Autonomous Refueling Strategies using Vehicle-to-Infrastructure Communication in Smart Cities

Kameron Carr[1], Jacob Rojas[2], Xiao Chen[2]
[1]Department of Computer Science and Engineering, University of California Los Angeles, Los Angeles, CA 90095
[2] Department of Computer Science, Texas State University, San Marcos, TX 78666
Email: KameronCarrCollege@gmail.com, jar528@txstate.edu, xc10@txstate.edu

***Abstract*—Recently self-driving vehicles have attracted tremendous attention from all walks of life. A problem facing self-driving vehicles is when to stop for gas. In this paper, we study the autonomous refueling strategies using vehicle-to-infrastructure communication in smart cities. We set three goals for our strategies: not to stop too late, nor too early, and get relatively cheap gas. To satisfy these goals, we relate our problem to the Gusein-Zade's version of the secretary problem and provide a solution framework where we divide the distance a vehicle can travel from a full tank to an empty tank into Density Observation Section, Secretary Section, and the Critical Section. To predict the number of gas stations a vehicle will encounter in the future, we use the Constant Density Approximation (CDA) method and the Machine Learning (ML) method. Simulation results comparing the proposed CDA and ML algorithms with the ground truth algorithm show that both perform nearly as well as the ground truth in satisfying all three goals.**

***Index Terms*—secretary problem, self-driving vehicle, smart city, stopping rule, V2I**

I. Introduction

Recently self-driving vehicles have attracted tremendous attention from all walks of life. They are part of the smart transportation in smart cities [6]. In self-driving vehicles, the driver hands over control to the vehicle and is no longer responsible for monitoring the system and handling the situations that may occur on the road. To make self-driving vehicles feasible, a vehicular network is required to let moving vehicles, road side units (RSUs), and pedestrians that carry communication devices communicate with each other. Among these, the communication between the vehicle and RSUs uses vehicle-to-infrastructure (V2I) [11] technologies. The technologies capture vehicle-generated traffic data and wirelessly provide information from the RSUs to the vehicle to inform it of safety, mobility, gas price, or other related information.

In this paper, we will work on a problem essential to self-driving vehicles using V2I that, to our best knowledge, has not been discussed before. It is the *Autonomous Vehicle Refueling* (AVR) problem. When a self-driving vehicle is low on gas, it needs to decide where to stop for gas. We assume that in a smart infrastructure, an RSU is deployed at each gas station to broadcast gas price to the passing vehicles. In each vehicle, an On Board Unit (OBU) is installed to receive gas prices from RSUs. Both RSU and OBU are dedicated short-range communication (DSRC) devices. DSRC works in 5.9 GHz band with bandwidth of 75 MHz and has an approximate range of 300m [2]. The context of our problem is depicted in Fig. 1.

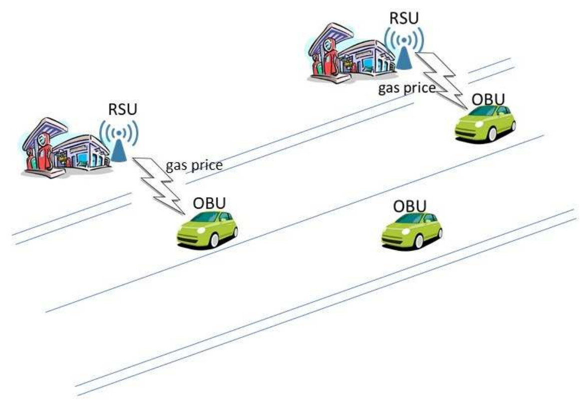


Fig. 1. RSUs at gas stations sending gas prices to OBUs in vehicles

In the AVR problem, we want to achieve three goals ranked by their priorities from high to low. First, we do not want to stop too late when the vehicle is out of gas and the driver is stranded on the road. Second, we do not want to stop too early either with a large amount of gas in the tank. And third, we hope to add gas at a station that offers a relatively low price.

