 | 197.4 | 0.31 | 0.33 | 23 | 40 | 6 | 268.9 | 101.9 | 0.30 | 1.7 | 420.7 | 565.1 | DA |
| | 4 | 176.2 | 197.4 | 0.32 | 0.33 | 21 | 38 | 10 | 253.5 | 182.5 | 0.34 | 1.8 | 425.8 | 579.1 | DA |
| | 5 | 176.2 | 197.4 | 0.28 | 0.33 | 20 | 38 | 11 | 254.2 | 190.8 | 0.37 | 1.9 | 435.5 | 601.4 | DA |
| #0346 | 1 | 152.9 | 319.1 | 0.20 | 0.26 | 21 | 37 | 11 | 175.8 | 276.6 | 0.43 | 1.8 | 640.6 | 926.4 | DA |
| | 2 | 152.9 | 319.1 | 0.19 | 0.26 | 22 | 38 | 9 | 176.2 | 121.9 | 0.29 | 1.7 | 668.5 | 935.4 | DA |
| | 3 | 152.9 | 319.1 | 0.30 | 0.26 | 19 | 36 | 14 | 167.5 | 269.2 | 0.37 | 1.9 | 691.4 | 996.1 | DA |
| | 4 | 152.9 | 319.1 | 0.26 | 0.26 | 21 | 37 | 11 | 174.7 | 204.7 | 0.38 | 1.8 | 663.2 | 944.8 | DA |
| | 5 | 152.9 | 319.1 | 0.24 | 0.26 | 21 | 39 | 9 | 180.9 | 229.9 | 0.29 | 1.9 | 674.6 | 948.7 | DA |
| #0431 | 1 | 186.5 | 251.5 | 0.33 | 0.31 | 14 | 21 | 11 | 224.7 | 188.7 | 0.37 | 1.5 | 519.7 | 691.1 | DA |
| | 2 | 186.5 | 251.5 | 0.30 | 0.31 | 14 | 21 | 11 | 228.4 | 146.3 | 0.36 | 1.5 | 531.3 | 704.3 | DA |
| | 3 | 186.5 | 251.5 | 0.24 | 0.31 | 17 | 24 | 5 | 253.1 | 58.6 | 0.24 | 1.4 | 494.8 | 635.4 | DA |
| | 4 | 186.5 | 251.5 | 0.21 | 0.31 | 15 | 21 | 10 | 226.1 | 85.5 | 0.31 | 1.4 | 529.8 | 690.0 | DA |
| | 5 | 186.5 | 251.5 | 0.25 | 0.31 | 18 | 25 | 3 | 264.2 | 41.7 | 0.30 | 1.4 | 475.2 | 608.7 | DA |
| #0684 | 1 | 150.2 | 320.7 | 0.28 | 0.26 | 17 | 32 | 9 | 145.2 | 236.0 | 0.43 | 1.9 | 722.5 | 1030.1 | DA |
| | 2 | 150.2 | 320.7 | 0.25 | 0.26 | 19 | 32 | 7 | 139.6 | 298.7 | 0.39 | 1.7 | 659.1 | 925.2 | DA |
| | 3 | 150.2 | 320.7 | 0.26 | 0.26 | 16 | 31 | 11 | 144.4 | 294.5 | 0.40 | 1.9 | 721.3 | 1040.6 | DA |
| | 4 | 150.2 | 320.7 | 0.29 | 0.26 | 19 | 32 | 7 | 142.0 | 156.2 | 0.43 | 1.7 | 684.9 | 956.7 | DA |
| | 5 | 150.2 | 320.7 | 0.26 | 0.26 | 19 | 34 | 5 | 153.3 | 298.6 | 0.34 | 1.8 | 672.6 | 938.3 | DA |

**Table EC.12 Detailed Calculations of the Comparative Metrics for the Amazon Instances.**