A Sequential Clustering Method for the Taxi-Dispatching Problem Considering Traffic Dynamics

Negin Alisoltani, Mahdi Zargayouna, Ludovic Leclercq

A Sequential Clustering Method for the Taxi-Dispatching Problem Considering Traffic Dynamics

Negin Alisoltani
Is with Université Gustave Eiffel, COSYS-GRETTIA, Marne-la-Vallée and Université Gustave Eiffel, Université de Lyon, ENTPE, LICIT, France.
Email: negin.alisoltani@univ-eiffel.fr

Mahdi Zargayouna
Is with Université Gustave Eiffel, COSYS-GRETTIA, Marne-la-Vallée, France.
Email: mahdi.zargayouna@univ-eiffel.fr

Ludovic Leclercq
Is with Université Gustave Eiffel, Université de Lyon, ENTPE, LICIT, France.
Email: ludovic.leclercq@univ-eiffel.fr

xxxxxxx

***Abstract*—**Taxis are an important transportation mode in many cities due to their convenience and accessibility. In the taxi-dispatching problem, sometimes it is more beneficial for the supplier if taxis cruise in the network after serving the first request to pick up the next passenger, while sometimes it is better that they wait in stations for new trip requests. In this article, we propose a rolling-horizon scheme that dynamically optimizes taxi dispatching considering the actual traffic conditions. To optimize passenger satisfaction, we define a limitation for passenger waiting time. To be able to apply the method to large-scale networks, we introduce a clustering-based technique that can significantly improve the computation time without harming the solution quality. Finally, we test our method on a real test case considering taxi requests with personal car trips to reproduce actual network loading and unloading congestion during peak hours.

Digital Object Identifier 10.1109/MITS.2020.3014444
Date of current version: 8 September 2020

*Corresponding author

1939-1390/20©2020IEEE

Taxis play an essential role as a transportation alternative in many cities because of their convenience and accessibility. The regulation of the taxi industry and how the taxi problem has been addressed in the literature have a long history. The authors in [36] presented a comprehensive review of the different models developed for the taxicab problem.

Today, the spread of mobile devices and the development of GPS make it possible for all of the transport operators to dynamically adapt the transportation supply to travel demand. These new technologies have made considerable changes in the transportation modes as well as taxis. With traditional taxi services, the demand is requested via a call center, and the drivers must rely on experience to handle the optimization process. With the app-based platforms, however, the requests are centralized, and more advanced optimization techniques can be implemented to assign passengers to the fleet [9]. Consequently, many pieces of research have been performed on different methods to improve the efficiency of taxis, especially approaches for taxi recommending under different conditions and objectives [17], [29].

The critical issue in dynamic assignment is to be able to solve the optimization process very quickly, especially for large-scale networks with thousands of requests, while respecting the current traffic situation in the network at the time. In the taxi-assignment problem, the objective is to reduce the taxi travel time and distance. Also, it is important to minimize the passengers' waiting times.

Another serious point is to predict the travel times accurately to determine the availability of the taxi and pickup/drop-off times. This important problem, to the best of our knowledge, has not yet received enough attention. In the literature, authors usually assume that the travel times during the assignment process stay the same during the execution of the vehicle schedules. However, the network congestion can have significant impacts on vehicle speed and travel time, which means that the number of private cars that are currently driving besides the taxis should be considered as well [1].

In this article, we present an algorithm at full spatial scale to optimize taxi operation, considering the current network traffic conditions over a given rolling horizon. This algorithm is capable of finding the routes with minimum travel times to serve all of the passengers with acceptable waiting times. Also, the system needs fewer taxis to serve a higher number of passengers because the optimization process adjusts the sequence of trip requests. As depicted in Figure 1, with this method, one taxi can serve three possible sequential trips from one taxi station to another in a shorter time than if these three requests were served with three different taxis.

To speed up the optimization computation time, especially for large-scale problems, we introduce a heuristic method based on clustering. We apply the clustering method based on the "Sequential Function" on the requests to put those who have more potential to be served successively by the same taxi in the same cluster. Then we employ the dynamic taxi-dispatching algorithm (DTaD) within each cluster.

The goal of this study is to assess the performance of a taxi service under the optimal situation on a large scale considering the traffic dynamics. The optimal situation is the scenario in which all of the total costs (for the taxis and passengers) are minimal. To be sure that we can find most of the sequential taxi trips for each taxi and minimize the taxi travel distance, we choose a rolling horizon of 10 min. With a short prediction horizon of next requests, it is not possible to optimally manage a taxi fleet, and we have to resort to a very simple strategy like assigning the nearest vehicle. As we want to dynamically optimize the fleet management, we need to assume that the requests are predicted over the rolling horizon. We do not question this assumption in this article.

To dynamically test the performance of our system, we use a traffic simulator that predicts the mean network speed with respect to the current accumulation in the system every second [2], [21], [27]. Such a simulator has been inspired by

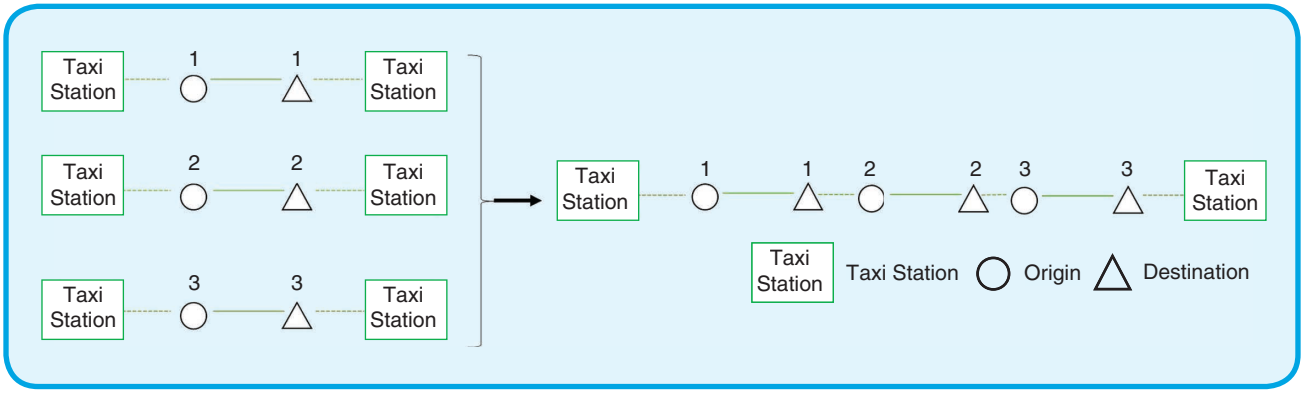

FIG 1 An example of serving the requests in sequence.

the concept of a macroscopic fundamental diagram (MFD) [14]. We use a real case study from Lyon, France, with the data about all of the trips (taxi trips and also personal cars in the network). Hence, we can assess the influence of personal trips in the network on the dynamic taxi-service performance and vice versa.

Literature Review

Taxi-recommendation systems aim to suggest a sequence of pickup points that can serve passengers with the shortest driving distance, as in [16], [20], [42], and [45]. The authors in [43] provide an optimized online querying subsystem to calculate the probability of getting a taxi.

Taxi dispatch systems map the customers with drivers for traveling a certain distance from the pickup locations, as in [4], [11], and [22]. Recently, many studies have focused on order dispatch in taxi networks [6], [13], [46]. Xu et al. [44] propose an order-dispatch algorithm to optimize the platform's long-term efficiency. However, their proposed method is costly in terms of computation time and not flexible to transfer knowledge across cities. Maciejewski et al. [24] model large-scale, real-time taxi dispatching and use a strategy that involves classifying the system state into two categories. If there is an oversupply of vehicles relative to unassigned passengers, requests are assigned to the nearest idle vehicle based on the first-in, first-served rule. If there is an undersupply of vehicles, when a vehicle becomes idle, it is assigned to the nearest unassigned traveler request.

The taxi-dispatching problem is solved exactly by modeling it as a mixed-integer program or by applying the heuristics in the literature. The main exact solutions are extensions of the branch-and-cut method, which is based on the branch-and-bound procedure, where cutting planes are added to the problems in the branch-and-bound tree [7], [31]. In this research, we propose an algorithm based on the branch-and-bound concept to explore all of the possible combinations of trips to be served in sequence.

