Proceedings of the 2022 Winter Simulation Conference
B. Feng, G. Pedrielli, Y. Peng, S. Shashaani, E. Song, C.G. Corlu, L.H. Lee, E.P. Chew, T. Roeder, and P. Lendermann, eds.

COMBINING SURVIVAL ANALYSIS AND SIMHEURISTICS TO PREDICT THE RISK OF DELAYS IN URBAN RIDESHARING OPERATIONS WITH RANDOM TRAVEL TIMES

Erika M. Herrera
Javier Panadero

Universitat Oberta de Catalunya
Computer Science Department
156 Rambla Poblenou
Barcelona, 08018, SPAIN

Angel A. Juan
Patricia Carracedo
Elena Perez-Bernabeu

Universitat Politècnica de València
Dept. of Applied Statistics and Operations Research
Plaza Ferrandiz-Carbonell
Alcoy, 03801, SPAIN

Rocio de la Torre

Universitat Politècnica de València
Department of Business Organization
Plaza Ferrandiz-Carbonell
Alcoy, 03801, SPAIN

ABSTRACT

More sustainable transportation and mobility concepts, such as ridesharing, are gaining momentum in modern smart cities. In many real-life scenarios, travel times among potential customers' locations should be modeled as random variables. This uncertainty makes it difficult to design efficient ridesharing schedules and routing plans, since the risk of possible delays has to be considered as well. In this paper, we model ridesharing as a stochastic team orienteering problem in which the trade-off between maximizing the expected reward and the risk of incurring time delays is analyzed. In order to do so, we propose a simulation-optimization approach that combines a simheuristic algorithm with survival analysis techniques. The aforementioned methodology allows us to generate not only the probability that a given routing plan will suffer a delay, but also gives us the probability that the routing plan experiences delays of different sizes.

1 INTRODUCTION

Transportation and mobility (T&M) activities represent a key sector in modern cities, and they significantly contribute to social and economic progress worldwide. The novel concepts of sharing and platform economy have promoted the offer of pay-per-use T&M services. Likewise, the emergence of the on-demand economy (services) and the rise of e-commerce (products) have boosted the number of pickups and deliveries in urban and metropolitan areas. This, in turn, increases the need for efficient and sustainable T&M operations. In modern cities, large quantities of data are gathered in real time via electronic devices located inside vehicles and infrastructures (computer chips, sensors, traffic cameras, drones, etc.), transmitted over the internet, and analyzed through intelligent algorithms that allow for predicting the evolution of traffic and making informed decisions. This allows for saving energy consumption and citizens' time, among other social, environmental, and economic benefits. Modern T&M systems include heterogeneous fleets consisting of traditional internal-combustion engine vehicles as well as other types of vehicles using 'greener' (less

978-1-6654-7661-4/22/$31.00 ©2022 IEEE

polluting) technologies, e.g.: bicycles, electric vehicles, plug-in hybrid vehicles, and even unmanned or self-driving vehicles. The increasing incorporation of these low-emission vehicles in T&M activities also raises additional challenges from the strategic, planning, operational, and environmental perspectives (Almouhanna et al. 2020). For instance, the limited driving range and load capacity of most electric vehicles impose additional constraints when designing efficient distribution and collection routes. Due to the social consciousness about promoting sustainability in all transportation means, which includes the need for reducing greenhouse gas emissions, mobility concepts such as ridesharing have gained enormous popularity during the last years (Faulin et al. 2019). The shared mobility for people is an advantageous approach in big cities, which usually present expensive prices and a high level of transportation demand. In ridesharing, people can make use of lower prices and achieve less pollution (Dolati Neghabadi et al. 2021). Since ridesharing mobility is going to be part of a shared activity performed by people in any modern city, the optimization of the trip selection and the vehicle assignment is revealed as essential to improve the customer experience in these scenarios (Martins et al. 2021). Since real-life travel times are random in nature, this profile of excellence in mobility will only be achieved by suitable optimization models that describe stochastic scenarios.

In this work, we model ridesharing operations as a stochastic team orienteering problem (TOP). Here, a set of drivers departing from a origin O have to select and pick up a set of passengers on their way to a destination location D. Each passenger will have to pay a fee for the trip (driver's reward). Hence, the goal is to maximize the total collected reward while taking into account constraints regarding the number of available vehicles as well as deadlines (i.e., the maximum time at which each vehicle should reach its destination). We also consider random traveling times, so the optimization problem becomes more realistic but also stochastic and increasingly challenging. Figure 1 shows a simple ridesharing scenario, modeled as a TOP. Nodes that are not included in any route are the non-visited ones (yellow nodes). Here, the random variable that models the travel time between node i and node j is defined as $T_{ij} = t_{ij} + D_{ij}$, where t_{ij} represents the minimum time necessary to complete the arc connecting both nodes and D_{ij} represents a random delay that follows a given probability distribution (Weibull in this case).

