AVL Based Settlement Algorithm and Reservation System for Smart Parking Systems in IoT-based Smart Cities

Hikmet Canli
Department of Computer Engineering,
Duzce University, Turkey
hikmetcnli@hotmail.com

Sinan Toklu
Department of Computer Engineering,
Gazi University, Turkey
stoklu@gazi.edu.tr

Abstract: *In Internet of Things (IOT)-based smart cities, negative reasons such as cost, energy and air pollution when searching for a parking space increase the importance of smart parking systems. In this study, a two-stage hybrid approach is proposed so that drivers can find a parking space that will consume the least time and energy. The first stage focuses on car parks having at least one free parking space located near the target address in an n diameter circumference, which are also open for business. An AVL tree-based hierarchical structure is created with driving time from the starting point to each car park and walking time from each car park to the destination, and it focuses on the most appropriate car park. In the second stage, the most suitable parking space is searched and made available, if found, in hierarchical parking monitoring system. In order to demonstrate the effectiveness of the approach, the results compared with hierarchical, hierarchical Binary Search Tree (BST) and non-hierarchical solutions in terms of energy and time performance are shown on a simulation. Proposed approach gave the best result with 99% energy efficiency. In addition, a dynamic cloud-based reservation system was proposed for the parking lot determined in the study.*

Keywords: *AVL tree, adjacency lists, smart cities, smart cities, cloud, internet of things*.

Received November 1, 2020; accepted November 24, 2021
https://doi.org/10.34028/iajit/19/5/11

1. Introduction

In large cities, traffic density and shortage of parking spaces are becoming more and more of a big problem. The increasing number of vehicles and the lack of parking spaces seemingly trigger this problem. For instance, start and end times of businesses or schools are around 8:30 and 17:30, respectively in Turkey. Heavy/peak traffic hours are experienced during and 30 minutes before and after the hours mentioned [7]. In the morning, the density of traffic in the direction of travel is high, while it is lower in the return path, and it is more or less similar in the evening hours.

Research show that in cities with a large number of vehicles, people spend around between 3.5 and 14 minutes to find a parking space [18]. The most important reason for this is that the drivers do not know where to park their vehicles on their way to their destination and this leads to traffic congestion. In addition, due to the congestion that occurs during these dense times, the time to find a parking space increases dramatically. This also leads to elongation of queues, loss of time and increase of vehicle costs due to searching and waiting.

The swift growth of IoT and its enormous capabilities make it useful to realize the goal of smart environment around us such as smart cities, smart education, intelligent transportation, smart healthcare, etc., [4]. IoT is heavily utilized in smart city applications. It is important here that the sensors, computers and in-vehicle devices or mobile devices communicate with each other in a robust and uninterrupted manner. In this study, questions such as which ways the vehicles would take from start to the destination and where they would park when they reach their destination are addressed to make determinations. A reservation-based approach is provided in order to ensure that the reserved places are vacant and the system works properly within the specified period of time. Thus, a hybrid structure was created. Road identification and parking spaces are determined dynamically using the AVL tree structure.

In the second part of this study, a literature review is made. The system proposed is presented in chapter 3 in detail and the results obtained are compared and presented in chapter 4. In chapter 5, conclusions and subsequent studies are discussed.

2. Related Work

In the age of Internet of Things (IoT) and smart city ecosystems, smart parking and related innovative solutions for more sustainable cities are required [1]. Parking space management in a city with an increasing number of vehicles is becoming a critical issue, especially in congested residential areas. The difficulty

of finding a parking space, especially during peak hours, can create mayhem for drivers [11]. In the literature, there are different solutions for smart parking systems.

In the study [16], a new cloud-based Smart Vehicle Parking System (SVPS) was proposed by means of ubiquitousVehicular Ad Hoc Network (VANETs). It offers a real-time, reservation-based solution. Thomas and Kovoor [19] also proposed a new prototype for the parking spaces of shopping centers, which are generally very busy at weekends. In order to address the issue of reserving the vehicle into the parking area, a genetic algorithm approach was utilized. The performances of the sensor types that would be used to detect the parking spaces in the shopping centers were examined. Atif *et al*. [2] offers an ongoing work agenda that contributes to new business solutions and the impact of cutting-edge research. It plays an intermediary role in advertising parking spaces on a common cloud platform with Park Service Providers (PSP).

Shin and Jun [17] mentions an approach that would assist drivers in finding the most suitable parking space, taking into account the real-time status of parking facilities in the city. In order to suggest the most suitable parking space, driving factors to the guided parking facility, walking distance from the guided parking to the destination, the expected parking cost and the traffic jam due to parking guidance are considered in the proposed algorithm. Bachani *et al*. [3], a hierarchic-based BST method was proposed for parking systems. First, the nearest parking space was searched for and then the nearest parking space in the nearest parking space was focused on. In terms of search time and energy efficiency of the approach, it was observed that non-hierarchical and hierarchical methods had been compared and better results had been obtained. Geng and Cassandras [8] included a study carried out in the garage of Boston University. A new approach to efficient use of parking spaces and guidance to drivers was mentioned.