It has been indicated in the literature that the patterns of demands and supplies are spatially–temporally dependent [41]. Much of the research in this domain uses different clustering methods to consider these dependencies, such as dividing the time into several time slots or dividing the space into several clusters, road segments, or cells [10], [15], [33], [45]. Qiang and Shuang-Shuang [34] propose an algorithm to use the data set of taxi get-off points to achieve the clustering of taxis on urban roads and compare their method with classical clustering methods. However, the taxi-clustering data in their study are conducted in a static environment. Bard and Jarrah [3] demonstrate that, for large-scale problems, an appropriate solution is clustering the demand nodes and downsizing the network.

Some researches try to limit the feasible region with clustering methods to speed up the computation. They usually divide the demand nodes in the network into geographically dense clusters [30], [35]. In our proposed method, we make the clusters based on a similarity function. Therefore, even the trips that are not geographically in the same cluster but have the potential to be picked up sequentially with the same vehicle can be in the same cluster; thus, the clustering method does not remove possible optimal solutions.

One of the recent works on the clustering of the trips is reported in [37]. The investigators introduce the notion of a shareability network to quantify the spatial and temporal compatibilities of individual trips in a dynamic environment. In their method, two trips are shareable if they would incur a delay of no more than 5 min. Then, Vazifeh et al. [39] modify the idea to model the sharing of vehicles instead of rides and address the minimum fleet problem in on-demand urban mobility.

In this research, we propose the concept of "sequential function" for the same purpose to assess the possibility of serving two trips with the same car in sequence. Our proposal employs a method that reduces the number of required taxis because possible sequences and not only initial matching are considered. We prove that the proposed method provides high-quality solutions very quickly and overcomes previous methods.

This is very important if we envision the shift from classical human-driven taxi fleets to autonomous vehicles. Different research focuses on different objectives in the problem related to passengers [38], drivers [8], [23], providers [18], [25], [32], or the network [5].

In the taxi problem, it is important balance all of the participants' objectives. Our method optimizes the travel time, minimizes the total travel distance, and reduces the number of cars while keeping the passenger waiting times at acceptable levels. Also, we assess the impact of the taxi service on traffic congestion to find the best configuration for the system to improve the network traffic condition.

App-Based Platform

The proposed system function is illustrated in Figure 2. The passenger launches his or her request in (a), which contains the origin point, the destination point, and the desired pickup time, via an e-hailing application in the smartphone (b). The app is connected to a server, which can access the supplier data (c) and network (e). These data encompass all of the waiting locations' situations, taxicab timetables, current location of each taxi, and current vehicle mean speed in the network. The mean speed is computed based on the current traffic situation. When the algorithm (d) receives the requests, it searches all of the waiting or moving taxicabs in the system to find the best assignment with the minimum taxi travel time and passenger waiting time for each request. Then, it sends the defined schedule to the fleet and a response to each passenger.

The DTaD

We have defined an algorithm for taxi dispatching based on the branch-and-bound concept to find the best schedule for the taxis and serve requests with minimum passenger waiting time. First, we present the primary algorithm, which can do the process for short- or medium-scale problems; then, we introduce our heuristic method based on clustering to speed up the algorithm and make it qualified to handle large-scale problems with thousands of requests.

The taxi leaves the closest waiting location to serve the first passenger. After dropping off the passenger at the destination, the cab faces two choices. The first option is to continue to another origin, and the second is to go back to the nearest waiting location to wait for the next passenger. The designed algorithm computes all of the possible routes for the received requests. If the waiting time for each passenger is acceptable, the algorithm calculates the car travel time. In the end, when it computes all of the possible options based on the exploration of a tree (branch and bound), it chooses the optimal situation. At each iteration, the algorithm removes the assigned requests and continues with the rest of the requests until it inserts all of the passengers into the taxi schedules. Table 1 lists the notations that we used, and Algorithm 1 provides the primary algorithm steps.

The algorithm works iteratively. The objective function is to minimize the taxi travel time. At each iteration, it builds a tree of routes for all of the trip requests that have not been assigned and, at the end, chooses the best branch of the tree. For the next iteration, it removes all of the trips of the previous iteration's selected branch and builds a new tree with the remaining requests. This process continues until all of the requests are served. Figure 3 depicts a brief example of the algorithm performance for five requests. First, the algorithm finds the nearest taxi to each request. Then, in the next step, it starts the iterations. In the first iteration, the algorithm finds the best route among the possible solutions. Afterward, it removes passengers 1 and 3 from the request set and continues the computations. Finally, taxi A serves passengers 1 and 3, taxi E serves passengers 4 and 5, and taxi B serves passenger 2.

The presented algorithm solves the static dispatching problem. To solve the dynamic and real-time problem, we employ a rolling-horizon solution approach. This reactive approach is based on a scheduling formulation that solves the static dispatching problem iteratively by moving forward the optimization horizon in each iteration.

The proposed algorithm optimizes taxi routes to serve all of the passengers. This algorithm minimizes travel time and distance and mobilizes fewer taxis to serve the passengers. However, since the algorithm explores all of the feasible routes, the computation time increases greatly and rapidly with increasing numbers of requests.

A Heuristic Method Based on Clustering

In large-scale problems, for dynamic dispatching, the system must be able to find the schedules on very short notice. To make the proposed algorithm fast enough to handle large-scale problems, we propose a heuristic method based on clustering. The originality of our proposal is that it reuses the DTaD method presented earlier while feeding it with fewer requests to optimize. This selection of requests must be performed such that it does not overlook promising pairs of requests, i.e., those that can be efficiently served together. In our method, limiting the algorithm to make branches with only the trips that have a greater possibility of creating the optimal assignment can narrow the search of feasible solutions. To apply this limitation, we define a clustering method to make clusters of requests that could conceivably be efficiently served with the same taxi. Then, the algorithm is executed with each cluster separately.

To do so, we define the "sequence function" (SF) between request i and request j ($i, j \in A$). We compute $SF_{i,j}$ for each pair of trips, and the function value is the difference between the travel time when the two trips are served sequentially with the same taxi and the travel time to serve each trip individually with two separate taxis. If the travel time between the first destination and the second origin is less than the summation of the travel time between the first origin and the closest waiting location and the travel time between the nearest waiting location and the second origin, then serving the trips in a sequence is more beneficial than serving them separately. It means that the SF here is the difference between the passengers' waiting times when the trips are in sequence and the passengers' waiting times when the trips are independent.

The following equations demonstrate how we compute $SF_{i,j}$ for each pair of trips:

$$T_i + T_j = WT_i + DTT_{\text{Origin}_i,\text{Destin}_i} + WT_j + DTT_{\text{Origin}_j,\text{Destin}_j}; \forall i, j \in N, \quad (1)$$

$$SF_{i,j} = TT_i + TT_j - (DTT_{\text{Origin}_i,\text{Destin}_i} + DTT_{\text{Origin}_j,\text{Destin}_j} + WT_i' + WT_j'), \quad (2)$$

and, finally,

$$SF_{i,j} = WT_i + WT_j - (WT_i' + WT_j'). \quad (3)$$

Afterward, we have the SF value for each pair of trips, and it creates a sequential matrix, which is a kind of dissimilarity matrix for the received requests that can be used in the clustering process. When we make clusters based on the SF, we put the trip requests that are more potential to be efficiently served by the same taxi in the same cluster (the trips that have a lower SF value). To create the clusters, we first use the multidimensional scaling method