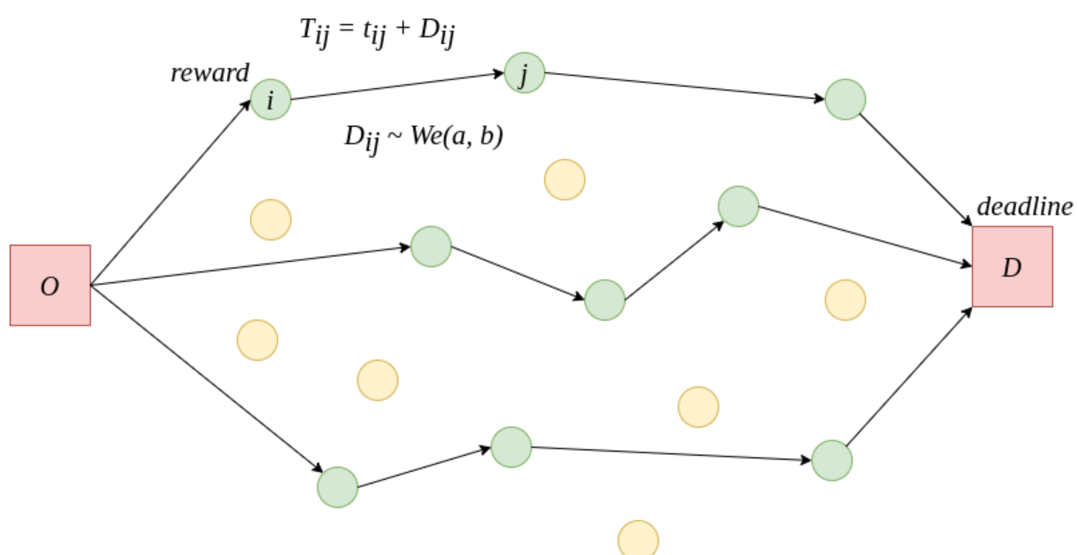

Figure 1: Modeling ridesharing operations as a stochastic team orienteering problem.

In order to deal with the stochastic TOP described above, we combine survival analysis concepts with a simheuristic algorithm (Chica et al. 2020). This combination allows us to obtain, for each proposed routing plan, its corresponding survival function, i.e.: the probabilities associated with delays of different sizes. Being able to generate the survival function associated with one routing plan allows us not only to compute the probability of that plan suffering a delay (i.e., with its last route being completed after t_{max}, so that the finishing time does not belong to the region of punctuality), but also to estimate the probability of suffering a delay of a given size (e.g., what is the probability of exceeding t_{max} by more than 5 minutes). Obviously, this can be quite useful when choosing a solution plan among several promising ones with different survival functions. Hence, the main contribution of this paper is twofold: *(i)* it proposes a simheuristic algorithm for solving a stochastic version of the ridesharing problem; and *(ii)* it shows how survival analysis can be employed to provide probabilistic (richer) information during the risk analysis.

The remaining sections of the paper are structured as follows: Section 2 briefly reviews related articles, while Section 3 defines and introduces the problem to solve. Section 4 describes the proposed simheuristic algorithm and its structure. Section 5 carries out a series of computational experiments to illustrate the performance of the proposed algorithm, while Section 6 analyzes the obtained results. Finally, the main findings and future research lines are given in Section 7.

2 RELATED WORK ON RIDESHARING PROBLEMS

Ridesharing is defined as the sharing of a vehicle driven by the owner, who makes available to other passengers going to the same destination the use of the available passenger capacity of his vehicle. It provides several proven benefits, such as decreasing the number of cars on the road and thus reducing traffic congestion, maximizing the actual use of vehicle capacity, and reducing the carbon footprint. Due to the benefits and increased public awareness of the environment, research on this topic has increased in recent years. The benefits have been described in Twumasi-Boakye et al. (2021) through a simulation for the city of Ann Arbor (USA), the economic and environmental effects have been analyzed for Kuwait in AlKheder (2021), highlighting the reduction in gas emissions in three scenarios, and in New York City by Barann et al. (2017) demonstrating the reduction of gas emissions and the total travel distance of all taxi rides. The effects of ridesharing were analyzed for the city of Shenzhen (China) by Tang et al. (2021) and the findings indicated that it replaces buses and cabs for short-distance trips in the center of the city, and in surrounding areas, it replaces buses and assists in accessing metro stations. Ridesharing also has an impact in areas where public transport is not fully deployed and driving is almost the only solution to move around the city, as is the case in Thessaloniki (Greece), where Ayfantopoulou et al. (2021) have evaluated the initial step of ridesharing implementation in the area. However, ridesharing is not only useful in densely populated areas, but in the rural environment it can be a very useful tool with high impact (Elting and Ehmke 2021).

Ridesharing can be classified as static or dynamic, depending on whether all passenger requests are known in advance or requests are added when passengers are already en-route. The different variants and applications of ridesharing have been detailed in a thorough review in Martins et al. (2021). Since ridesharing is considered a shared service that contributes to reducing the demand for vehicles on urban road networks, cases are usually studied mainly for cars, but the problem of bus ridesharing has also been studied by increasing vehicle capacity in highly populated cities such as Shanghai using exact and approximate algorithms to optimize the ride-matching service (Liu et al. 2019). The first works addressed the resolution of the problem by exact methods. Prominent among them are Agatz et al. (2011), who use an exact method based on a moving horizon strategy, in which drivers and suppliers are matched with the aim of maximizing the profits obtained by the suppliers. If several vehicles are considered, Hosni et al. (2014) proposes a solution based on the Lagrangian decomposition method for multi-vehicle ridesharing systems, in which both the passengers assigned to each vehicle and the routes to choose to reach the destination are optimized. Liang et al. (2020) also propose a solution approach based on a customized Lagrangian relaxation algorithm in order to identify a near-optimal solution for the automated ridesharing taxi problem, including the traffic congestion caused by them. Numerical experiments carried out for the city of Delft (The Netherlands) show that quality solutions can be obtained and that the penalty for delay in the profit target is an effective control parameter to ensure the quality of service while at the same time being profitable for the system. For complex systems involving stochastic scenarios, Naoum-Sawaya et al.

(2015) presents an exact integer programming model for solving a stochastic carpool problem, where the random variable is whether or not the vehicle is available at the time of the request. More recently, Li and Chung (2020) addressed travel time uncertainty, where they present an improved mixed-integer method for solving the problem which considers, in addition to route design and passenger/vehicle assignment, passenger meeting points and their preferred time slots. However, the long computation times made it necessary to transform it into a hybrid heuristic that uses an insertion algorithm with tabu search. Other works that use exact methods to solve variants of ridesharing systems are Masoud and Jayakrishnan (2017) and Chen et al. (2019).