Lan and Shih [12], a mass-sourced solution was proposed using sensors on smartphones to collect real-time parking availability information. A telephone-based system was designed to monitor a driver's route to determine when they would leave the parking space and focused on the efficiency of using a telephone to monitor the walking distance and a waist-mountedPersonal Dead Reckoning (PDR) method that can measure the driver's range of motion with high precision. As a result, it was stated that the user could achieve an accuracy of approximately 98% in estimating the walking distance and the total position error was approximately 0.48 m.

Chatzigiannakis *et al*. [5], security problems in the communication of IoT devices used in smart parking systems were highlighted and elliptical encryption method was used to increase security. It was pointed out that it was more efficient compared to other encryption methods. Giuffrè *et al*. [9] also proposed a smart parking system calledIntelligent Parking Assistant (IPA) for the management of parking spaces. It was assumed that advanced Information and Communication Technologies (ICT) solutions such as IPA, wireless networks and sensor communication would be used for the management of parking spaces. However, it is only at the preliminary stage as yet. In [14], smart parking system approach was looked upon from another point. Parking entrances and exits would be controlled by Radio Frequency Identification (RFID) readers, tags and barriers. This would significantly reduce personnel costs. Check-ins and check-outs would also be carried out in a faster manner and the congestion problem would be solved as cars would not have to stop. Chinrungrueng *et al*. [6] discussed the efficiency of the use of parking spaces. With Wireless Sensor Network (WSN), the vehicles according were grouped into their size and a more efficient parking was studied on.

Lu *et al*. [13] proposed a VANET-based Smart Parking plan (SPARK) system for large car parks. They monitored the entire parking space by means of the RSUs in the parking space. They used the data obtained for parking navigation, theft protection and the dissemination of friendly parking information. In their simulations, they demonstrated that they had managed to reduce time for parking search and save fuel. Khanna and Anand [10] proposed a IoT-based approach for the smart parking system. In the proposed system, the IoT module was used to monitor and report each parking space. In addition, a mobile application was developed for the end user to check and reserve the parking space. In [15], the results of a survey conducted for the needs of drivers from parking infrastructures were given. The latest trends in parking monitoring, parking reservation and dynamic pricing plans were discussed. In addition, how to integrate technologically advanced parking infrastructures was examined to benefit both drivers and parking operators.

In the studies examined above, various solutions have been proposed to solve parking problems in cities with high traffic density. The common goal of the studies carried out is minimizing the time people spend during parking, minimizing the energy used, reducing carbon emissions and preventing air pollution. The aim of this study is in line with the studies examined.

3. Methodology

A great deal of unnecessary time is consumed when searching for parking spaces and the traffic density increases while doing so. It also increases fuel consumption and carbon emissions, causing both material and environmental damages. The main objective of the application proposed is to determine which car park in the surrounding area could be used to

reach the target in the shortest time and the way to the destination. The vehicle will know where to go in advance and will take the most economical way and the fastest way to the parking space. Thanks to the reservation system created to make a booking in the parking space, drivers will be able to park in their own parking space without waiting. Thus, the congestion that would otherwise be created will be largely eliminated due to the hybrid structure provided. One of the most important issues here is the creation of a completely dynamic system.

First of all, in this study, "Cities Map Application" and "Open Data Portal" (ODP) services of Istanbul metropolitan municipality were used. In the following sections of the study, the word "ibbMap" will be used as an abbreviation instead of "Cities Map Application". The word "ODP" will be used instead of "Open Data Portal". ibbMap is an open source map service similar to Google maps. ODP, on the other hand, is a portal where information collected from IoT-based devices in the city is shared. There are data sets in different categories such as transportation, environment, life, mobility and economy. There are also dynamic web services such as parking capacity and traffic density.

To briefly explain the application, we first transfer the data we receive over ODP with the help of a web service to our cloud-based server in certain periods. In this way, available car parks are determined at the driver's destination. Then, the double-sided AVL tree algorithm is run for both the departure roads and for the parking spaces available in the car parks with minimum cost. According to the result, the most suitable parking place is reserved. In addition to the time that the vehicle can arrive for each reservation, +15 min. duration is defined. If the vehicle does not arrive during this time, the reservation is canceled.

Istanbul, the most populated city and most crowded traffic holder of Turkey, was selected for the application. Maltepe district was chosen as the plot region. The reason for this is the large number of parking spaces available in the region and that there are many alternative routes in the region for transportation. Therefore, the effectiveness of the study would be easier to understand. The map section of the interface design of the application is shown in Figure 1.