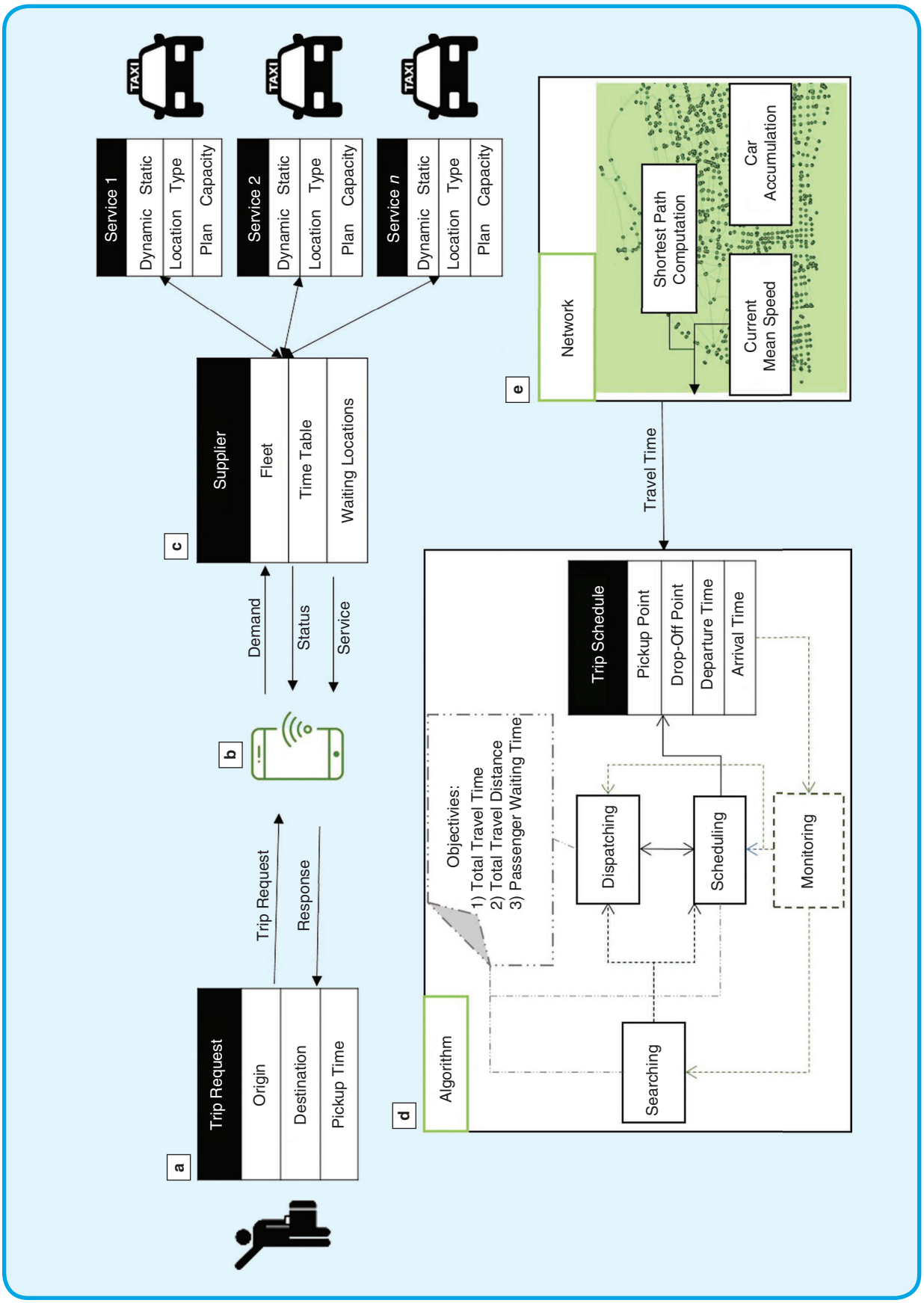

FIG 2 A real-time taxi-dispatching system includes the (a) trip request, (b) smartphone app, (c) supplier data, (d) algorithm, and (e) network.

Table 1. A description of the major notations used.

Variable	Description
C	Set of taxis
A	Set of requests
B	Set of branches in the algorithm
origin_i	Origin of request i
destin_i	Destination of request i
Results	Set of routes defined by the algorithm
WL_c	Closest waiting location to taxi c
WT_i	Waiting time of passenger i
WT'_i	Waiting time of passenger i when he or she is served individually
MWT	MWT for each passenger
TT_b	Total travel time of route b
T_i	Passenger i travel time
$\text{DTT}_{i,j}$	Direct travel time from points i to j
M	A large enough number

Algorithm 1. The DTaD.

```
while A is not empty do
    nob = 1;
    for request i ∈ A do
        minimum_distance = M;
        for taxi c ∈ C do
            if D_{o_i,WL_c} ≤ minimum_distance then
                minimum_distance = D_{o_i,WL_c};
                taxi_i = c;
        Create branch b_nob for taxi c to serve passenger i from WL_c;
        Put the branch b_nob in branch set B;
        Compute TT_nob;
        nob = nob + 1
    while B ≠ ∅ do
        for branch b ∈ B do
            createbranch = false;
            for request j ∈ A − Trips on b do
                Compute WT_j;
                if WT_j ≤ MWT then
                    Create branch b_nob by adding request j to branch b;
                    Compute TT_nob;
                    Put the branch b_nob in set B;
                    nob = nob + 1;
                    createbranch = true
            if createbranch = false then
                Add the closest WL from the last request on b to it;
                Compute TT_nob;
                Put branch b in Results set;
                Remove branch b from B
        minimum − TT = M;
        for branch R ∈ Results do
            if TT_R ≤ minimum_TT then
                minimum_TT = TT_R;
                Result = R
        Remove the trips on Result from A;
        Put route Result in Final Results set
return Final Results
```

(compare [40]) to convert the dissimilarity matrix to a distance matrix. The most common clustering approach is the k-means technique (compare [26]). To have clusters with the same size, we apply the modified k-means clustering method, which customizes the cluster sizes (compare [12]).

With this heuristic method, the algorithm is executed inside each cluster to make the branches using only the trips in the considered cluster. Hence, compared with the primary algorithm, we may lose some feasible solutions, but, on the other hand, the computation time should decrease significantly as the number of starting branches decreases. The idea is to find the best tradeoff between the size of clusters and computation time considering the objective function values.

Figure 4 provides an example of the clustering method for the case in Figure 3. First, we compute the SF between each pair of requests and build a 5 × 5 dissimilarity matrix [Figure 4(a)]. Then, with the multidimensional scaling method, we get the configuration of five points in two dimensions [Figure 4(b)]. In the next step, we apply the modified k-means clustering method with a cluster size of three. The result is two clusters of trips. The first cluster contains request numbers 1 and 3, and the second includes requests 2, 4, and 5. We apply DTaD inside each group and, finally, get the final routes for taxis [Figure 4(c)].

The important point in using the heuristic method is to make a decision about the size of the clusters. Bigger cluster sizes can give better results but with a high computational price. Therefore, it is necessary to determine the best tradeoff between the quality of the objective function and the computation time to choose the appropriate cluster size. In the next section, first, we launch the algorithm with different cluster sizes to find such a tradeoff.

The waiting time is a critical issue in taxi-dispatching problems. The objective of DTaD is also to minimize the travel time while satisfying a constraint on passengers' waiting times. The maximum acceptable waiting time (MWT) in our algorithm defines the passengers' waiting times, and it is important to choose a proper value for this parameter. Bigger values for *MWT* give more flexibility to the provider but, on the other side, lead to passenger dissatisfaction. Another goal of this article is to fix a suitable value for this parameter.

The last important parameter is the number of taxis in the fleet. This parameter has a crucial role for provider costs. In this article, we are interested in assessing the impact of different values for this parameter, as it has important influences on the passengers' waiting times and, also, the traffic situation.

Experiments

Case Study

In the proposed research, we apply our method on a realistic origins–destinations (ODs) matrix for the city of Lyon, France. The network is loaded with travelers of all ODs with a given departure time to represent 4 h of the network with more than 62,000 requests based on the study of [19]. This network has 1,883 nodes and 3,383 links. The area is depicted in Figure 5.

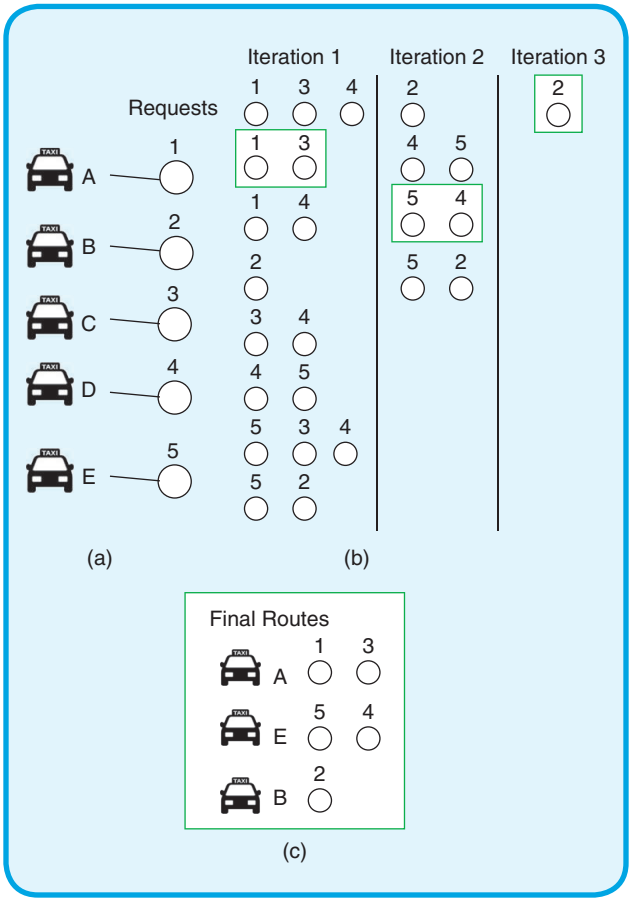

FIG 3 An example of the algorithm performance. (a) The algorithm finds the nearest taxi to each request. (b), Next, it starts the iterations. (c) Finally, it provides the optimized recommendation.

