Contents lists available at ScienceDirect

Applied Soft Computing

journal homepage: www.elsevier.com/locate/asoc

Dijkstra solution algorithm considering fuzzy accessibility degree for patch optimization problem

Resmiye Nasiboglu

Dokuz Eylul University, Faculty of Science, Department of Computer Science, Izmir, Turkiye

ARTICLE INFO

Article history:
Received 2 March 2021
Received in revised form 3 July 2022
Accepted 20 September 2022
Available online 5 October 2022

Keywords:
Public transport network
Optimal path recommendation
Fuzzy accessibility
Dijkstra algorithm

ABSTRACT

One of the most widely used applications for urban public transport passengers, especially for tourists, are route recommendation applications. These applications are generally based on criteria such as the shortest route, the least number of line transfers, and the least travel time. In some recent studies, there are fuzzy models that consider qualitative indicators such as comfort and accessibility in addition to these criteria. In our study, a new model and a new FuzzyDijkstra algorithm based on the Dijkstra algorithm are proposed to produce routes with the shortest path, the least transfer criteria, as well as the highest degree of accessibility. In the calculation of the accessibility degrees of the routes, concepts such as fuzzy stop neighborhoods, fuzzy intensity inside the bus, and fuzzy line accessibility were used. In the proposed algorithm, fuzzy penalty values are used to produce higher ranked routes. The validity of the proposed model and algorithm has been tested using Izmir city public transport network data. Optimal routes were generated for 100 source-target bus stop pairs that were handled randomly in the calculation experiments. Although the indicators such as the average distance traveled by the vehicle and the average number of transfers are approximately the same in the routes produced by the algorithms, it is seen that the FuzzyDijkstra algorithm gives better results compared to others with an average of 11.5% higher degree of accessibility and an average of 13.3% less walking distances between the transfer stops. In contrast, the FuzzyDijkstra algorithm used an average of 2.2% more time to solve. It is clear that this time delay, the actual value of which is less than 0.5 s, will not affect the practical use of the algorithm much.

1. Introduction

Today, one of the most important problems of cities is the traffic congestion problem. The traffic congestion problem started from environmental pollution and causes various negativities from the psychological conditions of people and the inefficient use of their time. In this respect, it is very important to plan the urban public transport network based on scientific foundations [1]. In the attempts made to solve this problem, the use of public transportation vehicles is predominant. It is important to encourage people to change their daily habits and use public transport. Route recommendation applications created to go from one place to another in the city are one of the most effective tools that serve this purpose. In this direction, various mobile applications have been developed especially in recent years. In the paper [2], an evaluation methodology for route planning applications in public transportation was given and existing web applications that aim people to use public transportation effectively were evaluated from various aspects. It is concluded that the factors such as ease of use of the application, suitability of the route recommended, the producing time of the optimal route are the most important factors for the users.

The solution performance of the algorithms used in route suggestion systems depends very much on the public transport network models and spaces on which they are based. There are many studies in the literature that deal with the structures of public transport networks (PTN) from a topological and theoretical perspective. Some of these studies generally deal with static network topology structures and are aimed at making network analysis based on various features. Some group studies have focused on making dynamic analyzes on smart card data used in urban transportation. Other studies include the shortest route, the least time, the least transfer, the most preferred route for public transport users. Costa et al. in the study [3], which is one of the studies based on the analysis of static network structure, examined the measurements and basic models that can be used in network topologies in general. The study includes representation of complex networks in terms of paths in measurement spaces, correlation analysis between some of the traditional measurements, perturbation analysis as well as multivariate statistical analysis for feature selection and network classification. In the studies [4,5], various topology spaces used

E-mail address: resmiye.nasiboglu@deu.edu.tr.

https://doi.org/10.1016/j.asoc.2022.109674
1568-4946/© 2022 Elsevier B.V. All rights reserved.

for public transport networks have been examined and useful properties that can be used in these spaces are specified. Based on these PTN features, analyzes of 14 major cities in the world were made. In the study [6], the structure of the public transportation system in Curitiba, Brasilia is analyzed with a static network topology approach. The transport network is considered as a P-space topology and modeled as a complex network with the exact geographic coordinates of the bus stops. A comparative analysis of the results obtained with 3 similar cities in China was made. Sienkiewicz and Holyst [7] analyzed the public transportation systems in 22 Polish cities. The sizes of these nets range from 152 to 2881. The distributions of path lengths in networks are given by asymmetric, single-mode functions. The properties of networks such as clustering, diversity and range were examined.

Xu et al. [8] reported the statistical properties of three bus-transportation networks (BTN) in three different cities of China. Network properties including degree distribution, clustering, and average path length in different network topologies are studied. In addition, a weighted network representation was created for BTN with the lines mapped to nodes and the number of common stations with weights between lines. In the study [9], based on the P-space concept, the public transport networks of 330 cities in China were examined. Statistical results showed that the degree distribution of all networks can be described with an exponential function. It has been revealed that the development of the public transport network is in the form of randomly generated connections over time instead of preferentially created connections.

In the study [10], the bus network including 722 lines and 5421 bus stops in Beijing was discussed and mainly L and P spaces were used to analyze the network topology properties. In the article, using L-space in the first stage, it has been shown that Beijing's bus network is an unscaled network and that more than 99% of the nodes have degree less than 10. In the second step, the P-space was used to analyze the transfer properties. It has been calculated that the average number of transfers of the network is 1.88 and the percentage that can be reached with a maximum number of 4 transfers between any two pairs of stops is 99.8%. In the study [11], bus lines in Tai, China were modeled and simulated using the P-space and the L-space approaches. By calculating the adjacency matrix and complex network topology properties, it was concluded that the Tai city network has small world character with short average path length and high clustering coefficient. In the study [12], theoretical analyzes and experimental tests were performed on degree distribution, average path length (APL) and clustering coefficient. It has been stated that any bus transportation network has small world characteristics, that is, short APL and high clustering coefficient, indicating that this bus network is efficient with low cost and low number of transfers.

In the studies [13,14] empirical research was conducted on the urban bus transport networks of four major cities in China. To reveal the randomly arranged architecture of bus networks, new measures such as the sum of mean degree–degree correlations of the closest neighbors and the edge-degree averages between the closest neighbors have been proposed. Zhu et al. [15] developed an evolutionary model of the bus transport network in the B-space. The bus transportation networks of Hangzhou and Nanjing cities in China have been simulated using this model. Kang [16] discusses the shortest path problem when the edge lengths of the oriented non-cyclic graph are given as interval numbers. In order to process the data in the form of intervals, minimax criterion was applied to the "regret" values. The original objective function with the interval coefficient has been transformed into the least-maximum-value worst-case regret finding function called the minimax regret criterion. An intuitive algorithm that takes advantage of features has been developed to reduce computation time and improve solution quality.

Other group studies are studies on the best route suggestion between specific source and destination points using the city public transport network. As an example of these studies, a min–max version and computational complexity were analyzed to solve the shortest path problem of Dijkstra's algorithm in a directed weighted hypergraph [17]. Bozyigit et al. [18] proposed an approach to recommend the route consisting of the minimum number of transfers in public transport use. In this approach, the P-space model is used, and the Breadth First Search approach is used to construct the ideal route based on the Pareto optimal solution on the network. Bozyigit et al. in [19] mentioned that the classic Dijkstra algorithm is inefficient in public transport route planning because it ignores the number of transfers and walking distances. In the study, the modified Dijkstra Algorithm has been proposed by adding the penalty function. The proposed algorithm has been tested based on Turkey's Izmir city transport network and the data were compared with the classic Dijkstra's algorithm results. It has been observed that the proposed algorithm is very efficient in terms of the least number of transfers. Zhao et al. [20], considering the travel time uncertainty caused by the traffic congestion situation, the shortest transmission time algorithm and the shortest passageway algorithm are designed in the stochastic transition network. The study also considers transfer times, travel time and cost of each transit route plan, based on the travel psychology analysis.

The optimal route recommendation models in the literature are mostly built on the static structure of the public transport network. However, recently in the public transport network a huge amount of dynamic data is collected through systems such as Automatic Vehicle Location (AVL), Automatic Passenger Counting (APC), Automatic Fare Collection (AFC) etc. These systems are becoming more and more common in urban public transport systems in the developed world. However, the data obtained from these systems can have wide-ranging applications in public transportation far beyond design applications. Of particular interest in planning public transport is the opportunity to leverage these increasingly ubiquitous databases to develop a better picture of how public transport systems work and are used. In some cases, although there are significant limitations to the detailed attributes typically obtainable in these systems, better estimates of certain performance metrics and usage attributes can be made at lower cost than traditional data collection methods.

The review article [21] examines various aspects of smart card data usage in the context of public transport. The hardware and information systems required to operate these tools are specified, privacy concerns and legal issues related to the spreading of smart card data, data storage and encryption are discussed. Various uses of data are demonstrated at three management levels: strategic (long-term planning), tactical (service adjustments and network development) and operational (passenger statistics and performance indicators). Wilson et al. [22] explain two applications that focus on system usage and passenger behavior, developed jointly between MIT and Chicago Transit Authority (CTA), using the CTA's AFC and AVL systems. Specific applications are discussed for the estimation of passenger origin–destination matrices for the CTA rail system and for the prediction of road selection patterns for CTA rail passengers. Bagchi and White [23] considered the potential role of smart card data for travel behavior analysis. The smart card experience in the UK, the retired privileged pass at Southport, Merseyside, and the case experience of the commercially operated program in Bradford were reviewed. Morency et al. [24] suggest various measures regarding the variability of travel behavior of public transport users. Analyzes are performed with smart card data collected over a ten-month period. The daily boarding and transit network behaviors are analyzed on the basis of travel days. Using data mining techniques, travel days are classified according to the similarity of ride intervals.

There are several studies in the literature to predict alighting stops from information in smart card data. In the studies [25–27], the travel chain method was used to estimate the alighting stops and to create the OD matrix for Izmir. In the study [25], the electronic toll collection data of urban public transport system in Izmir, Turkey are handled. In the study, an origin–destination (OD) matrix showing the boarding and arrival zones of the passengers was created and interpreted by estimating the alighting information from the smart card data containing only the boarding information. After estimating the alighting stop information, the inside bus density are estimated based on lines [27,28]. A methodology for estimating the public transport OD matrix from smart card and GPS data for Chile, Santiago is presented in the study [29]. By applying the proposed method to two 1-week datasets obtained for different time periods, an estimate of the time and location of alighting for more than 80% of the boarding operations could be made from information about the time and location of getting on public transport. In the study [30], a model is presented to predict the destination for each individual who gets on a bus using a smart card. In the implementation of the model, a success rate of 66% was achieved for arrival estimation and up to 80% during peak hours.