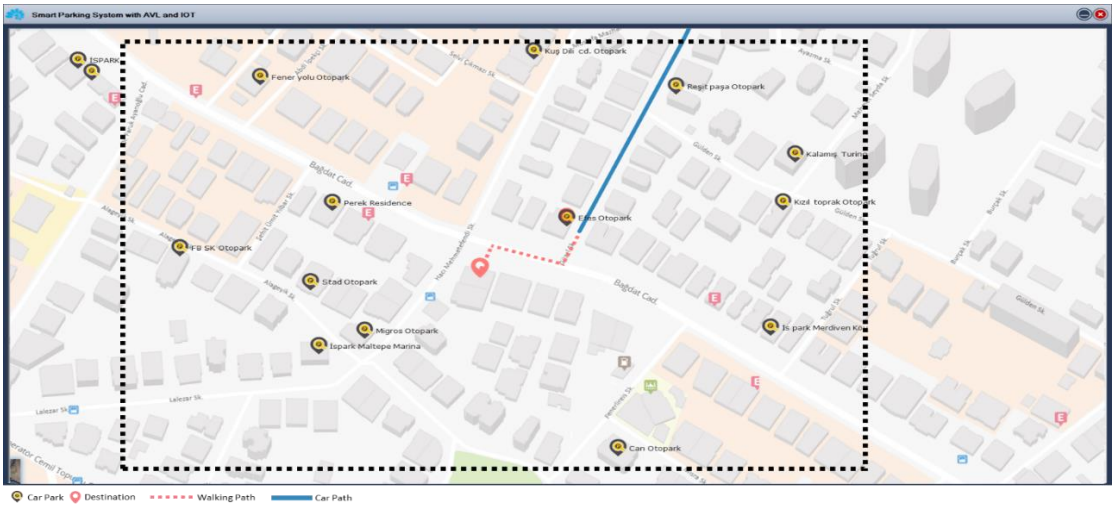


Figure 1. Region used in the study.

Figure 1 shows the area used for smart parking. The yellow and black markings indicated by the symbol P in Figure 1 are the carparks. The orange sign is the carpark where the vehicle will be parked. The sign shown in blue is the address you want to visit.

1. Destination address is entered into the application.
2. All car parks within the n-address of the destination address are searched. n is parametric. Its default value is 2 km.
3. Parking spaces and car parks outside the working hours are eliminated.
4. The driving time from the initial location to each car park (traffic intensity is effective) and the walking time from the car park to the destination address are taken from the API and placed hierarchically in the AVL tree at each step.
5. The AVL tree searches and determines the most suitable car park.
6. The reservation request for the most suitable car park is sent to the smart parking monitoring server on the cloud (the parking spaces are constantly updated on the server in real time).

Reservation is made by searching the nearest parking place for the vehicle concerned. The reservation will be canceled if the vehicle does not arrive at the parking place within the specified time or cancels the transaction.

3.1. AVL and BST

Tree structures are flexible structures. Even if they are not the best solution for a single purpose, the flexibility

of the tree structure for multiple solutions is the advantage of this structure. It uses truncated search performance engine to change search performance. It can be placed on the tree. Likewise, the parking spaces in these car parks can be placed in the tree based on their distance from the car park entrance, and a tree can be formed intertwined. Searching for car parks and parking spaces in such a structure is very flexible and efficient. However, not all tree structures are the same. Adding, deleting and searching processes are different, creating advantages and disadvantages in terms of efficiency. There are certain types such as Binary Tree, Binary Search Tree and AVL Tree. The Binary Search Tree is the Binary Tree, created with certain rules to increase the efficiency of search operations.

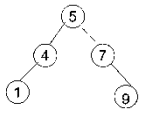
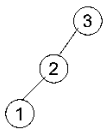
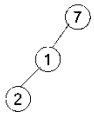
AVL is the initials of the inventors of the AVL Tree, "Adelson-Velsky and Landis". AVL is a special version of the binary search tree. The main difference between them is that AVL is a self-balancing tree. Balanced tree means that the difference between the heights of the right and left child for each node "I" in the tree is 0 or 1. So;

$$abs\left(height(left(i)) - height(right(i))\right) < 2 \quad (1)$$

The AVL tree must provide the Equation (1). The main purpose of the balancing process is an action to reduce the complexity of the algorithm. AVL trees eliminate the disadvantages of binary search trees. Adding in the AVL tree is like adding a regular binary search tree. After the addition, the AVL feature is corrected using left or right turns.

In Table 1, comparison of AVL and BST's complexity in the cases of search, addition and deletion as well as the tree structures are given. As can be seen in the tree structure, AVL is a very balanced tree. Although the complexity of AVL and BST is the same in average cases, in the worst case, AVL appears to be more advantageous.

Table 1. Time complexity in big O notation.

Tree Structure						
Algorithm	AVL Tree		Binary Search Tree		Binary Tree	
Case	Average	Worst Case	Average	Worst Case	Average	Worst Case
Search	Olog *n*	Olog *n*	Olog *n*	*O* (*n*)	*O* (*n*)	*O* (*n*)
Insert	Olog *n*	Olog *n*	Olog *n*	*O* (*n*)	*O* (*n*)	*O* (*n*)
Delete	Olog *n*	Olog *n*	Olog *n*	*O* (*n*)	*O* (*n*)	*O* (*n*)

The AVL tree needs more memory than BST. Because the height factor information must be stored for each node.

3.2. Design of the Proposed Study

Studies carried out in this section of the study is explained in two sections. In the first stage, how the most suitable car park is identified, the method flow diagram and algorithm used are explained. In the second stage, the reservation process for the parking space identified is explained.