To provide a holistic solution to the AVR problem, we will address the third goal first and then the other two can be dealt with accordingly. We relate our third goal to the Gusein-Zade's version [9] of the classic secretary problem [4], where a stopping rule is present to select one of the r best secretaries out of n rankable applicants arrived in random order with the maximum probability. Our third goal resembles the Gusein-Zade's version because we need a stopping rule to maximize the probability to get a gas price which is among the lowest few. But also at the same time, our third goal is different from the secretary problem in that the number of gas stations k a vehicle will encounter ahead is not known while the number of coming applicants n in the secretary problem is fixed and known. So we need a way to predict the number of gas stations ahead. Our method is to use the gas station information that a vehicle has already collected as it drives on the highway to predict the number of gas stations a vehicle will see in the future. After having the idea for the third goal, we provide a solution framework for the vehicle to decide where to add gas satisfying all three goals. In the framework, we divide the distance a vehicle can travel from a full tank to an empty tank into three sections: Density Observation Section, Secretary Section, and Critical Section. The purpose of the Density Observation Section is to watch the frequency and distribution of gas stations the vehicle passes and predict the number of gas stations k in the future sections. We propose two prediction methods, one is called *Constant Density Approximation* (CDA) and the other is based on a *Machine Learning* (ML) model. Another purpose of the Density Observation Section is to prevent the vehicle from stopping too early. In the Secretary Section, we will use the Gusein-Zade's stopping rule to stop at the gas station with a relatively low price. Finally, the role of the Critical Section is to prevent the vehicle from running out of gas. If the vehicle does not find a gas station in the Secretary Section, then it is critically low on gas and must add gas at the first gas station in the Critical Section. We evaluate the effectiveness of the proposed CDA and ML strategies by comparing them with the ground truth algorithm using simulations. Simulation results show that both perform nearly as well as the ground truth in satisfying all three goals.

The key contributions of our work are as follows:

- We are the first to discuss autonomous vehicle refueling (AVR) problem using V2I to the best of our knowledge.
- We relate our problem to the Gusein-Zade's version of the classic secretary problem and propose a solution framework to solve the AVR problem.
- We demonstrate the effectiveness of our strategies by simulations.

The rest of the paper is organized as follows: Section II references the related work; Section III defines the problem; Section IV describes our solution; Section V presents the simulations; and Section VI is the conclusion.

II. Related Work

The following two problems are related to our defined AVR problem.

A. Classic Secretary Problem

In the classic secretary problem (CSP) [4], a decision maker (DM) wants to hire the best secretary out of n rankable applicants for a position. The applicants are interviewed one by one in a random order. When the DM interviews the jth applicant in the sequence, she gains information sufficient to rank the applicant among all applicants interviewed so far, but is unaware of the quality of yet unseen applicants. Her objective is to find an optimal strategy or the stopping rule to maximize the probability of selecting the best applicant.

The problem has an elegant solution and the optimal search policy is to interview and reject the first $\frac{n}{e}$ applicants and then to accept the first one thereafter with a relative rank of 1 [8]. In $\frac{n}{e}$, e is the base of the natural logarithm, and the optimal policy selects the best applicant with probability $\frac{1}{e} \approx 0.3679$ as $n \to \infty$.

B. Gusein-Zade's Version of the Secretary Problem

In the Gusein-Zade's version of the secretary problem, the goal is to find the optimal stopping rule to select one of the best r candidates out of n applicants, not necessary the best one as in the CSP. This problem was first studied by Gusein-Zade [9] and then further studied by Frank and Samuels [7]. In [7], the authors give the stopping rule and the complete limiting form of the optimal stopping rule for each r up to $r = 10$, and for $r = 15, 20$ and 25. They show that, for large n and r, the optimal risk of not selecting one of the r best is approximately $(1 - t^*)^r$, where $t^* \approx 0.2834$ obtained as the root of a function which is the solution to a certain differential equation. Our problem is similar to this version and we will use their calculated results in the stopping rule to decide when to stop for gas in our solution framework.

III. Problem Statement

In this section, we define the problem we want to solve. We assume in a smart city, a self-driving vehicle is driving on the highway. When it comes close to a gas station, the OBU in the vehicle and the RSU at the gas station can communicate with each other and the gas price at the gas station can be transmitted to the vehicle.