Studies on the creation of origin–destination matrices for metro, train, etc. has also been made for transport data as well as bus transportation data. Compared to the bus network, although there is less variety of lines and stops in metro and train networks in general, such studies are sufficiently included in the literature. Barry et al. [31] presented a methodology for estimating the station-to-station OD matrix using MetroCard information in New York, USA. This estimation was done by sorting the MetroCard boarding information by serial number and time, and then subtracting the order of journeys for each MetroCard and the station used at the start of each journey. The algorithms used in forecasting are based on two basic assumptions. First, a high percentage of passengers return to the destination station of their previous trip to begin their next trip. Second, a high percentage of passengers finish their last trip of the day at the station where they started their first trip of the day. There are many other studies in the literature on subway and train networks [32–35].

With the widespread use of mobile devices, optimal route suggestion studies and applications have also become widespread. In these studies, the models for the most suitable route proposal considering the travel priorities of the passengers are discussed. In most of the studies, criteria such as the shortest route passing through the least number of stops between certain origin and destination points, the least travel time, the transfer by changing the least number of lines are considered. Some models also consider the shortest time criterion. In most of these studies, the stops or lines used are not evaluated qualitatively. However, Nasibov et al. [26] proposed a fuzzy approach to optimal travel planning in public transport. In this study, when the boarding and alighting stops are selected, a model is created that offers the most appropriate route suggestions. In the model proposed in the study, suitability concepts for stops are defined. Each stop was evaluated from the points of view such as proximity in terms of physical distance, intensity of boarding and alighting, and intensive crossing of the lines. The evaluations were carried out by grading with fuzzy logic approach. From all these fuzzy degrees, a general suitability degree for the stop was calculated and the route planning was made considering the stop suitability degrees.

There are also some other approaches using fuzzy concept in transportation network. For example, Mahdavi et al. [36] handle a graph with fuzzy distance for every edge, and propose a dynamic programming approach to solve the fuzzy shortest chain problem. Deng et al. [37] propose a generalized Dijkstra algorithm to handle shortest path problem with edges with fuzzy length. The graded mean integration representation of fuzzy numbers is used to adapt the classical Dijkstra algorithm to perform on fuzzy edges. Ji et al. [38] handle the shortest path problem with fuzzy arc lengths, and a hybrid intelligent approach using genetic algorithm is provided. Erbao and Mingyong [39] consider the open vehicle routing problem with fuzzy demands. A fuzzy chance-constrained program model is designed based on fuzzy credibility theory. Demands of customers are determined as fuzzy numbers and the “chance” of the next customer to receive service is calculated based on this fuzzy demand. Jovanovic et al. [40] calculate an Index of Performance (IP) value for each link, taking into account the operator costs, passenger costs and local environmental status variables. These values are given as fuzzy numbers. IP values are calculated as output of the ANFIS (Adaptive Neuro-Fuzzy Inference System) model. Finally, the maximum spanning tree of the network is calculated with the Kruskal’s algorithm, taking into account the IP values of the links. The model has been tested on a part of the public transport system in Belgrade. Cirovic et al. [41] also use the ANFIS model to determine the link performance in the transport network, but considering input variables such as logistics operating costs, exhaust emissions and noise. After obtaining the link performances via ANFIS a modified Clark–Wright algorithm was used to find the optimal route. The proposed model was tested on a network which simulates the conditions also in Belgrade.

In the route planning in previous studies, although the fuzzy preference degrees for the stops are considered, the lines are considered only with 0/1 logic in terms of suitability, i.e. it is only based on whether there is a line between certain stops or not. However, at a time when the contagious pandemic epidemic is important, taking into account the bus occupancy rates on the lines has become one of the critical factors. Therefore, it is very important to consider this factor in recommending a certain route. In the study [42], automatic fare collection, automatic vehicle location and general transit feed specification data were used to calculate vehicle occupancy in public transportation. Using data collected from the urban transport network in Hague, Netherlands, the spatio-temporal load profiles are visualized through space–time seat occupancy charts, creating a compact and powerful reference for improving services to public transport operators. Nasiboglu [28] also used vehicle occupancy degree to formulate the decision-making strategy of the passenger by the OWA (Ordered Weighted Averaging) operator. The techniques discussed in the study [27] were used to calculate the occupancy rates within the bus. In this study, by analyzing the boarding information recorded with the use of smart cards, it is possible to calculate the bus density for each line between the bus stops.

In this study, a line accessibility degree is created according to the predicted bus occupancy rates, and this approach is considered in the most appropriate route planning. A new modification of the Dijkstra algorithm, which considers the fuzzy neighborhood degree between stops and the line fuzzy accessibility degrees, in addition to the penalty values for extra walking and transfer lines, is proposed. An approach based on walking distance is used to calculate the degree of fuzzy neighborhood between stops, and the density inside the bus for calculating the fuzzy accessibility of the line. Data mining methodology based on smart card data is suggested to calculate the fuzzy degree based on intensity inside the bus.

In the rest of the paper, preliminary information about topological spaces used for modeling public transport networks is given in the second section. Next in the section, information on the modified Dijkstra algorithms using classical Dijkstra and penalty values is reminded. In Section 3, the concepts of fuzzy possible connection, fuzzy accessibility of a route are given and then the new proposed FuzzyDijkstra algorithm with fuzzy penalty is proposed. Steps of the algorithm are explained by a numerical example. In the next Section 4, meta information of Izmir city PTN data is given and the calculation results are discussed. The last section concludes the results obtained in the paper and gives hints about future studies.

2. Preliminaries

2.1. Topologies used in modeling of public transport networks

In public transport network studies, different topological spaces are used to model the relations of stops and lines with each other. In general, L, P, C and B spaces and their extensions L', P', C' and B' are used in the literature [4]. In a simple graphical representation of a public transport map, a stop is represented as a node, while stops one after another through which the line passes are joined by the edges. Each line can be given as an ordered list of the stops it passes through. In this case, the lines can normally be specified as two separate lines, one outgoing and one return direction. We can simply give such a situation as the L-space. In L-space, each stop is represented by a node, and the stops where a particular line passes are joined by successive edges. Neighbors of a particular node in L-space represent all the stations accessible to passengers at this stop (Fig. 1a). By expanding the L-space concept, either the stops can be added multiple edges between nodes depending on the number of lines between them, or edges can be added with different weights corresponding to different situations. Such a representation is called L'-space (Fig. 1b). Another notation that is particularly useful for modeling connectivity is called P-space. In this representation, if at least one common line is served between the stops, these stops are directly connected via the edge (Fig. 1c). The concept of P-space can be extended to include multiple or weighted links. Such a representation is called the P'-space (Fig. 1d).

A different model useful in analyzing collaborative networks is a two-part graph representation. In this notation we call B-space, both lines and stops are represented by nodes. Each line node is connected to all the stop nodes it passes through. There is no direct connection between nodes of the same type (Fig. 1e). Obviously, while in B-space, a given line node has all stops of its neighbors, the neighbors of a stop are all lines passing through it. Note that in the B-space representation, the order of the stops on which the line passes are unknown. The single-mode projection of the binary graph representation of the B-space into the set of stop nodes turns into a P-space, and in the case of multiple connections, it becomes a P'-space. The single-mode projection of the B-space onto the set of line nodes turns into a C-space. In C-space representation, all nodes represent lines, and neighbors of any line node are lines with which any common station shares (Fig. 1f). If more than one connection is handled, this representation is called the C'-space model (Fig. 1g).

2.2. Dijkstra algorithm

The Dijkstra algorithm, proposed by Edsger W. Dijkstra, is an algorithm used to find the shortest path between nodes in a graph that can represent road networks [43]. The algorithm is available in many variants. Dijkstra's original algorithm found the shortest path between two given nodes, but a more common variant is the variant that finds the shortest paths from a single source node to

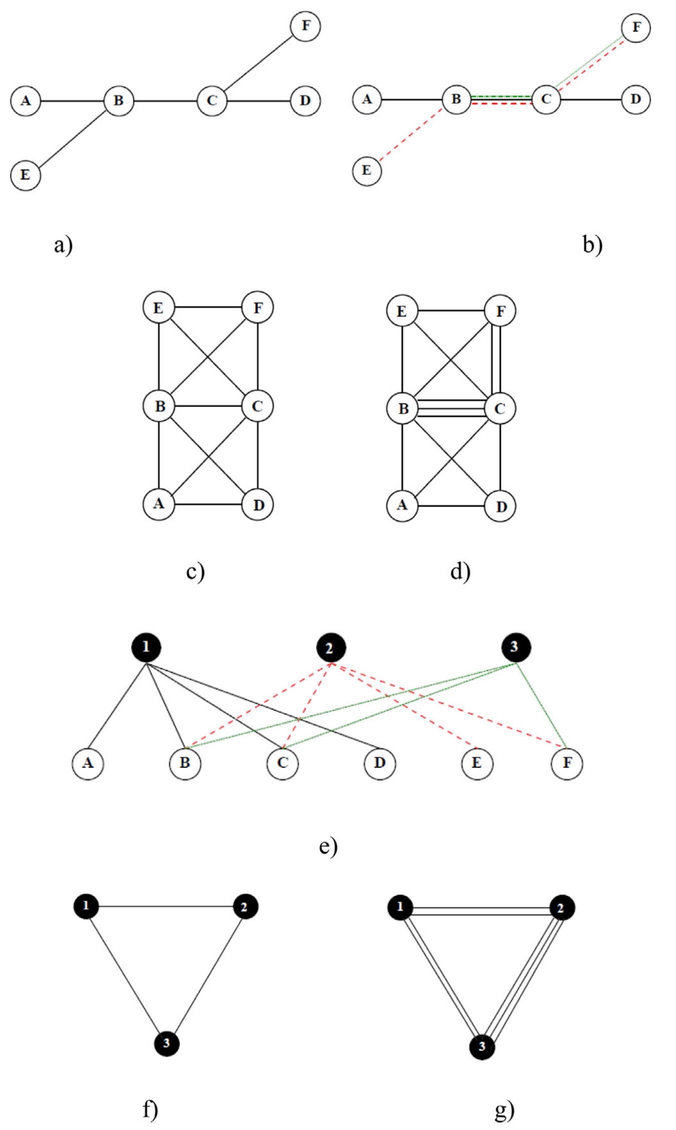


Fig. 1. Topologies used for modeling public transport networks: (a) L-space, (b) L'-space, (c) P-space, (d) P'-space, (e) B-space, (f) C-space, (g) C'-space.

all other nodes. The Dijkstra's algorithm uses the P-space model for public transport networks.

