Foggy-Park: A Dynamic Pricing and NSGA based Allocation Scheme for On-Street Parking System

Sandeep Saharan
AI Research Centre, Woxsen University
Hyderabad, Telagana, India
sandeepsaharan@outlook.com

Seema Bawa
Thapar Institute of Engineering and Technology
Patiala, Punjab, India
seema@thapar.edu

Neeraj Kumar
Thapar Institute of Engineering and Technology
Patiala, Punjab, India
neeraj.kumar@thapar.edu

Rajkumar Buyya
CLOUDS Lab, University of Melbourne
Melbourne, Australia
rbuyya@unimelb.edu.au

Abstract

In past decade parking and problems associated with it have attracted the researchers attention towards it. Some of the well known problems associated in the path of making parking smart are optimal parking resources usage, guaranteed parking reservation, identification of available parking slots, efficient communication protocols. This paper proposes a scheme, namely, *Foggy-Park* which deals with dynamic pricing and allocation aspects of the smart on-street parking system. While allocating the available parking slots, Non-dominated Sorting Genetic Algorithm (NSGA) is used to address the interests of both the parkers and parking authorities. The parkers always desire to pay less parking fees. Whereas, the parking authorities want to generate high revenue by renting out parking slots. In order to compute dynamic prices for the available parking slots, Seattle city parking and its prices data-sets are used. The former one is used to train random forest model which is used predict occupancy. Whereas, the later one is used to form base prices. *Foggy-Park* scheme is implemented on different computing paradigms, such as, cloud, fog, and edge using the concept of Zero Trust Network Access (ZTNA). The scheme implemented on fog computing paradigms shows its worth over others in terms of less communication overhead. The obtained results prove that the proposed *Foggy-Park* scheme minimizes the average parking prices, maximizes the generated revenue, maximizes the accepted requests, and maximizes the occupancy fairness by around 4%, 23%, 6%, and 11.28% respectively.

CCS Concepts

• **Networks** → **Network architectures**; *Network performance analysis*; • **Computing methodologies** → **Modeling and simulation**.

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for components of this work owned by others than the author(s) must be honored. Abstracting with credit is permitted. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee. Request permissions from permissions@acm.org.

ZTA-NextGen '24, August 4–8, 2024, Sydney, NSW, Australia

© 2024 Copyright held by the owner/author(s). Publication rights licensed to ACM.

ACM ISBN 979-8-4007-0715-5/24/08
https://doi.org/10.1145/3672200.3673873

Keywords

Dynamic Pricing, Parking Pricing, Machine Learning, NSGA, Resource Allocation, Cloud Computing, Fog Computing, Edge Computing, Intelligent Transportation System, Smart Cities.

ACM Reference Format:
Sandeep Saharan, Seema Bawa, Neeraj Kumar, and Rajkumar Buyya. 2024. *Foggy-Park:* A Dynamic Pricing and NSGA based Allocation Scheme for On-Street Parking System. In *SIGCOMM Workshop on Zero Trust Architecture for Next Generation Communications (ZTA-NextGen '24), August 4–8, 2024, Sydney, NSW, Australia.* ACM, New York, NY, USA, 6 pages. https://doi.org/10.1145/3672200.3673873

1 Introduction

Zero Trust Smart Parking (ZTSP), which employs effective algorithms, protocols, and cutting-edge hardware and software to meet the demands of drivers and parking authority, is urgently needed where no one trust anyone. Various issues with conventional parking systems are eliminated by the smart parking systems. These issues include load balancing, traffic congestion brought on by cruising while looking for parking, pollution brought on by cruising, effective pricing (less for drivers and more for parking authorities), optimal use of parking resources, automatic identification of available parking spaces, effective communication between various parking system entities, and others. The aforementioned problems have thus far been addressed using a variety of methods. One popular method effectively addresses the majority of these issues. That typical method is Dynamic Pricing (DP) [11]. Change in the cost or pricing of products or services, respectively, is referred to as DP [3]. The majority of people are motivated by money. Thus, DP can be a highly promising solution to the majority of issues that arise in the path to smart parking systems provided the system learns to modify rates correctly. So far, DP has been used in the Intelligent Transportation System (ITS) in a variety of locations, including fair pricing, electric vehicle (EV) charging and discharging pricing, parking pricing, congestion pricing, and more [11]. Various allocation procedures have been employed up to this point in order to better distribute the available parking spaces. One of them is the use of machine learning techniques. Machine learning is mostly used to forecast how full various parking lots will be and to provide directions to drivers so they can choose where to park. To determine the projected occupancy or cost of the available parking spaces, several machine learning algorithms are used. These models include

