*Dy*PARK: A Dynamic Pricing and Allocation Scheme for Smart On-Street Parking System

Sandeep Saharan, Neeraj Kumar, *Senior Member, IEEE*, and Seema Bawa, *Member, IEEE*

***Abstract*— In-advance availability of parking information plays an important role in parker/ traveller decision-making for parking, curbing congestion, and managing parking lots efficiently. Specifically on-street parking poses many challenges compared to the off-street ones. Many users such as, store owners, municipal authorities, and police demand slots for on-street parking Free of Charges (FoC) for a short duration. In last few years, parking authorities collected data to attract the attention of researchers to present data-centric solutions for various problems such as, minimization of parking prices, maximization of revenue, and balacing the congestion at parking lots associated with smart parking systems. Motivated from the aforementioned problem, this paper proposes a scheme based on machine learning and game theory for dynamic pricing and allocation of parking slots in on-street parking scenarios. The dynamic pricing and allocation problem is modeled as Stackelberg game and is solved by finding its Nash equilibrium. Two types of Parking Users (PUs), i.e., Paid Parking Users (PPUs) and Restricted Parking Users (RPUs) are considered in this work. RPUs avail parking slots FoC once a day. PPUs compete to minimize the prices, and RPUs compete to maximize the FoC granted duration. The Parking Controllers (PCs) compete to maximize revenue generated from PPUs and to minimize total FoC parking duration granted to the RPUs. The random forest model is used to predict occupancy, which in turn is used to generate parking prices. Seattle city parking and its prices data sets are used to predict occupancy and to generate prices, respectively. In order to test the performance of communication system, the proposed *Dy*PARK Pricing and Allocation Scheme (PAS) is compared with its four variants and is found worth. The proposed scheme is also compared with other state-of-the-art schemes using various performance evaluation metrics. Simulated results prove the superiority of the proposed scheme in comparison to the other state-of-the-art schemes.**

***Index Terms*— Resource allocation, dynamic pricing, smart parking, stackelberg game, machine learning, cloud computing, fog computing, edge computing, smart cities, intelligent transportation system.**

Manuscript received 1 April 2022; revised 15 July 2022 and 4 November 2022; accepted 12 December 2022. Date of publication 5 January 2023; date of current version 29 March 2023. The Associate Editor for this article was H. Jiang. *(Corresponding author: Neeraj Kumar.)*

Sandeep Saharan and Seema Bawa are with the Department of Computer Science and Engineering, Thapar Institute of Engineering and Technology, Patiala, Punjab 147004, India (e-mail: ssaharan_phd17@thapar.edu; seema@thapar.edu).

Neeraj Kumar is with the Department of Computer Science and Engineering, Thapar Institute of Engineering and Technology, Patiala, Punjab 147004, India, also with the School of Computer Science, University of Petroleum and Energy Studies, Dehradun, Uttarakhand 248007, India, also with the Department of Electrical and Computer Engineering, Lebanese American University, Beirut 1102 2801, Lebanon, and also with the Faculty of Computing and IT, King Abdulaziz University, Jeddah 21589, Saudi Arabia (e-mail: neeraj.kumar@thapar.edu).

Digital Object Identifier 10.1109/TITS.2022.3230851

I. Introduction

SMART PARKING (SP) has become one of the emerging areas with the inclusion of Information and Communication Technologies (ICT) in it. Growing awareness, new business opportunities, and evolving technology stretched out the role of SP in smart and sustainable cities. SP en-cash its utility with the application of the latest smart/ intelligent techniques, such as, machine learning, artificial intelligence, image processing, and various sensors, such as, infra-red, ultrasonic, magnetometer, and radar. Despite fast improvements in the ICT for SP systems, it is still a challenge to manage limited parking resources efficiently. Various challenges still exist even after the emergence of SP technologies, including congestion due to parking, cruising of vehicles, increase in pollution, driver's unwillingness to pay high prices, parker's satisfaction and wrong placement of vehicles. These are mainly due to the massive increase in vehicles compared to the total number of existing parking slots. It is a case of a common phenomenon, i.e., high demand and low supply. In many such situations, Dynamic Pricing (DP) has evolved as a promising solution [1].

Whether the DP is simple or complex but it has already replaced the static pricing. It addresses various problems associated with the SP systems. DP in the field of SP considers many affecting factors for efficient allocation of parking slots. These factors include occupancy, availability, weekday/ weekend, time of the day, geographical parameters (such as, area, sub-area, streets), climate conditions, parking violations, temporary arrangements (such as, functions, congregations in nearby places) [1]. Short-period (e.g., hourly) parking facilities pose more challenges on the parking authorities than long-period (e.g., daily) parking facilities. These challenges are mainly because of the uncertainty of parkers' arrivals and departures. The travelers find such short-period parking facilities on the streets which is known as on-street parking. In general, these facilities are associated by the stores, shops, offices, and small scale industries where the owners require such parking Free of Charges (FoC) for a brief duration to load and unload the merchandise. Many a times other authorities such as municipal department and state police also require parking for a brief duration of time.

Thus, there exist many stakeholders in the on-street parking facilities, such as, Paid Parking Users (PPUs), Restricted Parking Users (RPUs), and parking management authorities. PPUs want to pay minimum prices. RPUs want to have FoC parking for longer duration. The parking management authorities want

to collect maximum revenue even after providing FoC parking to the RPUs. In such a scenario where all the stakeholders (known as players of the game) negotiate their motives, game theory can play an important role in providing an equilibrium solution [2]. Such solution takes care of motives of all the players and it can be a unique or many in number depending upon the game formulations. While trading in the game, all stakeholders communicate with each other. Parking Users interact with Infrastructure (PU2X), Infrastructure interacts with the Parking Users (X2PU), and also, the Infrastructure interacts with the Infrastructure (X2X) [3].

The type of interactions defined above must be sequential. The one who moves first can be thought as of a leader, whereas, the one who moves second can be a follower. Thus, Stackelberg game which qualifies all of these conditions can be one formulation for allocation of parking facilities to the users. In game theory, the Stackelberg game is used more often to formulate the problem of resource allocation [4]. Hence, it can be used to model the virtual parking slots trading. Thereafter, the game is solved for its equilibrium outcome. As a result, the travelers have in-advance knowledge of reservation which mitigates uncertainty. Thus, SP systems can have the advantage of eliminating vehicles cruising for parking and hence, the vehicular congestion on the roads due to parking can be reduced.

A. Related Work

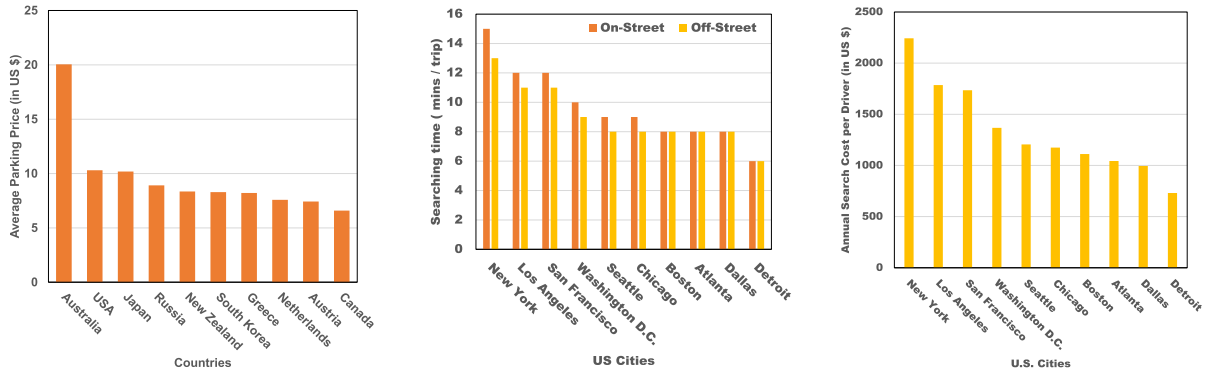
International Parking and Mobility Institute (IPMI) finds many gaps in SP systems through its research survey [5]. These gaps include "No utilization of data in deriving decisions", "Transportation, parking, and mobility are not seen as one connected entity," and "Shortage of investment in professional development and staff training." Many key barriers, market drivers, and current readiness levels have also been identified for end-to-end value parking while imaging parking 4.0 [1]. Excessive search for free parking slots when there is traffic congestion on the road due to less availability of parking slots or otherwise contributes additional problems, such as, negative socioeconomic impacts and travel delays. Traffic congestion in reverse also influences the parking and according to the IPMI survey [5] it is by 41%.

Over the last decade, many techniques, such as, optimization, dynamic programming, game theory, and queuing theory, have been applied in the area of dynamic parking pricing and allocation [6], [7], [8]. In the game theory, mainly auction have been used to accommodate dynamics in pricing. Data-driven study [9] illustrates the Spatio-temporal characteristics of curbside parking. Gaussian mixture model-based technique has been used to find out the places having similar spatial parking demands. Temporal and spatial distribution of parking prices can achieve many goals in real parking facilities [10], [11]. These goals include increasing revenue generated, optimizing balance in occupancy of different parking facilities, and reducing congestion due to cruising. Such distribution can be achieved by solving multi-period non-cooperative bi-level models with non-myopic approximate dynamic programming. Game-theoretic framework [12] analyzes parking slot assignment games in which analysis of both types of information scenario, i.e., complete and incomplete, can be done. To satisfy the stakes of both, i.e., travelers and parking agencies, the dynamic programming formulations of the problem can be solved using a stochastic look-ahead technique based on the Monte Carlo tree search algorithm [13].

The availability of parking occupancy in real-time can play a significant role in making parking choices [14]. Two types of parking choice, i.e., optimal and equilibrium, have been studied where drivers can pick parking spaces that reduce total driving distance and, hence, minimize the congestion. Coordinated and decentralized can be one view of the parking facility to reduce congestion [15]. Competition among parking vehicles can be modeled as a stochastic poisson game which is tested for real-world CBD covering Guicheng Community, Nanhai District at Foshan in China. Whereas, the uncoordinated parking space allocation in an urban environment has also been the area of study [16]. Every driver can have two choices, i.e., expensive over proper parking lots and inexpensive limited on-street parking slots. The resource selection game is formulated, and the impact of price on it is studied. Personalize parking guidance service can be modeled as a bi-level programming model where upper level modeled balanced and effective utilization of parking spaces and lower level minimizes the walking distance after parking [6]. This model has been solved using a nested particle swarm optimization algorithm. Similarly, alternating direction method of multipliers based approach has also been used in the formulation which minimizes the parking expenses and balances demand of parking overall parking lots [17].

The role of hardware such as sensors and associated software systems in solving problems related to the on-street parking system is well-known [18]. These problems include illegal parking, additional vehicle emissions, traffic congestion, and finding parking spaces on the streets. The hardware can sense or take pictures. Then, an object detection algorithm can be applied to such data-sets. The software can be used to manage occupancy and to provide real-time information to the parkers. The parking allocation has also been done by using matching theory where stability and strategy-proofness are discussed [19]. In this study, dynamic pricing is considered where three types of parking statuses (i.e., available, congested, and full) are used to compute the prices. This dynamicity in the prices acts as the load balancer in the parking system. An ensemble optimization strategy can be used for dynamic parking-space allocation [20]. In the first step of this strategy, an inefficient use of the parking spaces is found. Then this strategy optimizes the usage of parking spaces by removing entry conflicts. It is done by altering the parking sequence. This study has been carried out on the on-street parking facilities of Guilin city.

Now a days, the vehicles are connected with the cloud or Parking Service Provider (PSP) through internet. Such vehicles can also connect themselves with each other. Although continuous connectivity may be a challenge but the vehicles can form a coalition among themselves for the proper delivery of the messages [21]. The reliability of data forwarded in such vehicular networks can be ensured by using Bayesian Coalition Game (BCG) using Learning Automata [22]. In such game, the players learn from the reward/ penalty they get for their



(a) (b) (c)

Fig. 1. Motivation examples (a) Average parking price for 2-hour parking facility [24] (b) Average searching time for each trip in US cities (c) Average search cost for each driver in US cities.

TABLE I
COMPARISON WITH EXISTING PROPOSALS

Proposals	1	2	3	4	5	6	7	8	9	10	11	12
[6]	✗	✗	✓	✗	✓	✓	✓	✗	✗	✗	✗	✗
[12]	✗	✗	✓	✗	✓	✓	✗	✗	✗	✗	✗	✗
[14]	✓	✗	✗	✗	✓	✓	✗	✗	✗	✗	✗	✗
[15]	✗	✗	✓	✗	✓	✓	✓	✗	✗	✗	✗	✗
[16]	✓	✗	✓	✓	✓	✓	✗	✗	✗	✗	✗	✗
[13]	✓	✓	✓	✗	✓	✓	✓	✗	✗	✗	✗	✗
[17]	✓	✗	✓	✗	✓	✓	✓	✗	✗	✗	✗	✗
[25]	✓	✓	✓	✗	✓	✓	✓	✗	✗	✗	✗	✗
[29]	✓	✓	✓	✗	✗	✓	✗	✗	✗	✗	✗	✗
[30]	✓	✓	✗	✗	✗	✓	✗	✗	✗	✗	✗	✗
[31]	✓	✓	✓	✓	✓	✓	✓	✓	✗	✗	✗	✗
Proposed Scheme	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓

1:Parking prices, 2:Revenue generated, 3: Geo-distribution of parking facilities, 4: On-street parking facilities, 5: Congestion, 6: Parking allocation, 7: Occupancy, 8: Machine learning, 9: Seattle city parking data-set, 10: Restricted parking users, 11: SLA violations, 12: Stackelberg game, ✓: considered, and ✗: non-considered

actions from the environment. The solution to this game can improve parameters, such as, delay, throughput, and message overhead. The collaborative learning automata-based routing algorithm is useful in the scenario where congestion happens due to cruising of vehicles in search of parking [23]. This algorithm divides whole region into clusters and then learns from parameters, such as, distance between vehicle and nearest road side unit, vehicular congestion, and delay in the network.

Various Computing Paradigms (CPs), such as, Cloud Computing (CC), vehicular Fog Computing (FC) [25], [26], and vehicular Edge Computing (EC) [27] already find their place in SP research. Parking allocation can also be done using single-round auction. Further, fog capability of the parked vehicles have also been used to assist the delay-sensitive computing services [25], [26]. Directions of the parking research are not limited and, in fact, systems in which flexible reservation mechanism where reservation is no longer restricted to a specific location at one particular time but tolerates predetermined spatio-temporal flexibility instead are developed [28]. Table I compares the proposed scheme with the existing schemes based on various relevant parameters.

B. Motivation

The on-street parking is a very common and widely used parking across the globe. Its importance extends with the ease of accessibility to the destination and availability of other limited parking facilities. The restricted use of such parking spaces arise often due to local shopkeepers, municipal authorities, police, etc. Such restrictions should be readily available and given FoC. In the case of central business districts, the demand of parking is more than its supply. The on-street parking at such places is generally used for short time duration and average prices for such short parking are high as reported in Fig. 1(a). The cascading effect of charging high prices increases in search time for affordable spaces as presented in Fig. 1(b). It leads to an increase in the annual search cost per driver as depicted in Fig. 1(c). Also, the parking authorities aim to maximize their turnover/ revenue by charging higher prices which is unacceptable to the parking users in general. Thus, all the stakeholders fight for their own motives. Such above-mentioned problems have not been addressed jointly and proportionality in the reported literature.

C. Contributions of this article

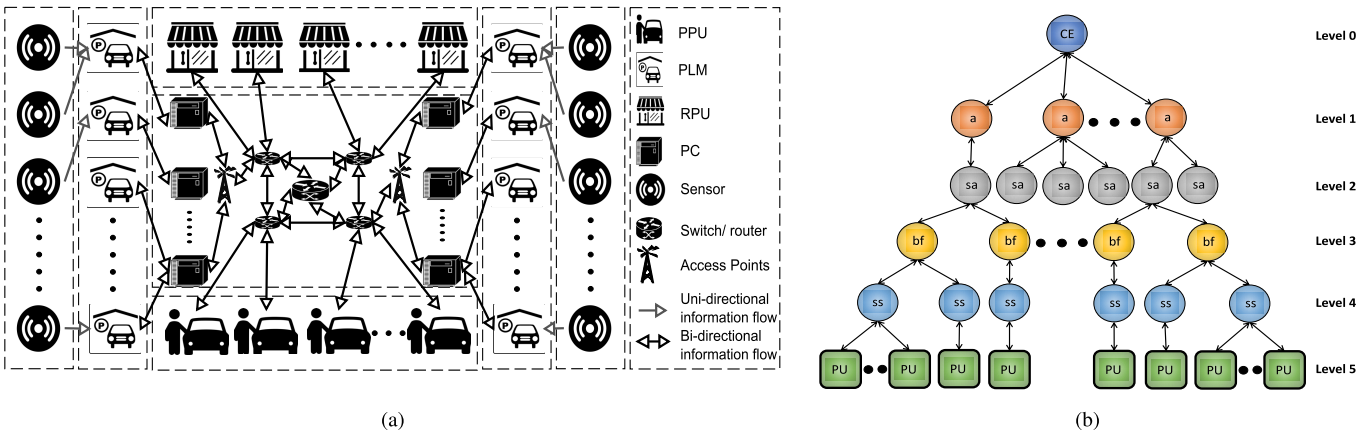
The significant contributions of this paper are as follows.

i) The two Stackelberg games are formulated in this work in order to solve the dynamic parking pricing and allocation problem of the on-street parking system. One is played between the PPUs and Parking Controllers (PCs) (i.e., PSPs) and second is played between RPUs and PCs. The proposed game is played to have a Nash Equilibrium (NE).

ii) The game played between PPUs and PCs decides the parking prices to be paid by the PPUs if their requests are accepted. In order to generate such prices, PCs uses random forest model to predicts occupancy of the requested parking lots. The Seattle city parking data-set [32] and parking prices data-set [33] are used in this reference.

iii) The game played between RPUs and PCs decides the FoC parking duration to be allocated to the RPUs if their requests are accepted. This work implements a service level agreement between PCs and RPUs where RPUs can access parking facility FoC once a day on priority basis. Also, the SLA provides assurance of allocating at least predefined threshold duration to the RPUs.

iv) Efficient algorithms are designed, simulated and tested on the Seattle on-street parking system environment which minimizes the parking prices for PPUs, maximizes the allocated FoC parking duration for RPUs, maximizes the revenue generated by the PCs, and



(a) (b)

Fig. 2. (a) System Model (b) Communication architecture of Seattle parking system.

minimizes the overall parking loss incurred by the PCs in allocating FoC parking to the RPUs. The efficiency of the proposed scheme has been compared with other state-of-the-art schemes.

D. Organization

Rest of the paper is organized as follows. Section II describes the overall model of the developed system. Section III describes the notations and objective functions of the problem solved in this work. Section IV describes the formulated game and proposed *Dy*PARK Pricing and Allocation Scheme (PAS) which optimizes the objective functions given in Section III. Section V shows the results obtained by using the proposed PAS scheme when compared to other state-of-the-art techniques. Finally, section VI concludes the article.

II. System Model

The proposed system model is divided into five layers, as shown in Fig. 2(a). These layers are discussed below.