Let us mark the set of stops as $i = 1, 2, \ldots, n$. Let us specify the length of the line, i.e., the count of stops passed through, between the pair of stops (i, j) with the $W : n \times n$ matrix. If there is no direct line between certain (i, j) pair of stops, it will take $w_{i,j} = INFINITY$:

$$W = \begin{bmatrix} 0 & w_{1,2} & w_{1,3} & \ldots & w_{1,n} \\ w_{2,1} & 0 & w_{2,3} & \ldots & w_{2,n} \\ & \ldots & \ldots & \ldots & \\ w_{n,1} & w_{n,2} & \ldots & w_{n,n-1} & 0 \end{bmatrix} \quad (1)$$

In the classical Dijkstra's algorithm, the cost of this inter-stops path for each (i, j) pair of stops is calculated only through the $W : n \times n$ weight matrix. Let us mark a path from a specific source stop s_O to the destination stop s_D with π_{s_O,s_D}. Let the lines used by this route be $l_i, i = 1, \ldots, m$. In this case, the total length of the road from the source stop to the destination stop can be

```
function Dijkstra(Graph, source):
  create vertex set Q
  for each vertex v in Graph:
    dist[v] ← INFINITY
    prev[v] ← UNDEFINED
    add v to Q
  dist[source] ← 0
  while Q is not empty:
    u ← vertex in Q with min dist[u]
    remove u from Q
    for each neighbor v of u:
      alt ← dist[u] + length(u, v)
      if alt < dist[v]:
        dist[v] ← alt
        prev[v] ← u
  return dist[], prev[]
```

Fig. 2. The pseudocode of the classical Dijkstra's algorithm.

```
S ← empty sequence
u ← target
while u is defined:
  insert u at the beginning of S
  u ← prev[u]
output S
```

Fig. 3. Constructing a sequence of stops on the optimal path.

calculated as follows:

$$w\left(\pi_{S_O,S_D}\right) = \sum_{i=1}^{m} w(l_i) \tag{2}$$

Here, $w(l_i)$ is the length of the segment of the path crossed by the line l_i. This length can be considered as the distance calculated using the W matrix. Generally, path length is used as the most important criterion in optimal route suggestion models. Classical Dijkstra algorithm, uses the following criterion to find the optimal route:

$$w\left(\pi_{S_O,S_D}\right) \longrightarrow min \tag{3}$$

The pseudocode of the Dijkstra algorithm is given in Fig. 2. As the output of the pseudocode designed as a function, the shortest paths from the source node to all other nodes and the lengths of these paths return as appropriate arrays.

We can read the stops in the *prev[]* array in reverse iteration to construct the shortest path from the source to a specific destination. The S sequence given to the output will contain the list of the stops on the shortest paths from the source to the destination, or the empty string if there is no path (Fig. 3).

As we mentioned above, the Dijkstra algorithm is one of the most used algorithms to find the shortest path between the nodes on the graph. However, the main shortcoming of Dijkstra-based algorithms working on the P-space model is the difficulties experienced in modeling the number of transfers between lines. For example, in the case indicated in Fig. 4, the shortest path from the stop v_1 to the stop v_5 can be realized through lines 2, 3 and 1 with 6 distance units. However, in this case, it will be necessary to change the line two times. Another route to the target stop v_5, is a route that can be reached with 7 distance units through lines 4 and 3 by changing only one line.

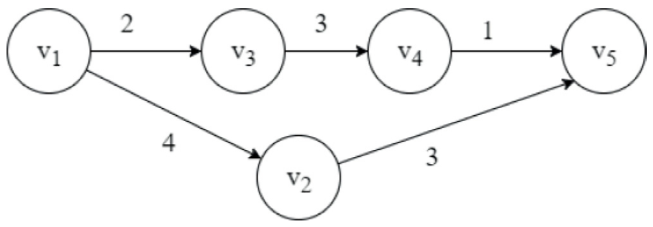


Fig. 4. Example of a solution with less line transfer [19].

In the classical Dijkstra's algorithm normally the number of transfers is not taken into account, so it will suggest the shortest route, $v_1 \longrightarrow v_3 \longrightarrow v_4 \longrightarrow v_5$. In the real world, many passengers prefer the second route $v_1 \longrightarrow v_2 \longrightarrow v_5$. To model this situation, the transfer penalty function was used in the study [19]. In the study, the problem of finding the most suitable route by considering not only the distances between the stops but also the transfers between the lines in the public transport graph was discussed and a Modified Dijkstra algorithm was proposed for the solution of this problem. In this algorithm, it is also taken into account that the transfer between the stops can be provided by different lines. Therefore, this algorithm is also based on the P'-space representation.

2.3. The modified Dijkstra algorithm using penalty

The penalty approach to the Dijkstra's algorithm, without a fuzzy model, is handled in the study [19]. This PTN model is a P'-space approach mentioned in Section 2.1. In the Modified Dijkstra algorithm, for each pair of stops (i, j), the weighting matrix $W : n \times n$, which holds the edge length between stops, as well as the $L : n \times n$ matrix consisting of lines that provide this path is handled as an input. The optimization model on which the Modified Dijkstra algorithm is based can be written as follows:

$$cost(\pi_{S_O,S_D}) = w\left(\pi_{S_O,S_D}\right) + walkPenalties + transPenalties \longrightarrow min \tag{4}$$

Here, *walkPenalties* are the total penalty values used to minimize uninterrupted repetition walkings between stops and *transPenalty* to minimize the number of transfers between lines. In the Modified Dijkstra algorithm, an $L : n \times n$ incidence matrix is also used, which contains lists of all lines serving between stops:

$$L = \begin{bmatrix} \varnothing & l_{1,2} & l_{1,3} & \dots & l_{1,n} \\ l_{2,1} & \varnothing & l_{2,3} & \dots & l_{2,n} \\ & \dots & \dots & \dots & \\ l_{n,1} & l_{n,2} & \dots & l_{n,n-1} & \varnothing \end{bmatrix} \tag{5}$$

Here $l_{i,j}$ is the list of all lines that pass from the stop i to the stop j. The pseudocode of the Modified Dijkstra algorithm can be written as in Fig. 5:

Here, in order to reduce transfer count and repeating walking situations, the *Penalty()* function is used to calculate penalty values added to the goal function as in Fig. 6.

3. Fuzzy optimal route recommendation model and a solution algorithm

3.1. Fuzzy optimal route recommendation model

Generally, accessibility from one stop to another can be achieved by walking or by using a bus line. Let $s_1, s_2 \in S$ be any two stops. In the study Nasibov et al. [26], the fuzzy neighborhood

```
function Dijkstra(W,L,s):
  create list unvisited
  create arrays lines[],cost[],path[]
  for each vertex v in Graph:
    cost[v] ← INFINITY
    path[v] ← EMPTY
    add v to unvisited
  cost[s] ← 0
  while unvisited is not empty:
    u ← vertex in unvisited with min cost
    remove u from unvisited
    for each neighbor v of u:
          create line list commonLines
          penalty,commonLines ← Penalty(lines[u],L[u,v])
      alt ← cost[u] + W[u,v] + penalty
      if alt < cost[v]:
        cost[v] ← alt
        path[v] ← path[u] + commonLines
  return cost[], path[]
```

Fig. 5. Pseudocode of the Modified Dijkstra algorithm.

```
function Penalty(curLines,adjLines):
  walkPenalty ← CONST1
  transPenalty ← CONST2
  create empty list commonLines
  penalty ← 0
  if curLines is empty:
    if adjLines is empty:
      penalty ← walkPenalty
    else:
      penalty ← transPenalty
      commonLines ← adjLines
  else:
    if adjLines is empty:
      penalty ← walkPenalty
    else:
      for each line in curLines:
        if adjLines contains line:
          add line to commonLines
      if commonLines is empty:
        penalty ← transPenalty
        commonLines ← adjLines
  return penalty, commonLines
```

Fig. 6. Calculation of penalty values in Modified Dijkstra algorithm.

relationship of the stop s_2 with respect to the stop s_1 is defined through the membership function $\mu_{s_1}(s_2) \in [0, 1]$. Therefore, in the case of walking from a certain stop s_1 to another stop s_2, the degree of fuzzy accessibility can be defined as follows:

$$\mu_{s_1}(s_2) = \max\left(0, 1 - \frac{d(s_1, s_2)}{d_{max}}\right) \tag{6}$$

Here, $d(s_1, s_2)$ is the walking distance between the stops s_1 and s_2, while d_{max} is the maximum acceptable walking distance.

In the case of reaching the stop s_2 from the stop s_1 by using the bus line, the fuzzy accessibility degree can be calculated on the basis of various indicators such as the length of the line, the comfort of the bus, and the density inside the bus. Nasiboglu in [28] proposed a fuzzy line preference degree based on the intensity rate inside the bus. The intensity rate, or occupancy rate in the bus, hence the fuzzy preference degree of the line and bus can be calculated from smart card data. Smart card data is a very important data source for analyzing the urban public transport network and for optimal planning. There are many studies based on the analysis of these data [21–25,29,30]. Usually, it is not known where passengers alight, as smart cards are only used on boarding. There are several studies in the literature to predict alighting stops using smart card data. There are some studies also on the estimation of alighting stops and the creation of origin–destination (OD) matrix for Izmir city [25,27].

Let us denote the occupancy rate of a bus just departing the stop s on the line l, in other words, the fuzzy density inside the bus as $\mu(l; s)$. This fuzzy degree can be calculated as the ratio of the number of passengers in the bus to the maximum number of passengers traveled for this type of bus, namely:

$$\mu(l; s) = \begin{cases} \dfrac{p_s}{p_{max}}, & if\ p_s \le p_{max}, \\ 1, & otherwise. \end{cases} \tag{7}$$

Here, p_s is the number of passengers in the bus just departing the stop s, and p_{max} is the maximum number of passengers for this bus type.