3.2.1. Determination of Optimal Car Park

There are many search algorithms in the literature. Search algorithms are used to search for information on various data structures. Methods such as linear and binary search etc. are widely used. In this study, a new search approach that uses the AVL tree is proposed. Instead of searching all the car parks, the search is made for the car parks on the fastest route from the starting point to the destination and from the car park to the destination place, then make a reservation for the nearest car park in this route. AVL algorithm is used to make the most efficient search. The flow diagram of the system proposed is shown in Figure 2.

As shown in Figure 2, the address of the vehicle is obtained first. Then, the destination address is received. The ibbMAP provides longitude and latitudevalues for these addresses. In all transactions, longitude and latitude are translated. All car parks within n-distance of the destination address are queried (this value was accepted to be 2 km). Information on ibbMAP car parks is given in Table 2.

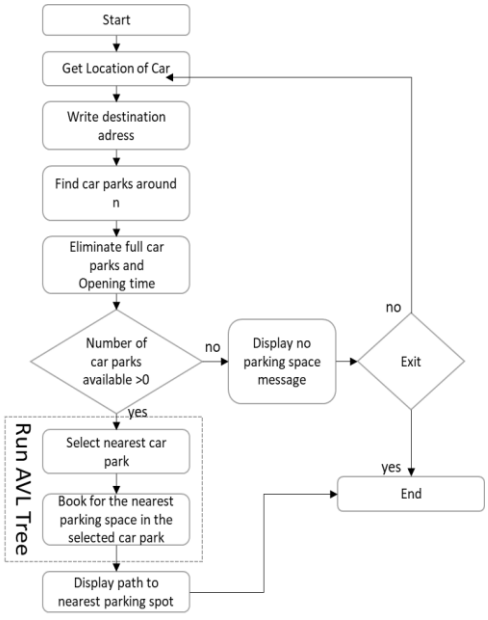


Figure 2. Flowchart of smart parking system with AVL tree.

According to the result of the query, the car parks that are completely full and closed are eliminated. For each car park located, information on distance and travel time from current location to the car park and from the car park to the destination is received. The parameters used to obtain this information are given in Table 3.

Table 2. Characteristics of the parking lot near the n diameter of the address.

Parameter	Explanation
lat	Contains latitude information of the address.
lng	Contains the longitude of the address.
id	It contains a unique value for the coordinates.
name	Provides a recognizable name. For enterprise results, it is usually the business name.
Opening-hours	The boolean value that indicates whether the location is currently open.
icon	Contains the URL of a proposed icon to show the result to the user.
Pluce-code	Post code
rating	It includes a rating of 1 to 5 based on the collected user data.
types	Contains a number of property types that define the given result.
vicinity	Contains the property name of a nearby location.
Place-id	The unique address of the address. Contains 27 characters of text.

Table 3. Information between the two objectives.

Parameter	Explanation
Destination-addresses	It contains the address information of the destination coordinates.
Origin-addresses	It contains the address information of the starting position coordinates.
distance-text	It includes the distance between the start and the end as text xyz km.
distance-value	It contains the distance between start and end as integer in meters.
duration-text	It includes the time between the start and the end as text in xyz mins format.
duration-value	Includes the time between start and end in seconds.

In this study, evaluation was made based on time, not distance. It would be more advantageous to use time instead of distance, as ibbMAP takes the traffic time into consideration. This is due to the fact that the shortest road is very busy in cities with high traffic. If it has already set off at an hour with low traffic density, the shortest route will be the route to reach the destination in the least time. Algorithm choose the shortest route and reduce the time it will wait in traffic. In this way, algorithm not only reduce fuel savings and carbon emissions, but also plan to avoid traffic stress.

- Start: Start address.
- Finish: The destination address.
- s: Time to travel by car between the starting address and the parking space.
- f: The walking time between the car park and the destination address.
- P: Parking
- m: (s+f) duration.
- k: Parking space.

Algorithm 1: Finding the optimal car park with AVL

```
1:  function insertAVL(k,e,T)
2:        Input:   A key-element pair, (k,e) and an AVL tree, T
3:        Output: An update of T to now contain the item (k,e)

4:        v ← IterativeTreeSearch(k,T)
5:        if v is not an external node than
6:             return "An item with key k is already in T"
7:       Expand v into an internal node with two external-node
8:  children
9:        v.key ← k
10:       v.element ← e
11:       v.height ← 1
          rebalanceAVL(v,T)
12:
13: function reblanceAVL(v,T)
14: Input: A node, v, where an imbalance may have occurred
15: in an  AVL tree, T
     Output: An update of T to now balanced
16:
17: v.height←1+max{v.leftChild().height,v.rightChild().height}
18:       while v is not the root of T do
19:            v ← v.parent()
20    if (v.leftChild().height - v.rightChild().height) > 0 then
21: Let y be the tallest child of v and let x be the tallest
22: child of y
23:     v ← restructure(x)      //tree node restructure operation
24: v.heigth←1+max{v.leftChild().height,
    v.rightChild().height}
25:
26: function search(k, T)
          if  k.leftchild != null then
               return search(k.leftchild,T)
          else
              return k


27: main:
28:  // map: Istanbul metropolitan municipality City Map API
29  Input: Finish: Destination address, Start:current location,
30: m:    2000 meters,f: walking time,s: drive time,key:"Car
31: Park"–keyword, T: tree
32        Output: P: optimal Car Park

33:       Finish ← 'write destination address:'
34:       Start ← map.GetLocation() // return longitude,latitude

35:  For each car park location i map.Nearby(Finish, key,m)
36:  if (i.emptyparkingslot)> 0 and (i.workinghour)==true then
37:  s= Map.Drive(Start , i)   // time from current location to
38: parking lot
39:            f= Map.Drive(i , Finish) // time from parking to
40: destination
41:            insertAVL(i,(s+f),T)
42:          End if
43:    End For

44:    P=Search(k,T) //Search optimal car park

45:    if (P) != null then
46:     print "Optimal Car Park:" + P
47:    else
48:     print "car park not found"
```