We have the following assumptions:

1) A vehicle can observe sequentially a list of gas prices offered by the gas stations as it drives by on the highway.
2) For each gas station j, the vehicle can only ascertain the *relative rank* of the gas station relative to the previous $j - 1$ viewed gas stations.
3) Once the vehicle passes a gas station, it cannot later come back.

We call our defined problem the Autonomous Vehicle Refueling (AVR) problem, whose objective is to find the best autonomous refueling strategy to satisfy the following three goals with priorities from high to low:

1) A vehicle should not run out of gas
2) A vehicle does not stop for gas too early
3) A vehicle should stop at a gas station with a relatively low price

These three conditions are chosen and given their respective priorities because this is generally how people decide when to stop for gas. People do not want to add gas too late to be stranded on the road. People do not want to add gas too early either with a lot of gas still left in the tank. Furthermore, they want to get a relatively low gas price.

IV. Our Solution

In this section, we provide a solution framework in Fig. 3 to find the gas station for gas. In the framework, there are three parts which correspond to the three sections (shown in Fig. 2) that a vehicle will drive through from a full tank to an empty tank.

1) Density Observation Section
2) Secretary Section
3) Critical Section

Fig. 2. Three sections of the total distance a vehicle can drive

In the framework in Fig. 3, lines 4 through 6 cover the Density Observation Section. In this section, the vehicle will simply watch the frequency and distribution of gas stations it passes. This information will later be used to predict the number of gas stations, k, that the vehicle will pass in the future sections which include the Secretary Section and the Critical Section. The prediction can be done with either the Constant Density Approximation (CDA) (Fig. 4) or the Machine Learning (ML) (Fig. 5) method explained in the next subsections.

The Secretary Section is found on lines 8 through 10. Here, we use the stopping rule stated in subsection IV-C to determine the station for gas. If such a gas station is not found, the vehicle keeps driving.

Finally, if the vehicle does not stop in the Secretary Section, it is now considered to be critically low on gas and will stop at the first gas station in the Critical Section. The Critical Section can be found on lines 12 through 15. The purpose of this section is to prevent the vehicle from running out of gas.

In the next few subsections, we will give the details of the prediction methods of k and the stopping rule.

A. Predicting k using Constant Density Approximation (CDA)

Fig. 4 gives the algorithm of predicting the number of gas stations k in the future sections (including the Secretary Section and the Critical Section) of the highway using Constant Density Approximation. The main premise behind the algorithm can be seen in line 8 of Fig. 4. Since we do not know the distribution of the gas stations, a reasonable assumption is that a gas station can appear at any location along the highway with the same probability. Then the density of the gas stations in the future sections is the same as the density in the Density Observation Section. So the number of gas stations we will see in the future sections is equal to the number of gas stations in the Density Observation Section times the total length of the future sections (Secretary Section and Critical Section) divided by the length of the Density Observation Section.

B. Predicting k using Machine Learning (ML)

The general premise behind the Machine Learning k-prediction method shown in Fig. 5 is to divide the Density Observation Section into ten segments and whose gas station numbers will be used as inputs for the machine learning model. We decide to divide the Density Observation Section into segments so that the machine learning model will be able to account for the distribution of the gas stations along the highway. The 11th and last input into the model is the total distance the vehicle can drive. To improve the accuracy of the model, we standardize the inputs that we train on. For this reason, the average and standard deviation of the test data must be included as a part of the given model. The inputs are then