Since ridesharing problems are *NP-hard*, exact methods are generally used for smaller instances, while heuristics and metaheuristics are used for larger instances. Among the most used metaheuristic approaches to solve this problem are tabu search (Li et al. 2018), local search (Chen et al. 2019), etc. In this regard, Jung et al. (2016) combine a nearest vehicle dispatch algorithm, a hybrid-simulated annealing, and an insertion heuristic to solve a dynamic shared-taxi dispatch problem. To solve the variant of the problem with time-dependent travel time uncertainty, Long et al. (2018) use Monte-Carlo simulation (MCS) to estimate the cost of the trip and the time of departure. For the same ridesharing variant, Li and Chung (2020) present a new deterministic formulation as a mixed-integer optimization problem. However, due to the long computation time required to solve an instance with just 44 nodes, the authors proposed a hybrid method based on an insertion algorithm together with a tabu search method. For the case of sharing a taxi between a passenger and parcels with speed window considerations, Do et al. (2018) make a classification of these speed windows by different zones and congestion levels during a day resulting in a dynamic model. To solve it they used a greedy algorithm combined with a local search. On the basis of experimental data from Tokyo taxis, they analyzed the total profit, the cumulative travel time at the end of the day, and the number of shared requests. Wang and Li (2021) show that the shared taxi system solved using a heuristic algorithm and an approximate algorithm succeeded in reducing the number of trips in two real datasets by approximately 30%. Reinforcement learning is also being employed to deal with ridesharing optimization (Qin et al. 2021). In addition to the customer-vehicle matching, there is also the pricing and dispatching decisions (Haliem et al. 2021), which refers to the use of directing drivers to the areas with the highest demand. Hence, for instance, Kim et al. (2022) propose the use of reinforced learning for shared autonomous electric vehicles. Notice, however, that the described articles refer either to simulation or to optimization approaches, but there is a lack of studies combining both. This is precisely what our work achieves by proposing a simheuristic algorithm.

3 PROBLEM DESCRIPTION

The urban ridesharing problem is conceived for its implementation in smart cities. In our case, this system can be represented by an origin-destination graph $G = (V,A)$, in which V includes an origin O, a destination D, and a set of pick-up points for users P. Each of the nodes $v \in V$ included within the network is defined by a coordinate (x_v, y_v). In addition, each of the arcs $(i, j) \in A$ has an associated travel time $t_{ij} > 0$. There is a vehicle $v \in V$ for each of the drivers $c \in C$. Each of these vehicles is associated with a single origin O and a single destination D. At the beginning ($t = 0$), each vehicle $v \in V$ is available at the origin. Starting from there, it should reach its destination D, either on or before of a pre-established arrival time, at_v. Related to the pick-up points $p \in P$, each of them has an associated number of passengers n_p, available at $t = 0$, to be picked up. Also, each of them has the same destination D, and an associated reward r_p, which is offered as a fee to the driver. This reward is proportional to the distance between the collection point and the destination. A solution to this deterministic version of the problem consists in determining a set of routes R ($|V| = |R|$), to be covered by each vehicle and driver. The objective is, therefore, to maximize the reward obtained, within the established deadline for the routes to be completed.

In the stochastic version of the problem considered here, travel times are modeled as positive random variables T_{ij}. These random travel times will follow a probability distribution with known mean $E[T_{ij}] > 0$. Hence, our main goal will be to maximize the expected reward collected by the fleet of vehicles, while trying not to exceed the deadline pre-established for the arrival times. However, since the travel times are now random, delays might occur and some of the vehicles might arrive late to their destination. Whenever this happens, we will consider that a route failure has occurred and, as a consequence of this failure, the reward collected in that route is lost. Notice that other, less severe, penalty costs can also be applied, but for the experiments considered in this paper we have assumed an all-or-nothing value for the collected reward. In addition to this main goal, we will also be interested in obtaining probabilistic information about the size of the possible delays associated with each solution or routing plan.

4 A SIMHEURISTIC APPROACH

To solve the stochastic optimization problem described in the previous section, we have developed a simheuristic algorithm that extends a biased-randomized constructive heuristic. The main concepts of our methodology are explained next.

4.1 A Biased-Randomized Algorithm for the Deterministic Problem

Given an origin O, a destination D, and a set of pick-up points P with their respective rewards, we will start by building an efficiency list of edges e_{ij} (pairs of pick-up points). As proposed in Panadero et al. (2020), this list is then sorted using an efficiency criterion, which is defined as a linear combination of the travel time required to traverse each edge e_{ij} and the aggregated reward generated by visiting the two extreme nodes, i and j. In our experiments, Euclidean distances are used as travel times between each pair of nodes. The low level details and a Python implementation of these concepts are provided in Listing 1. The function *generateeffList* receives as parameters the set of nodes and a tuning parameter $\alpha \in (0,1)$, which is chosen as the one that best performs for each instance. In lines 3 and 4 the starting and finishing depot are set. From line 6 to line 14 the edges connecting the origin O with each pick-up point $p \in P$, as well as the edges connecting $p \in P$ with the destination D, are defined. From this point on, we initialize the efficiency list and, for each pair of nodes $(i,j) \in A$, with $\{i,j\} \notin \{O,D\}$, the 'enriched savings' (efficiency criterion) are computed as follows. First, we compute the cost of the edge as the Euclidean distance (lines 29 and 30). Then, the reward of the edge is computed to be incorporated in the efficiency value (lines 35 and 38) for each of the edge's directions. In line 43 the efficiency list is sorted from higher to lower, and finally returned by the function. Notice that the formula implemented in lines 35 and 38 can be expressed as: $s_{ij} = \alpha(t_{id} + t_{0j} - t_{ij}) + (1-\alpha)(r_i + r_j)$, where t_{ij} represents the traveling time between i and j, d is the destination node, and $(r_i + r_j)$ accounts for the aggregated reward.