A. Paid Parking Users Layer

The PPUs layer consists of one or many parkers/ travelers who want parking slots in exchange for parking prices. Such parking slots are provided by the PSP(s) to the PPUs. The PPUs send requests for the required parking slots to the PSP(s). After receiving such a request, PSP(s) computes the price to be charged for the requested parking slot. After price computation, PSP(s) announces it to the PPUs. If the prices suit the PPUs' pocket, they pay the price and have the parking slot; otherwise, PPUs either wait or withdraw the request. This process can last longer depending on the type of request.

B. Restricted Parking Users Layer

The RPUs layer plays an important role in existing parking facilities. The major part of the world's parking facilities is located on the sides of the streets. Thus, RPUs (such as, store owners, police, and official authorities) who are not the owner or regular users of such parking facilities but need it for various purposes, such as, loading/ unloading of goods, inauguration, functions, etc. Such RPUs have the fundamental right to use parking facilities reasonably without paying the prices. Thus, RPUs send parking requests to the PSP(s). After receiving such requests, PSP(s) compute the FoC parking duration for which these requests can be granted requested parking slots. After computation, PSP(s) announces it to the RPUs. If the announced time duration suits RPUs' requirements, then the term is accepted. Otherwise, RPUs either wait or withdraw the request. Again, this process can also last longer depending upon the type of request.

C. Parking Lot Layer

The parking lot layer consists geo-distributed parking lots located at side of the streets, i.e., level 4 as depicted in Fig. 2(b) along with Parking Lot Managers (PLMs) and various sensors. PLM keeps the record of occupancy, accepted requests, arrival and departure of vehicles. Sensors are used for automatic detection of occupied/unoccupied parking slots or arrival/ departure of vehicles. This information helps the PSP(s) in better allocation of parking slots to the PPUs/ RPUs.

D. Parking Service Providers Layer

In this layer PSPs coordinate with PPUs/RPUs and PLMs. Each PSP has installed one PC which takes decision on his behalf. PC will get the requests from PPUs/ RPUs. It generates the prices/FoC durations and announce them. Finally, the PC decides request. The PCs decides prices and FoC parking duration based on several parameters, such as (i) day of the week, (ii) time of the day, (iii) area, (iv) sub-area, (v) block face, (vi) side of the street where requested parking slot is located, (vii) predicted occupancy, and (viii) observed occupancy of the requested parking lot. Further PC informs the PLMs regarding acceptance of requests and gets the status of parking from PLMs. In this work each sub area of the Seattle city handled by one PSP. Hence, PCs are installed at sub-areas, i.e., level 2 in depicted in Fig. 2(b) and takes care of one sub-area only. This placement of PCs at sub-areas mimics fog computing environment.

E. Parking Communication Layer

In the proposed system, there are two modes of communication, i.e., PU2X/ X2PU and X2X. Infrastructure consists

of PCs, PLMs, access points, routers and sensors. The communication between PPUs/ RPUs and PC happens through access points and routers. The sensors installed at parking lot communicate with PLMs and PLMs in turn communicate with PCs. Fig. 2(a) shows the interaction between the PPUs, RPUs, PCs, PLMs, and sensors in the proposed system model.

III. Problem Formulation

The problems that have been solved in this work are formulated and presented as objective functions in this section.

A. Notations

Let the sets are denoted by one or two uppercase letters. Let '$\mathtt{A}$' denotes the set of areas. '$\mathtt{A}$' can have one or more sub-area under it. Let '$\mathtt{SA}$' denotes the set of those sub-areas. On the similar lines one '$\mathtt{SA}$' can have one or more block face (street) under it. Let '$\mathtt{BF}$' denotes the set of those block faces. In turn, one '$\mathtt{BF}$' can have one or more side of the street under it. Let '$\mathtt{SS}$' denotes the set of those side of the streets. Let '$\mathtt{T}$' be the set of distinct timestamps. Let '$\mathtt{PU}$' denotes the set of all distinct registered parking users. Let '$\mathtt{PC}$' be the set of all parking controllers. Let all parking lots considered in this paper comes under the set '$\mathtt{PL}$'. Let '$\mathtt{PS}$' denotes all parking slots present in each parking lot. Let '$\mathtt{X}$' be the set of unique request IDs. Let '$\mathtt{N}$', '$\mathtt{R}$' and '$\mathtt{B}$' denote the set of natural, real, and binary numbers respectively. Let finite set '$\mathtt{M}$' denotes moves of the players in the game. The elements of a set are denoted by the lowercase of letter(s) which have been used for the set. The position of any element in the set is denoted by subscript. For example '$\mathtt{a}_\mathtt{i}$' is the '$\mathtt{i}^{\mathtt{th}}$' element of the set $\mathtt{A}$. In order to access element '$\mathtt{a}_\mathtt{i}$' of the set $\mathtt{A}$, $(\mathtt{A})_\mathtt{i}$ notation is used. Let the subset is denoted by same uppercase letter followed by number. For example $\mathtt{A1}$ and $\mathtt{A2}$ are two subsets of the same set $\mathtt{A}$. In this paper roster notation for sets have been used whenever required.

The relations are denoted by $\mathtt{S}^\mathtt{n}$ in general, where *n* is a natural number that distinguishes between two relations. And some functions have been used in this work on such relations. The generalized notation of the function applied on relation $\mathtt{S}^\mathtt{n} = \{\{\mathtt{I}_1, \mathtt{O}_1\}, .., \{\mathtt{I}_\mathtt{k}, \mathtt{O}_\mathtt{k}\}, .., \{\mathtt{I}_{|\mathtt{S}|}, \mathtt{O}_{|\mathtt{S}|}\}\}$ is given as $\mathcal{F}_{\mathtt{S}^\mathtt{n}}(\mathtt{I}_\mathtt{k}) \rightarrow (\mathtt{O}_\mathtt{k})$. $\mathtt{I}_\mathtt{k}$ is one or group of consecutive elements (at-most $|\mathtt{s}^\mathtt{n}|\text{-}1$) from starting in $\mathtt{s}^\mathtt{n}$. Whereas, $\mathtt{O}_\mathtt{k}$ is an ordered set computed using $\mathtt{s}^\mathtt{n} \backslash \mathtt{I}_\mathtt{k}$. The relations that are used in this work, are defined as follows: Let $\mathtt{S}^1 = \{\mathtt{s}^1 \mid \mathtt{s}^1$ is of type $\{\mathtt{pu}, \mathtt{b}\}\}$ be a relation where $\mathtt{s}^1$ depicts type of parking user (PPU if $\mathtt{b} = 1$ and RPU if $\mathtt{b} = 0$). $\mathtt{S}^2 = \{\mathtt{s}^2 \mid \mathtt{s}^2$ is of type $\{\mathtt{pu}, \mathtt{r}\}\}$ shows wallet value (amount or duration if '$\mathtt{pu}$' is PPU or RPU respectively) of the parking user. $\mathtt{S}^3 = \{\mathtt{s}^3 \mid \mathtt{s}^3$ is of type $\{\mathtt{a}, \mathtt{sa}, \mathtt{bf}, \mathtt{ss}, \mathtt{pl}\}\}$ where $\mathtt{s}^3$ provides mapping between parking lot id and its physical location. $\mathtt{S}^4 = \{\mathtt{s}^4 \mid \mathtt{s}^4$ is of type $\{\mathtt{pl}, \mathtt{ps}, \mathtt{ps}, \ldots.\}\}$ shows parking slots present in particular parking lot. $\mathtt{S}^5 = \{\mathtt{s}^5 \mid \mathtt{s}^5$ is of type $\{\mathtt{pl}, \mathtt{pc}\}\}$ depicts parking controller of the particular parking lot. Let $\mathtt{S}^6 = \{\mathtt{s}^6 \mid \mathtt{s}^6$ is of type $\{\mathtt{a}, \mathtt{sa}, \mathtt{t}, \mathtt{t}\}\}$ depicts the starting $(\mathtt{s}^6)_3$ and the ending $(\mathtt{s}^6)_4$ timestamp between which parking is charged in area '$\mathtt{a}$' and sub area '$\mathtt{sa}$'. $\mathtt{S}^7 = \{\mathtt{s}^7 \mid \mathtt{s}^7$ is of type $\{\mathtt{a}, \mathtt{sa}, \mathtt{n}\}\}$ depicts maximum time duration for which parking can be sought in one single request in specific area and sub area. $\mathtt{S}^8 = \{\mathtt{s}^8 \mid \mathtt{s}^8$ is of type $\{\mathtt{a}, \mathtt{sa}, \mathtt{t}, \mathtt{r}\}\}$ where $\mathtt{s}^8$ represents parking prices '$\mathtt{r}$' to be charged for one hour in area '$\mathtt{a}$', sub-area '$\mathtt{sa}$' at timestamp '$\mathtt{t}$'.

Let $\mathtt{S}^9 = \{\mathtt{s}^9 \mid \mathtt{s}^9$ is of type $\{\mathtt{x}, \mathtt{pu}, \mathtt{t}, \mathtt{pl}, \mathtt{ps}, \mathtt{t}, \mathtt{n}, \mathtt{b}, \mathtt{b}\}\}$ depicts ordered set (based on $(\mathtt{s}^9)_3$) of received requests. '$\mathtt{x}$' is the request id of the request received from '$\mathtt{pu}$'. $(\mathtt{s}^9)_3$ depicts request arrival timestamp. Parking is requested for slot '$\mathtt{ps}$' (if '$\mathtt{pu}$' is RPU otherwise it will be -1 (not defined)) of parking lot '$\mathtt{pl}$'. $(\mathtt{s}^9)_6$ depicts vehicle arrival time, and the requested duration of stay will be '$\mathtt{n}$' seconds. Here $(\mathtt{s}^9)_8 = \mathtt{0/1}$ represents offline/ online request and $(\mathtt{s}^9)_9 = \mathtt{0/1}$ shows immediate/ advance request respectively. Parking Users (PUs) can send requests either using devices with internet connection (*such request hereinafter referred to as 'online request'*) or from physical pay stations installed at various places of Seattle city (*such requests hereinafter referred to as 'offline request'*). PUs can demand parking slot with vehicle arrival time either equals to current timestamp (*such request hereinafter referred to as 'immediate request'*) or up to $\Delta \mathtt{t}^{\mathtt{cmp}}_{\mathtt{adv}}$ duration ahead of the current timestamp (*such request hereinafter referred to as 'advance request'*). The vehicle arrival time must be in the parking chargeable hours of the day of particular area and sub-area where requested parking lot is situated as given in Eq. 1.

$$(\mathtt{s}^9)_6 \in [(\mathcal{F}_{\mathtt{S}^6}(\mathtt{a}, \mathtt{sa}))_1, (\mathcal{F}_{\mathtt{S}^6}(\mathtt{a}, \mathtt{sa}))_2) \tag{1}$$

The requested parking duration by the PPUs and RPUs must be in accordance to Eqs. 2 and 3, as shown at the bottom of the next page, respectively. The values of area and sub-area used in Eqs. 1, 2, and 3 are given in Eq. 4.

$$\mathtt{a} = (\mathcal{F}^{-1}_{\mathtt{S}^3}((\mathtt{s}^9)_4))_1 \ , \ \mathtt{sa} = (\mathcal{F}^{-1}_{\mathtt{S}^3}((\mathtt{s}^9)_4))_2 \tag{4}$$

$\mathtt{S}^{10} = \{\mathtt{s}^{10} \mid \mathtt{s}^{10}$ is of type $\{\mathtt{t}, \mathtt{b}, \mathtt{b}\}\}$ where $\mathtt{s}^{10}$ represents whether the '$\mathtt{pc}$' has announced the FoC duration (if $(\mathtt{s}^{10})_2 = 1$) or not (if $(\mathtt{s}^{10})_2 = 0$) and also the prices (if $(\mathtt{s}^{10})_3 = 1$) or not (if $(\mathtt{s}^{10})_3 = 0$) at timestamp '$\mathtt{t}$'. $\mathtt{S}^{11} = \{\mathtt{s}^{11} \mid \mathtt{s}^{11}$ is of type $\{\mathtt{x}, \mathtt{t}, \mathtt{r}\}\}$ depicts prices or FoC duration '$\mathtt{r}$' (when sender of request having id '$\mathtt{x}$' is PPU or RPU respectively) generated by the parking controller at timestamp '$\mathtt{t}$'. $\mathtt{S}^{12} = \{\mathtt{s}^{12} \mid \mathtt{s}^{12}$ is of type $\{\mathtt{x}, \mathtt{t}, \mathtt{b}\}\}$ where $\mathtt{s}^{12}$ represents whether '$\mathtt{pu}$' requested parking with request id '$\mathtt{x}$' has accepted the prices/FoC duration or accepted waiting for next announcement at timestamp '$\mathtt{t}$' (when '$\mathtt{b}$' = 1) offered by the '$\mathtt{pc}$' or not (when '$\mathtt{b}$' = 0). $\mathtt{S}^{13} = \{\mathtt{s}^{13} \mid \mathtt{s}^{13}$ is of type $\{\mathtt{x}, \mathtt{n}, \mathtt{t}, \mathtt{r}, \mathtt{n}, \mathtt{ps}\}\}$ depicts details of the request decision. '$\mathtt{x}$' depicts request id. $(\mathtt{s}^{13})_2$ shows request decision ($\mathtt{-1/0/1}$ depicts rejected/ pending/ accepted decision respectively), '$\mathtt{t}$' shows decision timestamp, '$\mathtt{r}$' depicts decided prices per hour to be charged, $(\mathtt{s}^{13})_5$ and '$\mathtt{ps}$' shows allocated parking duration and parking slot (if the parking user is RPU then allocated parking slot will be the same as requested parking slot.) respectively. $\mathtt{S}^{14} = \{\mathtt{s}^{14} \mid \mathtt{s}^{14}$ is of type $\{\mathtt{ps}, \mathtt{t}, \mathtt{b}\}\}$ depicts occupancy status of the parking slot '$\mathtt{ps}$' at timestamp '$\mathtt{t}$' (occupied if '$\mathtt{b}$' = 1 or unoccupied if '$\mathtt{b}$' = 0). Let $\mathtt{S}^{15} = \{\mathtt{s}^{15}_\mathtt{i} \mid \mathtt{i} \in [1, 4] \wedge \mathtt{s}^{15}_\mathtt{i} = \{\{\} \vee \mathtt{x}_\mathtt{j} \mid \mathtt{j} \geq 1\}\}$ be a Multi-Level Queue (MLQ) where '$\mathtt{i}$' defines the priority level. The lesser value of level '$\mathtt{i}$' describes higher priority. Let $\mathtt{S}^{16} = \{\mathtt{s}^{16} \mid$

$\mathtt{s}^{16}$ is of type $\{\mathtt{x}, \mathtt{t}, \mathtt{m}, \mathtt{m}\}$ depicts the priority moves of PU having request id as 'x' at timestamp 't'. Let $\mathtt{S}^* = \{\mathtt{s}^* \mid \mathtt{s}^*$ is of type $\{\mathtt{pc}, \mathtt{S}^9, \mathtt{S}^{10}, \mathtt{S}^{11}, \mathtt{S}^{12}, \mathtt{S}^{13}, \mathtt{S}^{14}, \mathtt{S}^{15}, \mathtt{S}^{16}\}\}$ be a special relation which depicts sets related to particular 'pc'. Let $\mathcal{Z}(\mathtt{pc}) \rightarrow (\{\mathtt{S}^9, \mathtt{S}^{10}, \mathtt{S}^{11}, \mathtt{S}^{12}, \mathtt{S}^{13}, \mathtt{S}^{14}, \mathtt{S}^{15}, \mathtt{S}^{16}\})$be a special function that can be applied only on special relation $\mathtt{S}^*$.

B. Service Level Agreement

In this work, there is a SLA between PSPs and RPUs that if RPUs request for parking then parking will be granted to them i) FoC, ii) on priority basis, and iii) for at-least $\Delta\mathtt{t}^{\mathtt{thr}}_{\mathtt{rpu}}$ duration. The details of the priority of RPUs' requests are explained in section IV. Let $\Delta\mathtt{t}^{\mathtt{min}}_{\mathtt{rpu}}$ and $\Delta\mathtt{t}^{\mathtt{max}}_{\mathtt{rpu}}$ are the minimum and maximum time duration for which RPUs can ask parking facility in general. Then, $\Delta\mathtt{t}^{\mathtt{thr}}_{\mathtt{rpu}}$ can calculated as given in Eq. 5.

$$\Delta\mathtt{t}^{\mathtt{thr}}_{\mathtt{rpu}} = \begin{cases} \Delta\mathtt{t}^{\mathtt{min}}_{\mathtt{rpu}}, & \mathtt{if}\ \mathtt{s}^9_7 \in [\Delta\mathtt{t}^{\mathtt{min}}_{\mathtt{rpu}}, \Delta\mathtt{t}^{\mathtt{max}}_{\mathtt{rpu}}] \\ \mathtt{s}^9_7, & \mathtt{if}\ \mathtt{s}^9_7 < \Delta\mathtt{t}^{\mathtt{min}}_{\mathtt{rpu}} \end{cases} \tag{5}$$

C. List of Objectives

In light of the formulations mentioned above, the main objectives of the present work are defined as follows.

$\mathcal{O}1$) *Minimization of parking prices:* The proposed *Dy*PARK PAS minimizes the parking prices offered to the PPU(s). The objective function '$\mathcal{O}1$' is given in Eq. 6.

$$\mathcal{O}1 : \forall\, \mathtt{i}, \mathtt{j}, \mathtt{t}\ \ \mathtt{Minimize}\ \mathcal{F}_{(\mathcal{Z}(\mathtt{pc}_\mathtt{i}))_3}(\mathtt{x}_\mathtt{j}, \mathtt{t}) \tag{6}$$

Various constraints used while solving $\mathcal{O}1$ are $\mathcal{C}1$, $\mathcal{C}2$, $\mathcal{C}4$, $\mathcal{C}6$, and $\mathcal{C}7$.

$\mathcal{O}2$) *Maximization of granted parking duration:* The proposed *Dy*PARK PAS maximizes the FoC parking duration offered to the RPU(s). The objective function '$\mathcal{O}2$' is given in Eq. 7.

$$\mathcal{O}2 : \forall\, \mathtt{i}, \mathtt{j}, \mathtt{t}\ \ \mathtt{Maximize}\ \mathcal{F}_{(\mathcal{Z}(\mathtt{pc}_\mathtt{i}))_3}(\mathtt{x}_\mathtt{j}, \mathtt{t}) \tag{7}$$

The constraints used while solving $\mathcal{O}2$ are $\mathcal{C}1$, $\mathcal{C}3$, $\mathcal{C}4$, $\mathcal{C}5$, $\mathcal{C}6$, and $\mathcal{C}7$.