Solution Framework to the AVR Problem

1: **Inputs:** the total mileage $Total\_Mileage$ a vehicle can drive from a full tank until empty, the distance driven so far $Driven\_Mileage$, the length of the density observation section $Observation\_Length$, the length of the Secretary Section $Secretary\_Length$
2: **Output:** the gas station j where the vehicle stops for gas
3: // **Density Observation Section**
4: **while** $Driven\_Mileage < Observation\_Length$ **do**
5: Keep driving and observe the number of the gas stations along the way to predict the number of gas stations k in the future sections which include the Secretary and the Critical Sections. Value k will be predicted using either the algorithm in Fig. 4 or Fig. 5
6: **end while**
7: // **Secretary Section**
8: **while** $Driven\_Mileage < (Secretary\_Length + Observation\_Length)$ **do**
9: Keep driving and for every passing gas station, use the stopping rule stated in subsection IV-C to find a gas station j for gas. If j exists, stop and add gas; if not, keep driving.
10: **end while**
11: // **Critical Section**
12: **while** $Driven\_Mileage < Total\_Mileage$ **do**
13: Keep driving until you hit a gas station j
14: Stop and add gas at station j
15: **end while**
16: If you reach this point without finding a gas station j, then you have run out of gas

Fig. 3. Our solution framework to the AVR problem.

Constant Density Approximation (CDA) Prediction

1: **Inputs:** the distance driven so far $Driven\_Mileage$, the length of the Density Observation Section $Observation\_Length$, the length of the Secretary Section $Secretary\_Length$, and the length of the Critical Section $Critical\_Length.$
2: **Output:** a number k that is an approximation of the number of gas stations in the Secretary Section and the Critical Section
3: $gas\_stations\_passed \leftarrow 0$
4: **while** $Driven\_Mileage < Observation\_Length$ **do**
5: Drive to the next gas station
6: $gas\_stations\_passed \leftarrow gas\_stations\_passed + 1$
7: **end while**
8: $k \leftarrow (gas\_stations\_passed) * (Secretary\_Length + Critical\_Length) / (Observation\_Length)$
9: **return** k

Fig. 4. Predicting k gas stations in the future sections of the highway by assuming a constant density of gas stations on the highway.

Machine Learning (ML) Prediction

Inputs: the total mileage $Total\_Mileage$ a vehicle can drive from a full tank until empty, the distance driven so far $Driven\_Mileage$, the length of the Density Observation Section $Observation\_Length$, the machine learning model given by: A a 1x11 vector, b a scalar bias, Avg a 1x11 vector of input averages, and Dev a 1x11 vector of input standard deviations
Output: a number k that is an approximation of the number of gas stations in the Secretary Section and the Critical Section
$segment\_length \leftarrow Observation\_Length/10$
for segment i = 1..10 in $Observation\_Length$ **do**
  $gas\_stations[i] \leftarrow$ number of gas stations passed in segment i
end for
$Inputs \leftarrow [gas\_stations[1], \ldots, gas\_stations[10], Total\_Mileage]$
for i=1..11 **do**
  $Std\_Inputs[i] \leftarrow (Inputs[i] - Avg[i])/Dev[i]$
end for
$k \leftarrow A * Std\_Inputs' + b$
return k

Fig. 5. Predicting k gas stations in the future sections of the highway by machine learning method

r	j	$t_j(r)$
1	1	0.3679
...		
3	1	0.3367
	2	0.5868
	3	0.7746

TABLE I
SAMPLE ASYMPTOTIC FORM OF THE OPTIMAL STOPPING RULES

adjusted by the same average and standard deviation as the training data to produce the prediction for k.

The weights in the model, A and b, are found by training the model using TensorFlow [5]. The general procedure when training the model is to initially set these weights to random values and then slowly modify them to improve the accuracy of the prediction. When training the model used in our simulations, we use Elastic Net Regression and the loss function described by McClure [10].

C. Stopping Rule

In their paper [7] on the Gusein-Zade's version of the secretary problem, Frank and Samuels stated the stopping rule to get one of the best r individuals as "select arrival of relative rank j only after $t_j(r) \cdot n$ previous arrivals". The authors also calculated the values of $t_j(r)$ for different values of r and j after solving a differential equation. We therefore can directly use the $t_j(r)$ values from their paper. Here, Table I shows some sample values and we will use them to interpret the meaning of the stopping rule.