deep neural networks, Long Short Term Memory (LSTM), neural networks, decision trees, random forests, and linear regression [14] [12] [13]. To assign parking places to the drivers, predicted parking occupancy driven fees can also be employed. On a parking data set of the Seattle city, this allocation is examined [13]. Despite its intended purpose, predictive parking allocation can occasionally make matters worse. This is mostly because such solutions only take past data into account and ignore actual circumstances. Potential remedies include adaptive pricing and virtual voting, the former of which can increase parking authority' revenue and the latter of which, when combined with the hash-graph consensus technique, can aid in the real distribution of parking spaces [6].
Other methods, including game theory, optimization, auction theory, and mixed integer linear programming (MILP), are also used to better distribute the parking spaces that are available. A multi-agent auction that takes into account the preferences of both drivers and parking authority can be used to distribute parking spaces [10]. Such a system may look for available parking spaces, negotiate rates with both parties, and allow drivers to reserve a space. In most of the cases, the private parking spaces are inefficiently managed by their owners and can be managed efficiently by renting them out to the public authorities. Public authorities can use cloud platform to present every information for the use of public. Vickrey-Clarke-Groves auction can be used to rent out such private parking spaces [7]. The same private parking sharing problem can also be modeled as social welfare maximization problem. By using mixed integer non-linear programming, the distance between the allotted parking space and the parker's destination may be reduced for the winners of the auction for these private parking spaces [4]. The usage of parked cars is one way to get around the Road Side Units' intrinsic constraint in the vehicular networks, which is that they cannot be employed extensively. Energy efficiency may be one of the problems in such a circumstance that may be resolved by creating an appropriate optimization problem. Reverse auction theory may be used to choose the best vehicle for the RSU, and then nonlinear fractional programming can be used to solve the joint resource allocation problem and maximize the transmission power [9]. Parkers require assured parking reservations at the most affordable pricing in the present smart parking system. Parkers want to spend less time looking for open parking spaces. The parking authority, on the other hand, desire increased income and the best possible use of their resources. These objectives can be created and resolved using mixed-integer linear programming [8]. The best distribution of parking spaces may be represented as a driver's cost function that includes pricing and distance to destination as variables [5]. MILP may be used to solve this sort of model. Today, the relationship between parking lots and EVs' charging and discharging capabilities is crucial. This connection aids in determining how customers park and charge or discharge their electric vehicles, which has an impact on parking authorities' planning [15]. The smart parking system in the context of the vehicular fog computing environment may also benefit from the help of parked cars, and in exchange for this support, the parked cars receive incentives [16]. These services are referred to as delay-sensitive computer support. The use of sensors, communication protocols, and hardware-software interfaces are examples of existing technologies that may have drawbacks and require modification when newer technology develops. However, the DP is such a strong solution that may needs an update due to inclusion of newer parameters otherwise it is robust [3].

1.1 Motivation

Compared to all other types of parking facilities available across the world, on-street parking has a particular significance. It offers several benefits, such as being often utilized for shorter periods of time and being close to the user's destination. On the other side, it has a lot of drawbacks as well when managed ineffectively. These include increased traffic cruising for parking, increased pollution from vehicles cruising, and heavy traffic during peak hours. As a result, the administration of the on-street parking system needs a special consideration. In addition, although parking service providers want to make a lot of money by renting out available parking spaces, parkers always prefer to pay less. Most people quickly check for open spots in any on-street parking lot. Therefore, the booking procedure must move quickly enough. The cited literature does not adequately and comprehensively address the aforementioned issues.

1.2 Contributions

The main contributions of this work are as follows.

(i) This study develops a dynamic distributed pricing mechanism that takes into account both actual and anticipated parking lot occupancy. The random forest approach is used to forecast occupancy.
(ii) A Non-dominated Sorting Genetic Algorithm (NSGA) is utilised to distribute the available parking places to the incoming requests. Both parking users and authorities are taken care of by this system. It gives parking customers a slot so they may pay less. On the other side, it guarantees the parking authority greater profit.
(iii) In order to verify communication overhead, the developed scheme "*Foggy-Park*" is implemented in a variety of computing environments, including cloud, fog, and edge. This paper formulates three distinct variations of fog computing environments.
(iv) Design, simulation, and testing of efficient algorithms are done in the parking environment of Seattle. In order to calculate the dynamically distributed prices for the parking lots, Seattle city prices data [2] is used as a base price and Seattle city parking data [1] is used to predict occupancy. Because of this scheme, parking authority will make more money while charging less from the parkers. The effectiveness of the implemented scheme is evaluated in comparison to other cutting-edge schemes.