$\mathcal{O}3$) *Maximization of expected revenue:* The proposed *Dy*PARK PAS maximizes the expected revenue generated from the PPU(s). The objective function '$\mathcal{O}3$' is given in Eq. 8.

$$\mathcal{O}3 : \forall\, \mathtt{i}, \mathtt{j}, \mathtt{t}\ \ \mathtt{Maximize} \sum_{\mathtt{i}} \sum_{\mathtt{j}} \mathcal{F}_{(\mathcal{F}_{\mathtt{S}^*}(\mathtt{pc}_\mathtt{i}))_3}(\mathtt{x}_\mathtt{j}, \mathtt{t}) \times \left\lceil \frac{\left(\mathcal{F}_{(\mathcal{F}_{\mathtt{S}^*}(\mathtt{pc}_\mathtt{i}))_1}(\mathtt{x}_\mathtt{j})\right)_6}{3600} \right\rceil \tag{8}$$

The constraints used while solving $\mathcal{O}3$ are $\mathcal{C}1$, $\mathcal{C}2$, $\mathcal{C}4$, $\mathcal{C}6$, and $\mathcal{C}7$.

$\mathcal{O}4$) *Minimization of expected parking loss:* The proposed *Dy*PARK PAS minimizes the overall expected parking loss that would incur due to FoC allocation of parking slots to the RPUs. The objective function '$\mathcal{O}4$' is given in Eq. 9.

$$\mathcal{O}4 : \forall\, \mathtt{i}, \mathtt{j}, \mathtt{t}\ \ \mathtt{Minimize} \sum_{\mathtt{i}} \sum_{\mathtt{j}} \mathcal{F}_{(\mathcal{F}_{\mathtt{S}^*}(\mathtt{pc}_\mathtt{i}))_3}(\mathtt{x}_\mathtt{j}, \mathtt{t}) \tag{9}$$

The constraints used while solving $\mathcal{O}4$ are $\mathcal{C}1$, $\mathcal{C}3$, $\mathcal{C}4$, $\mathcal{C}5$, $\mathcal{C}6$, and $\mathcal{C}7$.

$$\begin{aligned}
\mathcal{C}1 &: \ \mathtt{x}_\mathtt{j} = (((\mathcal{F}_{\mathtt{S}^*}(\mathtt{pc}_\mathtt{i}))_1)_\mathtt{j})_1\,,\ \left(\mathcal{F}_{(\mathcal{F}_{\mathtt{S}^*}(\mathtt{pc}_\mathtt{i}))_5}(\mathtt{x}_\mathtt{j})\right)_1 = 0 \\
\mathcal{C}2 &: \ \mathcal{F}_{\mathtt{S}^1}((\mathcal{F}_{(\mathcal{F}_{\mathtt{S}^*}(\mathtt{pc}_\mathtt{i}))_1}(\mathtt{x}_\mathtt{j}))_1) = 1,\ (\mathcal{F}_{(\mathcal{F}_{\mathtt{S}^*}(\mathtt{pc}_\mathtt{i}))_2}(\mathtt{t}))_2 = 1 \\
\mathcal{C}3 &: \ \mathcal{F}_{\mathtt{S}^1}((\mathcal{F}_{(\mathcal{F}_{\mathtt{S}^*}(\mathtt{pc}_\mathtt{i}))_1}(\mathtt{x}_\mathtt{j}))_1) = 0,\ (\mathcal{F}_{(\mathcal{F}_{\mathtt{S}^*}(\mathtt{pc}_\mathtt{i}))_2}(\mathtt{t}))_1 = 1 \\
\mathcal{C}4 &: \ \mathcal{F}_{(\mathcal{F}_{\mathtt{S}^*}(\mathtt{pc}_\mathtt{i}))_4}(\mathtt{x}_\mathtt{j}, \mathtt{t}) = 1 \\
\mathcal{C}5 &: \ \mathcal{F}_{(\mathcal{F}_{\mathtt{S}^*}(\mathtt{pc}_\mathtt{i}))_3}(\mathtt{x}_\mathtt{j}, \mathtt{t}) \\
&\in \ [0, (\mathcal{F}_{(\mathcal{F}_{\mathtt{S}^*}(\mathtt{pc}_\mathtt{i}))_1}(\mathtt{x}_\mathtt{j}))_6 - \Delta\mathtt{t}^{\mathtt{thr}}_{\mathtt{rpu}}] \\
\mathcal{C}6 &: \ \sum_{\mathtt{k}=1}^{|\mathcal{F}_{\mathtt{S}^4}(\mathtt{pl})|} \mathcal{F}_{(\mathcal{F}_{\mathtt{S}^*}(\mathtt{pc}_\mathtt{i}))_6}\left(\left(\mathcal{F}_{\mathtt{S}^4}(\mathtt{pl})\right)_\mathtt{k}, \mathtt{t}\right) \leq |\ \mathcal{F}_{\mathtt{S}^4}(\mathtt{pl})\ | \\
\mathcal{C}7 &: \ \mathtt{i} \in [1, |\mathtt{PC}|],\ \mathtt{j} \in [1, |(\mathcal{F}_{\mathtt{S}^*}(\mathtt{pc}_\mathtt{i}))_1|],\ \mathtt{pl} \\
&\in \ \mathtt{PL},\ \mathtt{t} \in \mathtt{T}
\end{aligned}$$

The constraint $\mathcal{C}1$ defines the request id 'x' received at $\mathtt{i}^{th}$ parking controller and it should not be decided. $\mathcal{C}2$ represents that parking user of request id 'x' should be a PPU and parking controller must have announced the parking prices to the PPU at timestamp 't'. Whereas, $\mathcal{C}3$ represents that parking user of request id 'x' should be a RPU and parking controller must have announced the FoC parking duration to the RPU at timestamp 't'. The constraint $\mathcal{C}4$ represents that the requester must accept the offered parking prices or FoC duration (whichever applicable) at timestamp 't'. The constraint $\mathcal{C}5$ depicts that the offered FoC parking duration to the RPUs must be in range '0' to the difference between requested and threshold duration. The constraint $\mathcal{C}6$ defines

$$(\mathtt{s}^9)_7 \in \begin{cases} [1, \mathcal{F}_{\mathtt{S}^7}(\mathtt{a}, \mathtt{sa})], & \mathtt{if}\ (\mathtt{s}^9)_6 + \mathcal{F}_{\mathtt{S}^7}(\mathtt{a}, \mathtt{sa}) \leq (\mathcal{F}_{\mathtt{S}^6}(\mathtt{a}, \mathtt{sa}))_2 \\ [1, (\mathcal{F}_{\mathtt{S}^6}(\mathtt{a}, \mathtt{sa}))_2 - (\mathtt{s}^9)_6], & \mathtt{otherwise} \end{cases} \tag{2}$$

$$(\mathtt{s}^9)_7 \in \begin{cases} [\Delta\mathtt{t}^{\mathtt{min}}_{\mathtt{rpu}}, \Delta\mathtt{t}^{\mathtt{max}}_{\mathtt{rpu}}], & \mathtt{if}\ (\mathtt{s}^9)_6 + \Delta\mathtt{t}^{\mathtt{max}}_{\mathtt{rpu}} \leq (\mathcal{F}_{\mathtt{S}^6}(\mathtt{a}, \mathtt{sa}))_2 \\ [1, (\mathcal{F}_{\mathtt{S}^6}(\mathtt{a}, \mathtt{sa}))_2 - (\mathtt{s}^9)_6], & \mathtt{if}\ (\mathtt{s}^9)_6 + \Delta\mathtt{t}^{\mathtt{min}}_{\mathtt{rpu}} > (\mathcal{F}_{\mathtt{S}^6}(\mathtt{a}, \mathtt{sa}))_2 \\ [\Delta\mathtt{t}^{\mathtt{min}}_{\mathtt{rpu}}, (\mathcal{F}_{\mathtt{S}^6}(\mathtt{a}, \mathtt{sa}))_2 - (\mathtt{s}^9)_6], & \mathtt{otherwise} \end{cases} \tag{3}$$

that the occupancy of any parking lot at each timestamp '$\mathtt{t}$' should not be more than the total available slots in the parking lot. The constraint $\mathcal{C}7$ defines the bounds on used variables.

IV. *DyPARK:* THE PROPOSED SCHEME

This section describes the formulation of both the stackelberg games and the *Dy*PARK PAS implemented to find the NE of both the games.

A. Scheme Description

The PUs send their requests using algorithm 1 which runs on their devices. Whereas, PCs forms decision using algorithm 2 which executes on their servers. A MLQ '$\mathtt{S}^{15}$' is implemented at each PC. In this queue, the level 1 and 2 stores RPUs' request ids. Whereas, level 3 and 4 store PPUs' request ids. The levels 1 and 3 are High-Priority Levels (HPLs) for RPUs and PPUs respectively. Whereas, the levels 2 and 4 are Low-Priority Levels (LPLs) for RPUs and PPUs respectively. The priority of levels meant for RPUs is higher than that of meant for PPUs in general. The priority of offline-advance, offline-immediate and online-immediate requests are high. Whereas, online-advance requests have low priority. Thus, at the request arrival timestamp, only online-advance requests are en-queued into the LPLs and rest all other types of reqests are en-queued into HPLs of the MLQ (algorithm 2: *lines 13-26*). PCs can delay the decision of high priority requests by compulsory game time duration, i.e., $\Delta\mathtt{t}_{\mathtt{gd}}^{\mathtt{cmp}}$ only. Whereas, if the requested parking slots are unavailable then the decision of low priority requests can be further delayed up to their respective soft decision deadline timestamp. Such timestamp can be calculated using method $\mathcal{M}^6:(\mathtt{pc},\mathtt{x},\gamma^{\mathtt{d}}) \rightarrow \mathtt{t}$ given in Eq. 10. The priority of the request changes from low to high at decision deadline timestamp using method $\mathcal{M}^8:(\mathtt{pc},\mathtt{k},\mathtt{i}) \rightarrow ((\mathcal{Z}(\mathtt{pc}))_7)_{\mathtt{k}-1}$ given in Eq. 11 (as mentioned in algorithm 2: *lines 3-11*), as shown at the bottom of the page. The priority defines allocation order of the requests.

$$\mathcal{M}^6:\left\{\left(\mathcal{F}_{(\mathcal{Z}(\mathtt{pc}))_1}(\mathtt{x})\right)_2 + \left\lfloor\left(\left(\mathcal{F}_{(\mathcal{Z}(\mathtt{pc}))_1}(\mathtt{x})\right)_5 - \left(\mathcal{F}_{(\mathcal{Z}(\mathtt{pc}))_1}(\mathtt{x})\right)_2\right)\times\gamma^{\mathtt{d}}\right\rfloor\right\} \tag{10}$$

B. Game Formulation

In this work, two *single-leader multiple-followers* Stackelberg games are formulated for the proposed *Dy*PARK PAS. One *single-leader multi-follower* Stackelberg game (*hereinafter referred to as 'G1'*) is played between PC and PPUs and second (*hereinafter referred to as 'G2'*) is played between PC and RPUs. There are 33 PCs located at level 2 as depicted in Fig. 2(b). Thus, both the games, i.e., '*G1*' and '*G2*', are played differently at each PC. Therefore, overall, 66 Stackelberg games are played. '*G1*' and '*G2*' are explained as follows

C. G1: Single-Leader Multi-Follower Stackelberg Game for Dynamic Pricing Based Parking Allocation

All the players, i.e., PC and PPUs play to increase their expected utilities. The expected utility of PC (when plays with PPUs) is the total expected revenue and can be calculated using method $\mathcal{M}^3:(\mathtt{pc},\mathtt{t},\mathtt{b}) \rightarrow \mathtt{r}$ given in Eq. 12, as shown at the bottom of the page. Whereas, the expected utility of each PPU is the remaining wallet amount if generated prices are assumed to be paid and it can be calculated using method $\mathcal{M}^4:(\mathtt{pc},\mathtt{x},\mathtt{t}) \rightarrow \mathtt{r}$ given in Eq. 13, as shown at the bottom of the page.

Various components of '*G1*' are as follows. *Players:* It is played between PC and PPUs where PC is the leader and PPUs are the followers. *Information:* The leader and followers have knowledge that opponent will observe their moves. *Moves:* In the game PC can play one out of two moves (i.e., $\mathtt{m}_1^1$ and $\mathtt{m}_2^1$) at a time. Let $\mathtt{m}_1^1$ and $\mathtt{m}_2^1$ represents "PC announces prices" and "PC doesn't announce prices" respectively. Whereas, each PPU can play one out of four moves (i.e., $\mathtt{m}_3^1$, $\mathtt{m}_4^1$, $\mathtt{m}_5^1$ and $\mathtt{m}_6^1$) at a time. Let $\mathtt{m}_3^1$, $\mathtt{m}_4^1$, $\mathtt{m}_5^1$ $\mathtt{m}_6^1$ represents "PPU accepts prices",

$$\mathcal{M}^8:\Big\{((\mathcal{Z}(\mathtt{pc}))_7)_{\mathtt{k}-1} \cup = \{((((\mathcal{Z}(\mathtt{pc}))_7)_{\mathtt{k}})_{\mathtt{i}}\} \mid \exists \mathtt{j}\ ((((\mathcal{Z}(\mathtt{pc}))_7)_{\mathtt{k}})_{\mathtt{i}} = (((\mathcal{Z}(\mathtt{pc}))_7)_{\mathtt{k}-1})_{\mathtt{j}}) \wedge (\mathcal{F}_{(\mathcal{Z}(\mathtt{pc}))_1}((((\mathcal{Z}(\mathtt{pc}))_7)_{\mathtt{k}-1})_{\mathtt{j}-1}))_2 \leq (\mathcal{F}_{(\mathcal{Z}(\mathtt{pc}))_1}((((\mathcal{Z}(\mathtt{pc}))_7)_{\mathtt{k}-1})_{\mathtt{j}}))_2 < (\mathcal{F}_{(\mathcal{Z}(\mathtt{pc}))_1}((((\mathcal{Z}(\mathtt{pc}))_7)_{\mathtt{k}-1})_{\mathtt{j}+1}))_2)\Big\} \tag{11}$$

$$\mathcal{M}^3:\begin{cases}\sum_{\mathtt{k}=3}^{4}\sum_{\mathtt{i}=1}^{|((\mathcal{Z}(\mathtt{pc}))_7)_{\mathtt{k}}|}\mathcal{F}_{(\mathcal{Z}(\mathtt{pc}))_3}((((\mathcal{Z}(\mathtt{pc}))_7)_{\mathtt{k}})_{\mathtt{i}},\mathtt{t}) \times \lceil * \rceil \dfrac{(\mathcal{F}_{(\mathcal{Z}(\mathtt{pc}))_1}((((\mathcal{Z}(\mathtt{pc}))_7)_{\mathtt{k}})_{\mathtt{i}}))_6}{3600} & \mathtt{if}\ \mathtt{b}=1\\ \sum_{\mathtt{k}=1}^{2}\sum_{\mathtt{i}=1}^{|((\mathcal{Z}(\mathtt{pc}))_7)_{\mathtt{k}}|}\mathcal{F}_{(\mathcal{Z}(\mathtt{pc}))_3}((((\mathcal{Z}(\mathtt{pc}))_7)_{\mathtt{k}})_{\mathtt{i}},\mathtt{t}) & \mathtt{if}\ \mathtt{b}=0\end{cases} \tag{12}$$

$$\mathcal{M}^4:\begin{cases}\mathcal{F}_{\mathtt{S}^2}((\mathcal{F}_{(\mathcal{Z}(\mathtt{pc}))_1}(\mathtt{x}))_1) - \mathcal{F}_{(\mathcal{Z}(\mathtt{pc}))_3}(\mathtt{x},\mathtt{t}) & \mathtt{if}\ \mathcal{F}_{\mathtt{S}^1}((\mathcal{F}_{(\mathcal{Z}(\mathtt{pc}))_1}(\mathtt{x}))_1) = 1\\ \mathcal{F}_{\mathtt{S}^2}((\mathcal{F}_{(\mathcal{Z}(\mathtt{pc}))_1}(\mathtt{x}))_1) + \mathcal{F}_{(\mathcal{Z}(\mathtt{pc}))_3}(\mathtt{x},\mathtt{t}) & \mathtt{if}\ \mathcal{F}_{\mathtt{S}^1}((\mathcal{F}_{(\mathcal{Z}(\mathtt{pc}))_1}(\mathtt{x}))_1) = 0\end{cases} \tag{13}$$

"PPU rejects prices", "PPU accepts waiting for next round", and "PPU rejects waiting for next round" respectively. *Payoffs:* The PC and PPUs get payoffs when they play their moves. Let function $\mathcal{U}^1_{\mathtt{pc}}(\{\mathtt{m}^1_{\mathtt{p}}, \mathtt{m}^1_{\mathtt{q}}\})$ determines the payoff value for leader (i.e., PC) and $\mathcal{U}^1_{\mathtt{x}}(\{\mathtt{m}^1_{\mathtt{p}}, \mathtt{m}^1_{\mathtt{q}}\})$ determines the payoff value for follower (i.e., PPU who have sent request having id 'x') when leader moves by using move $\mathtt{m}^1_{\mathtt{p}}$ and follower moves by using move $\mathtt{m}^1_{\mathtt{q}}$. If PC expects more revenue at timestamp 't' than at 't-1' then $\mathcal{U}^1_{\mathtt{pc}}(\{\mathtt{m}^1_1, \mathtt{m}^1_3\})$ and $\mathcal{U}^1_{\mathtt{pc}}(\{\mathtt{m}^1_1, \mathtt{m}^1_4\})$ are equal to the expected prices at timestamp 't' and 't-1' respectively otherwise they are equal to '0'. If the request arrives at timestamp 't' and PC expects more revenue at timestamp 't' than at 't-1' then only $\mathcal{U}^1_{\mathtt{pc}}(\{\mathtt{m}^1_1, \mathtt{m}^1_4\})$ is equal to 'τ'. $\mathcal{U}^1_{\mathtt{x}}(\{\mathtt{m}^1_1, \mathtt{m}^1_3\})$ and $\mathcal{U}^1_{\mathtt{x}}(\{\mathtt{m}^1_1, \mathtt{m}^1_4\})$ are equal to the remaining wallet amount if generated prices at timestamp 't' and 't-1' are assumed to be paid. If the generated prices at timestamps 't-1' and 't' for any request having id 'x' are same then $\mathcal{U}^1_{\mathtt{x}}(\{\mathtt{m}^1_1, \mathtt{m}^1_3\})$ is additionally 'τ' lesser than the remaining wallet amount if generated prices at timestamp 't' are assumed to be paid. If timestamp 't' is not the hard decision deadline for request having id 'x' then $\mathcal{U}^1_{\mathtt{pc}}(\{\mathtt{m}^1_2, \mathtt{m}^1_5\})$ and $\mathcal{U}^1_{\mathtt{x}}(\{\mathtt{m}^1_2, \mathtt{m}^1_5\})$ are equal to 'α' otherwise they will be equal to '0'. Similarly, if timestamp 't' is not the hard decision deadline for request having id 'x' then $\mathcal{U}^1_{\mathtt{pc}}(\{\mathtt{m}^1_2, \mathtt{m}^1_6\})$ and $\mathcal{U}^1_{\mathtt{x}}(\{\mathtt{m}^1_2, \mathtt{m}^1_6\})$ are equal to '0' otherwise they will be equal to 'α'. *Strategies:* Let $\Gamma^1_{\mathtt{pc}}$, $\Gamma^1_{\mathtt{x}_1}$, $\Gamma^1_{\mathtt{x}_2}$,....,$\Gamma^1_{\mathtt{x}_n}$ are the set of strategies for PC and all PPUs who sent requests having ids as '$\mathtt{x}_1$', '$\mathtt{x}_2$',...., and '$\mathtt{x}_n$' respectively. $\Gamma^1_{\mathtt{pc}} = \{\mathtt{m}^1_1, \mathtt{m}^1_2\}$ and $\Gamma^1_{\mathtt{x}_i} = \{\mathtt{m}^1_3, \mathtt{m}^1_4, \mathtt{m}^1_5, \mathtt{m}^1_6\}$. The objective here is to find $\gamma^{1*}_{\mathtt{pc}}$ and all $\gamma^{1*}_{\mathtt{x}_i}$ using Eqs. 14 and 15 respectively where $\mathtt{i} \in [1, \mathtt{n}]$ and $\mathtt{x}_i \in [\mathtt{x}_1, \mathtt{x}_n]$. Thus, $\{\gamma^{1*}_{\mathtt{pc}}, \gamma^{1*}_{\mathtt{x}_1}, \gamma^{1*}_{\mathtt{x}_2}, \ldots, \gamma^{1*}_{\mathtt{x}_n}\}$ is a solution at NE.