According to Table I, if $r = 1$, this is the classic secretary problem and the special case of the Gusein-Zade's version. In this case, you only select the best individual. So the relative rank j can only be 1. The stopping rule is to pass the first $0.3679 \cdot n$ candidates and then pick the next candidate (with a relative rank 1) who is better than all the previous ones. If $r = 3$, the relative rank j can be $1, 2$, and 3. Then the stopping rule is: after you pass the first $0.3367 \cdot n$ candidates, you can pick the next candidate with a relative rank $j = 1$. If you do not get anyone, you can pick the next candidate with a relative rank $j \leq 2$ after you pass the first $0.5868 \cdot n$ candidates. And if you still do not get one, you can pick the next candidate with a relative rank $j \leq 3$ after you pass the first $0.7746 \cdot n$ candidates. Now we can see from the stopping rule that, as you continue searching, you will consider people with a lower relative rank to increase the chance of getting one of the r best candidates.

V. SIMULATIONS

In this section, we evaluate the effectiveness of our algorithms by comparing them with the ground truth algorithm using a customized simulator written in MatLab and Python.

A. Data Collection

Since there is no existing data for our needs to the best of our knowledge, we explored several methods of retrieving real-life gas station data. In our collection, the locations and rates of gas stations which are less than 300 meters away from the path a vehicle would travel were recorded. First, manual collection was done by using GasBuddy [3], a website which displays gas station locations and prices based on location or along a given route. We used GasBuddy's 'Plan Your Trip' option with 'mile per gallon' setting to 0.5 miles a gallon tank on a one way trip. This effectively captures gas station information every 0.5 miles along a route. We selected routes in between 300 and 500 miles to represent how far a vehicle could drive on a full tank of gas. After setting these configurations, a route is calculated and station information is displayed on the screen. We used a Python script which utilizes Beautiful Soup [1], a library for web scraping, to pull gas station rates and locations. With these data, we obtained 151 routes across the United States to test our refueling algorithms. These routes were captured as matrices in Matlab with each index representing a unit distance and the element in each index being the price of gas at that location. A value of zero was used if no gas station was present. It should be noted that GasBuddy routes do no perfectly capture the frequency of gas stations along every route. We found that certain locations along our generated routes would show no stations when in reality there were plenty. Because we found no suitable alternative, we accept that our routes are more conservative than reality.

B. Algorithms Compared

We compared the following algorithms.

1) Constant Density Approximation (CDA): where k is predicted by assuming the density of the gas stations to be nearly a constant.
2) Machine Learning (ML): where k is predicted based on Machine Learning.

3) Ground Truth (GT): where the actual number of gas stations in the future sections is known.

In the above algorithms, the CDA and ML methods use the same framework in Fig. 3. The GT method also follows the same framework except that it replaces the Critical Section part with the last gas station. This is because we know from the ground truth where the last gas station is. So if the stopping rule does not return any gas station for the vehicle to stop earlier, we will stop to add gas at the last gas station in GT.

C. Metrics

The quality of the comparing algorithms is evaluated by the following three metrics that correspond to our three goals.

1) The probability of running out of gas
2) The percentage of the gas left in the tank when refueling
3) The probability of getting cheap gas

Here, a cheap gas price is defined as the price which is less than 80% of all the gas prices along the route.

D. Setting

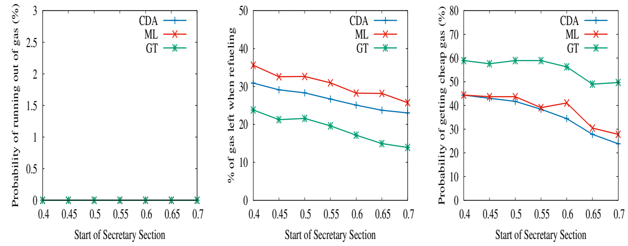
In our simulations, we set the start of the Secretary Section from 0.4 to 0.70 of the total distance a vehicle could travel from a full tank to an empty tank. We set the start of the Critical Section to be 75% and 82% of the whole route. If the start of the Critical Section exceeds 82%, we will start to see vehicle running out of gas based on our data. So we do not want to go beyond that. Using the 151 routes collected from GasBuddy.com, we calculated the average of the three metrics.