The origins set contains 94 points, the destinations set includes 227 points, and the waiting locations set contains 237 points on the network. We do the experiments for the morning peak hour from 6:30 a.m. to 10:30 a.m. for 4 h. The trips number during this period is 62,450. Some trips start from within or end outside of the network. They are considered only from their entering point or up to their existing location. We serve the trips that are completely included inside the network with taxis in our assessment, which contains 11,235 trips.

The MWT for the passengers is defined based on their trip length. It is equal to a fixed value plus 1 min for each 1 km. We do a sensitivity analysis to find the best value for the fixed part of the MWT, which can provide a reasonable compromise between the passengers' waiting times and total taxi travel distance. To detect the best tradeoff between the size of clusters and the computation time, we compute the MWT for each trip with a fixed value equal to 6 min.

We have defined two kinds of waiting locations: the local waiting locations and the central depot. In the beginning, there are 142 taxis in the local waiting locations and 500 taxis in the central depot. For each trip, the algorithm ranks the waiting locations based on the distance to the first origin and then sends a taxi from the closest nonempty stop location to serve the first passenger. When the assigned trips end, the taxi goes to the nearest waiting location to the last destination. We consider a 1-min service time for each passenger to get in and out of the taxi, which is computed in the total travel time.

One of the goals of this research is to figure out the performance of a taxi-dispatching system under the optimal situation. In this case study, we fully monitor traffic dynamics as we assess both service and personal trips in the network. To find the near-optimal dispatching of taxi services in the dynamic traffic conditions, we choose a longer

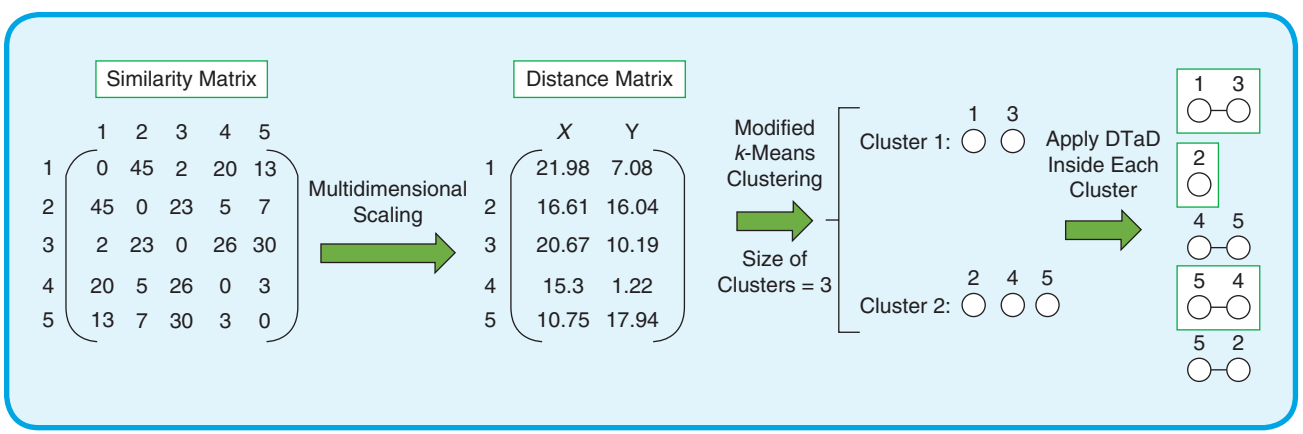

FIG 4 An example of the clustering method.

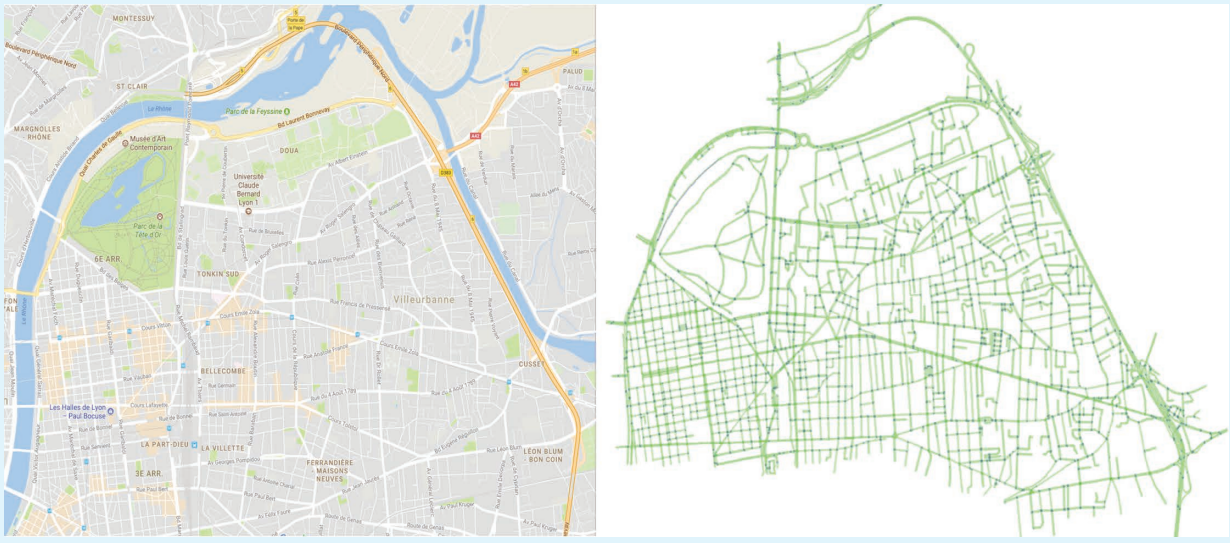

FIG 5 The Lyon 6e + Villeurbanne. (a) Mapping data from Google 2018. (Source: Google.) (b) The traffic network.

time step compared to other dynamic methods to guarantee that we can find most of the optimal sequential trips for the taxis. Hence, to solve the dispatching problem dynamically, we apply the DTaD method every 10 min, considering we have a perfect knowledge of all requests over such a time horizon. The computations are carried out on a desktop with an Intel Xeon core E5-2620 with two processors, 64 GB of random-access memory, and a Windows 10 operating system running C++ Visual Studio 2013.

Simulation Model

The goal of this study is to assess the performance of DTaD considering the traffic dynamics, so a simulation platform, which is able to simulate the time evolution of traffic flows and speeds on the road network, is used to simulate the function of DTaD. In this research, we use the trip-based MFD to accommodate individual trips while keeping a very simple description of traffic dynamics [21], [27], [28]. This method can compute the distance that a car can pass during a particular time based on the vehicle's mean speed at the time. In addition to the taxis that are moving in the network, personal cars are also circulating in the network. Therefore, the accumulation of vehicles in the system and, consequently, the mean speed depend on the accumulation of cars in the network at each specific time. We update the situation of all of the vehicles every second based on the current mean speed of cars in the simulations. Also, we compute the travel times in DTaD based on this speed to consider the traffic dynamics in the network.

Therefore, we use the trips-based MFD as the dynamic simulator and the predicted speed at the beginning of each horizon as the prediction model, which can be calibrated, to do the optimization. The mean speed is changing over the full horizon but not that much over the next 10 min. During the loading of the network, the estimation of the real value of the mean speed is a little overconfident, and, during the unloading, it is a little conservative. Figure 6 illustrates the predicted speed at the beginning of each horizon and the mean speed during the 4-h simulation. It is clear that the difference is very small, but, to estimate the speed more precisely, we define the coefficient factor α for the speed during the loading and unloading. Considering the difference between the current speed and mean speed, we define α equal to 0.95 during loading and 1.05 during unloading in the simulations.