Figure 3 shows the hierarchical structure of the study. In the first step, m values are added to the tree. A new node is created that contains the value to be added.

Starting from the root, the new node to be added on the tree is compared with other nodes. If the new node is larger than the comparable node, it moves to the right or left. If the node being compared is the last node, a new node is added to this node as a child. The advantage of AVL trees is that it balances the tree after the insertion process is finished. This reduces search complexity to a minimum. See finding the optimal car park with AVL in Algorithm (1);

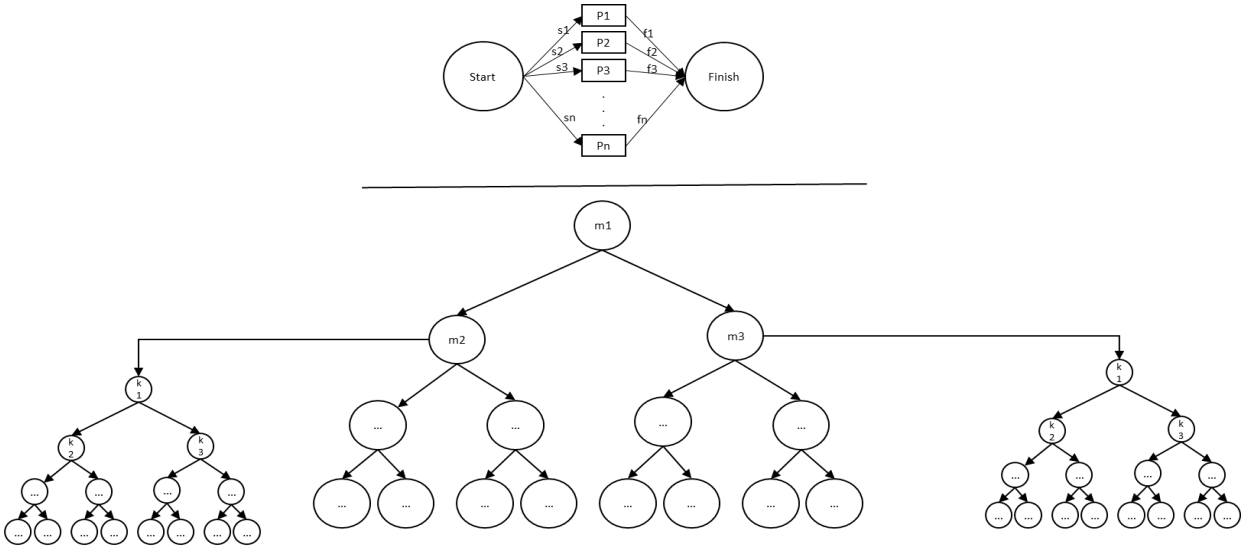


Figure 3. Hierarchical structure.

After the AVL tree is created, the search for the road to be traveled is carried out as soon as possible and the most suitable parking space is decided upon. After finding the most suitable parking space, the parking space is booked in the second stage.

3.2.2. Dynamic Parking Reservation

In the second stage of this study, cloud based parking monitoring system was studied. Various sensor types are available for monitoring vehicle parking spaces. Light Dependable Resistor (LDR) and Infra-Red (IR) sensors are the most widely used types. Based on the literature review made, the accuracy of LDR is highly affected by the change in light intensity throughout the day. The shadow of the object further enhances the mis-detection of the LDR sensor and therefore precisely affects its accuracy [3]. It is, on the other hand, costly and too complicated to control the parking capacity with sensors. Therefore, the Open Data Portal (ODP) was used in this study for the detection of empty parking spaces instead of sensors. This portal provides access to parking spaces available in the city via a web service.

In Table 4, the web service's url, return parameters, data types and descriptions are given. First of all, all parking information was extracted from https://api.ibb.gov.tr/ispark/Park and saved in the cloud database. The empty capacities of the car parks were updated dynamically fetching the detailed information in certain periods for each registered car park.

Table 4. Car Park web service return data structure.

Parameter	Description
ParkID	Includes Park Id information. Unique area
ParkName	Includes park name information
Latitude	Latitude information of the park
Longitude	Longitude information of the park
Capacity	Total capacity of the park
EmptyCapacity	Empty capacity of the park
ParkType	Type of the park (on the road, open, closed)
District	District where the park is located
UpdateDate	Update date and time of the information
Workinghours	Working hours of the park
Free Parking	Free parking time (min)
Subscription Fee	Monthly Subscription Fee
Address	Car park address
AreaPolygon	Park polygon information
Recipe	Properties of the Tariff Information object. Parking Holds the name information of the Tariff slice.
Price	Properties of the Tariff Information object. money
LocationName	Location information of the park. One and above parks are connected to a location.