1.3 Organization

The rest of the paper is organized as follows. Section 2 defines the system model and presents the problem solved in this work. Then, Section 3 presents the proposed *Foggy-Park* scheme. Thereafter, Section 4 evaluates the performance of the proposed *Foggy-Park* scheme. Finally, Section 5 concludes this paper.

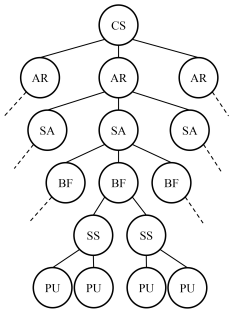

Figure 1: Seattle city multifurcation tree

2 System Model and Problem Formulation

The multi-furcation of the Seattle is given in Fig. 1. Let the single central server is located at topmost position in the hierarchy. One area have multiple sub-areas, one sub-area have many block-faces and similarly one block-face may have one or many side of the streets. The parking slots are present on the side of the streets. The slots located at one side of the street belongs to one parking lot. The users send their requests using internet applications which are then captured by the nearest receiver located on the side of streets only after validation of parking user. Then, the requests are forwarded to the nearest linked computing device for its processing after due validation as per the concept of ZTA. Thereafter, the decision on requests are notified back to the users by validating the identities of all the devices in same way.

Let the set of elements is denoted by using uppercase letter. Whereas, the elements are depicted using same small case letters. The subset of any set is denoted same uppercase letter followed by a digit. Let AR, SA, BF, SS, PU, PL, PS, TS, ID, NN, RR are the set of areas, sub-areas, block-faces, side of streets, parking users, parking lots, slots, timestamps, unique IDs, natural, and real numbers respectively. Let 'D' denotes the set of Computing Devices (CD) present at each sub-area of the city and 'Q^d' is the ready/waiting queue of arrived request at CD 'd'. Let RQ^d depicts the details of requests arrived at CD 'd'. Each 'rq^d' is of type (id, ts, pu, pl, ts, nn) where 'id' is the unique id given to the request. 'ts' is the request arrival timestamp. 'pu' is the parking user of the request. 'pl' is the lot where slot is requested. 'ts' is the vehicle arrival timestamp. and 'nn' is the requested parking duration in seconds. Let RA^d depicts the details of allocation done at CD 'd'. Each 'ra^d' is of type (id, ts, nn, pl, ps, rr) where 'id' is the unique id of the request. 'ts' is the allocation timestamp. 'nn' depicts request decision. Its value is 0, 1 or 2 if the request is undecided, rejected and accepted respectively. 'pl' and 'ps' are the allocated parking lot and slot respectively. 'rr' is the parking prices charged per hour. Let the functions be denoted by 'Fn'. $Fn_1(pl) \rightarrow (ar, sa, bf, ss)$ depicts the location of the given parking lot. $Fn_2(sa) \rightarrow PL1$ depicts the subset of parking lots that belongs to the given sub-area. $Fn_3(pl, ts_1, ts_2) \rightarrow (nn_1, nn_2)$ denotes the maximum of real and predicted occupied slots of parking lot between two given timestamps. $Fn_4(pl, ts) \rightarrow rr$ depicts the prices that are charged at present by the Seattle Transport Department (STD)[2]. $Fn_5(pl) \rightarrow PS1$ depicts the subset of parking slots that belongs to the given parking lot. $Fn_6(d) \rightarrow RQ^d$ depicts the set of requests received at CD 'd'. $Fn_7(d) \rightarrow RA^d$ depicts the set of allocation done at CD 'd'. $Fn_8(ps) \rightarrow pl$ depicts the parking lot of given slot. $Fn_9(ps, ts_1, ts_2) \rightarrow bb$ depicts whether given slot is occupied (if 'bb' = 1) between given timestamps or not.

2.1 Dynamic Pricing Model

This model computes dynamic prices (p^d) to be charged for the slot in given parking lot 'pl' at time 'ts_1'. The formula to compute 'p^d' is given in Eq. 3.

2.2 Average Parking Prices Calculation

The average parking prices charged at each CD 'd' is calculated using Eq. 1 when Eq. 2 is satisfied.

$$p^a = \frac{\sum_{i=1}^{|Fn_7(d)|}((Fn_7(d))_i)_6}{\sum_{i=1}^{|Fn_7(d)|} 1} \tag{1}$$

$$((Fn_7(d))_i)_2 \in [ts_j, ts_k] \wedge ((Fn_7(d))_i)_3 = 1 \tag{2}$$

2.3 Generated Revenue Calculation

The revenue generated at each CD 'd' is calculated using Eq. 4.