The starting point is a 'dummy' solution, which assigns one vehicle $v \in V$ per node. Next, an iterative merging process starts: edges are selected from the sorted list and the associated routes are merged as far as this merge does not violate any capacity or time constraint. Finally, the list of merged routes is decreasingly sorted according to the total reward. Routes with the highest rewards are selected, and the number of selected routes equals the number of available vehicles in the fleet. This constitutes a first routing plan for the deterministic version of the problem. The greedy heuristic described above can be extended into a probabilistic algorithm by using a skewed probability distribution during the edge-selection process. This allows us to quickly generate a huge number of alternative solutions, all of them based on the efficiency criterion defined by the heuristic, but using a pseudo-greedy behavior instead of a greedy one (Belloso et al. 2019).

```python
""" Construct edges with costs and efficiency list from nodes """
def generateeffList(nodes, alpha):
    start = nodes[0] # first node is the start depot
    finish = nodes[-1] # last node is the finish depot

    for node in nodes[1:-1]: # excludes both depots
        snEdge = Edge(start, node) # creates the (start, node) edge (arc)
        nfEdge = Edge(node, finish)
        # compute the Euclidean distance as cost
        snEdge.cost = math.sqrt((node.x - start.x)**2 + (node.y - start.y)**2)
        nfEdge.cost = math.sqrt((node.x - finish.x)**2 + (node.y - finish.y)**2)
        # save in node a reference to the (depot, node) edge (arc)
        node.dnEdge = snEdge
        node.ndEdge = nfEdge

    effList = []
    for i in range(1, len(nodes) - 2): # excludes the start and finish depots
        iNode = nodes[i]
        for j in range(i + 1, len(nodes) - 1):
            jNode = nodes[j]
            ijEdge = Edge(iNode, jNode) # creates the (i,j) edge
            jiEdge = Edge(jNode, iNode)
            ijEdge.invEdge = jiEdge # sets the inverse edge (arc)
            jiEdge.invEdge = ijEdge
            # compute the Euclidean distance as cost
            ijEdge.cost = math.sqrt((jNode.x-iNode.x)**2 + (jNode.y-iNode.y)**2)
            jiEdge.cost = ijEdge.cost # assume symmetric costs
            # compute efficiency as proposed in Panadero et al. (2020)
            ijSavings = iNode.ndEdge.cost + jNode.dnEdge.cost - ijEdge.cost
            edgeReward = iNode.reward + jNode.reward
            ijEdge.savings = ijSavings
            ijEdge.efficiency = alpha * ijSavings + (1 - alpha) * edgeReward
            jiSavings = jNode.ndEdge.cost + iNode.dnEdge.cost - jiEdge.cost
            jiEdge.savings = jiSavings
            jiEdge.efficiency = alpha * jiSavings + (1 - alpha) * edgeReward
            # save both edges in the efficiency list
            effList.append(ijEdge)
            effList.append(jiEdge)
            # sort the list of edges from higher to lower efficiency
            effList.sort(key=operator.attrgetter("efficiency"),reverse=True)

    return effList
```

Listing 1: Computing the efficiency list.

4.2 A Simheuristic for the Stochastic Problem

In order to address the stochastic version of the problem, the biased-randomized algorithm described before can now be extended into a full simheuristic approach. This can be achieved by introducing the biased-randomized algorithm into a multi-start framework and then making use of a simulation component that provides estimates on the expected time employed by each ‘promising’ solution proposed by the optimization component (Hatami et al. 2018). Listing 2 provides a basic example of how this simulation component could be implemented in Python. First, we initialize the stochastic profit of the solution, the list of simulated results, and a variable *totProfit*. Then, in line 6 we start the simulation runs. For each run, we compute the time each route in the current solution takes to be completed, as well as the simulated profit for the solution. To achieve this, we iterate over the routes of the solution. For each edge e_{ij} in a route, we add the reward of node j in the edge. Also, we generate a random value for the travel time T_{ij}, according to the corresponding probability distribution (line 16). The stochastic value is then added to the edge travel time, and this result is accumulated into the route travel time. Now, in line 19 we define a penalty cost: if the travel time of the route is greater than the pre-established deadline (*maxRouteCost*), then the profit of that specific route will be reset to 0 (i.e., if the route fails and suffer a delay, no reward is gained). In lines 25 and 26, we update the maximum travel time inside a solution. Finally, the average reward is computed and stored in line 29. Rabe et al. (2020) offers a discussion on how more advanced simheuristic algorithms can be designed.

```
1  def simulation(sol, nSimulations, maxRouteCost, test):
2      sol.stochasticProfit = 0
3      sol.simResults = []
4      totProfit = 0
5  
6      for i in range(0, nSimulations):
7          maxTimeSol = 0
8          simProfit = 0
9          for route in sol.routes: #iterating over the routes
10             profit = 0
11             routeCost = 0
12             for e in route.edges: #iterating over each edge
13                 customer = e.end
14                 custProfit = customer.reward
15                 if custProfit > 0:
16                     delta = getStochasticValue(scale=test.scale, shape=test.shape)
17                     stochRoute = e.cost + delta #stochastic travel time
18                     routeCost += stochRoute #accumulating stochastic travel time
19                     if routeCost > maxRouteCost:
20                         profit = 0 #penalty cost
21                         break
22                     else:
23                         profit+= custProfit
24             simProfit += profit #updating solution profit
25             if routeCost > maxTimeSol:
26                 maxTimeSol = routeCost #update maximum route travel time
27         totProfit += simProfit
28     totProfit = totProfit / nSimulations #computing average
29     sol.stochasticProfit = totProfit
```

Listing 2: Implementing the simulation component in Python.