During the trip, a passenger is exposed to different levels of intensity inside the bus between the boarding the bus and the alighting stops. The average, the maximum, the minimum etc. aggregated values of these different intensity levels can be used in evaluations [28]. For example, the fuzzy density degree inside the bus collected along the line l between the boarding, s_b, and alighting, s_a, stops can be calculated as follows:

$$\mu^{dens}(s_b, l, s_a) = \max_{s \in S(s_b, l, s_a)} \mu(l; s) \tag{8}$$

Here $S(s_b, l, s_a)$ is the list of stops between the stops s_b and s_a on the line l.

If the intensity inside the bus is taken into account in a certain possible connection, there may be a fuzzy preference degree in terms of passenger preference. Obviously, the degree of passenger preference is inversely proportional to the intensity of the bus. For example, the fuzzy preference degree of a certain possible connection (s_b, l, s_a) can be determined in the simplest way as follows:

$$\mu^{line}(s_b, l, s_a) = 1 - \mu^{dens}(s_b, l, s_a) \tag{9}$$

Here, the value of $\mu^{dens}(s_b, l, s_a)$ is the degree of intensity inside the bus calculated according to the formula (8). As it can be seen from the formula (9), the fuzzy preference degree of the possible connection (s_b, l, s_a) in the case where the bus is fully loaded will be 0, so this situation will definitely not take place in the solution space. However, in real life, some passengers may prefer this situation as well. In this respect, it may be more reasonable to use a parametric formula as follows instead of formula (9) in order to ensure that the degree of preference in this situation is a certain value greater than 0:

$$\mu^{line}(s_b, l, s_a) = \frac{1}{(1 + \mu^{dens}(s_b, l, s_a))^n} \tag{10}$$

By giving various values to the parameter $n > 0$ in the formula (10), it can be guaranteed that the preference degree is greater than 0 for the fully loaded bus (Fig. 7).

Let us consider a possible route π_{S_O,S_D} from a given s_O source stop to the s_D destination stop with m possible connections (Fig. 8):

$$\pi_{S_O,S_D} = \langle (s_{b_1}, l_1, s_{a_1}), (s_{b_2}, l_2, s_{a_2}), \ldots, (s_{b_m}, l_m, s_{a_m}) \rangle \tag{11}$$

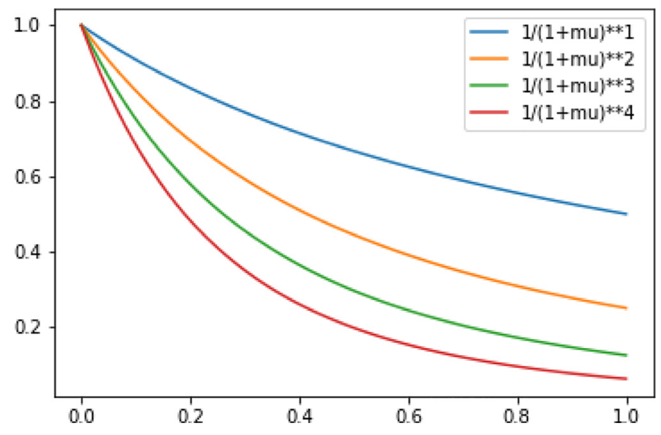


Fig. 7. Determining the line preference degree parametrically depending on the density ratio inside the bus.

Obviously, for this route to be preferable, the following conditions will need to be met:

$$\mu_{s_{a_{t-1}}}\left(s_{b_t}\right) > 0, t = 2, \ldots, m. \tag{12}$$

$$\mu^{line}\left(s_{b_t}, l_t, s_{a_t}\right) > 0, t = 1, \ldots, m, \tag{13}$$

$$s_{b_1} = s_O, \tag{14}$$

$$s_{a_m} = s_D, \tag{15}$$

The notation $\mu_{s_{a_{t-1}}}\left(s_{b_t}\right) > 0,\ t = 2, \ldots, m$, in the above Eq. (12) indicates that there is a walking distance between the last stop $s_{a_{t-1}}$ of the possible connection number $(t-1)$ on the route and the first stop s_{b_t} of the next possible connection. Eq. (13) indicates that each connection that the route includes, must be a possible connection. Eq. (14) shows that the starting stop s_O of the route is equal to the source stop, and similarly, (15) equals the last stop of the route to the stop s_D. Obviously, the number of transfers on the route π_{s_O,s_D} consisting of m possible connections will be equal to $m-1$. Thus, the degree of fuzzy accessibility from the origin stop s_O to the destination stop s_D using the possible route can be calculated as the minimum of the connection degrees of the possible connections that create it. So, the accessibility degree of the route can be defined as follows:

$$\mu(\pi_{s_O,s_D}) = \left[\bigwedge_{t=1}^{m} \mu^{line}\left(s_{b_t}, l_t, s_{a_t}\right)\right] \bigwedge \left[\bigwedge_{t=2}^{m} \mu_{s_{a_{t-1}}}\left(s_{b_t}\right)\right] \tag{16}$$

Here and after, the symbol "$\bigwedge$" means the minimum. As a result, the optimal route recommendation model from a certain origin stop s_O to the destination stop s_D that we have discussed in this study can be written as the following multi-criteria model:

$$w(\pi_{s_O,s_D}) \rightarrow min \tag{17}$$

$$\mu(\pi_{s_O,s_D}) \rightarrow max \tag{18}$$

$$m \rightarrow min \tag{19}$$

s.t.:

$$\mu(\pi_{s_O,s_D}) = \left[\bigwedge_{t=1}^{m} \mu^{line}\left(s_{b_t}, l_t, s_{a_t}\right)\right] \bigwedge \left[\bigwedge_{t=2}^{m} \mu_{s_{a_{t-1}}}\left(s_{b_t}\right)\right] \tag{20}$$

$$s_{b_1} = s_O \tag{21}$$

$$s_{a_m} = s_D \tag{22}$$

$$s_{b_t}, s_{a_t} \in S, t = 1, \ldots, m, \tag{23}$$

$$l_t \in L, t = 1, \ldots, m. \tag{24}$$

Eq. (17) above, ensures that the recommended route between the origin and destination stops is the shortest. Here the value $w\left(\pi_{s_O,s_D}\right)$ is given as in Eq. (2). Eq. (18) ensures that the proposed route is preferable as possible as highest degree. Eq. (19), on the other hand, ensures that the number of line transfers required on the proposed route is the least. Eqs. (20)–(22) are the conditions that require the route to be possible. Eqs. (23) and (24) show that the solution to the problem is sought on the set of all stops S, and on the set of all lines L. To solve the problem (17)–(24), a new FuzzyDijkstra algorithm is proposed by adding fuzzy accessibility penalties to the modified Dijkstra algorithm in the next section.

3.2. FuzzyDijkstra algorithm using fuzzy penalty

There are not many fuzzy approaches to the Dijkstra algorithm in the literature. One of them is the algorithm proposed in the study Kang [16], where the lengths of the edges in the graph are considered as fuzzy intervals and a solution algorithm for this problem is given. However, in our study, the concept of fuzzifying is not addressed as the length of the distance between stops, but as the fuzzy accessibility of the route from the source stop to the destination stop. For this purpose, in addition to the route length and number of transfers used in the modified Dijkstra algorithm, the model created by considering the stops' fuzzy preference degrees and line preference degrees is taken as basis. In the developed algorithm, which we call FuzzyDijkstra algorithm, a penalty calculating via fuzzy walking and fuzzy line accessibility degrees are used. The basic differences between the FuzzyDijkstra approach proposed in this study and other studies in the literature is given in Table 1.

In the FuzzyDijkstra algorithm, in addition to the matrixes W and L, the following $M: n \times n$ matrix, which reflects the preference degrees of the lines between the stops, is also used:

$$M = \begin{bmatrix} 1 & \mu_{1,2} & \mu_{1,3} & \ldots & \mu_{1,n} \\ \mu_{2,1} & 1 & \mu_{2,3} & \ldots & \mu_{2,n} \\ & \ldots & \ldots & \ldots & \\ \mu_{n,1} & \mu_{n,2} & \ldots & \mu_{n,n-1} & 1 \end{bmatrix} \tag{25}$$

In order to calculate the values of $\mu_{i,j}$, the ways of computing fuzzy degrees of preference, which we have outlined in the previous section, can be used. In addition, the distances between the stops' matrix, $D: n \times n$, is also used to calculate the fuzzy neighborhood degrees depending on the walking distance between the

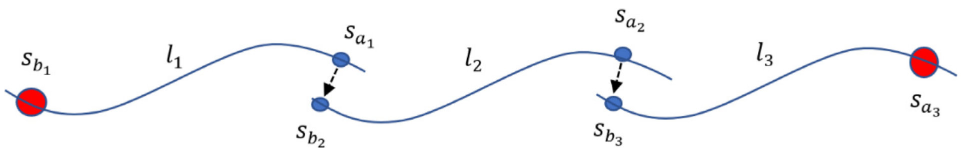


Fig. 8. Example of a possible route involving two transfers with three possible connections.

```
function FuzzyDijkstra(s,t,L,W,M,D):
  create list unvisited
  create arrays lines[],cost[],path[], fuzzyDegree[]
  for each vertex v in Graph:
    cost[v] ← INFINITY
    path[v] ← UNDEFINED
    fuzzyDegree[v] ← MaxInt
    add v to unvisited
    cost[s] ← 0
  while unvisited is not empty:
    u ← vertex in unvisited with min cost
    remove u from unvisited
    for each neighbor v of u:
        conDegree,fuzzyPenalty,commonLines ← FuzzyPenalty(cur,adj,lines[u],L,W,M,D)
      alt ← cost[u] + W[u,v] + fuzzyPenalty
      if alt < cost[v]:
        cost[v] ← alt
        fuzzyDegree[adj] ← min(fuzzyDegree[cur], comDegree)
        path[v] ← path[u] + commonLines
  return fuzzyDegree[],cost[],path[]
```

Fig. 9. Pseudocode of the FuzzyDijkstra algorithm.

stops:

$$D = \begin{bmatrix} 0 & d_{1,2} & d_{1,3} & \dots & d_{1,n} \\ d_{2,1} & 0 & d_{2,3} & \dots & d_{2,n} \\ & \dots & \dots & \dots & \\ d_{n,1} & d_{n,2} & \dots & d_{n,n-1} & 0 \end{bmatrix} \tag{26}$$

While the geographical latitude and longitude information is known for the stops i and j, $d_{i,j}$ distance between them can be calculated using the haversine formula [44].