$$p^d = Fn_4(pl, ts_1)\Bigg(1 + \frac{1}{\mid Fn_5(pl) \mid} * (((Fn_3(pl, ts_1, ts_2))_1 > (Fn_3(pl, ts_1, ts_2))_2)?(Fn_3(pl, ts_1, ts_2))_1 : (Fn_3(pl, ts_1, ts_2))_2) + 1 - \Big(\frac{1 + \mid Fn_5(pl) \mid}{2}\Big)\Bigg) \tag{3}$$

$$R^t = \sum_{i=1}^{|Fn_7(d)|} ((Fn_7(d))_i)_6 \times \left\lceil \frac{((Fn_6(d))_n)_6}{3600} \right\rceil \mid ((Fn_7(d))_i)_2 \in [ts_j, ts_k] \wedge ((Fn_7(d))_i)_1 = ((Fn_6(d))_n)_1 \wedge ((Fn_7(d))_i)_3 = 1 \wedge n \in [1, \mid Fn_6(d) \mid] \tag{4}$$

2.4 Average Occupancy Calculation

The average occupancy 'O^{avg}' at all parking lots between two timestamps is calculated using Eq. 5.

$$O^{avg} = \Big(\sum_{ts=ts_j}^{ts_k} \sum_{i=1}^{|PL|} \frac{Fn_3(pl_i, ts, ts)}{\mid Fn_5(pl_i) \mid}\Big) \times 100 \tag{5}$$

2.5 Problem Formulation

The objective functions that are solved in this work are given in Eqs. 6 and 7.

$$Minimize\ P^a \tag{6}$$

$$Maximize\ R^t \tag{7}$$

subject to constraints:

$$C1 : ((Fn_7(d))_i)_3 = 2 \tag{8}$$

$$C2 : i \in [1, \mid Fn_7(d) \mid], \quad d \in [(D)_1, (D)_{|D|}] \tag{9}$$

$$C3 : ts_j = ts_k \in [1, (TS)_{|TS|}] \tag{10}$$

The constraint 'C1' depicts the request must be accepted in order to reduce prices and generate revenue out of it. The constraint 'C2' defines the boundaries of the variable 'i' and 'd'. Whereas, the constraint 'C3' defines the boundaries of various timestamps used in order to computes values of the objectives given in Eqs. 6 and 7.

3 *Foggy-Park:* The Proposed Scheme

This section provides the details of the proposed scheme, i.e., '*Foggy-Park*'.

3.1 Solution Representation and Initial Population

Each sub-area have its own 'CD' that run the *Foggy-Park* scheme on the requests it have in its waiting queue denoted by $\mathtt{Q_d}$. The front and rear pointers of the queue '$\mathtt{Q_d}$' are denoted using '$\mathtt{F}^{\mathtt{Q_d}}$' and '$\mathtt{R}^{\mathtt{Q_d}}$' respectively. Let the initial population of random solutions be denoted using $\mathtt{S_d}$, where individual single solution is represented using $\mathtt{s_d}$. One potential 'ps' is a random solution for each 'id' present in the '$\mathtt{Q_d}$'. Thus, random '$\mathtt{s_d}$' is a collection of 'ps'(s) where $|\ \mathtt{s_d}\ |$ is equal to $|\ \mathtt{Q_d}\ |$. Further, the '*ps*' must not be already allocated for the requested period of time and should belongs to the sub-area of requested parking lot. In a single solution two '*ps*'(s) should not be same. Further, definition of the feasible solution is provided in Eq. 11.

$$\begin{aligned} S_d = \{s_d \mid (s_d)_i = ps_j || -1 \wedge i \in [1, |\ s_d\ |] \wedge j \in [1, |\ PS\ |] \wedge \\ |\ s_d\ | = |\ Q_d\ | \wedge Fn_8((s_d)_i) \in Fn_2((Fn_1((Fn_6((Q_d)_i))_3))_2) \wedge \\ (s_d)_i \neq (s_d)_k \wedge i \neq k \wedge \forall i (Fn_9((s_d)_i, (Fn_6((Q_d)_i))_5, \\ (Fn_6((Q_d)_i))_5 + (Fn_6((Q_d)_i))_6) = 0)\} \end{aligned} \tag{11}$$

3.2 Fitness of Objective Functions

The fitness of any random solution with respect to the first objective can be calculated using Eqs. 12, 13 and 14. In this average parking prices is calculated for the slots that are given in solution.

$$V_p = \begin{cases} rr_{ap} & if\ \exists i \mid (s_d)_i \neq -1 \\ \beta & else \end{cases} \tag{12}$$

$$rr_{ap} = \frac{\sum_{i=1}^{|s_d|} rr_p}{\sum_{i=1}^{|s_d|} bb} \tag{13}$$

$$\begin{aligned} (rr_p, bb) = (p^d(Fn_8((s_d)_i), (Fn_6((Q_d)_i))_5, \\ (Fn_6((Q_d)_i))_5 + (Fn_6((Q_d)_i))_6), 1) \end{aligned} \tag{14}$$