Results

In the proposed algorithm, it is important to define the configuration correctly. The clustering-based heuristic method can speed up the computations and make the algorithm capable of sending the response to the passenger very fast. On the other hand, it narrows the search of feasible space. Therefore, we may roll away from the optimal solution by choosing the wrong value for cluster sizes. The first step is, then, to determine the size of clusters that can ensure a near-optimal solution. For our case study, in the peak hour, the number of requests received every 5 min exceeds 400. We do our simulations with different cluster sizes from 20 to 160 to extract the proper cluster size and find a tradeoff between the algorithm computation time and the cluster size.

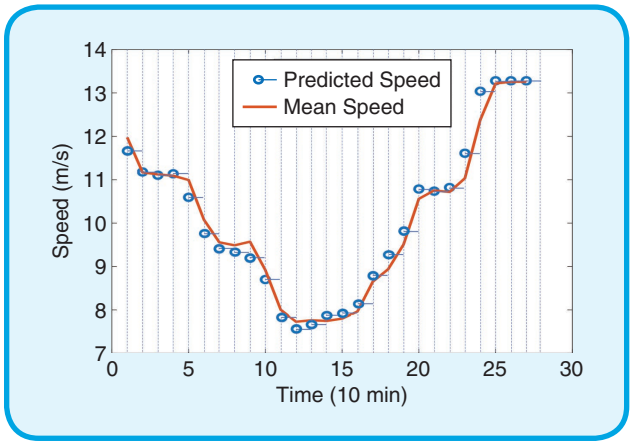

FIG 6 The predicted and mean speeds during the simulation horizon.

Figure 7 presents the simulation time and the objective function (total travel time) for different cluster sizes. The simulation times for up to cluster size 100 are under 100 min. Considering the simulation horizon (4 h for the morning peak hour) with 100 trips inside each cluster, the computation time for each cluster is approximately 35 s. Therefore, with DTaD working on parallel trends, the system can optimize the assignment and provide a response to the passenger in less than 1 min.

The total travel time decreases by increasing the size of clusters because the algorithm can explore a bigger space. However, the difference is mainly marginal, as the difference between the optimal solution for 20 clusters and for 160 is less than 0.1%.

Considering these results, the best tradeoff between the computation time and the objective function happens when the size of the clusters is 100. In the following, we do all of the simulations with a cluster size of 100.

Different clustering methods have been implemented in the literature for large-scale problems. They usually divide the space geographically and use a spatial clustering to downsize the problem. We have compared the sequential clustering with such a spatial clustering method. For spatial clustering, if the distance between two origins is less than a specified value, which we call the *spatial factor*, we put the two corresponding trips in the same cluster. To increase the size of the clusters, we increase the value of the "spatial factor." Also, we try to cluster the trips based on the time in a temporal clustering method. For temporal clustering, we define the "temporal factor," which is related to the starting time of each of the two trips, and then we cluster the trips based on this factor. Finally, we compare these three methods to substantiate the quality of our proposition.

Figure 8(a) depicts the different clustering methods' total travel times for different sizes of clusters. Point zero for the size of the clusters shows the optimal solution for the problem. The spatial clustering method is not comparable to the other two methods. The temporal clustering and sequential clustering methods can provide better results, as they can consider the combination of trips that have the potential to be combined but do not have spatially close origins. For cluster sizes of 20, 40, and 60, the results are almost the same for the temporal and sequential clustering because, with small clusters, sequential clustering cannot explore all of the trips' combination opportunities. With bigger clusters, however, the sequential clustering performance overcomes temporal clustering, and it can significantly provide near-optimal solutions. Figure 8(b) depicts the comparison for the computation time. The temporal clustering is the cheapest method in terms of computation time. Still, for example, for the size of cluster 100, with less than 0.08-s more computation time for each trip, the SF can improve the results by more than 32 min at this scale.

The passenger MWT is a determinant parameter from a passenger's perspective. In the algorithm, we have to define an appropriate value for this parameter. The algorithm minimizes the taxi's travel time while putting a constraint on the passenger's waiting time.

Figure 9 illustrates the results for different MWT values. We simulate five different scenarios for different MWTs from 2 to 10 min. Figure 9(a) demonstrates the total waiting time for passengers for different MWTs. When the maximum waiting time is 2 min, the passenger total waiting time is 3.83 h, and, then, when we increase the MWT to 10 min, the total waiting time is 4.43 h. Each passenger takes the taxi less than 2 s after her or his desired pickup time, on average, in both scenarios.

By increasing the MWT, the algorithm can find more trips in sequence, and after dropping off each passenger, the taxi moves to the next passenger instead of cruising in the network or going back to the waiting locations to wait for new passengers. Thus, the total travel distance for the taxi decreases. Figure 9(b) presents the total travel distance for different MWT values. With MWT equal to 2 min, the total travel distance is 38,437 km, and it reduces to 38,418.3 km for an MWT equal to 10 min. This means the system serves the same requests, saving 18.7 km. We have to set a value for the MWT that is acceptable to and beneficial for both the provider and passengers.

Another critical parameter in DTaD is the number of taxis in waiting locations in the network. Increasing the number of taxis adds flexibility for the algorithm to choose the closest cruising taxi, but, on the other hand, it increases the number of taxis cruising in the network and, consequently, the congestion. Here, we investigate the impact of the number of taxis on the travel time and passengers' waiting times.

We have 237 allowed waiting or stop locations for the taxis in this network. The fleet size is 500, and these 500

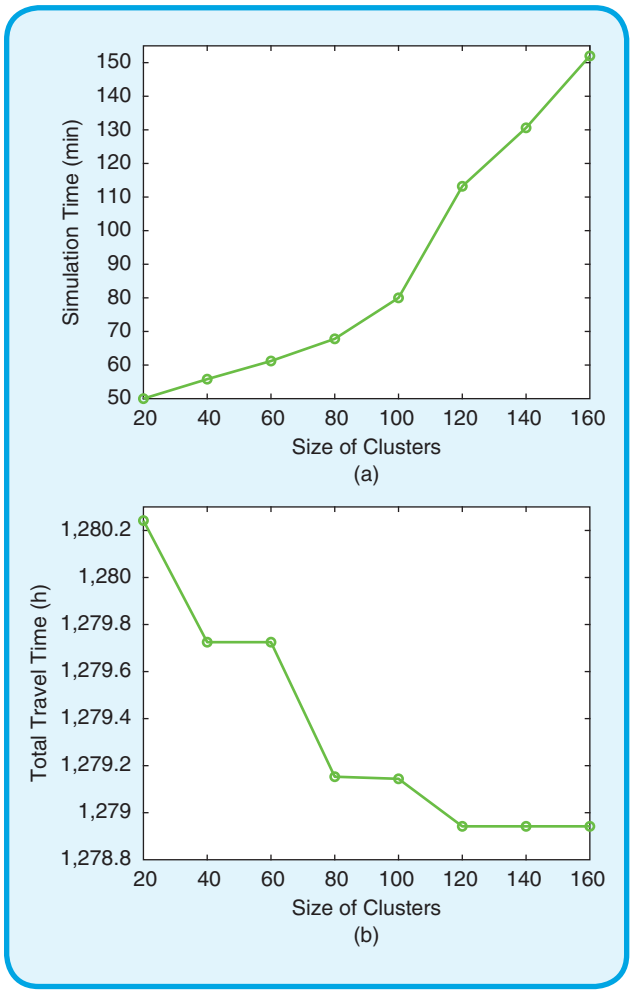

FIG 7 (a) The simulation time for different sizes of clusters. (b) The objective function for different sizes of clusters.