Figure 4 shows the smart parking monitoring and booking system. The IBB server maintains detailed information of all car parks in the city in real time. It also provides an ODP for users who want to access this data through a web service agent. This data is transferred to our database Cloud based-Smart Parking Server (CB-SPS) in certain periods, instead of controlling the car park and accessing the portal every time a request is made by the users. This way, the network density is decreased by not making too many queries to the web service. This is tolerable even though additional memory costs are included. When the users require a reservation, the available capacity of the relevant car park is lessened via CB-SPS and the relevant reservation table is updated according to the

time created by adding another 15 minutes (additional waiting time) in addition to the time the user will take to reach the parking space. If the user does not arrive within the expected period, the available capacity is increased and the reservation is cancelled.

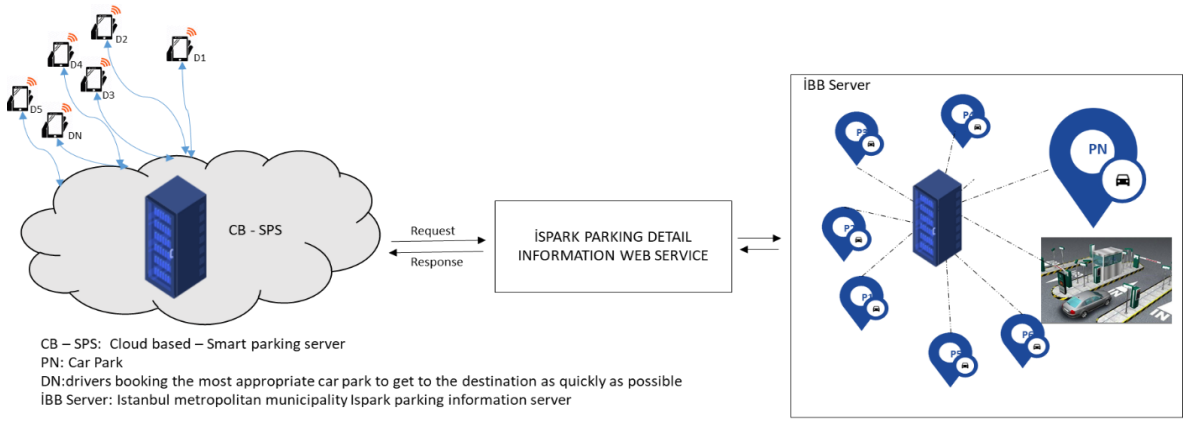


Figure 4. Smart parking monitoring and booking.

4. Experimental Results

Based on the studies made in literature, three different search methods were analyzed in this field. These are non-hierarchical, hierarchical, and hierarchical BST methods. The hierarchical BST method yields the best results. The BST tree is not a balanced tree, and in some cases very poor results can be achieved, which is a great risk. For example, if you placed a series of numbers that were usually large to small in the BST tree, the height of the tree would increase to the maximum size and would perform poorly in searches. AVL trees are balanced trees. The height difference between the two branches is maximum 1, which prevents bad undesired results. While the BST algorithm has a complexity of O (log (n)) in the average case, it can be O (n) in poor cases, as well. On the contrary, the complexity of AVL Trees in both cases is O (log (n)).

In the non-hierarchical method, the data are stored in linked lists. In other hierarchical methods, the data are stored in a hierarchical data structure. The only difference is that the hierarchical structure is created when adding data. In Figure 5, the cases where these four methods are used to find the car parks on the route, which are accessible and available for parking in the shortest time are compared. In non-hierarchical algorithms, the number of parking lots and the number of nodes visited increases linearly as the parking space increases. Hierarchical and hierarchical BST methods were not affected by the increase in the number of nodes. However, since they do not have a balanced structure, they do not give as good results as AVL, the method we recommend.

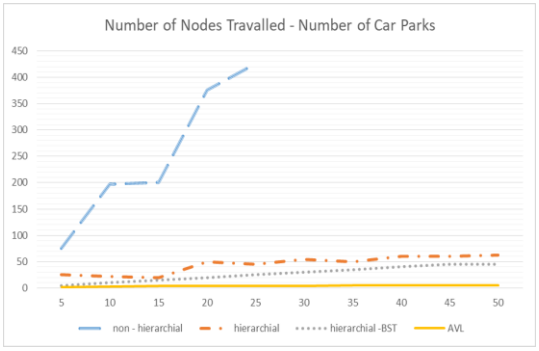


Figure 5. Number of nodes travalled vs. number of car parks.