The fitness of any random solution with respect to the second objective can be calculated using Eqs. 15, 16 and 17. In this the expected revenue is calculated that would be generated by renting out the slots that are given in solution.

$$V_r = \begin{cases} rr_r & if\ \exists i \mid (s_d)_i \neq -1 \\ \gamma & else \end{cases} \tag{15}$$

$$rr_r = \sum_{i=1}^{|s_d|} rr_p \times \left\lceil \frac{nn}{3600} \right\rceil \tag{16}$$

$$\begin{aligned} (rr_p, nn) = (p^d(Fn_8((s_d)_i), (Fn_6((Q_d)_i))_5, \\ (Fn_6((Q_d)_i))_5 + (Fn_6((Q_d)_i))_6), (Fn_6((Q_d)_i))_5) \end{aligned} \tag{17}$$

3.3 Crossover and Mutation

The uniform crossover technique is used in this work as depicted in Fig. 2. In order to mutate the solutions, random mutation technique

Parent 1	ps_6	ps_8	ps_7	ps_{10}	ps_{12}	ps_4
Parent 2	ps_4	ps_2	ps_9	ps_3	ps_5	ps_1
Coin Toss	1	0	1	1	0	1
Child 1	ps_4	ps_8	ps_9	ps_3	ps_{12}	ps_1
Child 2	ps_6	ps_2	ps_7	ps_{10}	ps_5	ps_4

Figure 2: Uniform crossover

is followed in which particular '*ps*' is replaced by the other '*ps*'. This replacement is carried out in such a way that the mutated solution satisfies the definition of feasible solution given in the Eq. 11.

3.4 Parking Allocation

The algorithm 1 executes on each 'CD' located at different sub-areas of the city. Each 'CD' is ready with its best solution for the requests it have in its waiting queue after the execution of above mentioned schemes. Then, the 'CD' will look for the lock if it have at-least one pending request. The locks are granted randomly. Once the lock is granted, then that 'CD' allocates the slots according to its best solution. Before allocation, every 'CD' checks whether the slot to be allocated is free or not for the desired amount of time. Although it has been already checked before generating 'S_d', but still there may be a case where any other 'CD' acquired the lock first and granted the said slot. This case is possible because the request may get decided by the 'CD' under which the requested parking lot is not present. Further, there is a compulsory time duration ('Δt_{cmp}') up-to which request can be present in the waiting queue if not accepted. After this, the request gets rejected if it is not accepted. Moreover, the advance requests are also allowed in this work where the 'CD' defers its decision (up-to $rr_{adv}\%$ of advance time) on such request if it doesn't find match with the requested parking lot. After such deferred period, 'CD' can allocate any parking lot under the common sub-area during time duration equals to 'Δt_{cmp}'.

4 Performance Evaluation

In this section, the performance of the proposed *Foggy-Park* scheme is evaluated with respect to various state of the art schemes.

4.1 Simulation Parameters

In the present simulation, the Seattle city parking environment is created using realistic parameters and tested. The Seattle parking data-set [1] and prices data-set [2] are used in the simulation. The values used for various parameters, such as, nn_{pop}, $|\ \mathtt{S_d}\ |$, Δt_{cmp}, rr_{adv}, nn_{iter}, rr_c, rr_m, α, β, and γ are $|\ \mathtt{S_d}\ |$, $|\ \mathtt{Q_d}\ | \times 10$, 5 seconds, 70, 100, 1, 0.05, 5555555555, 1000, and 0 respectively.

4.2 Results and Discussion

The compared schemes are as follows. STD Pricing and Allocation Scheme (PAS) which mimics the scheme implemented by the STD. Further, Spatial1 and Spatial2 PAS have same average prices for

Algorithm 1 *Foggy-Park* scheme

Input: AR, SA, BF, SS, PU, PL, PS, TS, ID, NN, RR, D, Q, RQ, $Fn_1, Fn_2, Fn_3, Fn_4, Fn_5, Fn_6, Fn_7, Fn_8, Fn_9$
Output: S_d