PPU designs its request (algorithm 1: *line 1*). If amount is there in the PPU's wallet then request is forwarded to the appropriate PC (algorithm 1: *lines 6-7*). Thereafter, PC generates the parking prices for each undecided request (algorithm 2: *line 28*). These prices have been generated using algorithm 4 by taking actual and predicted occupancy into consideration. A random forest method $\mathcal{M}^9 : (\mathtt{t}, \mathtt{a}, \mathtt{bf}, \mathtt{ss}) \rightarrow \mathtt{n}$ is used to predict occupancy. Then, in order to solve this game Backward Induction (BI) method is used. According to BI method, the PPU determine his best moves, i.e. one each for all possible moves of the PC (algorithm 1: *lines 9-19*). Then, PC is informed about such moves (algorithm 1: *line 20*). Afterwards, PC finds its best move based on the moves submitted by all the PPUs (algorithm 2: *lines 29-40*). If all such moves are same then finally, PC plays that move and based on which it will either announce the prices or not to the PPUs (algorithm 2: *lines 41-43*). Thereafter, PPU checks whether PC has announced the prices or not, (algorithm 1: *line 21*). If PC has announced the prices then PPU plays its best move based on which it will either accept or reject the prices (algorithm 1: *lines 22-26*). On the other hand if PC has not announced the prices then also, PPU plays its best move and based on which it will either accept or reject waiting for next announcement (algorithm 1: *lines 28-32*). PPUs keep on checking the request decision until they are decided (algorithm 1: *lines 34-42*).

Algorithm 1 *DyPARK* PAS (for PUs)

Input: $\mathtt{A}, \mathtt{SA}, \mathtt{BF}, \mathtt{SS}, \mathtt{T}, \mathtt{X}, \mathtt{PU}, \mathtt{PS}, \mathtt{N}, \mathtt{B}, \Delta\mathtt{t}^{\min}_{\mathtt{rpu}}$

Output: $\mathtt{s}^9_{\mathtt{n}}, \mathcal{F}_{(\mathcal{Z}(\mathcal{F}_{\mathtt{S}^5}((\mathtt{s}^9_{\mathtt{n}})_4)))_5}((\mathtt{s}^9_{\mathtt{n}})_1)$

1: $\mathtt{s}^9_{\mathtt{n}}$ = new_Request();
2: $\mathtt{b} = \mathcal{F}_{\mathtt{S}^1}((\mathtt{s}^9_{\mathtt{n}})_2)$, $\mathtt{x} = (\mathtt{s}^9_{\mathtt{n}})_1$;
3: **if** $\mathtt{b} == 0$ && $\mathcal{F}_{\mathtt{S}^2}((\mathtt{s}^9_{\mathtt{n}})_2) > 0$ && $(\mathtt{s}^9_{\mathtt{n}})_7 < \Delta\mathtt{t}^{\min}_{\mathtt{rpu}}$ **then**
4: $\left(\mathtt{S}^2 \setminus= \{(\mathtt{s}^9_{\mathtt{n}})_2, \Delta\mathtt{t}^{\min}_{\mathtt{rpu}}\}\right) \cup= \{(\mathtt{s}^9_{\mathtt{n}})_2, (\mathtt{s}^9_{\mathtt{n}})_7\}$;
5: **end if**
6: **if** $\mathcal{F}_{\mathtt{S}^2}((\mathtt{s}^9_{\mathtt{n}})_2) > 0$ **then**
7: $(\mathcal{Z}(\mathcal{F}_{\mathtt{S}^5}((\mathtt{s}^9_{\mathtt{n}})_4)))_1 \cup= \mathtt{s}^9_{\mathtt{n}}$;
8: **while** $\mathtt{t} \geq (\mathcal{F}_{(\mathcal{Z}(\mathcal{F}_{\mathtt{S}^5}((\mathtt{s}^9_{\mathtt{n}})_4)))_1}((\mathtt{s}^9_{\mathtt{n}})_1))_2$ **do**
9: $\mathtt{Y} = \{\mathtt{x}, \mathtt{t}\}$;
10: **if** $(\mathcal{U}^{\mathtt{b}}_{\mathtt{x}}(\mathtt{m}^{\mathtt{b}}_1, \mathtt{m}^{\mathtt{b}}_3) > \mathcal{U}^{\mathtt{b}}_{\mathtt{x}}(\mathtt{m}^{\mathtt{b}}_1, \mathtt{m}^{\mathtt{b}}_4))$ **then**
11: $\mathtt{Y} \cup= \{\mathtt{m}^{\mathtt{b}}_3\}$;
12: **else**
13: $\mathtt{Y} \cup= \{\mathtt{m}^{\mathtt{b}}_4\}$;
14: **end if**
15: **if** $(\mathcal{U}^{\mathtt{b}}_{\mathtt{x}}(\mathtt{m}^{\mathtt{b}}_2, \mathtt{m}^{\mathtt{b}}_5) > \mathcal{U}^{\mathtt{b}}_{\mathtt{x}}(\mathtt{m}^{\mathtt{b}}_2, \mathtt{m}^{\mathtt{b}}_6))$ **then**
16: $\mathtt{Y} \cup= \{\mathtt{m}^{\mathtt{b}}_5\}$;
17: **else**
18: $\mathtt{Y} \cup= \{\mathtt{m}^{\mathtt{b}}_6\}$;
19: **end if**
20: $(\mathcal{Z}(\mathcal{F}_{\mathtt{S}^5}((\mathtt{s}^9_{\mathtt{n}})_4)))_8 \cup= \mathtt{Y}$;
21: **if** $\left(\mathcal{F}_{(\mathcal{Z}(\mathcal{F}_{\mathtt{S}^5}((\mathtt{s}^9_{\mathtt{n}})_4)))_2}(\mathtt{t})\right)_{(\mathcal{F}_{\mathtt{S}^1}((\mathtt{s}^9_{\mathtt{n}})_2)+1)} == 1$ **then**
22: **if** $((\mathtt{Y})_3 == \{\mathtt{m}^{\mathtt{b}}_3\})$ **then**
23: $(\mathcal{Z}(\mathcal{F}_{\mathtt{S}^5}((\mathtt{s}^9_{\mathtt{n}})_4)))_4 \cup= \{\mathtt{x}, \mathtt{t}, 1\}$;
24: **else**
25: $(\mathcal{Z}(\mathcal{F}_{\mathtt{S}^5}((\mathtt{s}^9_{\mathtt{n}})_4)))_4 \cup= \{\mathtt{x}, \mathtt{t}, 0\}$;
26: **end if**
27: **else**
28: **if** $((\mathtt{Y})_4 == \{\mathtt{m}^{\mathtt{b}}_5\})$ **then**
29: $(\mathcal{Z}(\mathcal{F}_{\mathtt{S}^5}((\mathtt{s}^9_{\mathtt{n}})_4)))_4 \cup= \{\mathtt{x}, \mathtt{t}, 1\}$;
30: **else**
31: $(\mathcal{Z}(\mathcal{F}_{\mathtt{S}^5}((\mathtt{s}^9_{\mathtt{n}})_4)))_4 \cup= \{\mathtt{x}, \mathtt{t}, 0\}$;
32: **end if**
33: **end if**
34: **if** $\left(\mathcal{F}_{(\mathcal{Z}(\mathcal{F}_{\mathtt{S}^5}((\mathtt{s}^9_{\mathtt{n}})_4)))_5}(\mathtt{x})\right)_1 == 0$ **then**
35: display("Request is pending");
36: **else if** $\left(\mathcal{F}_{(\mathcal{Z}(\mathcal{F}_{\mathtt{S}^5}((\mathtt{s}^9_{\mathtt{n}})_4)))_5}(\mathtt{x})\right)_1 == -1$ **then**
37: display("Request decision: Rejected, ↪ Details: $\mathtt{s}^9_{\mathtt{n}}, \mathcal{F}_{(\mathcal{Z}(\mathcal{F}_{\mathtt{S}^5}((\mathtt{s}^9_{\mathtt{n}})_4)))_5}(\mathtt{x})$");
38: break;
39: **else if** $\left(\mathcal{F}_{(\mathcal{Z}(\mathcal{F}_{\mathtt{S}^5}((\mathtt{s}^9_{\mathtt{n}})_4)))_5}(\mathtt{x})\right)_1 == 1$ **then**
40: display("Request decision: Accepted, ↪ Details: $\mathtt{s}^9_{\mathtt{n}}, \mathcal{F}_{(\mathcal{Z}(\mathcal{F}_{\mathtt{S}^5}((\mathtt{s}^9_{\mathtt{n}})_4)))_5}(\mathtt{x})$");
41: break;
42: **end if**
43: **end while**
44: **else**
45: display("Insufficient balance");
46: **end if**

$$\gamma^{1*}_{\mathtt{pc}} = \underset{\gamma^1_{\mathtt{pc}} \in \Gamma^1_{\mathtt{pc}}}{\arg\max}\, \mathcal{U}^1_{\mathtt{pc}}(\mathtt{B}),\ \mathtt{B} = \{\gamma^1_{\mathtt{pc}}\} \cup \{\mathcal{R}^{1*}_{\mathtt{x}_k}(\gamma^{1*}_{\mathtt{pc}}, \ldots, \gamma^{1*}_{\mathtt{x}_{i-1}}, \gamma^{1*}_{\mathtt{x}_{i+1}}, \ldots) | \forall \mathtt{k} \in [1, \mathtt{n}] \wedge \mathtt{i}{=}\mathtt{k} \wedge \mathtt{x}_i \in [\mathtt{x}_1, \mathtt{x}_n]\} \quad (14)$$

$$\gamma^{1*}_{\mathtt{x}_i} = \underset{\gamma^1_{\mathtt{x}_i} \in \Gamma^1_{\mathtt{x}_i}}{\arg\max}\, \mathcal{U}^1_{\mathtt{x}_i}(\{\gamma^{1*}_{\mathtt{pc}}, \ldots, \gamma^{1*}_{\mathtt{x}_{i-1}}, \gamma^1_{\mathtt{x}_i}, \gamma^{1*}_{\mathtt{x}_{i+1}}, \ldots\}),\ \mathtt{i} \in [1, \mathtt{n}] \wedge \mathtt{x}_i \in [\mathtt{x}_1, \mathtt{x}_n] \quad (15)$$

Algorithm 2 *DyPARK* PAS (for PCs)

Input: $\mathtt{pc}, \mathtt{t}, \rho^{\mathtt{f}}, \eta^{\circ}, \eta^{\mathtt{f}}, \eta^{\mathtt{d}}, \gamma^{\mathtt{d}}, \Delta\mathtt{t}^{\mathtt{cmp}}_{\mathtt{gd}}, \Delta\mathtt{t}^{\mathtt{min}}_{\mathtt{rpu}}$
Output: $\mathtt{S}^2, (\mathcal{Z}(\mathtt{pc}))_2, (\mathcal{Z}(\mathtt{pc}))_3, (\mathcal{Z}(\mathtt{pc}))_5, (\mathcal{Z}(\mathtt{pc}))_6, (\mathcal{Z}(\mathtt{pc}))_7, (\mathcal{Z}(\mathtt{pc}))_8$

1: **while** t **do**
2: **for** (k = 2; k ≤ 4; k = k+2) **do**
3: i = 1;
4: **while** i ≤ | $((\mathcal{Z}(\mathtt{pc}))_7)_{\mathtt{k}}$ | **do**
5: **if** t == $\mathcal{M}^6(\mathtt{pc}, (((\mathcal{Z}(\mathtt{pc}))_7)_{\mathtt{k}})_{\mathtt{i}}, \gamma^{\mathtt{d}})$ **then**
6: $((\mathcal{Z}(\mathtt{pc}))_7)_{\mathtt{k}-1} = \mathcal{M}^8(\mathtt{pc}, \mathtt{k}, \mathtt{i})$;
7: $((\mathcal{Z}(\mathtt{pc}))_7)_{\mathtt{k}} \setminus = \{(((\mathcal{Z}(\mathtt{pc}))_7)_{\mathtt{k}})_{\mathtt{i}}\}$;
8: **else**
9: i++;
10: **end if**
11: **end while**
12: **end for**
13: **for** i ← | $(\mathcal{Z}(\mathtt{pc}))_1$ | to 1 **do**
14: **if** $(((\mathcal{Z}(\mathtt{pc}))_1)_{\mathtt{i}})_3$ == t **then**
15: j = $((((\mathcal{Z}(\mathtt{pc}))_1)_{\mathtt{i}})_8$ == 1) && $((((\mathcal{Z}(\mathtt{pc}))_1)_{\mathtt{i}})_9$ == 1);
16: **if** $\mathcal{F}_{\mathtt{S}^1}((((\mathcal{Z}(\mathtt{pc}))_1)_{\mathtt{i}})_2)$ == 0 **then**
17: $((\mathcal{Z}(\mathtt{pc}))_7)_{\mathtt{j}+1} \cup = \{(((\mathcal{Z}(\mathtt{pc}))_1)_{\mathtt{i}})_1\}$;
18: **else**
19: $((\mathcal{Z}(\mathtt{pc}))_7)_{\mathtt{j}+3} \cup = \{(((\mathcal{Z}(\mathtt{pc}))_1)_{\mathtt{i}})_1\}$;
20: **end if**
21: **else**
22: break;
23: **end if**
24: $(\mathcal{Z}(\mathtt{pc}))_5 \cup = \{(((\mathcal{Z}(\mathtt{pc}))_1)_{\mathtt{i}})_1, 0, \{\}, \{\}, \{\}, \{\}\}$;
25: $(\mathcal{Z}(\mathtt{pc}))_3 \cup = \{(((\mathcal{Z}(\mathtt{pc}))_1)_{\mathtt{i}})_1, \mathtt{t}-1, !(\mathcal{F}_{\mathtt{S}^1}((((\mathcal{Z}(\mathtt{pc}))_1)_{\mathtt{i}})_2))*$ $((((\mathcal{Z}(\mathtt{pc}))_1)_{\mathtt{i}})_7 - \mathcal{F}_{\mathtt{S}^2}((((\mathcal{Z}(\mathtt{pc}))_1)_{\mathtt{i}})_2))$;
26: **end for**
27: *generate_Duration*(pc,t,1,2,$\rho^{\mathtt{f}}, \eta^{\mathtt{d}}, \Delta\mathtt{t}^{\mathtt{min}}_{\mathtt{rpu}}$);
28: *generate_Prices*(pc,t,3,4,$\rho^{\mathtt{f}}, \eta^{\circ}, \eta^{\mathtt{f}}$);
29: **for** (k = 1; k ≤ 4; k = k+1) **do**
30: i = 1;
31: **while** i ≤ | $((\mathcal{Z}(\mathtt{pc}))_7)_{\mathtt{k}}$ | **do**
32: b = $\mathcal{F}_{\mathtt{S}^1}((\mathcal{F}_{(\mathcal{Z}(\mathtt{pc}))_1}((((\mathcal{Z}(\mathtt{pc}))_7)_{\mathtt{k}})_{\mathtt{i}}))_1)$
33: C = $\mathcal{F}_{(\mathcal{Z}(\mathtt{pc}))_8}((((\mathcal{Z}(\mathtt{pc}))_7)_{\mathtt{k}})_{\mathtt{i}}, \mathtt{t})$
34: **if** $(\mathcal{U}^{\mathtt{b}}_{\mathtt{pc}}(\mathtt{m}^{\mathtt{b}}_1, (\mathtt{C})_1)$ > $(\mathcal{U}^{\mathtt{b}}_{\mathtt{pc}}(\mathtt{m}^{\mathtt{b}}_2, (\mathtt{C})_2)$ **then**
35: $\mathtt{Yb} \cup = \{\mathtt{m}^{\mathtt{b}}_1\}$;
36: **else**
37: $\mathtt{Yb} \cup = \{\mathtt{m}^{\mathtt{b}}_2\}$;
38: **end if**
39: **end while**
40: **end for**
41: **if** | Y0 | == 1 && | Y1 | == 1 **then**
42: $(\mathcal{Z}(\mathtt{pc}))_2 \cup = \{\mathtt{t}, (\mathtt{Y1})_1 == \mathtt{m}^{\mathtt{b}}_1, (\mathtt{Y0})_1 == \mathtt{m}^{\mathtt{b}}_1\}$;
43: **end if**
44: *allocate_Park*(pc,t,$(\mathcal{F}_{(\mathcal{Z}(\mathtt{pc}))_2}(\mathtt{t}))_1$,1,2,$\gamma^{\mathtt{d}}$, $\Delta\mathtt{t}^{\mathtt{cmp}}_{\mathtt{gd}}, \Delta\mathtt{t}^{\mathtt{min}}_{\mathtt{rpu}}$);
45: *allocate_Park*(pc,t,$(\mathcal{F}_{(\mathcal{Z}(\mathtt{pc}))_2}(\mathtt{t}))_2$,3,4,$\gamma^{\mathtt{d}}$, $\Delta\mathtt{t}^{\mathtt{cmp}}_{\mathtt{gd}}, \Delta\mathtt{t}^{\mathtt{min}}_{\mathtt{rpu}}$);
46: **end while**