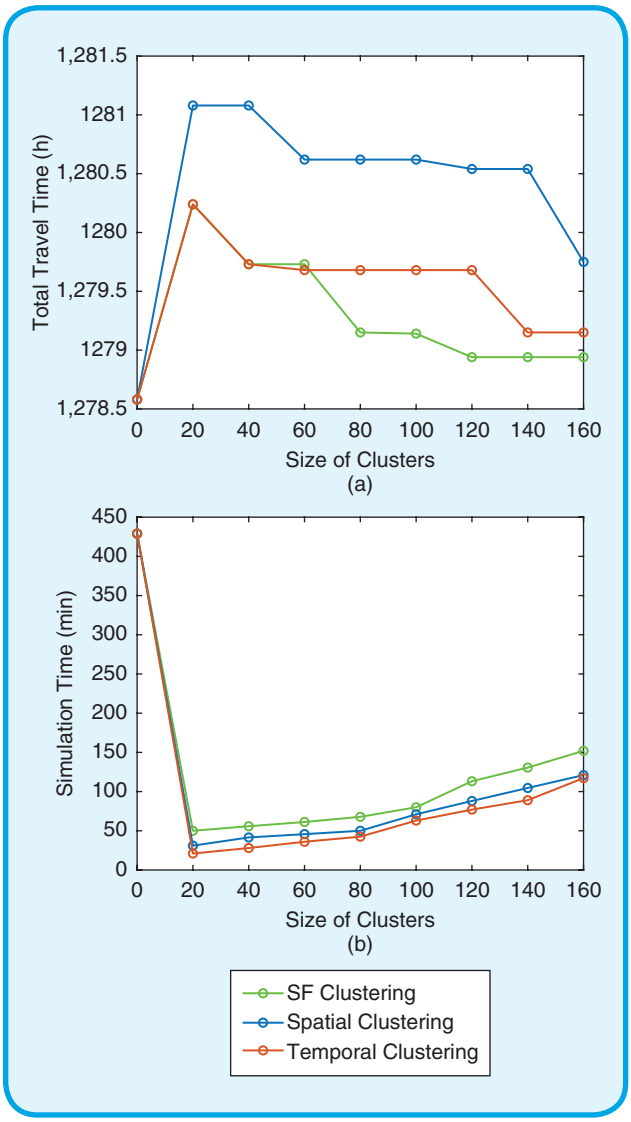

FIG 8 A comparison of (a) the total travel time for different cluster sizes of different clustering methods and (b) the simulation time for different cluster sizes of different clustering methods.

taxis are located in the central depot. On the one hand, distributing taxis over the waiting locations decreases the waiting time to be picked up for passengers. On the other hand, in the peak hour, if many taxis are circulating in the network, the congestion increases, and it leads to more travel time for taxis and passengers. Therefore, we analyze the number of taxis in the waiting locations over the network to decide the best distribution of taxis.

To locate the taxis at the beginning of the simulations, we use the historical data for the network demand to estimate the demand distribution over the network. Then, we specify the number of taxis at each location based on the demand around this waiting location. Therefore, if the demand is high for a stopping point, we consider more cars at that point, and, if the demand is low, we put fewer taxis at that location. Thus, considering the demand distribution, we feed 14 waiting locations with two taxis and 114 locations with one taxi, and we put 14 empty waiting locations at the beginning of the simulations, for a total of 142 taxis. Then, we increase the number of taxis to analyze the waiting time and travel time based on the demand distribution. Therefore, we have different scenarios with 157, 172, and 187 taxis. Figure 9 provides the results for 142, 157, 172, and 187 taxis in 237 waiting locations in the network.

Figure 10(a) illustrates the total waiting time for passengers with different numbers of taxis. When the number of taxis is 142, the total waiting time is 3.97 h, and, then, when we increase the number of taxis to 187, the total waiting time is 3.89 h. It means that, by adding 60 more taxis, we can reduce the passengers' waiting times by up to 5 min.

When the system has access to more taxis, to reduce the travel time between the waiting location and the first origin, it chooses more taxis to serve the trips. Then, in

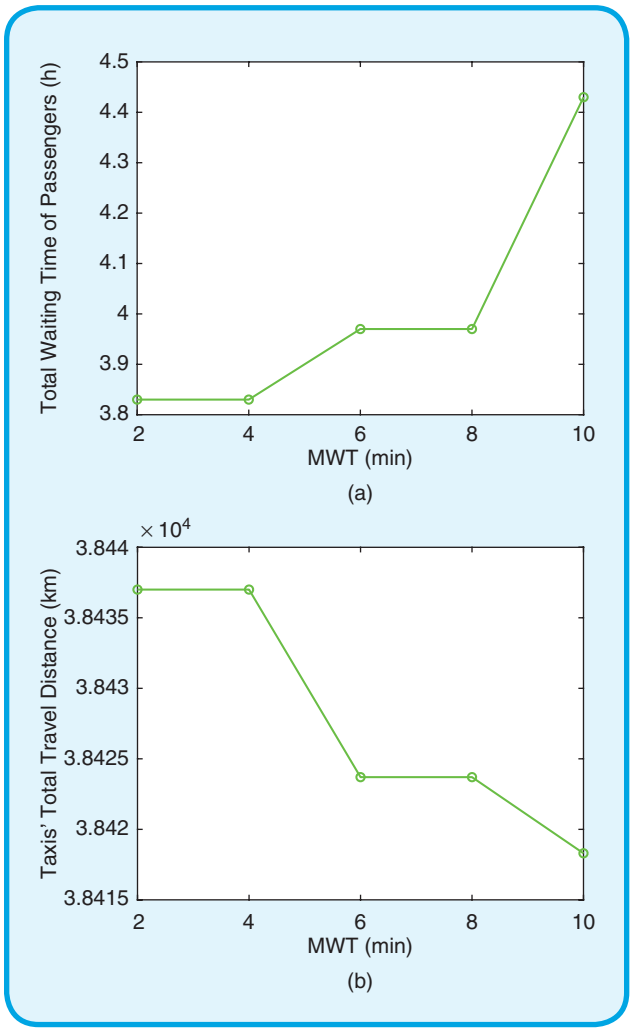

FIG 9 (a) The total waiting time for different MWTs. (b) The total travel distance for different MWTs.

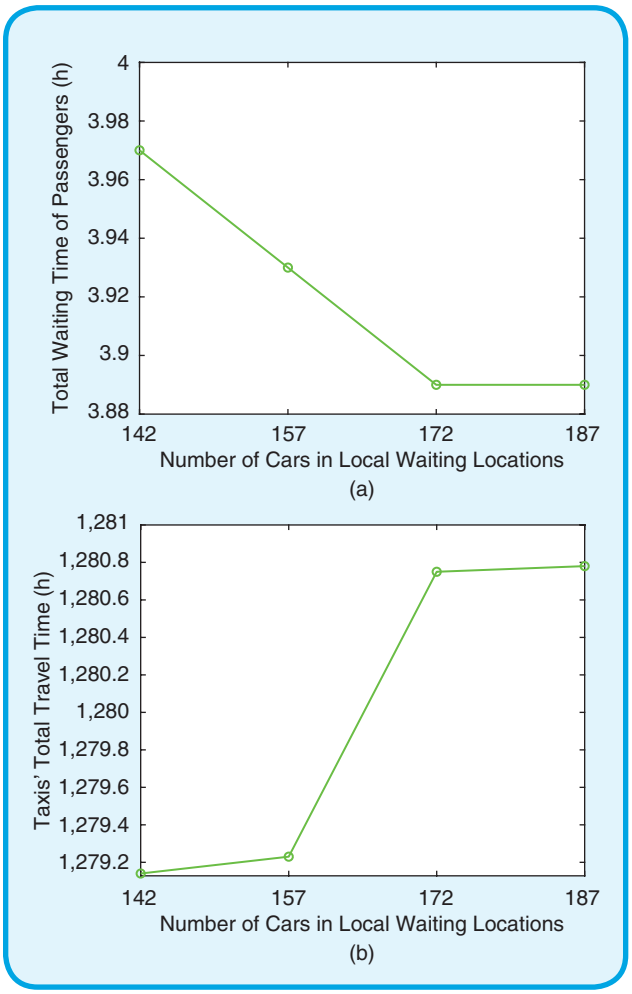

FIG 10 (a) The total waiting time and (b) total travel time for different numbers of taxis.

the peak hour, the number of taxis cruising in the network increases, and the speed decreases for the cars. Therefore, with an increase in the number of taxis, the total travel time increases because of the congestion. Figure 10(b) represents the total travel time for different numbers of taxis. The results indicate that, by having 45 more taxis in waiting locations (187 taxis) at the beginning of the day, the travel time is 1.64 h longer than with fewer cars (142 taxis).

Besides the taxis that are cruising in the network, there are also personal cars circulating in the network. Figure 11 depicts the private cars' travel times for different cluster sizes. It is significant that increasing the cluster size can reduce the travel distance on the taxis. Therefore, the taxis stay less in the network, and the congestion decreases. Hence, the vehicles' speeds increases in the system when the size of clusters increases. The total travel time for personal cars is 7,135.72 h when the size of the clusters is 20, and it decreases to 7,131.03 h for a cluster size of 100 and to 7,131 for cluster sizes of 120, 140, and 160.

Conclusions

In this article, we propose a new taxi-dispatching system called *DTaD*. The proposed DTaD assigns fleet taxis to the trip requests that correspond to a rolling horizon of 10 min. The method investigates all of the feasible assignments considering the maximum threshold on the passenger waiting time and, finally, chooses the optimal route for each taxi that can minimize the taxi's travel time. To speed up the method, we propose a heuristic method based on the clustering to put in the same cluster the trips with greater potential to be served in sequence by the same taxi. Therefore, we introduce the concept of the SF.