C = {}, CD = {}, C^n = {}, CD^n = {}, FP = {}, FR = {}, FP^n = {}, FR^n = {}, $iter$ = 1;
while $\exists\ rq_{new}$ **do**
  $Fn_6(d) = Fn_6(d) \cup rq_{new}$;
  $Fn_7(d) = Fn_7(d) \cup \{((Fn_6(d))_{|Fn_6(d)|})_1, 0, 0, 0, 0, 0\}$;
  $Q_d = Q_d \cup \{((Fn_6(d))_{|Fn_6(d)|})_1\}$;
end while
generate S_d where $|\ S_d\ | = nn_{pop}$
while $iter < nn_{iter}$ **do**
  for (i = 1; i ≤ nn_{pop}; i++) **do**
    $(FP)_i$ = calculate fitness of $(S_d)_i$ using Eqs. 12, 13, and 14.
    $(FR)_i$ = calculate fitness of $(S_d)_i$ using Eqs. 15, 16, and 17.
  end for
  C = non_dominated_sort(FP, FR)
  for (i = 1; i ≤ $|\ C\ |$; i++) **do**
    $CD = CD \cup$ {crowding_distance(FP, FR, $(C)_i$)}
  end for
  $S_d^n = S_d$;
  while $|\ S_d^n\ | \neq 2 \times nn_{pop}$ **do**
    (j,k) = (rand_int(1,nn_{pop}), rand_int(1,nn_{pop}))
    $S_d^n = S_d^n \cup$ crossover_mutation($(S_d)_j$, $(S_d)_k$, rr_c, rr_m)
  end while
  for (i = 1; i ≤ 2 × nn_{pop}; i++) **do**
    $(FP^n)_i$ = calculate fitness of $(S_d^n)_i$ using Eqs. 12, 13, and 14.
    $(FR^n)_i$ = calculate fitness of $(S_d^n)_i$ using Eqs. 15, 16, and 17.
  end for
  C^n = non_dominated_sort(FP^n, FR^n)
  for (i = 1; i ≤ $|\ C^n\ |$; i++) **do**
    $CD^n = CD^n \cup$ {crowding_distance(FP^n, FR^n, $(C^n)_i$)}
  end for
  X = {};
  for (i = 1; i ≤ $|\ C^n\ |$; i++) **do**
    for (j = 1; j ≤ $|\ (C^n)_i\ |$; j++) **do**
      $C^{nn} = C^{nn} \cup \{index\_of(((C^n)_i)_j, (C^n)_i)\}$;
    end for
    F^n = sort_by_values(C^{nn}, $(CD^n)_i$);
    for (j = 1; j ≤ $|\ (C^n)_i\ |$; j++) **do**
      $F = F \cup \{((C^n)_i)_{(F^n)_j}\}$;
    end for
    F = reverse(F);
    for (j = 1; j ≤ $|\ F\ |$; j++) **do**
      $X = X \cup \{(F)_j\}$
      if $|\ X\ |$ == nn_{pop} **then**
        break;
      end if
    end for
    if $|\ X\ |$ == nn_{pop} **then**
      break;
    end if
  end for
  S_d = {};
  for (i = 1; i ≤ $|\ X\ |$; i++) **do**
    $S_d = S_d \cup \{(S_d^n)_{(X)_i}\}$;
  end for
end while

Table 1: Comparison between different computing paradigms

Parameter→	Communication cost (Transferred data in MBs)				
Links ↓	*CS*	*AR*	*Foggy-Park*	*BF*	*SS*
cs ↔ ar	7.19	1.25	1.16	1.33	1.35
ar ↔ sa	7.19	7.72	2.32	2.66	2.69
sa ↔ bf	7.19	7.72	7.22	5.16	5.22
bf ↔ ss	7.19	7.72	7.22	8.25	8.41
ss ↔ pu	5.01	5.36	5.04	5.72	5.79
Total	33.77	29.77	22.96	23.12	23.46

each area and sub-area respectively. Lastly, Occupancy PAS is based on research work [13].

Fig. 3(a) depicts the number of requests accepted and rejected during different hours of the day. It clearly depicts that more number of requests are accepted during different hours of the day when '*Foggy-Park*' scheme is used in comparison to other schemes. Fig. 3(b) shows average parking prices paid by the travelers during different hours of the day where '*Foggy-Park*' comes out as a better scheme than others. The less average prices charged by the proposed scheme compared with Occupancy PAS is due to the allocation of parking not at desired parking lot but at nearby parking lots where occupancy is less and hence, prices. Fig. 3(c) represents revenue collected by renting out the parking slots during different hours of the day. Here also, the '*Foggy-Park*' scheme outperforms other compared schemes by generating more revenue. The '*Foggy-Park*' balances occupancy among different sub-areas due to which more requests are accepted and hence, more revenue is generated. Fig. 4(a) presents occupancy fairness among sub-areas during different hours of the day. This is calculated using Jain's fairness index using Eq. 18.

$$\mathtt{JF}(\mathtt{o}_1, \mathtt{o}_2, ..., \mathtt{o}_\mathtt{n}) = \frac{\left(\sum_{\mathtt{i}=1}^{\mathtt{n}} \mathtt{o}_\mathtt{i}\right)^2}{\mathtt{n} \times \sum_{\mathtt{i}=1}^{\mathtt{n}} \mathtt{o}_\mathtt{i}^2} \tag{18}$$