Let us mark a path from the origin s_O stop to the destination s_D stop with π_{s_O,s_D}. In the FuzzyDijkstra algorithm, a *fuzzyPenalty* value consisting of the total fuzzy walking and fuzzy accessibility degrees is used:

$$fuzzyPenalty = 1 - \mu(\pi_{s_O,s_D}), \tag{27}$$

where $\mu(\pi_{s_O,s_D})$ is calculated as in Eq. (16). So, the total cost of the path from the origin stop to the destination stop is calculated as follows:

$$cost\left(\pi_{s_O,s_D}\right) = w\left(\pi_{s_O,s_D}\right) + c_1 * m + c_2 * \left(1 - \mu\left(\pi_{s_O,s_D}\right)\right), \tag{28}$$

where c_1 and c_2 are the fixed penalty coefficients of line transfers and connections, respectively.

The pseudocode of the *FuzzyDijkstra* algorithm is given in Fig. 9. In this algorithm, *fuzzyPenalty* value is added to reduce transfer count and repetitive walking situations and to highlight the roads with higher accessibility. The *fuzzyPenalty* value includes the fuzzy neighborhood values of the stops and the fuzzy accessibility values of the lines, as well as the repetitive walking and transfer penalties. For this purpose, a *FuzzyPenalty()* function whose pseudocode is given in Fig. 10 is used. The time complexity of the *FuzzyDijkstra* algorithm is $O(|V|^2)$, similar to the time complexity of the simple Dijkstra algorithm. Here $|V|$ is the count of nodes of the graph, i.e. the total count of the bus stops in our case. But the *FuzzyPenalty* function is called when each neighbor of the current node is visited. The *FuzzyPenalty* function, on the other hand, has no loops, so its time complexity is a constant value. As a result, the time complexity of the *FuzzyDijkstra* algorithm will also be $O(|V|^2)$, since a fixed multiplier does not affect the time complexity.

3.3. Numerical example

An example network is given in Fig. 11 which consists of the lines as follows:

Black line: $l_1 = (1) \rightarrow (4) \rightarrow (7) \rightarrow (15) \rightarrow (16) \rightarrow (14) \rightarrow (13)$

Blue line: $l_2 = (1) \rightarrow (3) \rightarrow (5) \rightarrow (9) \rightarrow (11) \rightarrow (15) \rightarrow (18)$

Orange line: $l_3 = (2) \rightarrow (4) \rightarrow (7) \rightarrow (11) \rightarrow (17) \rightarrow (18)$

Green line: $l_4 = (2) \rightarrow (4) \rightarrow (6) \rightarrow (8) \rightarrow (12) \rightarrow (15) \rightarrow (18)$

Red line: $l_5 = (10) \rightarrow (9) \rightarrow (11) \rightarrow (15) \rightarrow (16) \rightarrow (14) \rightarrow (13)$

The arc lengths between any nodes j and k, and the fuzzy connectivity degrees of the arcs are given in Table 2.

Let us calculate the optimal path from the origin node (1) to the target node (18). Computational results with some of the first and the last steps of FuzzyDijkstra algorithm are given in Figs. 12 and 13, respectively. In the figures, the number of transfers (m) and the connectivity degree of the path (μ) are given in parentheses next to the cost values. To avoid confusion, each link has a preference degree of 1.0, except for $\mu((7) \rightarrow (15)) = 0.6$, $\mu((11) \rightarrow (15)) = 0.6$, and $\mu((15) \rightarrow (18)) = 0.6$. The cost values are calculated according to the Eq. (28) with $c_1 = 10$ and $c_2 = 20$ coefficients.

As can be seen from Fig. 13, the route (1) →(4) →(7) →(11) →(17) →(18) with total cost value of 72, consisting of 62 path length, 1.0 fuzzy connectivity degree and 1 line transfer, was found to be the optimal route. Note that, the path (1) →(4) →(7) →(15) →(18) with 56 length, 0.6 fuzzy connectivity degree and 1 line transfer is another pareto optimal path (cost value is 74).

4. Experimental results and discussion

Experimental trails were made using the data of Izmir urban public transport network. The public transportation management authority (ESHOT) of Izmir provides services to Izmir city center and environmental life units within the area of Izmir Metropolitan Municipality. Izmir city is the 3th biggest metropolitan of

Table 1
Taxonomy of the referenced studies on path optimization and the public transport network analysis.

Study	Used transport network	Used fuzzy approach	Used optimization approach	Used method detailed
Ceder [1]	General theory and methods with numerical examples	No fuzzy approach is used.	Different PTN optimization models	Optimization models and methods are analyzed.
Nasiboglu et al. [2]	Existing web applications on PTN are evaluated in terms of running speed, ease of use, effectiveness etc.	No fuzzy approach is used.	Multicriteria approach using properties of web applications	Analytic Hierarchy Process (AHP) and Technique for Order Preference by Similarity to Ideal Solution (TOPSIS) are used to comparisons.
Costa et al. [3]	Theoretical models with simulated examples.	No fuzzy approach is used.	No optimization problem.	Network analysis techniques and methods are used.
Ferber et al. [4,5]	The public transport networks of 14 major cities of the world is analyzed.	No fuzzy approach is used.	No optimization problem.	Different network topologies are handled and analyzed.
Bona et al. [6]	The PTN system of Curitiba, Brazil is compared to three cities from China (Shanghai, Beijing, and Guangzhou), and three cities from Poland (GOP, Warszawa, and Łod'z).	No fuzzy approach is used.	No optimization problem.	The theory of complex networks in a static approach of network topology is used.
Sienkiewicz and Holyst [7]	Public transport systems in 22 Polish cities have been analyzed	No fuzzy approach is used.	No optimization problem.	The L-space and P-space network topologies with statistical analysis are used.
Xu et al. [8]	The statistical properties of three bus-transport networks (BTN) in three different cities of China is used.	No fuzzy approach is used.	No optimization problem.	The L-space and P-space network topologies with statistical analysis are used.
Xu et al. [9]	The public transport networks of 330 cities in China is used.	No fuzzy approach is used.	No optimization problem.	The P-space network topology with statistical analysis is used.
Zhang et al. [10]	The bus network of Beijing, China is used.	No fuzzy approach is used.	No optimization problem.	The L-space and P-space network analysis is used.
Zhang et al. [11]	The Tai'an city bus network is used.	No fuzzy approach is used.	No optimization problem.	The L-space and P-space network analysis is used.
Zhang et al. [12]	The Taian city, Shandong province, was selected as the experiment data	No fuzzy approach is used.	No optimization problem.	The complex network analysis and graph theory is used.
Chen et al. [13,14]	The urban bus transport networks of four major cities in China are used.	No fuzzy approach is used.	No optimization problem.	The graph topology with statistical analysis is used.
Zhu et al. [15]	BTNs for Hangzhou and Nanjing, China are simulated.	No fuzzy approach is used.	No optimization problem.	The B-space topology with statistical analysis is used
Kang [16]	Simulated data are used.	Lengths of arcs are given in the form of intervals.	One criterial optimization.	The shortest path algorithm is proposed.
Lossow [17]	A numerical example is used.	No fuzzy approach is used.	One criterial optimization.	A min–max version of Dijkstra's algorithm for solving the shortest path problem in a directed weighted hypergraph is proposed.
Bozyigit et al. [18]	The public transport network of Izmir, Türkiye is used.	No fuzzy approach is used.	Multicriteria problem is used according to the length of the path and the number of line transfers.	The P-space topology is handled. The Breadth First Search is proposed.
Bozyigit et al. [19]	The public transport network of Izmir, Türkiye is used.	No fuzzy approach is used.	Multicriteria problem is used according to the length of the path and the number of line transfers.	Dijkstra's algorithm using penalty is proposed.
Zhao et al. [20]	Numerical example is used.	Stochastic uncertainty of travel time caused by traffic congestion condition is used.	Shortest paths optimization.	The least transfer times algorithm and the K shortest transit paths search algorithm in the stochastic transit network is proposed.
, Nasibov et al. [26]	Numerical example is used.	Fuzzy route preference degree based on fuzzy line and stop neighborhood concepts is used.	A multicriteria problem according to the length and the preference degree of the path is used.	A brute force search approach is used.
Diker [27]	The public transport network of Izmir, Türkiye is used.	Fuzzy route preference degree is used.	A multicriteria problem according to the length and the preference degree of the path is used.	A brute force search approach is used.

(continued on next page)

Table 1 (*continued*).

Study	Used transport network	Used fuzzy approach	Used optimization approach	Used method detailed
Nasiboglu [28]	Numerical example is used.	Fuzzy route preference degree based on bus occupancy density is used.	One criteria optimization model is used	A brute force search approach is used.
Seaton and Hackett [32]	The bus networks of Boston and Vienna are used.	No fuzzy logic used	No optimization problem.	The bipartite graphs are used. The small-world properties of the network are analyzed.
Sen et al. [33]	Indian railway network is used.	No fuzzy logic used	No optimization problem.	The small-world properties of the network are analyzed.
Derrible and Kennedy [34]	33 metro systems worldwide are used.	No fuzzy logic used	No optimization problem.	By using graph theory concepts, a method to collect different topological properties of metro networks are proposed.
Zhang et al. [35]	Shanghai subway network of China is investigated.	No fuzzy logic used	No optimization problem.	Graph theory and complex network theory are adopted to investigate the connectivity, robustness and reliability of the subway network
Mahdavi et al. [36]	Numerical example is used.	Edge lengths are fuzzy numbers.	One criteria shortest path problem is used.	A dynamic programming approach to solve the fuzzy shortest chain problem using a suitable ranking method is used.
Deng et al. [37]	Numerical example is used.	Edge lengths are fuzzy numbers.	One criteria shortest path problem is used.	The graded mean integration representation of fuzzy numbers is adopted to improve the classical Dijkstra algorithm.
Ji et al. [38]	Numerical example is used.	Edge lengths are fuzzy numbers.	One criteria shortest path problem is used.	The concepts of expected shortest path, a-shortest path and the most shortest path concepts are proposed, and three types of models are formulated. A hybrid intelligent algorithm integrating simulation and genetic algorithm is provided.
Erbao and Mingyong [39]	Numerical example is used.	A triangular fuzzy number representing demand of a customer is used.	Multicriteria model minimizing total planned travel distance and minimizing total additional travel distance due to route failures is used.	A hybrid intelligent algorithm integrating stochastic simulation and a differential evolution algorithm to solve the fuzzy chance-constrained problem is proposed.
Jovanovic et al. [40], Cirovic et al. [41]	Simulating conditions of Belgrade, Serbia is used	A fuzzy inference system is used to calculate the performance value of a branch of the network.	A one criteria optimization approach is used.	An adaptive neural network and maximum spanning tree techniques are used.
The proposed approach	A numerical example and the public transport network of Izmir, Türkiye is used.	The fuzzy connectivity degree concept of the path is used.	Multicriteria problem is used according to the length of the path, the number of line transfers and the connectivity degree of the path.	A Fuzzy Dijkstra algorithm using fuzzy penalty is proposed.