One of the most important criteria used in the comparison of search algorithms is energy. The more exploration search is made, the more energy is consumed and the lower the efficiency would be. “s” is the time that elapses from the starting point to the destination, with each of the car parks near the destination. “f” is the walking time between the car park and the destination. “m” is the total time taken on each path from the start to the destination. “k” is the time spent traveling to each parking space.

$$m_n = s_n + f_n \tag{2}$$

E_S represents the total energy and is the sum of the m and k values. That is, the total energy is equal to the sum of the nodes created on the way to the destination and the nodes created when searching for a parking space.

$$E_S = \sum_{j=0} m \sum_{i=0} k \tag{3}$$

E_t refers to the energy used. n is the number of nodes navigated during the search. The sum of the n values yields the energy used.

$$E_t = \sum_{j=0} n \tag{4}$$

E_r represents the remaining energy. The remaining energy is equal to the difference between the total energy and the energy used. In this study, we remaining energy is used (%).

$$E_{r(\%)} = \frac{E_s - E_t}{Es} x100 \quad (5)$$

The remaining energy graph is given in Figure 6. In the four comparative approaches, initial energy assumed to have been consumed is 100%. Visiting each node causes a space of energy and reduces the efficiency of the system. In non-hierarchical methods, it seems that all the nodes consume too much energy, as some of the nodes are viewed more than once.

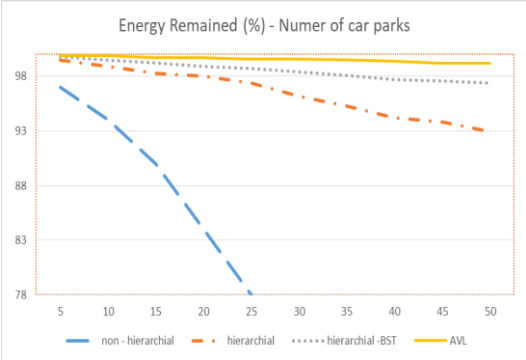


Figure 6. Energy remained (%) vs. number of car parks.

The hierarchical method and the hierarchical BST method use up far less energy than the non-hierarchical method does. AVL, which is the method recommended, yields the best results compared to other methods. Energy usage is very low and efficiency is very high. The energy of BST, which is the best method compared to others, decreases to 97% and AVL remains at 99%.

$$T_{avg} = \frac{\sum_{j=0}^{n} s + \sum_{i=0}^{n} m}{n} \quad (6)$$

$$T_{min} = (s+m)_{min} \quad (7)$$

T_{avg} gives the average value of all total driving and walking times from the starting position to the destination. The data type of the times is minutes. The T_{min} value gives the shortest driving and walking time in total from the starting position to the destination.

$$P_{avg} = \frac{T_{avg} * 100}{T_{min}} - 100 \quad (8)$$

P_{avg} is how much efficient the time to go from the start to the destination is as much as a percentage of the average of the times on all routes.

The large number of car parks that are available near the target location helps to create more alternatives. As the number of car parks around the target location increases, the time between the minimum arrival time to the destination and the average arrival time gradually decreases.

Figure 7 shows the relationship between the average arrival time and the minimum arrival time of the routes where the number of km and traffic density are close to each other but the number of car parks nearby is different.

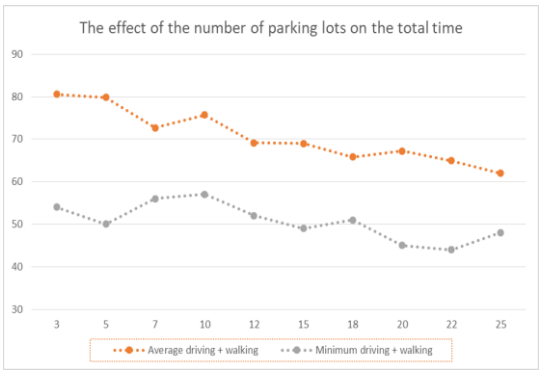


Figure 7. The effect of the number of parking spaces on the total time.

5. Conclusions and Future Studies

The development of smart cities in the world has accelerated rapidly. Researchers continue to work more and more each day for the problem of traffic density and parking space, which are important topics of smart cities. They tried to find a solution to the problem of searching for a parking space with different perspectives and methods, and as a result, they tried to prevent carbon emissions, increased stress, loss of energy and time.

In this study, an innovative different perspective is presented for smart parking systems and reservation process. The design of the study, the extraction of data through API, and the data accessed are explained in detail. The comparison of the AVL approach used in the proposed method made with the other methods in the literature is also presented. According to the simulation studies, non-hierarchical methods yielded the worst results, whereas hierarchical and hierarchical BST yielded better results. The AVL tree method proposed, proved to be the most efficient approach in terms of energy use and exploration compared to other approaches. In addition, a dynamic reservation system has been created. Thus, the time to search, wait and find the parking space is eliminated. The multivariate parking data set collected during the study, the detailed literature review on smart parking systems, the dynamic reservation system, the proposed method, and the 99% energy efficiency obtained in the experiments made a serious contribution to the literature.

At the end of this study, large amounts of data will be collected in databases after a certain period of time. It is thought that this data would be a basis for smart city applications, especially in many other subjects. In addition, it is predicted, for future studies, that a better and more successful system can emerge by using regression models or classification methods to create appropriate datasets from the data obtained. It is also contemplated that deep learning techniques may be used on the data obtained.