Here, $\mathtt{o}_\mathtt{i}$ is the average occupancy of $\mathtt{i}^{\mathtt{th}}$ sub-area during particular hour of the day. There are total '$\mathtt{n}$' = 33 sub-areas considered in this study. The plot shows that '*Foggy-Park*' scheme is more fair than other schemes in balancing occupancy of the parking lots. Table 1 presents the communication cost in terms of data transferred between different links present in the considered system model. It is evident from the results that less data is transferred under '*Foggy-Park*' scheme when compared with other computing paradigms implementation of the proposed scheme. Fig. 4(b) depicts the total number of hops passed by the messages exchanged between entities during different hours of the day. Whereas, the Fig. 4(c) shows total hops passed by the messages generated due to requests received for different parking lots. Both the figures proved efficacy of the '*Foggy-Park*' scheme as the less number of hops are passed in it. The notations, such as, '*CS*', '*AR*', '*SA*', '*BF*', and '*SS*' used in Fig. 4(b), Fig. 4(c), and Table 1 depicts the places where CDs are placed.

5 Conclusion

In this work, ZTSP allocation problem is formulated in terms of minimization of parking prices for the travelers and maximization of revenue generated by renting out the parking spaces. The dynamic parking prices are computed using base prices (used by STD)

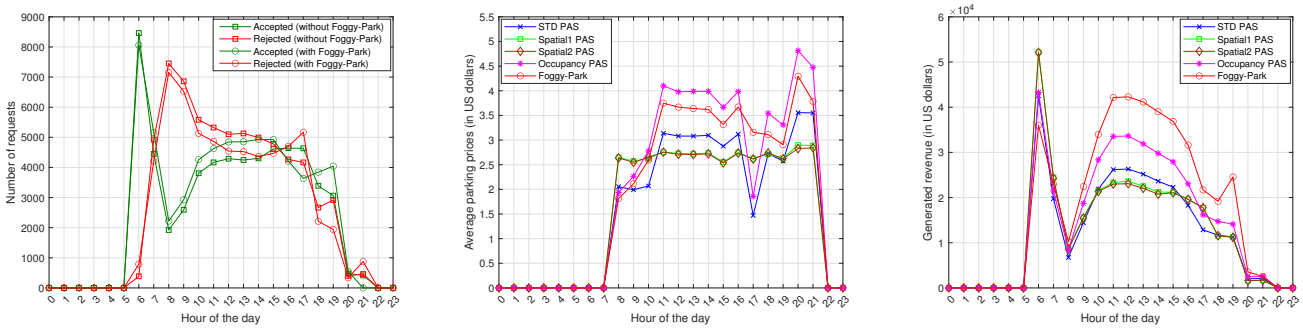

Figure 3: (a) Requests decision vs hour of the day (b) Average parking prices vs hour of the day (c) Revenue generated vs hour of the day

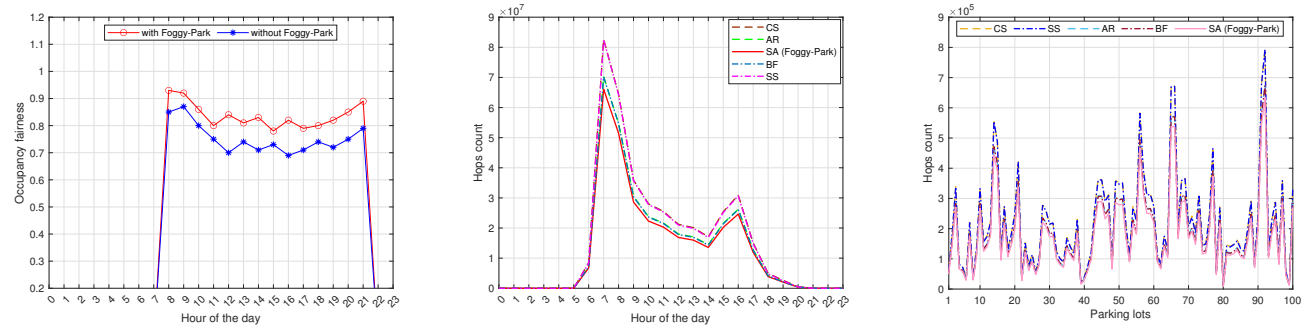

Figure 4: (a) Occupancy fairness vs hours of the day (b) Hops count vs hour of the day (c) Hops count vs parking lots

and random forest model of machine learning. The dynamic prices are fair in sense that it increases or decreases with increase or decrease in parking lot occupancy. The formulated multi-objective optimization problem. then solved by using NSGA-II algorithm. The obtained results prove that the proposed *Foggy-Park* scheme minimizes the average parking prices, maximizes the generated revenue, maximizes the accepted requests, and maximizes the occupancy fairness by around 4%, 23%, 6%, and 11.28% respectively. Further, the *Foggy-Park* scheme proved its worth over other computing paradigms by sending less amount of data over the network and passing less number of hops. In future, parameters such as parkers' type (paid or restricted), satisfaction, willingness to pay can be added while solving for the framed objectives.