Table 2
Arc lengths between, and the fuzzy connectivity degrees of the arcs.

Arc	Length	Fuzzy connectivity	Arc	Length	Fuzzy connectivity	Arc	Length	Fuzzy connectivity
(1,3)	10	1 (with l_2)	(7,11)	10	1 (with l_3)	(13,14)	10	1 (with l_1)
(1,4)	14	1 (with l_1)	(7,15)	14	0.6 (with l_1)	(13,14)	10	1 (with l_5)
(2,4)	10	1 (with l_3)	(8,12)	10	1 (with l_4)	(14,16)	10	1 (with l_1)
(2,4)	10	1 (with l_4)	(9,10)	14	1 (with l_5)	(14,16)	10	1 (with l_5)
(3,5)	10	1 (with l_2)	(9,11)	11	1 (with l_2)	(15,16)	10	1 (with l_1)
(4,6)	10	1 (with l_4)	(9,11)	11	1 (with l_5)	(15,16)	14	1 (with l_5)
(4,7)	14	1 (with l_1)	(11,15)	10	0.6 (with l_2)	(15,18)	14	0.6 (with l_2)
(4,7)	14	1 (with l_3)	(11,15)	10	0.6 (with l_5)	(15,18)	14	0.6 (with l_4)
(5,9)	14	1 (with l_2)	(11,17)	14	1 (with l_3)	(17,18)	10	1 (with l_3)
(6,8)	10	1 (with l_4)	(12,15)	10	1 (with l_4)			

```
function FuzzyPenalty(cur, adj, curLines,L,W,M,D):
  walkPenalty ← CONST1
  transPenalty ← CONST2
  adjLines ← L[cur,adj]
  adjDegree ← max{M[cur,adj][l]: l in adjLines}
  walkDegree ← max{0, 1-D[cur,adj]/maxWalk}
  create empty list commonLines
  penalty ← 0
  conDegree ← 1
  if curLines is empty:
    if adjLines is empty:
      penalty ← walkPenalty
      conDegree ← walkDegree
    else:
      commonLines ← adjLines
      penalty ← transPenalty
      conDegree ← adjDegree
  else:
    if adjLines is empty:
      penalty ← walkPenalty
      conDegree ← walkDegree
    else:
      for each line in curLines:
        if adjLines contains line:
           add line to commonLines
      if commonLines is empty:
        commonLines ← adjLines
        penalty ← transPenalty
        conDegree ← max{M[cur,adj][l]: l in commonLines}

    if commonLines is empty:
        fuzzyPenalty = penalty * (1 – walkDegree)
    else:
        fuzzyPenalty = penalty * (1 – conDegree)
  return conDegree, fuzzyPenalty, commonLines
```

Fig. 10. The pseudocode of the *FuzzyPenalty()* function.

Turkey, and it is among the cities with a developed public transport network in the world with a total area of 11,900 km^2, a population of 4.4 million, 6475 stops and 574 round-trip lines. Izmir bus fleet provides 13,884 daily services with 1396 buses [45]. Various public transport networks are used in Izmir, including buses, subways, trains, trams, and ferries.

Since 1999, smart cards have been used for boarding all public transportation vehicles in Izmir. Approximately 1.5 million transactions are processed daily with smart cards. PTN data used in this study can be accessed via the github website [46]. The travel chain method was used to estimate the alighting stops and to create the OD matrix. After estimating the alighting stop information on lines, the inside bus density also is estimated based on lines. This information also is used to calculate the line-based accessibility degrees.

To keep the calculations simple, only simulation values for the matrix M were used in experiments. The $m_{i,j}$ values, which are the degrees of line accessibility between stops, are generated as uniformly produced random real numbers between [0.5, 1.0]. The degrees of accessibility by walking between the stops were calculated according to the haversine formula using the actual coordinates of the stops. Walking distances between stops is limited to 300 m. The list of the actual lines between the stops and the number of the stops passed between the origin and destination stops are loaded from a statically recorded data file. The calculations were performed with the application written in Python 3 language, on the i7, 2.5 GHz, 8 GB RAM, 256 GB SSD featured laptop. The outputs of the optimal route recommended by the program between sample two stops are given in Fig. 14 and its reflection on the map is shown in Fig. 15. The test results of the optimal routes produced with different penalty values among 100 random stop pairs based on Izmir city PTN data are given in Table 3.

The *wPenalty* and *trPenalty* values reflected in Table 3 are the penalty values used in walking and changing the transit line accordingly. The case where $wPenalty = 1$ and $trPenalty = 0$ corresponding to the first row in Table 3 corresponds to the classical Dijkstra algorithm. It is seen that when fuzzy penalty is used in the classical Dijkstra algorithm, an increase of approximately 16.9% is achieved in the average accessibility degrees of the produced paths. Other cases where $wPenalty > 0$ and $trPenalty > 0$ are the cases corresponding to the Modified Dijkstra and the FuzzyDijkstra algorithms with penalties. The average accessibility degrees and average walking distances of the paths produced for different values of *wPenalty* and *trPenalty* parameters for each algorithm are also given in Fig. 16a, b. As can be seen from Table 3

Table 3
Average indicators of optimal routes generated for 100 randomly generated origin–destination pairs.

Model	Average distance driven by vehicle (km)	Average walking distance (m)	Average number of transfers	Average accessibility degree	Average processing time to find the optimal route (s)
Classical Dijkstra (wPenalty = 1, trPenalty = 0)	39.01	1104.68	11.35	**0.59**	6.04
Classical Dijkstra with Fuzzy penalty (wPenalty = 1, trPenalty = 0)	38.92	873.79	11.40	**0.69**	6.58
With penalty (wPenalty = 1, trPenalty = 1)	38.22	1194.22	10.11	**0.61**	5.92
With fuzzy penalty (wPenalty = 1, trPenalty = 1)	38.42	922.81	9.09	**0.67**	6.04
With penalty (wPenalty = 3, trPenalty = 3)	38.33	441.40	6.35	**0.63**	5.72
With fuzzy penalty (wPenalty = 3, trPenalty = 3)	38.15	426.28	6.35	**0.68**	6.07
With penalty (wPenalty = 5, trPenalty = 5)	38.26	366.27	5.80	**0.62**	5.72
With fuzzy penalty (wPenalty = 5, trPenalty = 5)	38.25	353.09	5.80	**0.69**	6.24
With penalty (wPenalty = 10, trPenalty = 10)	38.76	308.98	5.27	**0.62**	5.80
With fuzzy penalty (wPenalty = 10, trPenalty = 10)	38.77	299.53	5.27	**0.69**	6.13

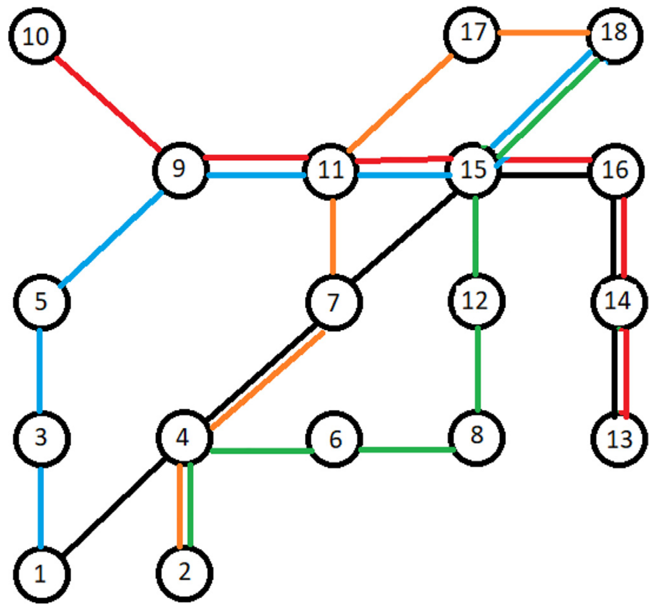


Fig. 11. An example network.

and Fig. 16, in each of these cases, an increase of approximately 11.5% has been achieved in the average accessibility of the paths produced by the FuzzyDijkstra algorithm using fuzzy penalty. At the same time, it is seen that an average of 13.3% reduction has been achieved in walking distances. In contrast, the FuzzyDijkstra algorithm used an average of 2.2% more time to solve. This time delay, whose actual value is less than 0.5 s, will not affect the practical use of the algorithm much.

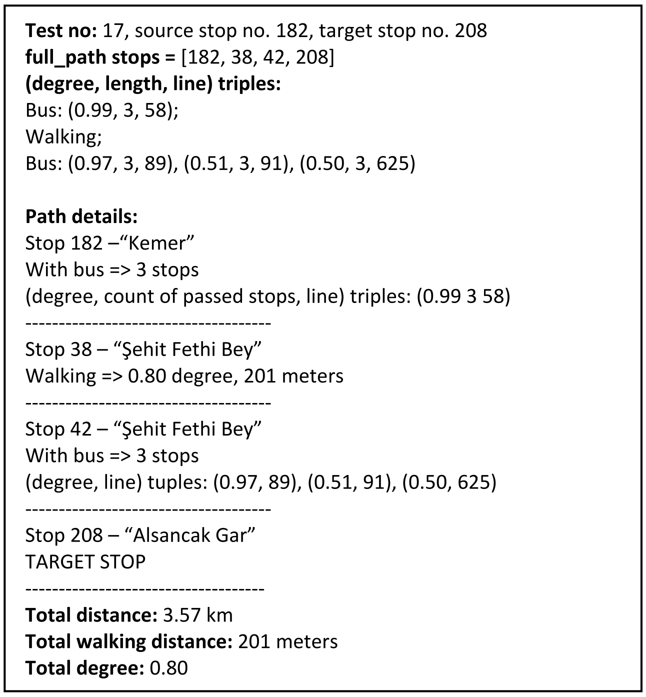
As it is seen from Table 3, the accessibility degrees of the paths produced by both the classical Dijkstra and the Modified Dijkstra algorithm are very low because they do not consider the accessibility degrees at all. There are two reasons for this: first, the lengths of the walking distances are not considered and second, the accessibility of the lines used is not taken into account. For example, although some walking distance between the transfer stops on the route is 500 m, another walking distance may be very small, and this situation does not differ in terms of optimizing criteria. This is based on the 0/1 classical logic of the criteria used in walking distance. In other words, at a certain threshold walking distance limit (e.g. 500 m), all walking trips are equally suitable, and walking distance outside this limit is not at all suitable. But in the newly proposed model in our study, walking distance is considered based on fuzzy degree, and penalty values are produced by taking into consideration this fuzzy degree. Therefore, the longer the walking distance, the higher the penalty for it. As a result of this approach, routes with less walking distances are produced. Experimental results using actual PTN data also confirm the accuracy of this approach.