References

[1] Al-Turjman F. and Malekloo A., "Smart Parking in Iot-Enabled Cities: A Survey," *Sustainable Cities and Society*, vol. 49, pp. 101608, 2019.

[2] Atif Y., Ding J., and Jeusfeld M., "Internet of Things Approach to Cloud-based Smart Car Parking," *Procedia Computer Science*, vol. 98, pp. 193-198, 2016.

[3] Bachani M., Qureshi U., and Shaikh F., "Performance Analysis of Proximity and Light Sensors for Smart Parking," *Procedia Computer Science*, vol. 83, pp. 385-392, 2016.

[4] Bashir A. and Mir A., "Lightweight Secure MQTT for Mobility Enabled E-Health Internet of Things," *The International Arab Journal of Information Technology*, vol. 18, no. 6, 773-781, 2021.

[5] Chatzigiannakis I., Vitaletti A., and Pyrgelis A., "A Privacy-Preserving Smart Parking System Using an Iot Elliptic Curve Based Security Platform," *Computer Communications*, vol. 89-90, pp. 165-177, 2016.

[6] Chinrungrueng J., Sunantachaikul U., and Triamlumlerd S., "Smart Parking: an Application of Optical Wireless Sensor Network," *in Proceeding of International Symposium on Applications and the Internet Workshops*, Hiroshima, pp. 66, 2007.

[7] Dener M., Akcayol M., Toklu S., and Bay O., "Zamana Bağli Dinamik En Kisa Yol Problemi Için Genetik Algoritma Tabanli Yeni Bir Algoritma," *Journal of the Faculty of Engineering and Architecture of Gazi University*, vol. 26, no. 4, pp. 915-928, 2011.

[8] Geng Y. and Cassandras C., "A new "Smart Parking" System Infrastructure and Implementation," *Procedia-Social and Behavioral Sciences*, vol. 54, pp. 1278-1287, 2012.

[9] Giuffrè T., Siniscalchi S., and Tesoriere G., "A Novel Architecture of Parking Management for Smart Cities," *Procedia-Social and Behavioral Sciences*, vol. 53, pp. 16-28, 2012.

[10] Khanna A. and Anand R., "IoT based Smart Parking System," *in Proceeding of International Conference on Internet of Things and Applications (IOTA)*, Pune, pp. 266-270, 2016.

[11] Kizilkaya B., Caglar M., Al-Turjman F., and Ever E., "Binary Search Tree Based Hierarchical Placement Algorithm for IoT Based Smart Parking Applications," *Internet of Things*, vol. 5, pp. 71-83, 2019.

[12] Lan K. and Shih W., "An Intelligent Driver Location System for Smart Parking," *Expert Systems with Applications*, vol. 41, no. 5, pp. 2443-2456, 2014.

[13] Lu R., Lin X., Zhu H., and Shen X., "SPARK: A New VANET-Based Smart Parking Scheme for Large Parking Lots," *in Proceedings of IEEE INFOCOM*, Rio de Janeiro, pp. 1413-1421, 2009.

[14] Pala Z. and Inanç N., "Smart Parking Applications using RFID Technology," *in Proceedings of 1st Annual RFID Eurasia*, Istanbul, pp. 1-3, 2007.

[15] Polycarpou E., Lambrinos L., and Protopapadakis E., "Smart Parking Solutions for Urban Areas," *in Proceedings of 14th International Symposium on A World of Wireless, Mobile and Multimedia Networks*, Madrid, pp. 1-6, 2013.

[16] Safi Q., Luo S., Pan L., Liu W., Hussain R., and Bouk S., "SVPS: Cloud-Based Smart Vehicle Parking System Over Ubiquitous Vanets," *Computer Networks*, vol. 138, pp. 18-30, 2018.

[17] Shin J. and Jun H., "A Study on Smart Parking Guidance Algorithm," *Transportation Research Part C: Emerging Technologies*, vol. 44, pp. 299-317, 2014.

[18] Shoup D., "Cruising for Parking," *Transp Policy*, vol. 13, no. 6, pp. 479-486, 2006.

[19] Thomas D. and Kovoor B., "A Genetic Algorithm Approach to Autonomous Smart Vehicle Parking System," *Procedia Computer Science*, vol. 125, pp. 68-76, 2018.

Hikmet Canli was born in Ordu, Turkey in 1992. He received the B.S. (in 1st place) in Computer engineering from Duzce University, Duzce in 2015 and M.S. degree in computer engineering from Duzce University, Duzce, Turkey in 2017, and he is currently PhD. student in computer engineering from Duzce University, Duzce, Turkey. His research interest Internet of Things, Cyber Security, Computer Networking, Deep Learning, Machine Learning and Data Mining. His teaching areas cloud computing, information and network security and programming languages at undergraduate degree.

Sinan Toklu received his B.Sc degree in computer engineering in 2004 from Eastern Mediterranean University, G. Magosa, TRNC and a M.Sc. degree in computer engineering in 2007 from Gazi University, Ankara, Turkey and his Ph.D. in electric-electronic education in 2013 from the Gazi University, Ankara, Turkey. His research interests include wireless sensor networks, energy optimization, Internet of Things, Deep Learning and Smart City.