5 COMPUTATIONAL EXPERIMENTS

The proposed simheuristic has been implemented using Python 3.7 and tested on a standard PC with a multi-core processor Intel *i*7 and 16 GB of RAM. To perform the experiments, we have extended to the stochastic scenario the well-known deterministic benchmarks for the TOP proposed in Chao et al. (1996). The deterministic benchmark used contains a total of 320 instances, which are distributed in 7 subsets. The instances are identified following the nomenclature ‘pa.b.c’, where ‘a’ represents the subset, ‘b’ defines the number of available vehicles, and ‘c’ identifies the specific instance under study. To extend these instances into a stochastic scenario, instead of the original deterministic travel times, $t_{ij} > 0$, we have considered random travel times, T_{ij}. These random travel times increase the deterministic ones by adding a random delay associated with each edge, $D > 0$, i.e.: $T_{ij} = t_{ij} + D$. In the context of the numerical example included next, we will assume that these delays will follow a Weibull distribution with parameters α and β. The Weibull probability distribution has been selected since, due to its flexibility, it can model almost any random variable with positive values. Due to the existence of random travel times, some vehicles might reach their destination once the deadline is over. Hence, we will be interested not only in maximizing the expected reward collected, but also in considering the reliability of the proposed solution (i.e., the probability that the corresponding routing plan can be completed on or before the pre-established deadline). In addition, we will also use the simulation outcome, in combination with the Kaplan-Meier estimator, to obtain the survival function associated with each 'elite' solution. In this context, elite solutions are the three best solutions obtained by the simheuristic, in terms of the expected reward collected. The survival function will give us valuable information about the probability that each considered plan can be completed on or before any future time. This allows us to answer questions such as "what is the probability that all customers reach their destination up to 5 minutes after the deadline?"

Table 1 shows a summary of the computational results. For each instance, the value associated with our best deterministic (OBD) solution refers to the best-found value in a deterministic scenario. When this solution is employed in a stochastic scenario, we obtain the OBD-S expected (average) value. The table also provides the expected value associated with our best solution (OBS) under a stochastic scenario, i.e., the one obtained by our simheuristic approach. Notice that OBS outperforms OBD-S, which supports the need for employing our simulation-optimization approach in order to take into account the existing uncertainty in the travel times. This effect can be clearly visualized in Figure 2, where OBD acts as a baseline for an ideal (but unrealistic) scenario without uncertainty, and OBS can provide better values than those provided by OBD when the latter is utilized in a stochastic scenario (OBD-S).

	Deterministic Scenario		Stochastic Scenario		
Instance	OBD	OBD Time (s)	OBD-S	OBS	OBS Time (s)
p1.2.i	130	0.2	125.2	128.7	8.7
p2.2.d	160	52.1	73.6	80	112.7
p2.2.i	230	18.8	52.8	152.8	72.2
p2.4.e	70	42.5	58.8	62.8	94.6
p3.2.r	780	27.85	69.9	677	97
p3.4.g	220	31.8	155.7	165.5	111.5
p5.3.f	110	39.9	64.2	102	111.8
p5.4.g	140	13.6	109	139.5	86.2
Average:	**230.0**	**28.3**	**88.7**	**188.5**	**86.8**

Table 1: Rewards obtained by each approach and scenario.

6 SURVIVAL ANALYSIS OF RESULTS

In this section, we will study further the results provided by the simulation component in combination with the Kaplan-Meier estimator that allows us to build the survival function of each routing plan, i.e., a function that returns, for each target time, the probability that the routing plan has not yet been completed (notice that, in this case, we are interested in routing plans with a low probability of survival by the deadline). Hence, this survival function provides probabilistic information on the duration of each routing plan, which allows us to compare different routing plans not only in terms of expected reward but also in terms of the probability that they have been completed at any target time in the future. For our analysis we will focus on the top 3 solutions that our approach generated for instances $p2.4.e$ and $p2.2.d$. For instance $p2.4.e$, Figure 3 represents the survival functions associated with each of the top 3 stochastic solutions obtained (all of them with the same expected reward of 70). In order to enhance the visualization of the curves, we have subtracted 3 units from the times associated with each solution. Notice that $Sol3$ clearly outperforms $Sol2$ at any target time. In turn, $Sol2$ also outperforms $Sol1$ at any target time. Thus, for instance, the survival function of $Sol3$ at target time 4.8 (1.8 in the graph since we subtracted 3 time units) takes a value around 25%. In other words, the probability that this routing plan can finish on or before the aforementioned target time is about 0.75. However, for $Sol1$ and $Sol2$ this probability is nearly 0.

Similarly, for instance $p2.2.d$, Figure 4 represents the top 3 stochastic solution. For a better visualization, we have subtracted 12 units from the times of the solutions. Notice that, at target time 12.65 (0.65 in

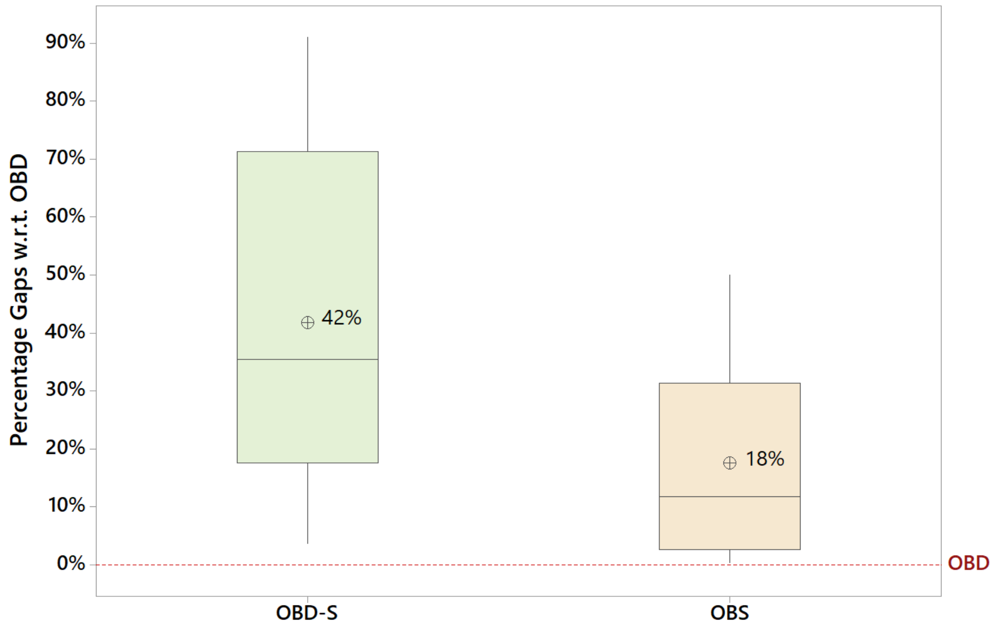

Figure 2: Percentage gaps of OBD-S and OBS w.r.t. OBD.