D. G2: Single-Leader Multi-Follower Stackelberg Game for Restricted Parking Allocation

All the players, i.e., PC and RPUs play to increase their expected utilities. The expected utility of PC (when plays with RPUs) is the total expected loss of parking duration and can be calculated using method $\mathcal{M}^3 : (\mathtt{pc}, \mathtt{t}, \mathtt{b}) \rightarrow \mathtt{r}$ given in Eq. 12. Whereas, the expected utility of each RPU is the sum of wallet and generated FoC duration if it is assumed to be accepted. It can be calculated using method $\mathcal{M}^4 : (\mathtt{pc}, \mathtt{x}, \mathtt{t}) \rightarrow \mathtt{r}$ given in Eq. 13. Various components of '*G2*' are as follows. *Players:* It is played between PC and RPUs where PC is the leader and RPUs are the followers. *Information:* The leader and followers have knowledge that opponent will observe their moves. *Moves:* In the game, PC can play one out of two moves (i.e., $\mathtt{m}^0_1$ and $\mathtt{m}^0_2$) at a time. Let $\mathtt{m}^0_1$ and $\mathtt{m}^0_2$ represents "PC announces FoC durations" and "PC doesn't announce FoC durations" respectively. Whereas, each RPU can play one out of four moves (i.e., $\mathtt{m}^0_3$, $\mathtt{m}^0_4$, $\mathtt{m}^0_5$ and $\mathtt{m}^0_6$) at a time. Let $\mathtt{m}^0_3$, $\mathtt{m}^0_4$, $\mathtt{m}^0_5$ $\mathtt{m}^0_6$ represents "RPU accepts FoC duration", "RPU rejects FoC duration", "RPU accepts waiting for next round", and "RPU rejects waiting for next round" respectively. *Payoffs:*The PC and RPUs get payoffs when they play their moves. Let function $\mathcal{U}^0_{\mathtt{pc}}(\{\mathtt{m}^0_{\mathtt{p}}, \mathtt{m}^0_{\mathtt{q}}\})$ determines the payoff value for leader (i.e., PC) and $\mathcal{U}^0_{\mathtt{x}}(\{\mathtt{m}^0_{\mathtt{p}}, \mathtt{m}^0_{\mathtt{q}}\})$ determines the payoff value for follower (i.e., RPU who have sent request having id 'x') when leader moves by using move $\mathtt{m}^0_{\mathtt{p}}$ and follower moves by using move $\mathtt{m}^0_{\mathtt{q}}$. If PC expects less FoC parking duration loss at timestamp 't' than at 't-1' then $\mathcal{U}^0_{\mathtt{pc}}(\{\mathtt{m}^0_1, \mathtt{m}^0_3\})$ and $\mathcal{U}^0_{\mathtt{pc}}(\{\mathtt{m}^0_1, \mathtt{m}^0_4\})$ are equal to '0' otherwise they are equal to the negative of expected FoC parking duration loss at timestamp 't' and 't-1' respectively. If the requested FoC parking duration is already there in sender's wallet and PC do not expect less FoC parking duration loss at timestamp 't' than at 't-1' then only $\mathcal{U}^0_{\mathtt{pc}}(\{\mathtt{m}^0_1, \mathtt{m}^0_3\})$ and $\mathcal{U}^0_{\mathtt{pc}}(\{\mathtt{m}^0_1, \mathtt{m}^0_4\})$ are equal to the negative of 'β'. $\mathcal{U}^0_{\mathtt{x}}(\{\mathtt{m}^0_1, \mathtt{m}^0_3\})$ and $\mathcal{U}^0_{\mathtt{x}}(\{\mathtt{m}^0_1, \mathtt{m}^0_4\})$ are equal to the sum of generated FoC parking duration and one that is present in the wallet at timestamp 't' and 't-1' respectively. Further, $\mathcal{U}^0_{\mathtt{x}}(\{\mathtt{m}^0_1, \mathtt{m}^0_3\})$ is β more than its normal value if requested FoC parking duration is already there in the sender's wallet. Similarly, $\mathcal{U}^0_{\mathtt{x}}(\{\mathtt{m}^0_1, \mathtt{m}^0_4\})$ is τ more than its normal value if requested FoC parking duration is not equal to the one present in the sender's wallet and if generated FoC parking durations at timestamp 't' and 't-1' are same. If timestamp 't' is not the hard decision deadline for request having id 'x' then $\mathcal{U}^0_{\mathtt{pc}}(\{\mathtt{m}^0_2, \mathtt{m}^0_5\})$ and $\mathcal{U}^0_{\mathtt{x}}(\{\mathtt{m}^0_2, \mathtt{m}^0_5\})$ are equal to negative of 'α' and 'α' respectively. Otherwise they both will be equal to '0'. Similarly, if timestamp 't' is not the hard decision deadline for request having id 'x' then both $\mathcal{U}^0_{\mathtt{pc}}(\{\mathtt{m}^0_2, \mathtt{m}^0_6\})$ and $\mathcal{U}^0_{\mathtt{x}}(\{\mathtt{m}^0_2, \mathtt{m}^0_6\})$ are equal to '0'. Otherwise they will be equal to negative of 'α' and 'α' respectively. *Strategies:* Let $\Gamma^0_{\mathtt{pc}}$, $\Gamma^0_{\mathtt{x}_1}$, $\Gamma^0_{\mathtt{x}_2}$,....,$\Gamma^0_{\mathtt{x}_n}$ are the set of strategies for PC and all RPUs who sent requests having ids as '$\mathtt{x}_1$', '$\mathtt{x}_2$',...., and, '$\mathtt{x}_n$' respectively. $\Gamma^0_{\mathtt{pc}} = \{\mathtt{m}^0_1, \mathtt{m}^0_2\}$ and $\Gamma^0_{\mathtt{x}_i} = \{\mathtt{m}^0_3, \mathtt{m}^0_4, \mathtt{m}^0_5, \mathtt{m}^0_6\}$. The objective here is to find $\gamma^{0*}_{\mathtt{pc}}$ and all $\gamma^{0*}_{\mathtt{x}_i}$ using Eqs. 16 and 17 respectively where $\mathtt{i} \in [1, \mathtt{n}]$ and $\mathtt{x}_i \in [\mathtt{x}_1, \mathtt{x}_n]$. Thus, $\{\gamma^{0*}_{\mathtt{pc}}, \gamma^{0*}_{\mathtt{x}_1}, \gamma^{0*}_{\mathtt{x}_2}, \ldots., \gamma^{0*}_{\mathtt{x}_n}\}$ is a solution at NE.

RPU designs its request (algorithm 1: *line 1*). If the RPU has not availed the FoC parking and requested duration of stay is less than $\Delta\mathtt{t}^{\mathtt{min}}_{\mathtt{rpu}}$ then wallet duration is updated with the requested duration (algorithm 1: *lines 3-5*). If threshold FoC duration is there in the RPU's wallet then request is forwarded to the appropriate PC (algorithm 1: *lines 6-7*). Thereafter, PC generates the FoC parking duration for each undecided request (algorithm 2: *line 27*). These FoC durations are generated fairly using algorithm 3 by considering RPU's utility, wallet value, and requests dynamics. Then, in order to solve this game BI method is used. According to it, the RPU determine his best moves, i.e., one each for all possible moves of the PC (algorithm 1: *lines 9-19*). Then, PC is

Algorithm 3 generate_Duration

Input: $\mathtt{pc}, \mathtt{t}, \mathtt{l}_1, \mathtt{l}_2, \rho^{\mathtt{f}}, \eta^{\mathtt{d}}, \Delta \mathtt{t}^{\mathtt{min}}_{\mathtt{rpu}}$
Output: $(\mathcal{Z}(\mathtt{pc}))_3$

for $\mathtt{i} = \mathtt{l}_1 : \mathtt{l}_2$ **do**
  $\mathtt{j} = 1;$
  while $\mathtt{j} \leq |((\mathcal{Z}(\mathtt{pc}))_7)_{\mathtt{i}}|$ **do**
    $\mathtt{Y} = \mathcal{F}_{(\mathcal{Z}(\mathtt{pc}))_1}((((\mathcal{Z}(\mathtt{pc}))_7)_{\mathtt{i}})_{\mathtt{j}});$
    if $\mathcal{F}_{\mathtt{S}^2}((\mathtt{Y})_1) == \Delta \mathtt{t}^{\mathtt{min}}_{\mathtt{rpu}}$ **then**
      if $\mathtt{rand}(0,1) \leq \rho^{\mathtt{f}}$ **then**
        $\mathtt{r}_{\mathtt{min}} = 0;$
        if $\mathcal{M}^4(\mathtt{pc}, (((\mathcal{Z}(\mathtt{pc}))_7)_{\mathtt{i}})_{\mathtt{j}}, \mathtt{t}\text{-}1) == (\mathtt{Y})_6$ **then**
          $\mathtt{r}_{\mathtt{max}} = \mathcal{M}^4(\mathtt{pc}, (((\mathcal{Z}(\mathtt{pc}))_7)_{\mathtt{i}})_{\mathtt{j}}, \mathtt{t}\text{-}1) \hookrightarrow -\mathcal{F}_{\mathtt{S}^2}((\mathtt{Y})_1) - \eta^{\mathtt{d}};$
        else
          $\mathtt{r}_{\mathtt{max}} = \mathcal{M}^4(\mathtt{pc}, (((\mathcal{Z}(\mathtt{pc}))_7)_{\mathtt{i}})_{\mathtt{j}}, \mathtt{t}\text{-}1) \hookrightarrow -\mathcal{F}_{\mathtt{S}^2}((\mathtt{Y})_1);$
        end if
      else
        if $\mathcal{M}^4(\mathtt{pc}, (((\mathcal{Z}(\mathtt{pc}))_7)_{\mathtt{i}})_{\mathtt{j}}, \mathtt{t}\text{-}1) == (\mathtt{Y})_6$ **then**
          $\mathtt{r}_{\mathtt{min}} = 0;$
          $\mathtt{r}_{\mathtt{max}} = \mathcal{M}^4(\mathtt{pc}, (((\mathcal{Z}(\mathtt{pc}))_7)_{\mathtt{i}})_{\mathtt{j}}, \mathtt{t}\text{-}1) \hookrightarrow -\mathcal{F}_{\mathtt{S}^2}((\mathtt{Y})_1) - \eta^{d};$
        else
          $\mathtt{r}_{\mathtt{min}} = \mathcal{M}^4(\mathtt{pc}, (((\mathcal{Z}(\mathtt{pc}))_7)_{\mathtt{i}})_{\mathtt{j}}, \mathtt{t}\text{-}1) \hookrightarrow -\mathcal{F}_{\mathtt{S}^2}((\mathtt{Y})_1) + \eta^{d};$
          $\mathtt{r}_{\mathtt{max}} = (\mathtt{Y})_6 - \mathcal{F}_{\mathtt{S}^2}((\mathtt{Y})_1);$
        end if
      end if
    else
      $\mathtt{r}_{\mathtt{min}} = \mathtt{r}_{\mathtt{max}} = 0;$
    end if
    $\mathtt{n}_{\mathtt{d}} = \mathtt{randi}[\mathtt{r}_{\mathtt{min}}, \mathtt{r}_{\mathtt{max}}];$
    $(\mathcal{Z}(\mathtt{pc}))_3 \cup = \{((((\mathcal{Z}(\mathtt{pc}))_7)_{\mathtt{i}})_{\mathtt{j}}, \mathtt{t}, \mathtt{n}_{\mathtt{d}}\};$
    $\mathtt{j}{+}{+};$
  end while
end for

Algorithm 4 generate_Prices

Input: $\mathtt{pc}, \mathtt{t}, \mathtt{l}_3, \mathtt{l}_4, \rho^{\mathtt{f}}, \eta^{\mathtt{o}}, \eta^{\mathtt{f}}$
Output: $(\mathcal{Z}(\mathtt{pc}))_3$

for $\mathtt{i} = \mathtt{l}_3 : \mathtt{l}_4$ **do**
  $\mathtt{j} = 1;$
  while $\mathtt{j} \leq |((\mathcal{Z}(\mathtt{pc}))_7)_{\mathtt{i}}|$ **do**
    $\mathtt{Y} = \mathcal{F}_{(\mathcal{Z}(\mathtt{pc}))_1}((((\mathcal{Z}(\mathtt{pc}))_7)_{\mathtt{i}})_{\mathtt{j}});$
    $\mathtt{n}_{\mathtt{a}} = \mathcal{M}^2(\mathtt{pc}, (\mathtt{Y})_3, (\mathtt{Y})_5, (\mathtt{Y})_5 + (\mathtt{Y})_6);$
    $\mathtt{n}_{\mathtt{p}} = \mathcal{M}^9((\mathtt{Y})_5, \mathcal{F}^{-1}_{\mathtt{S}^3}((\mathtt{Y})_3)_1, \mathcal{F}^{-1}_{\mathtt{S}^3}((\mathtt{Y})_3)_3, \hookrightarrow \mathcal{F}^{-1}_{\mathtt{S}^3}((\mathtt{Y})_3)_4);$
    if $\mathtt{n}_{\mathtt{a}} \geq \mathtt{n}_{\mathtt{p}}$ **then**
      $\mathtt{n}_{\mathtt{c}} = \mathtt{n}_{\mathtt{a}};$
    else
      $\mathtt{n}_{\mathtt{c}} = \mathtt{n}_{\mathtt{p}};$
    end if
    if $\mathtt{n}_{\mathtt{c}} \leq |\mathcal{F}_{\mathtt{S}^4}((\mathtt{Y})_3)| - \mathtt{n}_{\mathtt{c}}$ **then**
      $\mathtt{n}_{\mathtt{d}} = \mathtt{n}_{\mathtt{c}};$
    else
      $\mathtt{n}_{\mathtt{d}} = |\mathcal{F}_{\mathtt{S}^4}((\mathtt{Y})_3)| - \mathtt{n}_{\mathtt{c}};$
    end if
    if $\mathtt{rand}(0,1) \leq \rho^{\mathtt{f}}$ **then**
      $\mathtt{n}_{\mathtt{f}} = \mathtt{randi}[(\mathtt{n}_{\mathtt{c}} - \mathtt{n}_{\mathtt{d}}), \mathtt{n}_{\mathtt{c}}];$
    else
      $\mathtt{n}_{\mathtt{f}} = \mathtt{randi}[(\mathtt{n}_{\mathtt{c}} + \eta^{\mathtt{o}}), (\mathtt{n}_{\mathtt{c}} + \eta^{\mathtt{o}} + \mathtt{n}_{\mathtt{d}})];$
    end if
    $\mathtt{p}_1 = \mathcal{F}_{\mathtt{S}^8}((\mathcal{F}^{-1}_{\mathtt{S}^3}((\mathtt{Y})_3))_1, (\mathcal{F}^{-1}_{\mathtt{S}^3}((\mathtt{Y})_3))_2, \mathtt{t});$
    $\mathtt{p}_5 = \mathtt{p}_1 + \frac{((\mathtt{n}_{\mathtt{f}}+1) - (\eta^{\mathtt{f}} \times |\mathcal{F}_{\mathtt{S}^4}((\mathtt{Y})_3)|)}{|\mathcal{F}_{\mathtt{S}^4}((\mathtt{Y})_3)|} \times \mathtt{p}_1;$
    if $\mathtt{n}_{\mathtt{c}} \neq 0$ && $\mathtt{n}_{\mathtt{f}} \neq 0$ **then**
      $\mathtt{p}_5 \times = (\frac{\mathtt{n}_{\mathtt{f}}}{\mathtt{n}_{\mathtt{c}}});$
    end if
    $(\mathcal{Z}(\mathtt{pc}))_3 \cup = \{((((\mathcal{Z}(\mathtt{pc}))_7)_{\mathtt{i}})_{\mathtt{j}}, \mathtt{t}, \mathtt{p}_5\};$
    $\mathtt{j}{+}{+};$
  end while
end for

informed about such moves (algorithm 1: *line 20*). Afterwards, PC finds its best move based on the moves submitted by all the RPUs (algorithm 2: *lines 29-40*). If all such moves are same then finally, PC plays that move and based on which it will either announce the FoC parking durations or not to the RPUs (algorithm 2: *lines 41-43*). Thereafter, RPU checks whether PC has announced the FoC duration or not, (algorithm 1: *line 21*). If PC has announced the FoC duration then RPU plays its best move based on which it will either accept or reject the FoC duration (algorithm 1: *lines 22-26*). On the other hand if PC has not announced the FoC durations then also, RPU plays its best move and based on which it will either accept or reject waiting for next announcement (algorithm 1: *lines 28-32*). RPUs keep on checking the request decision until they are decided (algorithm 1: *lines 34-42*).

$$\gamma^{0*}_{\mathtt{pc}} = \underset{\gamma^0_{\mathtt{pc}} \in \Gamma^0_{\mathtt{pc}}}{\arg\max} \mathcal{U}^0_{\mathtt{pc}}(\mathtt{B}),\ \mathtt{B} = \{\gamma^0_{\mathtt{pc}}\} \cup \{\mathcal{R}^{0*}_{\mathtt{x_k}}(\gamma^{0*}_{\mathtt{pc}}, \ldots, \gamma^{0*}_{\mathtt{x_{i-1}}}, \gamma^{0*}_{\mathtt{x_{i+1}}}, \ldots) | \forall \mathtt{k} \in [1, \mathtt{n}] \wedge \mathtt{i}{=}\mathtt{k} \wedge \mathtt{x_i} \in [\mathtt{x_1}, \mathtt{x_n}]\} \quad (16)$$

$$\gamma^{0*}_{\mathtt{x_i}} = \underset{\gamma^0_{\mathtt{x_i}} \in \Gamma^0_{\mathtt{x_i}}}{\arg\max} \mathcal{U}^0_{\mathtt{x_i}}(\{\gamma^{0*}_{\mathtt{pc}}, \ldots, \gamma^{0*}_{\mathtt{x_{i-1}}}, \gamma^0_{\mathtt{x_i}}, \gamma^{0*}_{\mathtt{x_{i+1}}}, \ldots\}),\ \mathtt{i} \in [1, \mathtt{n}] \wedge \mathtt{x_i} \in [\mathtt{x_1}, \mathtt{x_n}] \quad (17)$$

E. Parking Resource Allocation Scheme

PC allocates (algorithm 2: *lines 44-45*) parking slots to the PUs using algorithm 5. While allocating the slots, priorities of the requests are taken into consideration. If the PC has announced the prices or FoC durations and at-least one undecided request is there in the MLQ then allocation procedure starts. The accept request procedure is called for such request if its PPU or RPU had accepted the announced prices or FoC duration respectively and when slot is available (algorithm 5: *line 12, 34*). In the case of RPUs, availability of particular parking slot is find out using method $\mathcal{M}^1 : (\mathtt{pc}, \mathtt{ps}, \mathtt{t}_1, \mathtt{t}_2) \rightarrow \mathtt{b}$ as given in Eq. 18. Whereas, in the case of PPUs, availability of slot is find out by finding occupancy in parking lot using method $\mathcal{M}^2 : (\mathtt{pc}, \mathtt{pl}, \mathtt{t}_1, \mathtt{t}_2) \rightarrow \mathtt{n}$ as given in Eq. 19.

$$\mathcal{M}^1 : \begin{cases} 1 \ \ \mathtt{if} \sum_{\mathtt{t}=\mathtt{t}_1}^{\mathtt{t}_2} \mathcal{F}_{(\mathcal{Z}(\mathtt{pc}))_6}(\mathtt{ps}, \mathtt{t}) \geq 1 \\ 0 \ \ \mathtt{if} \sum_{\mathtt{t}=\mathtt{t}_1}^{\mathtt{t}_2} \mathcal{F}_{(\mathcal{Z}(\mathtt{pc}))_6}(\mathtt{ps}, \mathtt{t}) = 0 \end{cases} \quad (18)$$

$$\mathcal{M}^2 : \Big\{\mathtt{n_i} = \sum_{\mathtt{k}=1}^{|\mathcal{F}_{\mathtt{S}^4}(\mathtt{pl})|} \Big(\sum_{\mathtt{t_i}=\mathtt{t}_1}^{\mathtt{t}_2} \big(\mathcal{F}_{(\mathcal{Z}(\mathtt{pc}))_6}((\mathcal{F}_{\mathtt{S}^4}(\mathtt{pl}))_{\mathtt{k}}, \mathtt{t_i})\big) > 0\Big)\Big\} \quad (19)$$