The algorithm can minimize the total travel time, total travel distance, and passengers' waiting times. First, we investigate the proper cluster size for DTaD

and, then, assess the impact of the maximum acceptable travel time and the number of taxis in local waiting locations. The results reveal that, with a cluster size of 100, the algorithm works very quickly, and the system can respond to the passenger request in less than 36 s. Also, we demonstrate that a proper decision about the constraint on the passengers' waiting times and the number of taxis in local waiting locations improve the performance of the taxi fleet, and optimal decisions computed by DTaD can be revealed to the providers. We also show that the proposed system can improve the traffic in the network.

In future work, we will investigate the pricing scheme for such a system and apply our method to a larger network with more than a million requests. Also, we are interested in applying our method to taxi-sharing and ride-sharing by changing the concept of the SF to a "shareability function" in our heuristic method.

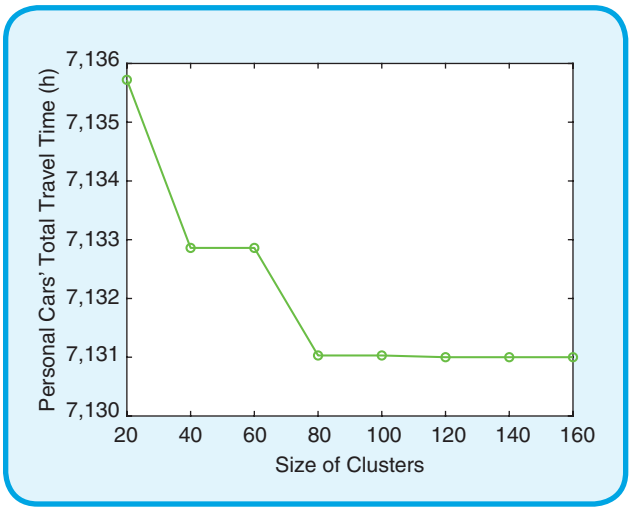

FIG 11 (a) The total travel time of personal cars for different cluster sizes.

Acknowledgment

This project is supported by the European Research Council under the European Union's Horizon 2020 research and innovation program (grant agreement 646592—MAGnUM project).

About the Authors

Negin Alisoltani earned her B.Sc. and M.Sc. degrees in industrial engineering (with distinction) from the University of Tehran, Iran, in 2014 and 2016, respectively, and her M.Sc. in mechanical engineering from Arts et Métiers Paristech, Paris, France, in 2016. She is currently working toward the Ph.D. degree at the Université Gustave Eiffel, GRETTIA, Paris, and Université Gustave Eiffel, ENTPE, LICIT, Lyon, France. Her research interests include modeling of mobility services, operation research, and transportation engineering.

Mahdi Zargayouna earned his M.Sc., Ph.D., and habilitation degrees in computer science and artificial intelligence from University of Paris Dauphine, France, in 2003, 2007, and 2019, respectively. He is a researcher at Université Gustave Eiffel, France, and deputy director of the GRETTIA Laboratory. He has published more than 60 papers in peer-reviewed journals and conference proceedings, and he is member of the reviewer boards of several international journals and conferences. His research interests include multiagent systems (languages, coordination models, simulation, and planning, among others) and complex transportation applications (traveler information, crisis management, dial a ride, and urban parking, among others).

Ludovic Leclercq earned his engineering and master's degrees in civil engineering in 1998, his Ph.D. degree in 2002, and his habilitation thesis (H.D.R.) in 2009. He is a research director (equivalent to a full professor) at Université Gustave Eiffel. He has coauthored 75 publications in top peer-reviewed journals, has supervised 12 Ph.D. candidates, and is currently supervising four Ph.D. students. In 2015, he was awarded the most prestigious research grant in Europe, i.e., a European Research Council Consolidator Grant in Social Science and Humanities. His research interests include multiscale and multimodal dynamic traffic modeling and the related environmental externalities; smart cities, mobility as a service, and sustainable and reliable transportation systems are some applications of his work.

References

[1] N. Alisoltani, L. Leclercq, M. Zargayouna, and J. Krug, "Optimal fleet management for real-time ride-sharing service considering network congestion," in *Proc. Transportation Research Board 98th Annu. Meeting (TRB 2019)*, 2019, p. 22.

[2] M. Ameli, J.-P. Lebacque, and L. Leclercq, "Cross-comparison of convergence algorithms to solve trip-based dynamic traffic assignment problems," *Comp.-Aided Civil Infrastructure Eng.*, vol. 35, no. 3, pp. 219–240, 2020. doi: 10.1111/mice.12524.

[3] J. F. Bard and A. I. Jarrah, "Large-scale constrained clustering for rationalizing pickup and delivery operations," *Transp. Res. B, Methodol.*, vol. 43, no. 5, pp. 542–561, 2009. doi: 10.1016/j.trb.2008.10.003.

[4] H. Billhardt, A. Fernández, S. Ossowski, J. Palanca, and J. Bajo, "Taxi dispatching strategies with compensations," *Expert Syst. Appl.*, vol. 122, pp. 173–182, May 2019. doi: 10.1016/j.eswa.2019.01.001.

[5] J. Bischoff and M. Maciejewski, "Simulation of city-wide replacement of private cars with autonomous taxis in berlin," *Procedia computer Sci.*, vol. 83, pp. 237–244, 2016. doi: 10.1016/j.procs.2016.04.121.

[6] B. Chen and H. H. Cheng, "A review of the applications of agent technology in traffic and transportation systems," *IEEE Trans. Intell. Transp. Syst.*, vol. 11, no. 2, pp. 485–497, 2010. doi: 10.1109/TITS.2010.2048313.

[7] C. E. Cortés, M. Matamala, and C. Contardo, "The pickup and delivery problem with transfers: Formulation and a branch-and-cut solu-

tion method," *Eur. J. Oper. Res.*, vol. 200, no. 3, pp. 711–724, 2010. doi: 10.1016/j.ejor.2009.01.022.

[8] G. Dai, J. Huang, S. M. Wambura, and H. Sun, "A balanced assignment mechanism for online taxi recommendation," in *Proc. 18th IEEE Int. Conf. Mobile Data Management (MDM)*, 2017, pp. 102–111. doi: 10.1109/MDM.2017.23.

[9] R. Darbéra, "Business-models for the taxi of the future," Ph.D. thesis, International Road Union (IRU), 2017.

[10] N. Davis, G. Raina, and K. Jagannathan, "Taxi demand forecasting: A hedge-based tessellation strategy for improved accuracy," *IEEE Trans. Intell. Transp. Syst.*, vol. 19, no. 11, pp. 3686–3697, 2018. doi: 10.1109/TITS.2018.2860925.

[11] M. Felt, N. Gharachorloo, and A. Moshrefi, "Mobile taxi dispatch system," U.S. Patent Appl. 12/607,782, Apr. 28 2011.

[12] N. Ganganath, C.-T. Cheng, and K. T. Chi, "Data clustering with cluster size constraints using a modified k-means algorithm," in *Proc. Int. Conf. Cyber-Enabled Distributed Computing and Knowledge Discovery*, 2014, pp. 158–161.

[13] G. Gao, M. Xiao, and Z. Zhao, "Optimal multi-taxi dispatch for mobile taxi-hailing systems," in *Proc. 45th Int. Conf. Parallel Processing (ICPP)*, 2016, pp. 294–303. doi: 10.1109/ICPP.2016.41.

[14] N. Geroliminis and C. F. Daganzo, "Macroscopic modeling of traffic in cities," in *Proc. Transportation Research Board 86th Annu. Meeting*, 2007.

[15] M. C. Gonzalez, C. A. Hidalgo, and A.-L. Barabasi, "Understanding individual human mobility patterns," *Nature*, vol. 453, no. 7196, pp. 779–782, 2008. doi: 10.1038/nature06958.

[16] Z. Huang, G. Shan, J. Cheng, and J. Sun, "TRec: an efficient recommendation system for hunting passengers with deep neural networks," *Neural Comput. Appl.*, vol. 31, pp. 209–222, Jan. 2019. doi: 10.1007/s00521-018-3728-2.