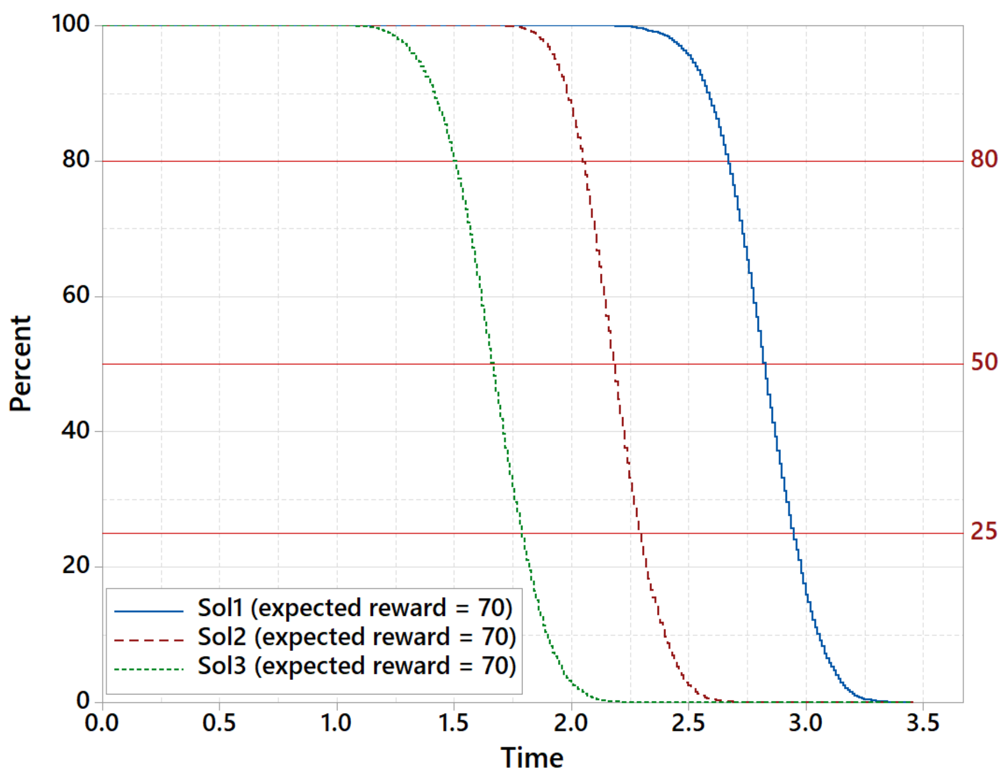

Figure 3: Survival function for instance P2.4.e.

the graph), *Sol*2 will have a higher probability of not being completed yet than *Sol*3. However, by time 13.40, *Sol*2 will have a much lower probability of not having finished (around 0.3) than *Sol*3 (around 0.9). Note also that *Sol*2 always outperforms *Sol*1 (i.e., for any target time, *Sol*2 will have ended with a higher probability than *Sol*1).

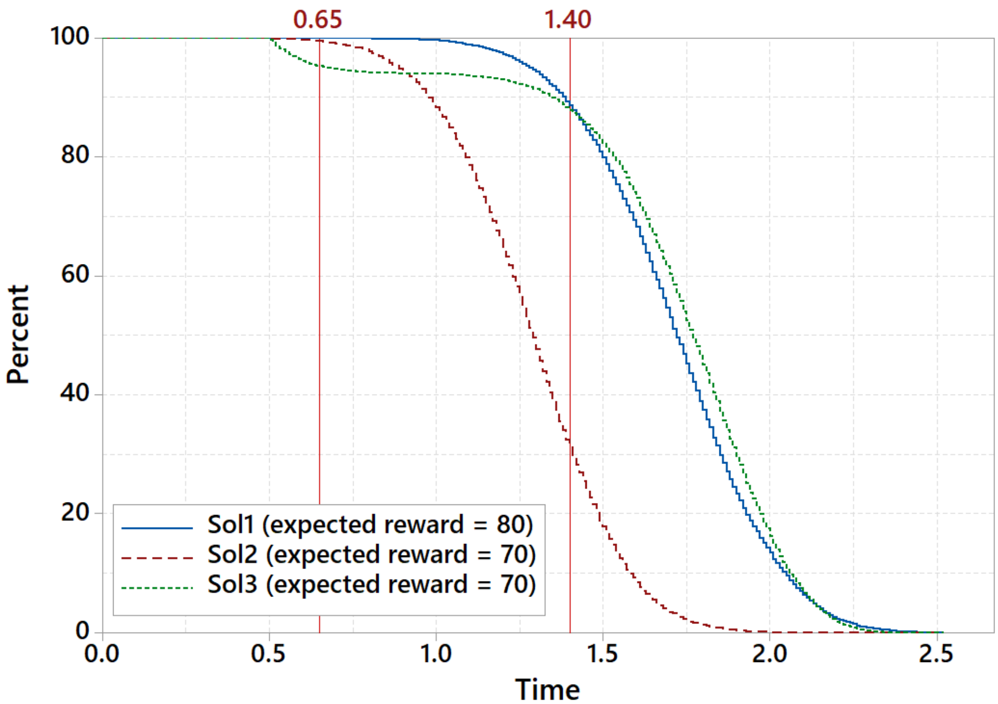

Figure 4: Survival function for instance P2.2.d.

7 CONCLUSIONS AND FUTURE WORK

In this paper, a stochastic version of the ridesharing problem with random travel times is considered. Ridesharing operations are modeled as a team orienteering problem. Here, drivers have to select which customers should be picked up in order to maximize the expected reward collected. At the same time, drivers should be able to reach their destination on or before a pre-established deadline. Of course the existence of random travel times might originate delays in some routing plans. Depending on their size, these delays might be associated with a penalty cost that jeopardizes the benefits of the driver.