References

[1] 2019. Seattle city on-street parking data. https://data.seattle.gov/Transportation/Paid-Parking-Occupancy-Last-30-Days-/rke9-rsvs. accessed on May 25, 2019.

[2] 2019. Seattle city on-street parking rates. https://www.seattle.gov/transportation/projects-and-programs/programs/parking-program/paid-parking-information/street-parking-rates. accessed on May 25, 2019.

[3] Meshari Aljohani, Stephan Olariu, Abrar Alali, and Shubham Jain. 2021. A survey of parking solutions for smart cities. *IEEE Transactions on Intelligent Transportation Systems* (2021).

[4] Dou An, Qingyu Yang, Donghe Li, Wei Yu, Wei Zhao, and Chao-Bo Yan. 2020. Where am i parking: Incentive online parking-space sharing mechanism with privacy protection. *IEEE Transactions on Automation Science and Engineering* (2020).

[5] Yanfeng Geng and Christos G Cassandras. 2013. New "smart parking" system based on resource allocation and reservations. *IEEE Transactions on intelligent transportation systems* 14, 3 (2013), 1129–1139.

[6] Vikas Hassija, Vikas Saxena, Vinay Chamola, and F Richard Yu. 2020. A parking slot allocation framework based on virtual voting and adaptive pricing algorithm. *IEEE Transactions on Vehicular Technology* 69, 6 (2020), 5945–5957.

[7] Xiang TR Kong, Su Xiu Xu, Meng Cheng, and George Q Huang. 2018. IoT-enabled parking space sharing and allocation mechanisms. *IEEE Transactions on Automation Science and Engineering* 15, 4 (2018), 1654–1664.

[8] Amir O Kotb, Yao-Chun Shen, Xu Zhu, and Yi Huang. 2016. iParker—A new smart car-parking system based on dynamic resource allocation and pricing. *IEEE transactions on intelligent transportation systems* 17, 9 (2016), 2637–2647.

[9] Peng Qin, Yang Fu, Xu Feng, Xiongwen Zhao, Shuo Wang, and Zhenyu Zhou. 2021. Energy-Efficient Resource Allocation for Parked-Cars-Based Cellular-V2V Heterogeneous Networks. *IEEE Internet of Things Journal* 9, 4 (2021), 3046–3061.

[10] Syed R Rizvi, Susan Zehra, and Stephan Olariu. 2020. Mapark: A multi-agent auction-based parking system in internet of things. *IEEE Intelligent Transportation Systems Magazine* 13, 4 (2020), 104–115.

[11] Sandeep Saharan, Seema Bawa, and Neeraj Kumar. 2020. Dynamic pricing techniques for Intelligent Transportation System in smart cities: A systematic review. *Computer Communications* 150 (2020), 603–625.

[12] Sandeep Saharan, Seema Bawa, and Neeraj Kumar. 2020. OP^{3}S: On-Street Occupancy based Parking Prices Prediction System for ITS. In *2020 IEEE Globecom Workshops (GC Wkshps*. IEEE, 1–6.

[13] Sandeep Saharan, Neeraj Kumar, and Seema Bawa. 2020. An efficient smart parking pricing system for smart city environment: A machine-learning based approach. *Future Generation Computer Systems* 106 (2020), 622–640.

[14] Sandeep Saharan, Neeraj Kumar, and Seema Bawa. 2023. DyPARK: A Dynamic Pricing and Allocation Scheme for Smart On-Street Parking System. *IEEE Transactions on Intelligent Transportation Systems* 24, 4 (2023), 4217 – 4234.

[15] Bo Zeng, Zhiwei Zhu, Hao Xu, and Houqi Dong. 2020. Optimal public parking lot allocation and management for efficient PEV accommodation in distribution systems. *IEEE Transactions on Industry Applications* 56, 5 (2020), 5984–5994.

[16] Yi Zhang, Chih-Yu Wang, and Hung-Yu Wei. 2019. Parking reservation auction for parked vehicle assistance in vehicular fog computing. *IEEE Transactions on Vehicular Technology* 68, 4 (2019), 3126–3139.