Algorithm 5 *allocate_Park*

Input: $\mathtt{pc}, \mathtt{t}, \mathtt{b_{pc}}, \mathtt{h}, \mathtt{l}, \gamma^{\mathtt{d}}, \Delta \mathtt{t}^{\mathtt{cmp}}_{\mathtt{gd}}, \Delta \mathtt{t}^{\mathtt{min}}_{\mathtt{rpu}}$
Output: $\mathtt{S}^2, (\mathcal{Z}(\mathtt{pc}))_5, (\mathcal{Z}(\mathtt{pc}))_6, (\mathcal{Z}(\mathtt{pc}))_7$

1: **if** $\mathtt{b_{pc}} == 1$ **then**
2: $\mathtt{i} = 1;$
3: **while** $\mathtt{i} \leq |\ ((\mathcal{Z}(\mathtt{pc}))_7)_{\mathtt{h}}\ |$ **do**
4: $\mathtt{Y} = \mathcal{F}_{(\mathcal{Z}(\mathtt{pc}))_1}((((\mathcal{Z}(\mathtt{pc}))_7)_{\mathtt{h}})_{\mathtt{i}});$
5: **if** $\mathcal{F}_{(\mathcal{Z}(\mathtt{pc}))_4}((((\mathcal{Z}(\mathtt{pc}))_7)_{\mathtt{h}})_{\mathtt{i}}, \mathtt{t}) == 1$ **then**
6: **if** $\mathcal{F}_{\mathtt{S}^1}((\mathtt{Y})_1) == 1$ **then**
7: $(\mathcal{M}^2(\mathtt{pc}, (\mathtt{Y})_3, (\mathtt{Y})_5, (\mathtt{Y})_5 + (\mathtt{Y})_6) <$
$\hookrightarrow\ |\ \mathcal{F}_{\mathtt{S}^4}((\mathtt{Y})_3)\ |)\ ?\ (\mathtt{b_a} = 1) : (\mathtt{b_a} = 0);$
8: **else**
9: $(\mathcal{M}^1(\mathtt{pc}, (\mathtt{Y})_4, (\mathtt{Y})_5, (\mathtt{Y})_5 + (\mathtt{Y})_6) == 0)\ ?$
$\hookrightarrow\ (\mathtt{b_a} = 1) : (\mathtt{b_a} = 0);$
10: **end if**
11: **if** $\mathtt{b_a} == 1$ **then**
12: *accept_Request*$(\mathtt{pc}, \mathtt{t}, \mathtt{h}, \mathtt{i});$
13: **else if** $\mathtt{t} == (\mathcal{M}^6(\mathtt{pc}, (((\mathcal{Z}(\mathtt{pc}))_7)_{\mathtt{h}})_{\mathtt{i}}, \gamma^{\mathtt{d}}) + \Delta \mathtt{t}^{\mathtt{cmp}}_{\mathtt{gd}})$ **then**
14: *reject_Request*$(\mathtt{pc}, \mathtt{t}, \mathtt{h}, \mathtt{i}, \Delta \mathtt{t}^{\mathtt{min}}_{\mathtt{rpu}});$
15: **else**
16: $\mathtt{i}{+}{+};$
17: **end if**
18: **else if** $\mathtt{t} == (\mathcal{M}^6(\mathtt{pc}, (((\mathcal{Z}(\mathtt{pc}))_7)_{\mathtt{h}})_{\mathtt{i}}, \gamma^{\mathtt{d}}) + \Delta \mathtt{t}^{\mathtt{cmp}}_{\mathtt{gd}})$ **then**
19: *reject_Request*$(\mathtt{pc}, \mathtt{t}, \mathtt{h}, \mathtt{i}, \Delta \mathtt{t}^{\mathtt{min}}_{\mathtt{rpu}});$
20: **else**
21: $\mathtt{i}{+}{+};$
22: **end if**
23: **end while**
24: $\mathtt{i} = 1;$
25: **while** $\mathtt{i} \leq |\ ((\mathcal{Z}(\mathtt{pc}))_7)_1\ |$ **do**
26: $\mathtt{Y} = \mathcal{F}_{(\mathcal{Z}(\mathtt{pc}))_1}((((\mathcal{Z}(\mathtt{pc}))_7)_1)_{\mathtt{i}});$
27: **if** $\mathcal{F}_{(\mathcal{Z}(\mathtt{pc}))_4}((((\mathcal{Z}(\mathtt{pc}))_7)_1)_{\mathtt{i}}, \mathtt{t}) == 1$ **then**
28: **if** $\mathcal{F}_{\mathtt{S}^1}((\mathtt{Y})_1) == 1$ **then**
29: $(\mathcal{M}^2(\mathtt{pc}, (\mathtt{Y})_3, (\mathtt{Y})_5, (\mathtt{Y})_5 + (\mathtt{Y})_6) <$
$\hookrightarrow\ |\ \mathcal{F}_{\mathtt{S}^4}((\mathtt{Y})_3)\ |)\ ?\ (\mathtt{b_a} = 1) : (\mathtt{b_a} = 0);$
30: **else**
31: $(\mathcal{M}^1(\mathtt{pc}, (\mathtt{Y})_4, (\mathtt{Y})_5, (\mathtt{Y})_5 + (\mathtt{Y})_6) == 0)\ ?$
$\hookrightarrow\ (\mathtt{b_a} = 1) : (\mathtt{b_a} = 0);$
32: **end if**
33: **if** $\mathtt{b_a} == 1$ **then**
34: *accept_Request*$(\mathtt{pc}, \mathtt{t}, \mathtt{l}, \mathtt{i});$
35: **else**
36: $\mathtt{i}{+}{+};$
37: **end if**
38: **end if**
39: **end while**
40: **else**
41: $\mathtt{i} = 1;$
42: **while** $\mathtt{i} \leq |\ ((\mathcal{Z}(\mathtt{pc}))_7)_{\mathtt{h}}\ |$ **do**
43: **if** $\mathtt{t} == (\mathcal{M}^6(\mathtt{pc}, (((\mathcal{Z}(\mathtt{pc}))_7)_{\mathtt{h}})_{\mathtt{i}}, \gamma^{\mathtt{d}}) + \Delta \mathtt{t}^{\mathtt{cmp}}_{\mathtt{gd}})$ **then**
44: *reject_Request*$(\mathtt{pc}, \mathtt{t}, \mathtt{h}, \mathtt{i}, \Delta \mathtt{t}^{\mathtt{min}}_{\mathtt{rpu}});$
45: **else**
46: $\mathtt{i}{+}{+};$
47: **end if**
48: **end while**
49: **end if**

Algorithm 6 *accept_Request*

Input: $\mathtt{pc}, \mathtt{t}, \mathtt{j}, \mathtt{i}$
Output: $\mathtt{S}^2, (\mathcal{Z}(\mathtt{pc}))_5, (\mathcal{Z}(\mathtt{pc}))_6, (\mathcal{Z}(\mathtt{pc}))_7$

1: $\mathtt{x_c} = (((\mathcal{Z}(\mathtt{pc}))_7)_{\mathtt{j}})_{\mathtt{i}},\ \mathtt{r_o} = \mathcal{F}_{(\mathcal{Z}(\mathtt{pc}))_3}(\mathtt{x_c}, \mathtt{t});$
2: $\mathtt{Y} = \mathcal{F}_{(\mathcal{Z}(\mathtt{pc}))_1}(\mathtt{x_c}),\ \mathtt{r_w} = \mathcal{F}_{\mathtt{S}^2}((\mathtt{Y})_1);$
3: $(\mathcal{Z}(\mathtt{pc}))_5 \setminus = \{\mathtt{x_c}, 0, \{\}, \{\}, \{\}, \{\}\};$
4: **if** $\mathcal{F}_{\mathtt{S}^1}((\mathtt{Y})_1) == 1$ **then**
5: $(\mathcal{Z}(\mathtt{pc}))_5 \cup = \{\mathtt{x_c}, 1, \mathtt{t}, \mathtt{r_o}, (\mathtt{Y})_6,$
$\hookrightarrow\ \mathcal{M}^5(\mathtt{pc}, (\mathtt{Y})_3, (\mathtt{Y})_5, (\mathtt{Y})_5 + (\mathtt{Y})_6)\};$
6: $\mathtt{S}^2 = (\mathtt{S}^2 \setminus \{(\mathtt{Y})_1, \mathtt{r_w}\}) \cup \{(\mathtt{Y})_1, (\mathtt{r_w} - \mathtt{r_o})\};$
7: **else**
8: $(\mathcal{Z}(\mathtt{pc}))_5 \cup = \{\mathtt{x_c}, 1, \mathtt{t}, 0, \mathtt{r_o} + \mathcal{F}_{\mathtt{S}^2}((\mathtt{Y})_1), (\mathtt{Y})_4\};$
9: $\mathtt{S}^2 = (\mathtt{S}^2 \setminus \{(\mathtt{Y})_1, \mathtt{r_w}\}) \cup \{(\mathtt{Y})_1, 0\};$
10: **end if**
11: $(\mathcal{Z}(\mathtt{pc}))_6 = \mathcal{M}^7(\mathtt{pc}, (\mathcal{F}_{(\mathcal{Z}(\mathtt{pc}))_5}(\mathtt{x_c}))_5, (\mathtt{Y})_5, (\mathtt{Y})_5 +$
$(\mathtt{Y})_6);$
12: $((\mathcal{Z}(\mathtt{pc}))_7)_{\mathtt{j}} \setminus = \{\mathtt{x_c}\};$

Algorithm 7 *reject_Request*

Input: $\mathtt{pc}, \mathtt{t}, \mathtt{j}, \mathtt{i}, \Delta \mathtt{t}^{\mathtt{min}}_{\mathtt{rpu}}$
Output: $\mathtt{S}^2, (\mathcal{Z}(\mathtt{pc}))_5, (\mathcal{Z}(\mathtt{pc}))_7$

1: $\mathtt{x_c} = (((\mathcal{Z}(\mathtt{pc}))_7)_{\mathtt{j}})_{\mathtt{i}},\ \mathtt{Y} = \mathcal{F}_{(\mathcal{Z}(\mathtt{pc}))_1}(\mathtt{x_c});$
2: $(\mathcal{Z}(\mathtt{pc}))_5 \setminus = \{\mathtt{x_c}, 0, \{\}, \{\}, \{\}, \{\}\};$
3: $(\mathcal{Z}(\mathtt{pc}))_5 \cup = \{\mathtt{x_c}, -1, \mathtt{t}, -1, -1, -1\};$
4: **if** $\mathcal{F}_{\mathtt{S}^1}((\mathtt{Y})_1) == 0$ **then**
5: $\mathtt{S}^2 = (\mathtt{S}^2 \setminus \{(\mathtt{Y})_1, \mathcal{F}_{\mathtt{S}^2}((\mathtt{Y})_1)\}) \cup \{(\mathtt{Y})_1, \Delta \mathtt{t}^{\mathtt{min}}_{\mathtt{rpu}}\};$
6: **end if**
7: $((\mathcal{Z}(\mathtt{pc}))_7)_{\mathtt{j}} \setminus = \{\mathtt{x_c}\};$

Otherwise, if the PU of the undecided request accepts the announced prices or FoC duration and parking slot is not available, and the request has already spent the necessary waiting time in the queue then the reject request procedure is invoked (algorithm 5: *line 14*). The reject procedure is also invoked for an undecided request when its PU did not accept the announced prices or FoC duration and it has already spent the necessary waiting time in the queue (algorithm 5: *line 19*). Further, it is also called when a request has spent the necessary time in the queue and the PC had not announced the prices or FoC durations (algorithm 5: *line 44*). When the accept request procedure is invoked then a parking slot is allocated (algorithm 6: *lines 5,8*), prices or FoC duration is deducted from the wallet (algorithm 6: *lines 6,9*), parking lots are apprised of changes in the status of parking slots (algorithm 6: *line 11*), and request id gets de-queued from the MLQ (algorithm 6: *line 12*). In the case of PPUs, the parking slot is allocated by finding it out using method $\mathcal{M}^5 : (\mathtt{pc}, \mathtt{pl}, \mathtt{t}_1, \mathtt{t}_2) \rightarrow \mathtt{ps}$ as given in Eq. 20. Whereas in the case of RPUs, the requested one is allocated. PC apprises the change in status of parking slot using method $\mathcal{M}^7 : (\mathtt{pc}, \mathtt{ps}, \mathtt{t}_1, \mathtt{t}_2) \rightarrow (\mathcal{Z}(\mathtt{pc}))_6$ as given in Eq. 21.

$$\mathcal{M}^5 : \Big\{ \mathtt{ps1}_{\mathtt{k}} \mid \mathtt{k} \in [1, |\ \mathtt{PS1}\ |] \wedge \mathtt{PS1} \subseteq \mathcal{F}_{\mathtt{S}^4}(\mathtt{pl}) \wedge \sum_{\mathtt{i}=1}^{|\mathtt{PS1}|} \sum_{\mathtt{t_j}=\mathtt{t_1}}^{\mathtt{t_2}} \left(\mathcal{F}_{(\mathcal{Z}(\mathtt{pc}))_6}((\mathtt{PS1})_{\mathtt{i}}, \mathtt{t_j})\right) = 0 \Big\} \tag{20}$$

$$\mathcal{M}^7 : \Big\{ \left(\mathcal{Z}(\mathtt{pc})\right)_6 \mid \forall \mathtt{t_i} \left(((\mathcal{Z}(\mathtt{pc}))_6 \setminus = \{\mathtt{ps}, \mathtt{t_i}, 0\}) \wedge \left((\mathcal{Z}(\mathtt{pc}))_6 \cup = \{\mathtt{ps}, \mathtt{t_i}, 1\}\right) \right) \wedge \mathtt{t_i} \in [\mathtt{t_1}, \mathtt{t_2}] \Big\} \tag{21}$$

When the reject request procedure is called then the status of request is updated to rejected with timestamp (algorithm 7:

line 3). Then, it is de-queued from the MLQ (algorithm 7: *line 7*). If the requester is RPU then its wallet is credited with daily default FoC threshold time duration (algorithm 7: *lines 4-6*).

V. Performance Evaluation

This section presents performance evaluation of the proposed *Dy*PARK PAS scheme.

A. Numerical Settings and System Modeling

In the simulation, the on-street parking system of the Seattle city (as depicted in Fig. 5(b)) is considered. It has 1471 on-street parking lots. These parking lots spread across 929 block-faces, where all of these block-faces are in 33 different sub-areas. In turn, these sub-areas are the part of 21 areas of the city. In order to predict occupancy by using random forest model, a parking data-set [32] released by the Seattle Transport Department (STD) is considered. The data-set contains 25.7 million rows of eight different features. This data-set is having entries for the period of April 01, 2019 to May 01, 2019. The parking prices charged at Seattle city have been used from data-set [33]. Seattle on-street parking data-set [32] is pre-processed using R Studio. Various activities, such as, encoding, imputation, normalization, and features selection have been carried out during pre-processing of the data-set. Random forest model is trained using MATLAB R2019b. In order to generate in-arrival parking requests, poission process is implemented. The arrival rate of PPUs' and RPUs' requests for each parking lot is 60% of the total slots available in the parking lot. The values of γ^{d}, α, β, and τ are 0.6, 0.00005, 1, and 1 respectively. Whereas, the values of $\Delta \mathrm{t}_{\mathrm{rpu}}^{\mathrm{min}}$, $\Delta \mathrm{t}_{\mathrm{rpu}}^{\mathrm{max}}$, and $\Delta \mathrm{t}_{\mathrm{gd}}^{\mathrm{cmp}}$ are 1200 seconds, 3600 seconds, and 5 seconds respectively. PUs can request advance booking for parking and they can send such requests up to 7200 seconds before their arrival at parking. Initially, all PPUs and RPUs have \$100 and $\Delta \mathrm{t}_{\mathrm{rpu}}^{\mathrm{min}}$ duration in their wallets respectively. Stackelberg games are formulated and solved in MATLAB R2019b.

Figs. 3(a), 3(b), 3(c), 3(d), 3(e), and 3(f) depict system modeling. Fig. 3(a) shows the arrival of PPUs' requests and vehicles during 10 minutes at all PCs, whereas Fig. 3(d) shows the same information about RPUs. Fig. 3(b) depicts scheduled and actual departures of PPUs' vehicles during 10 minutes at all PCs whereas, Fig. 3(e) depicts the same information about RPUs. The difference between the number of scheduled and actual departures at any point of time in Fig 3(b) is due to the implementation of early and late departure behavior of the PPUs. The requested and scheduled departure times for requests sent by the PPUs are the same because such requests are accepted only when the parking is available for the requested duration. As given in Fig. 3(e) the number of scheduled and actual departures of RPUs' vehicles during 10 minutes at all PCs are same because FoC parking durations granted to RPUs are dedicated. It means the sensors will not detect the early departure, and no late departure is allowed according to the SLA between PSPs and RPUs. Fig. 3(c) shows requested and utilized parking duration for requests made by PPUs arrived at different times of the day at all PCs whereas, Fig. 3(f) depicts the same for RPUs. Again, the non-overlapping plots in Fig. 3(c) depict the implementation of early, on-time, and late departure of vehicles. Whereas, in the case of RPUs (Fig. 3(f)), the utilized duration is less than or equal to the requested one. It is because FoC durations allocated to RPUs have been minimized by the PCs while solving objective function $\mathcal{O}4$.

B. Compared Techniques

In order to analyze the outcomes of the proposed scheme, we have compared the proposed scheme with four PASs as follows.

1) STD PAS [33]: This scheme mimics the PAS currently being used by the Seattle Transport Department where prices are defined by the STD and allocation is done based on FCFS.
2) Spatial-A PAS: This scheme is derived from STD PAS [33] where area wise average prices are charged from the users.
3) Spatial-SA PAS: This scheme is derived from STD PAS [33] where sub-area wise average prices are charged from the users.
4) Occupancy based PAS [31]: In this scheme, a machine learning and occupancy based parking pricing and allocation scheme is implemented.

Whereas, the communication results of the proposed scheme are compared with four other CPs defined as follows.

1) CC variant: In this *Dy*PARK is implemented by placing only one PC at level 0 (i.e., at Central Entity (CE)) of the Fig. 2(b).
2) FC_a variant: In this *Dy*PARK is implemented by placing PCs at level 1 (i.e., at a(s)) of the Fig. 2(b).
3) FC_{bf} variant: In this *Dy*PARK is implemented by placing PCs at level 3 (i.e., at bf(s)) of the Fig. 2(b).
4) EC variant: In this *Dy*PARK is implemented by placing PCs at level 4 (i.e., at ss(s)) of the Fig. 2(b).