In order to provide high-quality solutions to this challenging stochastic optimization problem, we combine a simheuristic algorithm with concepts from survival analysis. Thus, our optimization-simulation approach is not just able to generate 'elite' solutions with high expected rewards, but it also offers probabilistic information about the size of the delays associated with each of these elite solutions. This information might be valuable for managers since they have a more complete understanding of the probabilistic behavior or each proposed routing plan. Hence, questions such as "what is the probability that a specific routing plan causes some of our customers to be late by more than 10 minutes" can be properly answered. Regarding future work, we plan to carry out the following extensions: *(i)* to consider a more realistic scenario in which correlations among delays might occur –e.g., when a geographical area becomes congested, all paths in the area will be subject to high delays; and *(ii)* to extend the simheuristic approach by including a machine learning component that makes use of the simulation feedback to better guide the metaheuristic search in the space of solutions to the stochastic ridesharing problem.

ACKNOWLEDGEMENTS

This work has been partially funded by the Spanish Ministry of Science (PID2019-111100RB-C21-C22 /AEI/ 10.13039/501100011033), as well as by the Barcelona City Council and Fundació "la Caixa" under the framework of the Barcelona Science Plan 2020-2023 (grant 21S09355-001). Moreover, we appreciate the financial support of the Erasmus+ Program (2019-I-ES01-KA103-062602).

REFERENCES

Agatz, N. A., A. L. Erera, M. W. Savelsbergh, and X. Wang. 2011. "Dynamic Ride-Sharing: A Simulation Study in Metro Atlanta". *Transportation Research Part B: Methodological* 45(9):1450–1464.

AlKheder, S. 2021. "Taxi Ride Sharing in Kuwait: Econ-Enviro Study". *Energy* 225:120269.

Almouhanna, A., C. L. Quintero-Araujo, J. Panadero, A. A. Juan, B. Khosravi, and D. Ouelhadj. 2020. "The Location Routing Problem using Electric Vehicles with Constrained Distance". *Computers & Operations Research* 115:104864.

Ayfantopoulou, G., M. N. Konstantinidou, N. Boufidis, and J. M. Salanova Grau. 2021. *Evaluation of a Ride-Sharing Service in Thessaloniki: The Perspective of Both the Service Provider and the Users*, Volume 1278 of *Advances in Intelligent Systems and Computing*. Springer International Publishing.

Barann, B., D. Beverungen, and O. Müller. 2017. "An Open-Data Approach for Quantifying the Potential of Taxi Ridesharing". *Decision Support Systems* 99:86–95.

Belloso, J., A. A. Juan, and J. Faulin. 2019. "An Iterative Biased-Randomized Heuristic for the Fleet Size and Mix Vehicle Routing Problem with Backhauls". *International Transactions in Operational Research* 26(1):289–301.

Chao, I.-M., B. L. Golden, and E. A. Wasil. 1996. "The Team Orienteering Problem". *European Journal of Operational Research* 88(3):464–474.

Chen, W., M. Mes, M. Schutten, and J. Quint. 2019. "A Ride-Sharing Problem with Meeting Points and Return Restrictions". *Transportation Science* 53(2):401–426.

Chica, M., A. A. Juan, C. Bayliss, O. Cordón, and W. D. Kelton. 2020. "Why Simheuristics? Benefits, Limitations, and Best Practices when Combining Metaheuristics with Simulation". *Statistics and Operations Research Transactions* 44(2):311–334.

Do, P. T., N. V. D. Nghiem, N. Q. Nguyen, and Q. D. Pham. 2018. "A Time-Dependent Model with Speed Windows for Share-a-Ride Problems: A Case Study for Tokyo Transportation". *Data and Knowledge Engineering* 114:67–85.

Dolati Neghabadi, P., M.-L. Espinouse, and E. Lionet. 2021. "Impact of Operational Constraints in City Logistics Pooling Efficiency". *International Journal of Logistics Research and Applications* 0(0):1–25.

Elting, S., and J. F. Ehmke. 2021. "Potential of Shared Taxi Services in Rural Areas-A Case Study". In *Transportation Research Procedia*, Volume 52, 661–668.

Faulin, J., S. Grasman, A. A. Juan, and P. Hirsch. 2019. *Sustainable Transportation and Smart Logistics: Decision-Making Models and Solutions*. Amsterdam: Elsevier.

Haliem, M., G. Mani, V. Aggarwal, and B. Bhargava. 2021. "A Distributed Model-Free Ride-Sharing Approach for Joint Matching, Pricing, and Dispatching Using Deep Reinforcement Learning". *IEEE Transactions on Intelligent Transportation Systems* 22(12):7931–7942.

Hatami, S., L. Calvet, V. Fernández-Viagas, J. M. Framiñán, and A. A. Juan. 2018. "A Simheuristic Algorithm to Set Up Starting Times in the Stochastic Parallel Flowshop Problem". *Simulation Modelling Practice and Theory* 86:55–71.

Hosni, H., J. Naoum-Sawaya, and H. Artail. 2014. "The Shared-Taxi Problem: Formulation and Solution Methods". *Transportation Research Part B: Methodological* 70:303–318.

Jung, J., R. Jayakrishnan, and J. Y. Park. 2016. "Dynamic Shared-Taxi Dispatch Algorithm with Hybrid-Simulated Annealing". *Computer-Aided Civil and Infrastructure Engineering* 31(4):275–291.

Kim, S., U. Lee, I. Lee, and N. Kang. 2022. "Idle Vehicle Relocation Strategy Through Deep Learning for Shared Autonomous Electric Vehicle System Optimization". *Journal of Cleaner Production* 333:130055.

Li, X., S. Hu, W. Fan, and K. Deng. 2018. "Modeling an Enhanced Ridesharing System with Meet Points and Time Windows". *PloS One* 13(5):e0195927.