[17] R.-H. Hwang, Y.-L. Hsueh, and Y.-T. Chen, "An effective taxi recommender system based on a spatio-temporal factor analysis model," *Inf. Sci.*, vol. 314, pp. 28–40, Sept. 2015. doi: 10.1016/j.ins.2015.03.068.

[18] M. Hyland and H. S. Mahmassani, "Dynamic autonomous vehicle fleet operations: Optimization-based strategies to assign AVs to immediate traveler demand requests," *Transp. Res. C, Emerg. Technol.*, vol. 92, pp. 278–297, July 2018. doi: 10.1016/j.trc.2018.05.003.

[19] J. Krug, A. Burianne, and L. Leclercq, "Reconstituting demand patterns of the city of Lyon by using multiple GIS data sources," Tech. Rep., Univ. of Lyon, France, 2017.

[20] Y. Lai, Z. Lv, K.-C. Li, and M. Liao, "Urban traffic Coulomb's law: A new approach for taxi route recommendation," *IEEE Trans. Intell. Transp. Syst.*, vol. 20, no. 8, pp. 3024–3027, 2018. doi: 10.1109/TITS.2018.2870990.

[21] L. Leclercq, A. Sénécat, and G. Mariotte, "Dynamic macroscopic simulation of on-street parking search: A trip-based approach," *Transp. Res. B, Methodol.*, vol. 101, pp. 268–282, July 2017. doi: 10.1016/j.trb.2017.04.004.

[22] Z. Liao, "Taxi dispatching via global positioning systems," *IEEE Trans. Eng. Manag.*, vol. 48, no. 3, pp. 342–347, 2001. doi: 10.1109/17.946533.

[23] Y. Liu, J. Liu, Z. Liao, M. Tang, and J. Chen, "Recommending a personalized sequence of pick-up points," *J. Comput. Sci.*, vol. 28, pp. 382–388, Sept. 2018. doi: 10.1016/j.jocs.2017.05.004.

[24] M. Maciejewski, J. Bischoff, and K. Nagel, "An assignment-based approach to efficient real-time city-scale taxi dispatching," *IEEE Intell. Syst.*, vol. 31, no. 1, pp. 68–77, 2016. doi: 10.1109/MIS.2016.2.

[25] M. Maciejewski and K. Nagel, "Simulation and dynamic optimization of taxi services in matsim," Transport Systems Planning and Transport Telematics, TU Berlin, 2013.

[26] J. MacQueen, "Some methods for classification and analysis of multivariate observations," in *Proc. 5th Berkeley Symp. Mathematical Statistics and Probability*, Oakland, CA, 1967, vol. 1, pp. 281–297.

[27] G. Mariotte and L. Leclercq, "Flow exchanges in multi-reservoir systems with spillbacks," *Transp. Res. B, Methodol.*, vol. 122, pp. 327–349, Apr. 2019. doi: 10.1016/j.trb.2019.02.014.

[28] G. Mariotte, L. Leclercq, and J. A. Laval, "Macroscopic urban dynamics: Analytical and numerical comparisons of existing models," *Transp. Res. B, Methodol.*, vol. 101, pp. 245–267, July 2017. doi: 10.1016/j.trb.2017.04.002.

[29] L. Moreira-Matias, R. Fernandes, J. Gama, M. Ferreira, J. Mendes-Moreira, and L. Damas, "An online recommendation system for the taxi stand choice problem (poster)," in *Proc. IEEE Vehicular Networking Conf. (VNC)*, 2012, pp. 173–180. doi: 10.1109/VNC.2012.6407427.

[30] L. Özdamar and O. Demir, "A hierarchical clustering and routing procedure for large scale disaster relief logistics planning," *Transp. Res. E, Logistics Transp. Rev.*, vol. 48, no. 3, pp. 591–602, 2012. doi: 10.1016/j.tre.2011.11.003.

[31] S. N. Parragh, "Introducing heterogeneous users and vehicles into models and algorithms for the dial-a-ride problem," *Transp. Res. C, Emerg. Technol.*, vol. 19, no. 5, pp. 912–930, 2011. doi: 10.1016/j.trc.2010.06.002.

[32] J. W. Powell, Y. Huang, F. Bastani, and M. Ji, "Towards reducing taxicab cruising time using spatio-temporal profitability maps," in *Proc. Int. Symp. Spatial and Temporal Databases*, 2011, pp. 242–260.

[33] H. Qi and P. Liu, "Mining taxi pick-up hotspots based on spatial clustering," in *Proc. IEEE SmartWorld, Ubiquitous Intelligence & Computing, Advanced & Trusted Computing, Scalable Computing & Communications, Cloud & Big Data Computing, Internet of People and Smart City Innovation (SmartWorld/SCALCOM/UIC/ATC/CBDCom/IOP/SCI)*, 2018, pp. 1711–1717. doi: 10.1109/SmartWorld.2018.00290.

[34] X. Qiang and Y. Shuang-Shuang, "Clustering algorithm for urban taxi carpooling vehicle based on data field energy," *J. Adv. Transp.*, vol. 2018, pp. 1–8, Aug. 2018. doi: 10.1155/2018/3853012.

[35] D. Sáez, C. E. Cortés, and A. Núñez, "Hybrid adaptive predictive control for the multi-vehicle dynamic pick-up and delivery problem based on genetic algorithms and fuzzy clustering," *Comput. Oper. Res.*, vol. 35, no. 11, pp. 3412–3438, 2008. doi: 10.1016/j.cor.2007.01.025.

[36] J. M. Salanova, M. Estrada, G. Aifadopoulou, and E. Mitsakis, "A review of the modeling of taxi services," *Procedia-Soc. Behav. Sci.*, vol. 20, pp. 150–161, 2011. doi: 10.1016/j.sbspro.2011.08.020.

[37] P. Santi, G. Resta, M. Szell, S. Sobolevsky, S. H. Strogatz, and C. Ratti, "Quantifying the benefits of vehicle pooling with shareability networks," *Proc. Nat. Academy Sci.*, vol. 111, no. 37, pp. 13,290–13,294, 2014. doi: 10.1073/pnas.1403657111.

[38] W. Shen and C. Lopes, "Managing autonomous mobility on demand systems for better passenger experience," in *Proc. Int. Conf. Principles and Practice of Multi-Agent Systems*, 2015, pp. 20–35. doi: 10.1007/978-3-319-25524-8_2.

[39] M. M. Vazifeh, P. Santi, G. Resta, S. Strogatz, and C. Ratti, "Addressing the minimum fleet problem in on-demand urban mobility," *Nature*, vol. 557, no. 7706, pp. 534–538, 2018. doi: 10.1038/s41586-018-0095-1.

[40] Q. Wang and K. L. Boyer, "Feature learning by multidimensional scaling and its applications in object recognition," in *Proc. XXVI Conf. Graphics, Patterns and Images*, 2013, pp. 8–15.

[41] S. Wang, L. Li, W. Ma, and X. Chen, "Trajectory analysis for on-demand services: A survey focusing on spatial-temporal demand and supply patterns," *Transp. Res. C, Emerg. Technol.*, vol. 108, pp. 74–99, Nov. 2019. doi: 10.1016/j.trc.2019.09.007.

[42] X. Wang, H. Zhang, L. Wang, and Z. Ning, "A demand-supply oriented taxi recommendation system for vehicular social networks," *IEEE Access*, vol. 6, pp. 41,529–41,538, July 2018. doi: 10.1109/ACCESS.2018.2857002.

[43] X. Xu, J. Zhou, Y. Liu, Z. Xu, and X. Zhao, "Taxi-RS: Taxi-hunting recommendation system based on taxi GPS data," *IEEE Trans. Intell. Transp. Syst.*, vol. 16, no. 4, pp. 1716–1727, 2014. doi: 10.1109/TITS.2014.2371815.

[44] Z. Xu et al., "Large-scale order dispatch in on-demand ride-hailing platforms: A learning and planning approach," in *Proc. 24th ACM SIGKDD Int. Conf. Knowledge Discovery & Data Mining*, 2018, pp. 905–913.

[45] N. J. Yuan, Y. Zheng, L. Zhang, and X. Xie, "T-finder: A recommender system for finding passengers and vacant taxis," *IEEE Trans. Knowl. Data Eng.*, vol. 25, no. 10, pp. 2390–2403, 2012. doi: 10.1109/TKDE.2012.153.

[46] L. Zhang et al., "A taxi order dispatch model based on combinatorial optimization," in *Proc. 23rd ACM SIGKDD Int. Conf. Knowledge Discovery and Data Mining*, 2017, pp. 2151–2159.

ITS