C. Results and Discussion

1) Request Decision: Fig. 3(g) depicts the number of PPUs' requests accepted and rejected by all PCs during different hours of the day using proposed *Dy*PARK PAS and without using *Dy*PARK PAS. Whereas, Fig. 3(j) presents the same information about the RPUs' requests. It is evident from the plots that more requests are accepted when *Dy*PARK PAS is used. It is pertinent to mention here that the phrase "without *Dy*PARK" in the figures can be replaced by any one of the compared/ state-of-the-art techniques (i.e., STD, Spatial-A, Spatial-SA, Occupancy PAS) as and when required. During 24 hours simulation, 56,676 PPUs' requests are accepted, and only 38,294 are rejected when *Dy*PARK PAS is used. Whereas 55,421 PPUs' requests are accepted, and 39,549 are rejected when other state-of-the-art techniques are used. In the case of RPUs, 7,971 requests are accepted, and only 3,633 are rejected when *Dy*PARK PAS is used. Whereas 7,312 RPUs' requests are accepted, and 4,292 are rejected when other state-of-the-art techniques are used. Hence, more travelers will have the

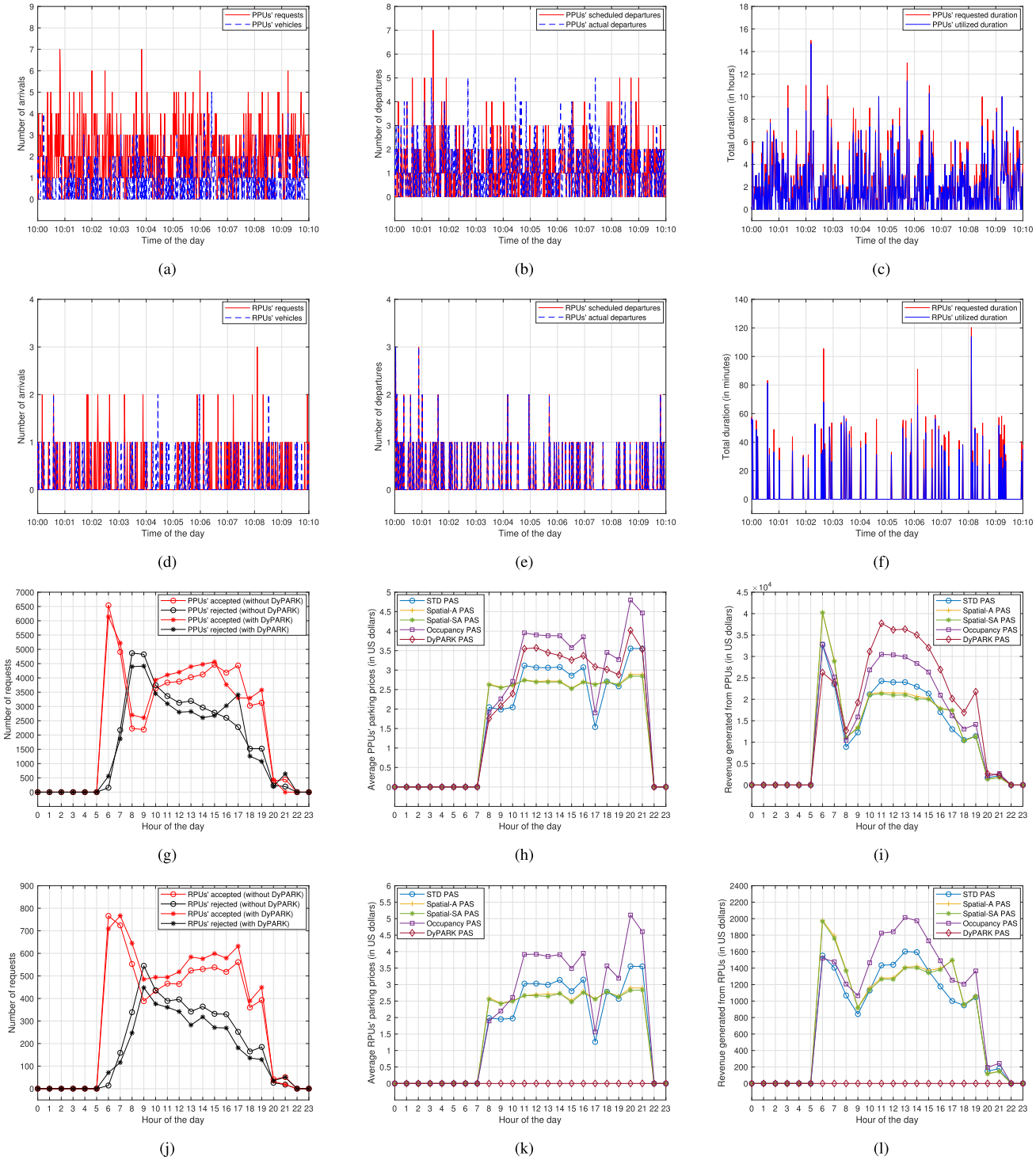


Fig. 3. (a) PPUs' requests and vehicles arrival vs time of the day (b) Scheduled and actual departures of PPUs' vehicles vs time of the day (c) PPUs' requested and utilized parking duration vs time of the day (d) RPUs' requests and vehicles arrival vs time of the day (e) Scheduled and actual departures of RPUs' vehicles vs time of the day (f) RPUs' requested and utilized parking duration vs time of the day (g) Number of accepted and rejected PPUs' requests vs hour of the day (h) Average parking prices for PPUs vs hour of the day (i) Revenue generated from PPUs vs hour of the day (j) Number of accepted and rejected RPUs' requests vs hour of the day (k) Average parking prices for RPUs vs hour of the day (l) Revenue generated from RPUs vs hour of the day.

in-advance knowledge of their parking reservation which in turn will eliminate the cruising of vehicles in search of parking. The objective $\mathcal{O}4$, i.e., minimization of FoC durations done by the PCs is the reason behind the increase in accepted RPUs' requests. This proves the reduction in total number of SLA violations, when *Dy*PARK PAS is used.

Fig. 4(i) represents the average processing time of the PUs' requests received during different time instances of the day.

It is evident from the plots that STD, Spatial-A, and Spatial-SA PASs take minimum request processing time because these PASs check occupancy and predefined prices only. Occupancy PAS has a slightly higher processing time where occupancy prediction operation is carried out in addition to things done in STD, Spatial-A, and Spatial-SA PASs. Although the processing time of the *Dy*PARK PAS is more than that of all other PASs but still it is within bearable limits. *Dy*PARK PAS performs all operations of the Occupancy PAS. Apart from that, it performs some extra operations where it generates prices and FoC parking durations. The significant hikes in the plot at some time instances are due to two major reasons. First is the presence of online advance requests. Such requests last in the ready queue for a long time, i.e., up-to their decision deadline. Second reason is compulsory game duration (Δt_{gd}^{cmp}) which provides a fair play to both PUs and PCs. For both reasons, requests are processed multiple times where prices or FoC parking durations are generated and communicated. Then PUs need to accept or reject such offerings. This whole procedure eventually adds more time to the processing time.

2) Average Parking Prices: The average parking prices paid by the PUs for their requests arrived during time interval $\Delta t = \{t_1, t_2\}$ is computed using Eq. 22. Fig. 3(h) and Fig. 3(k) shows average parking prices paid by the PPUs and RPUs respectively at different hours of the day using proposed *Dy*PARK PAS and various state-of-the-art techniques. The average prices charged from PPUs using *Dy*PARK PAS are less than prices charged using Occupancy PAS but more than prices charged using other PASs. At the end of 24 hours simulation, the average parking prices charged from PPUs using STD, Spatial-A, Spatial-SA, Occupancy, and *Dy*PARK PASs are \$2.54, \$2.66, \$2.64, \$3.11, and \$2.91 respectively. Due to SLA, the average parking prices charged from RPUs are zero. Thus, at the end of 24 hours simulation, the average prices charged from RPUs under STD, Spatial-A, Spatial-SA, Occupancy, and *Dy*PARK PASs are \$2.45, \$2.62, \$2.59, \$2.99, and \$0 respectively.

$$\mathcal{P}^{avg}(\Delta t) = \left\{ \frac{\sum_{t_k=t_1}^{t_2} \sum_{i=1}^{|PC|} \sum_{j=1}^{|(\mathcal{Z}(pc_i))_5|} (((\mathcal{Z}(pc_i))_5)_j)_4}{\sum_{t_k=t_1}^{t_2} \sum_{i=1}^{|PC|} \sum_{j=1}^{|(\mathcal{Z}(pc_i))_5|} (((\mathcal{Z}(pc_i))_5)_j)_2} \right| (((\mathcal{Z}(pc_i))_5)_j)_2 = 1 \wedge (\mathcal{F}_{(\mathcal{Z}(pc_i))_1} ((((\mathcal{Z}(pc_i))_5)_j)_1))_2 = t_k \Big\} \quad (22)$$

Fig. 4(b) shows average parking prices paid by the PUs at different hours of the day using proposed *Dy*PARK PAS and various state-of-the-art techniques. The average parking prices charged from PUs using STD, Spatial-A, Spatial-SA, Occupancy, and *Dy*PARK PASs are \$2.53, \$2.65, \$2.64, \$3.10, and \$2.55 respectively. It is evident that at the end of 24 hours simulation, the average parking prices are minimized to \$2.55, which is significantly less than the average prices of three PASs and only \$0.02 more than the STD PAS.

3) Revenue Generated: The revenue generated by the PCs from requests arrived during the time interval $\Delta t = \{t_1, t_2\}$ is computed using Eq. 23.

$$\mathcal{R}_{rev}^{T}(\Delta t) = \Bigg\{ \sum_{t_k=t_1}^{t_2} \sum_{i=1}^{|PC|} \sum_{j=1}^{|(\mathcal{Z}(pc_i))_5|} (((\mathcal{Z}(pc_i))_5)_j)_4 \times \left\lceil \frac{(((\mathcal{Z}(pc_i))_5)_j)_5}{3600} \right\rceil \Bigg| \Big(((\mathcal{Z}(pc_i))_5)_j \Big)_2 = 1 \wedge (\mathcal{F}_{(\mathcal{Z}(pc_i))_1} ((((\mathcal{Z}(pc_i))_5)_j)_1))_2 = t_k \Big\} \quad (23)$$

Fig. 3(i) and Fig. 3(l) show plots of revenue generated from parking prices paid by the PPUs and RPUs respectively at all PCs during different hours of the day using proposed *Dy*PARK and various other PASs. The generated revenue from PPUs is maximum using *Dy*PARK PAS during majority hours of the day even when the average parking prices charged using *Dy*PARK PAS are less than that of charged using Occupancy PAS. At the end of 24 hours simulation, the revenue generated from PPUs using STD, Spatial-A, Spatial-SA, Occupancy, and *DyPARK* PASs are \$2,70,866.50, \$2,80,894.72, \$2,77,847.46, \$3,25,883.20, and \$3,81,462.73 respectively. Whereas, the revenue generated from RPUs using STD, Spatial-A, Spatial-SA, Occupancy, and *Dy*PARK PASs are \$17,936, \$19,163.92, \$18,969.74, \$21,875.61, and \$0 respectively.

Fig. 3(c) shows the revenue generated from parking prices paid by the PUs at all PCs during different hours of the day using proposed *Dy*PARK and various state of the art PASs. The revenue generated from PUs during 24 hours simulation using STD, Spatial-A, Spatial-SA, Occupancy, and *Dy*PARK PASs are \$2,88,802.50, \$3,00,058.64, \$2,96,817.20, \$3,47,758.81, and \$3,81,462.73 respectively.

4) Average Parking Loss: The average parking loss incurred by the PCs due to RPUs' requests arrived during the time interval $\Delta t = \{t_1, t_2\}$ is computed using Eq. 24, as shown at the bottom of the next page.

Fig. 4(a) presents average FoC parking duration granted to the RPUs (i.e., average parking loss to the PSPs) by all PCs with and without using *Dy*PARK PAS. It is evident from the figure that the average FoC parking duration granted to the RPUs using *Dy*PARK PAS is less than granted without using *Dy*PARK PAS during all hours of the day. At the end of 24 hours simulation, the average FoC parking duration granted to the RPUs with and without *Dy*PARK PAS are 2057.20 and 2471.18 seconds, respectively, which is more than the general threshold duration (i.e., Δt_{rpu}^{min}) which is 1200 seconds. This minimization of average parking loss has been done when it has been ensured that if the requested duration is less than the Δt_{rpu}^{min}, then it will be granted as it is. And, if it is higher than Δt_{rpu}^{min}, then at least duration equal to Δt_{rpu}^{min} will be granted. It is evident from the figure that the average FoC parking duration granted using *Dy*PARK PAS is more than the general threshold duration, which proves that maximization of the FoC parking duration has been done from the RPUs' point of view. This fact shouts out the fairness of the proposed *Dy*PARK PAS.

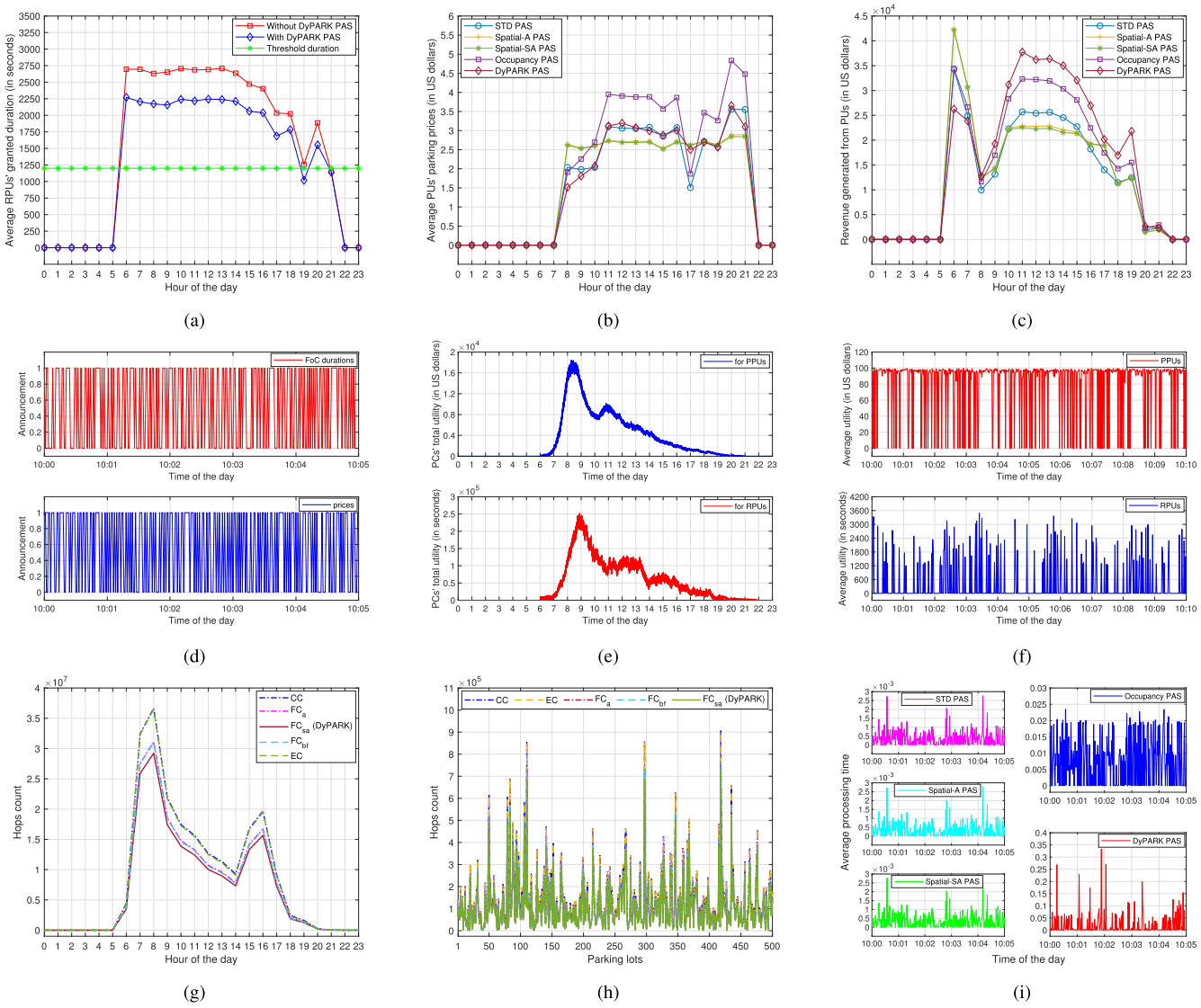


Fig. 4. (a) Average granted duration to RPUs vs hour of the day (b) Average parking prices for PUs vs hour of the day (c) Revenue generated from PUs vs hour of the day (d) Prices and durations announcement by PCs vs time of the day (e) PCs' total utility for PPUs and RPUs vs time of the day (f) Average utility of PPUs and RPUs vs time of the day (g) Hops count vs hour of the day (h) Hops count vs parking lots (500 parking lots) (i) Average processing time of PUs' requests vs time of the day.

Thus, in nutshell, the occupancy based PAS is an ideal and balanced solution when parking facilities are limited, and more vehicles need parking. In a similar situation simulated in this work, the Occupancy PAS-based average parking prices are more than STD, Spatial-A, and Spatial-SA PASs based average parking prices. This is an optimistic lead towards managing parking and revenue simultaneously. However, in the same situation, while accepting more requests, the *Dy*PARK PAS charged less average parking prices as compared to the ideal and balanced solution, i.e., Occupancy PAS during different hours of the day. And also, when *Dy*PARK PAS has provided FoC parking to the RPUs, it has generated the maximum revenue.

$$\mathcal{L}^{\mathtt{avg}}(\Delta \mathtt{t}) = \left\{ \frac{\sum_{\mathtt{t_k}=\mathtt{t}_1}^{\mathtt{t}_2} \sum_{\mathtt{i}=1}^{|\mathtt{PC}|} \sum_{\mathtt{j}=1}^{|(\mathcal{Z}(\mathtt{pc_i}))_5|} (((\mathcal{Z}(\mathtt{pc_i}))_5)_\mathtt{j})_5}{\sum_{\mathtt{t_k}=\mathtt{t}_1}^{\mathtt{t}_2} \sum_{\mathtt{i}=1}^{|\mathtt{PC}|} \sum_{\mathtt{j}=1}^{|(\mathcal{Z}(\mathtt{pc_i}))_5|} (((\mathcal{Z}(\mathtt{pc_i}))_5)_\mathtt{j})_2} \middle| (((\mathcal{Z}(\mathtt{pc_i}))_5)_\mathtt{j})_2 = 1 \wedge (\mathcal{F}_{(\mathcal{Z}(\mathtt{pc_i}))_1}((((\mathcal{Z}(\mathtt{pc_i}))_5)_\mathtt{j})_1))_2 = \mathtt{t_k} \wedge \mathcal{F}_{\mathtt{S}^1}((\mathcal{F}_{(\mathcal{Z}(\mathtt{pc_i}))_1}((((\mathcal{Z}(\mathtt{pc_i}))_5)_\mathtt{j})_1))_1) = 0 \right\} \tag{24}$$

TABLE II
COMPARISON BETWEEN DIFFERENT VARIANTS OF *Dy*PARK PAS

Parameter →	Communication cost									
Links ↓	CC		FC_a		FC_{sa} (*Dy*PARK)		FC_{bf}		EC	
	a	b	a	b	a	b	a	b	a	b
c ↔ a	52,970,954	329.79	20,803,192	141.64	20,803,192	144.12	20,803,192	154.04	20,803,192	156.52
a ↔ sa	52,970,954	329.79	52,970,954	361.37	41,631,444	288.43	41,631,444	308.28	41,631,444	313.24
sa ↔ bf	52,970,954	329.79	52,970,954	361.37	52,970,954	367.68	62,761,508	464.93	62,761,508	472.41
bf ↔ ss	52,970,954	329.79	52,970,954	361.37	52,970,954	367.68	52,970,954	392.94	84,096,416	633.24
ss ↔ pu	52,712,366	327.88	52,712,366	359.3	52,712,366	365.59	52,712,366	390.72	52,712,366	397
Total	264,596,182	1647.06	232,428,420	1585.05	221,088,910	1533.5	230,879,464	1710.92	262,004,926	1972.42

a: Transferred messages, b: Transferred data (in MBs)

5) Utilities: Fig. 4(d) depicts whether FoC durations and prices have been announced by the PCs to RPUs and PPUs, respectively. Thus, at any point in time, it can take either of two values, i.e., 0 (not announced) or 1 (announced). In this plot, '0s' are plotted when the majority of PCs have not announced the durations or prices. Whereas, '1s' are plotted when the majority of PCs have announced the durations or prices. The joint utility of all the PCs for PPUs (if b=1) and RPUs (if b=0) can be calculated using Eq. 25.

$$\mho^{\mathrm{PC}}(\mathrm{t}) \;=\; \sum_{\mathrm{i}=1}^{|\mathrm{PC}|} \mathcal{M}^{3}(\mathrm{pc_i}, \mathrm{t}, \mathrm{b}) \tag{25}$$

Fig. 4(e) presents the total utility of all the PCs during all-time instances of the day. At any time instance, the total utility of all PCs for PPUs is the revenue that would have been generated if all pending requests at that time instance had been accepted. Whereas, the total utility for RPUs is the FoC parking duration that would have been granted if all pending requests at that time instance had been accepted.