Step	**Set S** **(shortest paths with cost values)**	**Set U** **(unsettled)**
1	S={(1)} (1)→(1)=0 Seek from (1)	U={(2),(3),…,(18)} (1) → (3)=10 (1) →(4)=14 (1) →others=∞ Find out the path with minimal cost: **cost((1) →(3))=10+0*10+(1-1)*10=10** **(m=0, µ=1)**
2	S={(1),(3)} (1)→(1)=0 (1) →(3)=10 Seek from (1) →(3)	U={(2),(4),…,(18)} (1) → (3) →(5)=20 (1) →(3) →others=∞ Find out the path with minimal cost: **cost((1) →(4))=14+0*10+(1-1)*10=14** **(m=0, µ=1)**
3	S={(1),(3),(4)} (1)→(1)=0 (1) →(3)=10 (1) →(4)=14 Seek from (1) →(4)	U={(2),(5),(6),…,(18)} (1) → (4) →(6)=34 (1) →(4) →(7)=28 (1) →(4) →(2)=34 (1) →(4) →others=∞ Find out the path with minimal cost: **cost((1) →(3) →(5))=20+0*10+(1-1)*10=20** **(m=0, µ=1)**
4	S={(1),(3),(4),(5)} (1)→(1)=0 (1) →(3)=10 (1) →(4)=14 Seek from (1) →(3) →(5)	U={(2),(6),…,(18)} (1) → (3) →(5) →(9)=34 Find out the path with minimal cost: **cost((1) →(4) →(7))=28+0*10+(1-1)*10=28** **(m=0, µ=1)**

Fig. 12. The first four steps of FuzzyDijkstra algorithm in numerical example.

17	S={(1),(3),(4),(5),(7),(2),(6),(5),(8),(9),(15),(11),(12),(16),(10),(17)} (1)→(1)=0 (1) →(3)=10 (1) →(4)=14 (1) →(3) →(5)=20 (1) →(4) →(7)=28 (1) →(4) →(2)=24+10=34 (1) →(4) →(6)=24+10=34 (1) →(3) →(5) → (9)=34 (1) →(4) →(6) → (8)=34+10=44 (1) →(3) →(5) →(9) →(11)=44 (1) →(4) →(7) → (15)=42+4=46 (1) →(4) →(7) → (11)=38+10=48 (1) →(4) →(6) → (8) → (12)=44+10=54 (1) →(4) →(7) → (15) → (16)=52+4=56 (1) →(3) →(5) → (9) → (10)=48+10=58 (1) →(4) →(7) → (11) → (17)=52+10=62 Seek from (1) →(4) →(7) →(11) →(17)	U={(18)} (1)→(4)→(7)→(11)→(17)→(18)=62+10=72 Find out the path with minimal cost: **cost((1)→(4)→(7)→(11)→(17)→(18)) =** **62+10+(1-1)*10=72** **(m=1, µ=1)**
18	S={(1),(3),(4),(5),(7),(2),(6),(5),(8),(9),(15),(11),(12),(16),(10),(17),(18)} (1)→(1)=0 (1) →(3)=10 (1) →(4)=14 (1) →(3) →(5)=20 (1) →(4) →(7)=28 (1) →(4) →(2)=24+10=34 (1) →(4) →(6)=24+10=34 (1) →(3) →(5) → (9)=34 (1) →(4) →(6) → (8)=34+10=44 (1) →(3) →(5) →(9) →(11)=44 (1) →(4) →(7) → (15)=42+4=46 (1) →(4) →(7) → (11)=38+10=48 (1) →(4) →(6) → (8) → (12)=44+10=54 (1) →(4) →(7) → (15) → (16)=52+4=56 (1) →(3) →(5) → (9) → (10)=48+10=58 (1) →(4) →(7) → (11) → (17)=52+10=62 (1) →(4) →(7) →(11) →(17) →(18)=62+10=72	U={(13),(14)} Stop because the target (18) has been reached. The optimal path is: **(1)→(4)→(7)→(11)→(17)→(18)** **with cost=72, m=1 and µ=1.**

Fig. 13. The last two steps of FuzzyDijkstra algorithm in numerical example.



```
Test no: 17, source stop no. 182, target stop no. 208
full_path stops = [182, 38, 42, 208]
(degree, length, line) triples:
Bus: (0.99, 3, 58);
Walking;
Bus: (0.97, 3, 89), (0.51, 3, 91), (0.50, 3, 625)

Path details:
Stop 182 –"Kemer"
With bus => 3 stops
(degree, count of passed stops, line) triples: (0.99 3 58)
-------------------------------------
Stop 38 – "Şehit Fethi Bey"
Walking => 0.80 degree, 201 meters
-------------------------------------
Stop 42 – "Şehit Fethi Bey"
With bus => 3 stops
(degree, line) tuples: (0.97, 89), (0.51, 91), (0.50, 625)
-------------------------------------
Stop 208 – "Alsancak Gar"
TARGET STOP
-------------------------------------
Total distance: 3.57 km
Total walking distance: 201 meters
Total degree: 0.80
```

Fig. 14. Example of the optimal path output.

5. Conclusion

In this study, a new model is proposed using path accessibility concept when calculating the optimal route between the given origin and destination bus stops. The model is based on the P'-space representation of the public transport network. To find solutions for the proposed model, a new FuzzyDijkstra algorithm using fuzzy penalty values has been proposed. To calculate the degree of fuzzy accessibility between stops, a methodology that considers walking distances and inside the bus occupancy degrees is demonstrated. Comparative analysis of FuzzyDijkstra algorithm using fuzzy penalty is performed with classical Dijkstra and Modified Dijkstra algorithms using normal penalty. To carry out the analysis, optimal routes are calculated using different penalty values among 100 randomly selected pairs of stops. From the results of the calculation, it is seen that the FuzzyDijkstra algorithm, which uses fuzzy penalty, produces results with a higher accessibility degree than the optimal routes produced by both classical and modified Dijkstra algorithms. It is seen that there is an average 11.5% increase in the accessibility degree and an average 13.3% decrease in walking distances. In contrast, the FuzzyDijkstra algorithm used an average of 2.2% more time to solve. This time delay, whose actual value is less than 0.5 s, will not affect the practical use of the proposed algorithm much. Thus, it is observed that the FuzzyDijkstra algorithm is a more efficient algorithm.

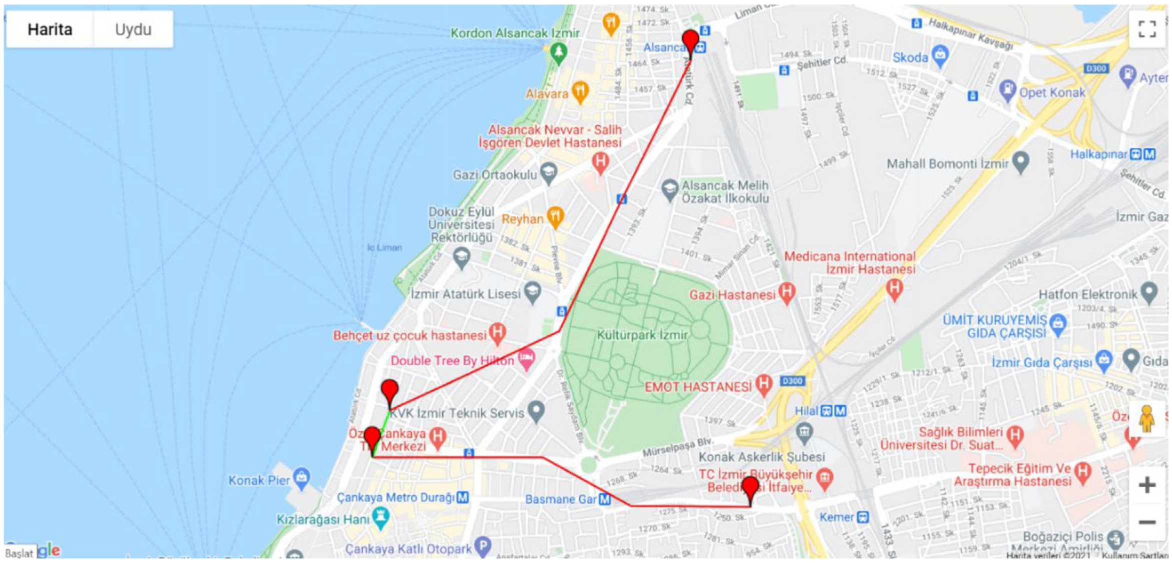


Fig. 15. Display of the calculated example of the optimal route on the map.

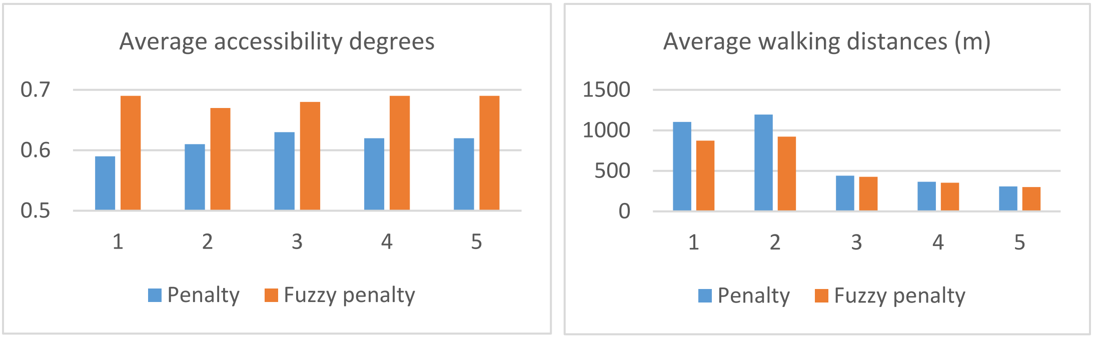


Fig. 16. Results produced by the Modified Dijkstra and the FuzzyDijkstra algorithms: (a) Average accessibility degrees of the routes, (b) Average walking distances.