E. Results

The simulation results are presented in Figs. 6 and 7. We can see that both CDA and ML performed nearly as well as GT by all three metrics. More specifically, setting the correct starting point of the Critical Section is crucial to avoid the risk of running out of gas. In our data, setting the starting point to be no more than 82% of the whole route allows all three algorithms to have a zero chance of being out of gas. The GT method has the least amount of gas left in the tank when the vehicle stops for gas and the highest probability of getting cheap gas. The ML method has more gas left in the tank than the CDA method, but has a little higher chance of getting a lower gas price. In terms of the cost effectiveness, we would say that the CDA method is better because of its simplicity and good performance.

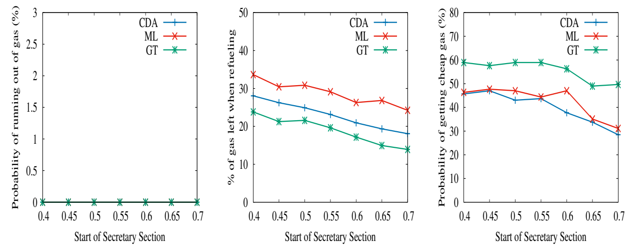
VI. Conclusion

In this paper, we have discussed the autonomous vehicle refueling strategies using vehicle-to-infrastructure communication in smart cities. We have set three goals for our strategies: not to stop too late nor too early, and get relatively cheap gas. To satisfy the goals, we have related our problem to the Gusein-Zade's version of the secretary problem and provided a solution framework where we have divided the route a vehicle can travel from a full tank to an empty tank into Density Observation Section, Secretary Section, and the Critical Section. To predict the number of gas stations a vehicle would encounter in the future, we have used the Constant Density Approximation (CDA) method and the Machine Learning (ML) method. Simulation results comparing the proposed algorithms with the ground truth algorithm have shown that both CDA and ML have performed nearly as well as the ground truth in satisfying all three goals. In this paper, the advantage of the ML method is not obvious comparing with the simple CDA method. One direction we can explore in the future is to obtain more real-life data to train the ML model.



(a) Probability of running out of gas (%) (b) % of gas left when refueling (c) Probability of getting cheap gas (%)

Fig. 6. Comparison of proposed strategies against ground truth, with the Critical Section starting from 75% of the whole route.



(a) Probability of running out of gas (%) (b) % of gas left when refueling (c) Probability of getting cheap gas (%)

Fig. 7. Comparison of proposed strategies against ground truth, with the Critical Section starting from 82% of the whole route.

Acknowledgements

This research was supported in part by NSF REU site grant 1757893.

References

[1] Beautiful Soup (HTML parser). https://en.wikipedia.org/wiki/Beautiful_Soup_(HTML_parser).
[2] Dedicated Short Range Communications (DSRC) Home. https://web.archive.org/web/20121119140100/http://www.leearmstrong.com/Dsrc/DSRCHomeset.htm.
[3] GasBuddy. https://www.gasbuddy.com/.
[4] Secretary problem. https://en.wikipedia.org/wiki/Secretary_problem.
[5] TensorFlow. https://www.tensorflow.org/.
[6] B. Bowerman, J. Braverman, and J. Taylor et al. The vision of a smart city. *2nd international life*, 2000.
[7] A. Q. Frank and S. M. Samuels. On An Optimal Stopping Problem of Gusein-Zade. *Stochastic Processes and their Applications*, 10:299–311, 1980.
[8] J. P. Gilbert and F. Mosteller. Recognizing the Maximum of a Sequence. *Journal of the American Statistical Association*, 61(313):35–73, 1966.
[9] S. M. Gusein-Zade. The problem of choice and the optimal stopping rule for a sequence of independent trials. *Theory of Probability & Its Applications*, 11(3):472–476, 1966.
[10] N. McClure. *TensorFlow Machine Learning Cookbook.* Packt Publishing, 2017.
[11] Y.-L. Tseng. LTE-advanced enhancement for vehicular communication. *IEEE Wireless Commun.*, 22(6):4–7, 2015.