Li, Y., and S. H. Chung. 2020. "Ride-Sharing under Travel Time Uncertainty: Robust Optimization and Clustering Approaches". *Computers & Industrial Engineering* 149:106601.

Liang, X., G. H. D. A. Correia, K. An, and B. van Arem. 2020. "Automated Taxis' Dial-a-Ride Problem with Ride-Sharing considering Congestion-Based Dynamic Travel Times". *Transportation Research Part C: Emerging Technologies* 112:260–281.

Liu, K., J. Zhang, and Q. Yang. 2019. "Bus Pooling: A Large-Scale Bus Ridesharing Service". *IEEE Access* 7:74248–74262.

Long, J., W. Tan, W. Szeto, and Y. Li. 2018. "Ride-Sharing with Travel Time Uncertainty". *Transportation Research Part B: Methodological* 118:143–171.

Martins, L. d. C., R. de la Torre, C. G. Corlu, A. A. Juan, and M. A. Masmoudi. 2021. "Optimizing Ride-Sharing Operations in Smart Sustainable Cities: Challenges and the Need for Agile Algorithms". *Computers & Industrial Engineering* 153:107080.

Masoud, N., and R. Jayakrishnan. 2017. "A Real-Time Algorithm to Solve the Peer-to-Peer Ride-Matching Problem in a Flexible Ridesharing System". *Transportation Research Part B: Methodological* 106:218–236.

Naoum-Sawaya, J., R. Cogill, B. Ghaddar, S. Sajja, R. Shorten, N. Taheri, P. Tommasi, R. Verago, and F. Wirth. 2015. "Stochastic Optimization Approach for the Car Placement Problem in Ridesharing Systems". *Transportation Research Part B: Methodological* 80:173–184.

Panadero, J., C. Currie, A. A. Juan, and C. Bayliss. 2020. "Maximising Reward from a Team of Surveillance Drones: A Simheuristic Approach to the Stochastic Team Orienteering Problem". *European Journal of Industrial Engineering* 14(4):485–516.

Qin, G., Q. Luo, Y. Yin, J. Sun, and J. Ye. 2021. "Optimizing Matching Time Intervals for Ride-Hailing Services using Reinforcement Learning". *Transportation Research Part C: Emerging Technologies* 129:103239.

Rabe, M., M. Deininger, and A. A. Juan. 2020. "Speeding Up Computational Times in Simheuristics Combining Genetic Algorithms with Discrete-Event Simulation". *Simulation Modelling Practice and Theory* 103:102089.

Tang, J., F. Gao, C. Han, X. Cen, and Z. Li. 2021. "Uncovering the Spatially Heterogeneous Effects of Shared Mobility on Public Transit and Taxi". *Journal of Transport Geography* 95:103134.

Twumasi-Boakye, R., X. Cai, J. Fishelson, and A. Broaddus. 2021. "Simulation of Potential Use Cases for Shared Mobility Services in the City of Ann Arbor". *Transportation Research Record* 2675(9):848–860.

Wang, Y., and M. Li. 2021. "Optimization Algorithm Design for the Taxi-Sharing Problem and Application". *Mathematical Problems in Engineering* 2021.

AUTHOR BIOGRAPHIES

ERIKA HERRERA is a predoctoral researcher in the ICSO group at the Universitat Oberta de Catalunya (Spain). She holds a BSc degree in Mathematics and a Master's degree in Computational Engineering and Mathematics. Her main research interests are theoretical and applied mathematics, simulation, and optimization. Her email address is eherreramac@uoc.edu.

ANGEL A. JUAN is a Full Professor in the Dept. of Applied Statistics and Operations Research at the Universitat Politècnica de València (Spain), as well as Invited Professor at University College Dublin (Ireland) and Universidade Aberta (Portugal). Dr. Juan holds a Ph.D. in Industrial Engineering and an M.Sc. in Mathematics. He completed a predoctoral internship at Harvard University and postdoctoral internships at the Massachusetts Institute of Technology and the Georgia Institute of Technology. His main research interests include applications of simheuristics and learnheuristics in computational logistics and finance. He has published over 125 articles in JCR-indexed journals and more than 300 papers indexed in Scopus. His website address is http://ajuanp.wordpress.com and his email address is ajuanp@upv.es.

JAVIER PANADERO is an Associate Professor of Simulation and Computer Science in the Computer Science, Multimedia and Telecommunication Department at the Universitat Oberta de Catalunya (Barcelona, Spain). He is also a Lecturer at the Euncet Bussines School, and a member of the ICSO research group. He holds a Ph.D. and a M.S. in Computer Science. His major research areas include Simulation-Optimization Algorithms, Artificial Intelligence, Data Science, and Parallel and Distributed Systems. He has co-authored more than 70 articles published in ISI JCR journals and conference proceedings. His email address is jpanaderom@uoc.edu.

PATRICIA CARRACEDO is an Assistant Professor in the department of Applied Statistics and Operations Research at the Universitat Politécnica de Valéncia. She holds a Ph.D. in Statistics and Operational Research from the same university. Her major research areas are: spatio-temporal statistics, panel data models, and text mining analysis. Her email address is pcarracedo@eio.upv.es.

ELENA PEREZ-BERNABEU is an Assistant Professor in the Applied Statistics & Operations Research Department at the Universitat Politècnica de València (Alcoi, Spain). Dr. Perez-Bernabeu holds a Ph.D. in Industrial Management from the Universitat Politècnica de València. Her major research interests lie in the area of optimization, simulation, vehicle routing problems, and sustainability. Her email address is elenapb@eio.upv.es.

ROCIO DE LA TORRE is an Assistant Professor in the Department of Business Organisation at the Universitat Politécnica de Valéncia. She holds a Ph.D. and a Bachelor Degree in Industrial Engineering from the Universitat Politécnica de Catalunya. Her major research areas are: mathematical programming for strategic planning decisions and the supply chain design. Her email address is mrtormar@omp.upv.es.