Otherwise, the solution time of the FuzzyDijkstra algorithm, which takes about 5–6 s on the computer, may take more time on mobile devices. In our future studies, we think that research should be conducted to increase the performance of the FuzzyDijkstra algorithm.

Funding

This research did not receive any specific grant from funding agencies in the public, commercial, or not-for-profit sectors.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Data availability

Data will be made available on request.

References

[1] A. Ceder, Public Transit Planning and Operation: Theory, Modelling and Practice, Oxford, Butterworth-Heinemann, 2007.
[2] E. Nasiboglu, A. Bozyigit, Y. Diker, Analysis and evaluation methodology for route planning applications in public transportation, in: 9th International Conference on Application of Information and Communication Technologies, AICT, Rostov on Don, 2015, pp. 477–481, http://dx.doi.org/10.1109/ICAICT.2015.7338605.
[3] L.F. Costa, F.A. Rodrigues, G. Travieso, P.R. Villas Boas, Characterization of complex networks: A survey of measurements, Adv. Phys. 56 (1) (2007) 167–242, http://dx.doi.org/10.1080/00018730601170527.
[4] C. von Ferber, T. Holovatch, Y. Holovatch, V. Palchykov, Network harness: metropolis public transport, Phys. A 380 (1–2) (2007) 585–591.
[5] C. von Ferber, T. Holovatch, Y. Holovatch, V. Palchykov, Public transport networks: empirical analysis and modeling, Eur. Phys. J. B 68 (2) (2009) 261–275.
[6] A.A. De Bona, K.V.O. Fonseca, M.O. Rosa, R. Lüders, M.R.B.S. Delgado, Analysis of public bus transportation of a Brazilian city based on the theory of complex networks using the P-space, Math. Probl. Eng. (2016) 3898762, http://dx.doi.org/10.1155/2016/3898762, 12 pages.
[7] J. Sienkiewicz, J.A. Holyst, Statistical analysis of 22 public transport networks in Poland, Phys. Rev. E 72 (4) (2005) 046127.
[8] X. Xu, J. Hu, F. Liu, L. Liu, Scaling and correlations in three bus-transport networks of China, Phys. A 374 (1) (2007) 441–448.
[9] Q. Xu, Z.-H. Zu, Z.-J. Xu, W.-D. Zhang, T. Zheng, Space P-based empirical research on public transport complex networks in 330 cities of China, J. Transp. Syst. Eng. Inf. Technol. 13 (1) (2013) 193–198.
[10] H. Zhang, P. Zhao, J. Gao, X.-M. Yao, The analysis of the properties of bus network topology in Beijing basing on complex networks, Math. Probl. Eng. (2013) 694956, 6 pages.
[11] L. Zhang, X. Ma, H. Wang, M. Feng, S. Xue, Modelling and optimization on bus transport system with graph theory and complex network, Int. J. Comput. Appl. Technol. 48 (1) (2013) 83–92.
[12] L. Zhang, H. Ma, B. Sun, Y. Li, M. Wang, S. Xue, Computing and application on complex bus network system, Int. J. Inf. Comput. Sci. 2 (7) (2013) 127–132.
[13] Y.-Z. Chen, N. Li, D.-R. He, A study on some urban bus transport networks, Phys. A 376 (1–2) (2007) 747–754.
[14] Y.-Z. Chen, N. Li, The randomly organized structure of urban ground bus-transport networks in China, Phys. A 386 (1) (2007) 388–396.
[15] Z.-T. Zhu, J. Zhou, P. Li, X.-G. Chen, An evolutionary model of urban bus transport network based on B-space, Chin. Phys. B 17 (8) (2008) 2874–2880.
[16] J.G. Kang, The minmax regret shortest path problem with interval arc lengths, Int. J. Control Autom. 6 (5) (2013) 171–180.
[17] M. von Lossow, A min–max version of Dijkstra's algorithm with application to perturbed optimal control problems, PAMM Proc. Appl. Math. Mech. 7 (2007) 4130027–4130028, http://dx.doi.org/10.1002/pamm.200700646.
[18] A. Bozyigit, G. Alankus, E. Nasibov, A public transport route recommender minimizing the number of transfers, Sigma J. Eng. Nat. Sci. 9 (4) (2018) 437–446.
[19] A. Bozyiğit, G. Alankuş, E. Nasiboğlu, Public transport route planning: Modified Dijkstra's algorithm, in: International Conference on Computer Science and Engineering, (UBMK), Antalya, Turkey, 2017, pp. 502–505, http://dx.doi.org/10.1109/UBMK.2017.8093444.
[20] L. Zhao, Y. Xiong, H. Sun, The k shortest transit paths choosing algorithm in stochastic transit network, in: others Wang G (Ed.), Proceedings of the 3rd International Conference on Rough Sets and Knowledge Technology, (RSKT'08), Springer-Verlag, Berlin, Heidelberg, 2008, pp. 747–754.
[21] M.P. Pelletier, M. Trepanier, C. Morency, Smart card data use in public transit: A literature review, Transp. Res. C 19 (2011) 557–568.
[22] N.H. Wilson, J. Zhao, A. Rahbee, The potential impact of automated data collection systems on urban public transportation planning, in: N.H. Wilson, A. Nuzzolo (Eds.), Schedule-Based Modeling of Transportation Networks, Springer, 2009, pp. 75–99.
[23] M. Bagchi, P.R. White, The potential of public transport smartcard data, Transp. Policy 12 (2005) 464–474.
[24] C. Morency, M. Trepanier, B. Agard, Analyzing the variability of transit users' behaviour with smart card data, in: Proceedings of the IEEE ITSC, 2006, pp. 44–49.
[25] E.N. Nasiboglu, U. Kuvvetli, M. Ozkilcik, U. Eliiyi, Origin-destination matrix generation using smart card data: Case study for izmir, in: Proceedings of the IV International Conference Problems of Cybernetics and Informatics, Vol. 1, PCI'2012 Baku, Azerbaijan, 2012, pp. 188–191.
[26] E.N. Nasibov, A. Diker, E. Nasibov, A multi criteria route planning model based on fuzzy preference degrees of stops, Appl. Soft Comput. 49 (2016) 13–26.
[27] Dokuz Eylul University,Turkey, A. Diker, Usage of Fuzzy Logic Based Data Mining Methods in Analysis of Public Transportation Data (Ph.D. thesis), 2015.
[28] R. Nasiboğlu, in: Gürbüz F. (Ed.), Fen Bilimleri ve Matematik Alanında Güncel Araştırmalar, Duvar Yayınları, ISBN: 978-625-7680-07-3, 2020, pp. 21–38,
[29] M.A. Munizaga, C. Palma, Estimation of a disaggregate multimodal public transport origin–destination matrix from passive smartcard data from Santiago Chile, Transp. Res. C 24 (2012) 9–18.
[30] M. Trepanier, N. Tranchant, R. Chapleau, Individual trip destination estimation in a transit smart card automated fare collection system, J. Intell. Transp. Syst. 11 (1) (2007) 1–14.
[31] J. Barry, R. Newhouser, A. Rahbee, S. Sayeda, Origin and destination estimation in new york city with automated fare system data, Transp. Res. Rec. 1817 (2002) 183–187.
[32] K.A. Seaton, L.M. Hackett, Stations, trains and small-world networks, Phys. A 339 (3–4) (2004) 635–644.
[33] P. Sen, S. Dasgupta, A. Chatterjee, P.A. Sreeram, G. Mukherjee, S.S. Manna, Small-world properties of the Indian railway network, Phys. Rev. E 67 (3) (2003) 036106.
[34] S. Derrible, C. Kennedy, The complexity and robustness of metro networks, Phys. A 389 (17) (2010) 3678–3691.
[35] J. Zhang, X. Xu, L. Hong, S. Wang, Q. Fei, Networked analysis of the shanghai subway network in China, Physica A 390 (23–24) (2011) 4562–4570.
[36] I. Mahdavi, R. Nourifar, A. Heidarzade, N.M. Amiri, A dynamic programming approach for finding shortest chains in a fuzzy network, Appl. Soft Comput. 9 (2009) 503–511.
[37] Y. Deng, Y. Chen, Y. Zhang, S. Mahadevan, Fuzzy Dijkstra algorithm for shortest path problem under uncertain environment, Appl. Soft Comput. 12 (2012) 1231–1237.
[38] X. Ji, K. Iwamura, Z. Shao, New models for shortest path problem with fuzzy arc lengths, Appl. Math. Model. 31 (2007) 259–269.
[39] C. Erbao, L. Mingyong, The open vehicle routing problem with fuzzy demands, Expert Syst. Appl. 37 (2010) 2405–2411.
[40] A.D. Jovanovic, D.S. Pamucar, S. Pejcic-Tarle, Green vehicle routing in urban zones – a neuro-fuzzy approach, Expert Syst. Appl. 41 (2014) 3189–3203.
[41] G. Cirovic, D. Pamucar, D. Bozanic, Green logistic vehicle routing problem: Routing light delivery vehicles in urban areas using a neuro-fuzzy model, Expert Syst. Appl. 41 (2014) 4245–4258.
[42] D. Luo, L. Bonnetain, O. Cats, H. van Lint, Constructing spatiotemporal load profiles of transit vehicles with multiple data sources, Transp. Res. Rec. 2672 (8) (2018) 175–186.
[43] E.W. Dijkstra, A note on two problems in connexion with graphs, Numer. Math. (1) (1959) 269–271, http://dx.doi.org/10.1007/BF01386390.
[44] G.R. Brummelen, Heavenly Mathematics: The Forgotten Art of Spherical Trigonometry, Princeton University Press, 2013.
[45] ESHOT official website, 2020, https://www.eshot.gov.tr/. (Accessed 16 November 2020).
[46] Izmir PTN data https://github.com/resilla/FuzzyDijkstra/blob/main/IzmirPTNData.rar.