Fig. 4(f) illustrates the average utility of PPUs and RPUs during the 10-minute interval. The only utility at the request decision timestamp has been considered for each request. The PPU's utility is the wallet amount (if the request is accepted) or it is the wallet amount when the offered prices are assumed to be paid (if the request is rejected). Whereas, the RPU's utility is the total FoC duration it gets (if the request is accepted) or it is the sum of the wallet and the offered duration (if the request is rejected). Thus, Fig. 4(f) is the average of all such utilities. "Time of the day" in the figure depicts requests decision time.

6) Parking Communication: An appropriate format for each type of message (as defined in section II) is considered in this work. Fig. 4(g) shows the hops count of all the messages passed during distinct hours of the day using *Dy*PARK PAS and other variants of the *Dy*PARK PAS. It can be seen in the figure that hops count are minimum when *Dy*PARK PAS is simulated. This proves that the used communication infrastructure in the case of proposed *Dy*PARK PAS is less than that in other variants. The same can be verified from Table II where overall transferred data across various links is presented. It can be easily observed that the less total number of messages and hence, data were passed through various links of the system using *Dy*PARK PAS when compared with its other variants, such as, CC, FC_a, FC_{bf}, and EC. *Dy*PARK PAS proves its worth by sending at least 4.24% and 3.25% fewer messages and bits of data, respectively compared to other variants. Fig. 4(h) represents the hops count of all the messages passed for requests related to 500 different parking lots. Here also, it can be seen that hops count are minimum when proposed *Dy*PARK PAS is used.

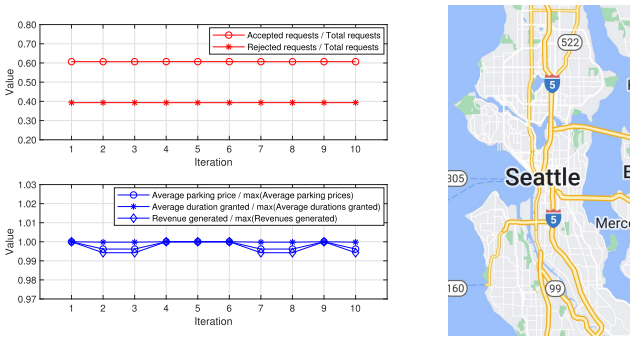


Fig. 5. (a) K-cross validation (for K=10) (b) Seattle city map.

7) Robustness: This work is based on the random generation of parking requests. Hence, k cross-validation of results has been performed to check the robustness of the system. k = 10 cross validation results of *Dy*PARK PAS are shown in Fig. 5(a) which prove the robustness of this work.

VI. CONCLUSION

In this paper, a dynamic pricing and allocation problem of on-street parking system is formulated as a Stackelberg game. The broad objectives of the formulated problem are minimization of parking prices for PPUs, maximization of granted FoC parking duration to RPUs, maximization of the generated revenue for PCs, and minimization of total parking loss for PCs. A realistic simulation environment of the Seattle parking system is created to verify the effectiveness of the proposed *Dy*PARK PAS which solves the problem to find NE. *Dy*PARK PAS is evaluated on several parameters, and the obtained results prove its worth over other state-of-the-art schemes. The results also prove the achievement of all objectives of this work. Further, the proposed *Dy*PARK PAS is compared with its four variants, namely CC, FC_a, FC_{bf}, and EC variants. It is learned from the results that less congestion or communication overhead was there in the Seattle city parking network when *Dy*PARK PAS was used. Hence the proposed scheme can be used when stakeholders (such as, PPUs, RPUs, PCs, parking authorities) negotiate for their immediate benefits. It can also be seen as a promising solution in order to decrease the network congestion. Further, the direct impact of the proposed scheme on congestion due to parking

is not evaluated in this work. Hence in the future work, the effect of the proposed *Dy*PARK PAS will be explored on the congestion which takes place due to the cruising of vehicles in search of parking. The other dimension of the future work will be to study the appropriateness of the proposed scheme with respect to the off-street parking facilities.

APPENDIX

The existence and uniqueness of the NE solutions for the formulated multiple single-leader and multi-follower Stackelberg games are explained in Appendix-A.

REFERENCES

[1] P. Misra, A. Vasan, B. Krishnan, V. Raghavan, and A. Sivasubramaniam, "The future of smart parking systems with parking 4.0," *GetMobile, Mobile Comput. Commun.*, vol. 23, no. 1, pp. 10–15, Jul. 2019.

[2] D. Guo and C. Zhou, "Realistic modeling of vehicle-to-grid in an enterprise parking lot: A Stackelberg game approach," in *Proc. IEEE Texas Power Energy Conf. (TPEC)*, Feb. 2018, pp. 1–6.

[3] P. Salyani, M. Abapour, and K. Zare, "Stackelberg based optimal planning of DGs and electric vehicle parking lot by implementing demand response program," *Sustain. Cities Soc.*, vol. 51, Nov. 2019, Art. no. 101743.

[4] G. S. Aujla, M. Singh, N. Kumar, and A. Y. Zomaya, "Stackelberg game for energy-aware resource allocation to sustain data centers using RES," *IEEE Trans. Cloud Comput.*, vol. 7, no. 4, pp. 1109–1123, Oct. 2019.

[5] *2018 Emerging Trends in Parking*. Accessed: Apr. 20, 2020. [Online]. Available: https://www.parking-mobility.org/wp-content/uploads/2019/01/IPMI-2018-Emerging-Trends-Report.pdf

[6] M. Duan, D. Wu, and H. Liu, "Bi-level programming model for resource-shared parking lots allocation," *Transp. Lett.*, vol. 12, no. 7, pp. 501–511, 2020.

[7] S. Saharan, S. Bawa, and N. Kumar, "Dynamic pricing techniques for intelligent transportation system in smart cities: A systematic review," *Comput. Commun.*, vol. 150, pp. 603–625, Jan. 2020.

[8] M. Aljohani, S. Olariu, A. Alali, and S. Jain, "A survey of parking solutions for smart cities," *IEEE Trans. Intell. Transp. Syst.*, vol. 23, no. 8, pp. 10012–10029, Aug. 2022.

[9] T. Fiez and L. Ratliff, "Data-driven spatio-temporal analysis of curbside parking demand: A case-study in Seattle," 2017, *arXiv:1712.01263*.

[10] C. Lei and Y. Ouyang, "Dynamic pricing and reservation for intelligent urban parking management," *Transp. Res. C, Emerg. Technol.*, vol. 77, pp. 226–244, Dec. 2017.

[11] S. Saharan, S. Baway, and N. Kumarz, "OP^{3}S: On-street occupancy based parking prices prediction system for ITS," in *Proc. IEEE Globecom Workshops (GC Wkshps*, Dec. 2020, pp. 1–6.

[12] D. Ayala, O. Wolfson, B. Xu, B. Dasgupta, and J. Lin, "Parking slot assignment games," in *Proc. 19th ACM SIGSPATIAL Int. Conf. Adv. Geographic Inf. Syst.*, 2011, pp. 299–308.

[13] A. Mirheli and L. Hajibabai, "Utilization management and pricing of parking facilities under uncertain demand and user decisions," *IEEE Trans. Intell. Transp. Syst.*, vol. 21, no. 5, pp. 2167–2179, May 2020.

[14] D. Ayala, O. Wolfson, B. Xu, B. DasGupta, and J. Lin, "Pricing of parking for congestion reduction," in *Proc. 20th Int. Conf. Adv. Geographic Inf. Syst.*, 2012, pp. 43–51.

[15] L. Du and S. Gong, "Stochastic Poisson game for an online decentralized and coordinated parking mechanism," *Transp. Res. B, Methodol.*, vol. 87, pp. 44–63, May 2016.

[16] E. Kokolaki, M. Karaliopoulos, and I. Stavrakakis, "Leveraging information in parking assistance systems," *IEEE Trans. Veh. Technol.*, vol. 62, no. 9, pp. 4309–4317, Nov. 2013.

[17] O. T. T. Kim, N. H. Tran, C. Pham, T. LeAnh, M. T. Thai, and C. S. Hong, "Parking assignment: Minimizing parking expenses and balancing parking demand among multiple parking lots," *IEEE Trans. Autom. Sci. Eng.*, vol. 17, no. 3, pp. 1320–1331, Jul. 2020.

[18] Y.-H. Huang and C.-H. Hsieh, "A decision support system for available parking slots on the roadsides in urban areas," *Exp. Syst. Appl.*, vol. 205, Nov. 2022, Art. no. 117668.

[19] T. Nakazato, Y. Fujimaki, and T. Namerikawa, "Parking lot allocation using rematching and dynamic parking fee design," *IEEE Trans. Control Netw. Syst.*, early access, Apr. 5, 2022, doi: 10.1109/TCNS.2022.3165015.

[20] Y. Chen, T. Wang, X. Yan, and C. Wang, "An ensemble optimization strategy for dynamic parking-space allocation," *IEEE Intell. Transp. Syst. Mag.*, early access, Jan. 19, 2022, doi: 10.1109/MITS.2022.3163506.

[21] N. Kumar, N. Chilamkurti, and J. J. P. C. Rodrigues, "Bayesian coalition game as-a-service for content distribution in Internet of Vehicles," *IEEE Internet Things J.*, vol. 1, no. 6, pp. 544–555, Dec. 2014.

[22] N. Kumar, R. Iqbal, S. Misra, and J. J. P. C. Rodrigues, "Bayesian coalition game for contention-aware reliable data forwarding in vehicular mobile cloud," *Future Generat. Comput. Syst.*, vol. 48, no. 7, pp. 60–72, 2015.

[23] N. Kumar, S. Misra, and M. S. Obaidat, "Collaborative learning automata-based routing for rescue operations in dense urban regions using vehicular sensor networks," *IEEE Syst. J.*, vol. 9, no. 3, pp. 1081–1090, Sep. 2015.

[24] *Global Parking Index 2017*. Accessed: Apr. 28, 2020. [Online]. Available: https://www.parkopedia.com/static/reports/global_parking_index2017-parkopedia.pdf

[25] Y. Zhang, C.-Y. Wang, and H.-Y. Wei, "Parking reservation auction for parked vehicle assistance in vehicular fog computing," *IEEE Trans. Veh. Technol.*, vol. 68, no. 4, pp. 3126–3139, Apr. 2019.

[26] Y. Zhang, C.-Y. Wang, and H.-Y. Wei, "Parked vehicle assisted VFC system with smart parking: An auction approach," in *Proc. IEEE Global Commun. Conf. (GLOBECOM)*, Dec. 2018, pp. 1–7.

[27] J. Zhang, X. Huang, and R. Yu, "Optimal task assignment with delay constraint for parked vehicle assisted edge computing: A Stackelberg game approach," *IEEE Commun. Lett.*, vol. 24, no. 3, pp. 598–602, Mar. 2020.

[28] X. Wang and X. Wang, "Flexible parking reservation system and pricing: A continuum approximation approach," *Transp. Res. B, Methodol.*, vol. 128, pp. 408–434, Oct. 2019.

[29] V. Hassija, V. Saxena, V. Chamola, and F. R. Yu, "A parking slot allocation framework based on virtual voting and adaptive pricing algorithm," *IEEE Trans. Veh. Technol.*, vol. 69, no. 6, pp. 5945–5957, Jun. 2020.

[30] Z. Sadreddini, S. Guner, and O. Erdinc, "Design of a decision-based multicriteria reservation system for the EV parking lot," *IEEE Trans. Transport. Electrific.*, vol. 7, no. 4, pp. 2429–2438, Dec. 2021.

[31] S. Saharan, N. Kumar, and S. Bawa, "An efficient smart parking pricing system for smart city environment: A machine-learning based approach," *Future Gener. Comput. Syst.*, vol. 106, pp. 622–640, May 2020.

[32] *Seattle City on-Street Parking Data*. Accessed: May 25, 2019. [Online]. Available: https://data.seattle.gov/Transportation/Paid-Parking-Occupancy-Last-30-Days-/rke9-rsvs

[33] *Seattle City on-Street Parking Rates*. Accessed: May 25, 2019. [Online]. Available: https://www.seattle.gov/transportation/projects-and-programs/programs/parking-program/paid-parking-information/street-parking-rates

Sandeep Saharan received the B.Tech. in computer science and engineering from M.D. University, Rohtak, Haryana, India, and the M.E. degree in computer science and engineering from Thapar University, Patiala, Punjab. He is a Research Scholar with the Department of Computer Science and Engineering, Thapar Institute of Engineering and Technology, Patiala. He has published research articles in reputed international journals as well as in international conferences, such as, IEEE TRANSACTIONS ON INTELLIGENT TRANSPORTATION SYSTEMS, *Future Generation Computer Systems*, *Computer Communications*, *Applied Mathematics and Information Sciences*, and IEEE Globecom. His research interests are in the areas of evolutionary optimization, game theory, and intelligent transportation system. He is an Active Member of various organizations, such as, IEEE, ACM, and CSI. He is currently Chair of ACM chapter of TIET.

Neeraj Kumar (Senior Member, IEEE) is working as a Full Professor with the Department of Computer Science and Engineering, Thapar Institute of Engineering and Technology (Deemed to be University), Patiala, Punjab, India. He is also an Adjunct Professor with King Abdul Aziz University, Jeddah, Saudi Arabia and Newcatle University, UK. He has highly-cited researcher from WoS, from 2019 to 2021. He has published more than 500 technical research papers (DBLP:https://dblp.org/pers/hd/k/Kumar_0001:Neeraj) in top-cited journals and conferences which are cited more than 35271 times from well-known researchers across the globe with current h-index of 105 (Google scholar: https://scholar.google.com/citations?hl=en&user=gL9gR-4AAAAJ. He has guided many research scholars leading to Ph.D. and M.E./M.Tech. He has also edited/authored 10 books with International/National Publishers like IET, Springer, Elsevier, CRC. Security and Privacy of Electronic Healthcare Records: Concepts, paradigms and solutions (ISBN-13: 978-1-78561-898-7), Machine Learning for cognitive IoT, CRC Press, Blockchain, Big Data and Machine learning, CRC Press, Blockchain Technologies across industrial vertical, Elsevier, Multimedia Big Data Computing for IoT Applications: Concepts, Paradigms and Solutions (ISBN: 978-981-13-8759-3), Proceedings of First International Conference on Computing, Communications, and Cyber-Security (IC4S 2019) (ISBN 978-981-15-3369-3). Probabilistic Data Structures for Blockchain based IoT Applications, (CRC Press). One of the edited text-book entitled, "Multimedia Big Data Computing for IoT Applications: Concepts, Paradigms, and Solutions" published in Springer in 2019 is having 3.5 million downloads till 06 June 2020. It attracts attention of the researchers across the globe. (https://www.springer.com/in/book/9789811387586). His research is supported by funding from various competitive agencies across the globe. His broad research areas are Green computing and Network management, IoT, Big Data Analytics, Deep learning and cyber-security.

He is serving as an Editors of ACM Computing Survey, IEEE TRANSACTIONS ON SUSTAINABLE COMPUTING, IEEE TNSM, Elsevier Computer Communication, Wiley International Journal of Communication Systems. Also, he has organized various special issues of journals of repute from IEEE, Elsevier, Springer. He has been a workshop chair at IEEE Globecom 2018, IEEE Infocom 2020 (https://infocom2020.ieee-infocom.org/workshop-blockchain-secure-software-defined-networking-smart-communities) and IEEE ICC 2020 (https://icc2020.ieee-icc.org/workshop/ws-06-secsdn-secure-and-dependable-software-defined-networking-sustainable-smart) and track chair of Security and privacy of IEEE MSN 2020 (https://conference.cs.cityu.edu.hk/msn2020/cf-wkpaper.php). He has won the outstanding leadership award from IEEE Trsutcom in 2021. Moreover He has also won best paper award from Elsevier JNCA in 2021 and IEEE Comsoc IWCMC 2021. He won the best researcher award from parent organization every year from last eight consecutive years. He is also TPC Chair and member for various International conferences such as IEEE MASS 2020, IEEE MSN2020. He has won the best papers award from IEEE Systems Journal in 2018, in 2020, and IEEE ICC 2018, Kansas-city in 2018. He has also won best paper award from Elsevier JNCA in 2021 and IEEE Comsoc IWCMC 2021.

Seema Bawa (Member, IEEE) received the M.Sc. degree in statistics, from IIT Kanpur, India, the M.Tech. degree in computer science and data processing from IIT Kharagpur, India, and the Ph.D. degree in computer science and engineering from the Thapar Institute of Engineering and Technology Patiala, Punjab, India. She is currently a Professor with the Department of Computer Science and Engineering, Thapar Institute of Engineering and Technology, Patiala. She has guided many Research Scholars Leading to Ph.D. and M.E./M.Tech. degrees. She has authored of more than 160 technical research papers in leading journals and conferences, including the ACM Computing Surveys, Journal of Network and Computer Applications, Future Generation Computer Systems, Artificial Intelligence Reviews, Telematics and Informatics, Neurocomputing etc. She is an Active Member of various professional bodies, such as, IEEE, ACM, and CSI.