Available online at www.sciencedirect.com

ScienceDirect

Procedia Computer Science 35 (2014) 614 – 623

Procedia Computer Science

18th International Conference on Knowledge-Based and Intelligent Information & Engineering Systems - KES2014

Unscheduled public transport intelligent navigation system

José Sendra Salcedo*, Osvaldo Cairó Battistuti

Instituto Tecnologico Autonomo de Mexico,Río Hondo 1, Mexico City 04000, Mexico

Abstract

This paper tackles the problem of searching unscheduled public transport fast routes for Mexico City proposing as solution a Public Transport Navigation System (PTNS) for mobile devices. The intelligent system proposed in this work finds fast public transport routs to a destination by using a search algorithm with a knowledge based time dependent heuristic. This heuristic captures the knowledge of public transport expert users and combines it with data given by transport companies to find the fastest routes available. The algorithm generates an estimated time of arrival (ETA) and finds the best rout in less than 10 seconds.

Keywords: Heuristic; Graph; Experts Knowledge; Navigation System;Mobile Application

1. Introduction

Mexico City is one of the largest cities in Latin America[1]; it has more than 20 million habitants in 7,850 km^2. Having so many people in such a large extension of terrain generates great infrastructure and services challenges; one of them is public transport. How do you transport efficiently so many people from one point to another in one of the biggest cities of the world?

Public transport systems in large cities are formed by many trolley, bus and subway lines. In the case of Mexico City, there are 12 subway lines and over 350 bus lines that do more than 12 million trips daily. Most of these lines are not scheduled and have no accurate map of their route. As a result, it is hard for users to make efficient trips and calculate the time of arrival to their destinations. Only expert users that have lived long enough

* Corresponding author. Tel.: +52-55-5416-4295
E-mail address: jsendrasa@itam.mx

doi:10.1016/j.procs.2014.08.143

in the city to know many transport lines and their average speed and waiting-time, can choose a fast rout to their destination predicting an accurate estimated time of arrival (ETA). This situation makes inexpert users avoid public transport services, preferring taxis or using their own car. Consequently, there are more traffic jams and pollution, some of the major problems in the city.

In this work a PTNS based on public transport expert users knowledge is proposed as a solution to the chaos in Mexico City public transport network. This system is able to find fast routs to destinations by using a search algorithm with a discrete time dependent heuristic. The PTNS uses a database with information of all subway, rail and regulated bus stations in the city. The public transport lines in the database were chosen because they are the most used in the city and with more information available.

At the present time if a person wants to go to a location using public transport, he will schedule the trip based on the remoteness of his destination, average speed of the transports to be used, transfer time between different lines and the estimated waiting-time for vehicles. The estimation of the travel time is more accurate if the person uses the same route frequently. The PTNS copies this human behavior when finding the optimal route resuming it in four knowledge-based factors:

- Time that it takes to change of transport services, e.g. subway to bus.
- Average speed of each transport.
- The date and hour in which the trip will be made.
- Waiting-time for each transport line.

Given this factors the system estimates travel times while generating each possible route, always choosing the one with less estimated travel time and delivering at the end an optimal rout with the shortest ETA according to the criteria before stated.

The goal of this paper is to show the reader a unique approach for solving a problem many path finding search implementations avoid. Finding fast routes in the chaotic and uncertain public transport environment. Expert users´ knowledge can be collected and combined with transit services providers' information to create a heuristic search algorithm that can find fast routes from point to point with a short calculation time; In addition, this paper will provide the reader with an understanding of the PTNS route search heuristic construction.

In first place this work will describe the existing technologies and past works, describing numerical route search algorithms and solutions that use real time data applied in public transport navigation systems in several countries, although this solutions could seem apply able to Mexico City transport system, they aren't the best options to be implemented in a chaotic environment, as seen later in the tests and results. Afterwards, a description of the PTNS proposed in this work will be made, deepening on the way the knowledge was extracted from public transport expert users, the methods followed to group it and how was it used to build the search heuristic. Afterwards, a description of how does the fastest route search algorithm works will be made. Next, the heuristic of the PTNS is validated showing results obtained of tests made to 200 public transport users. Finally conclusions about the search algorithm heuristic are made and future work and applications are proposed.

2. Existing technologies

The evolution of navigation applications has been accelerated in the last years, many new car navigation apps have flooded the market for mobile devices, this has brought a lot of attention into algorithms that can find optimal routes in a map given certain conditions and restrictions like traffic, trip time and users preferences.

Given the data of the public transport network, the problem of finding the fastest path between two stations is generally modeled as a shortest path, minimum delay problem in a graph, where the nodes of the graph represent the transport stations, the edges the connections between the station and the weights the time between stations.

There is a lot of literature about shortest path algorithms that has been studied widely; readers interested in theory about algebraic path problems are referred to[2,3]. Some of these algorithms model the weight of the edges as a time dependent variable[4]. This kind of problems can be solved with linear programming modified Dijkstra Algorithm[5].

Though Dijkstra can find the solution, the process takes a lot of time. Nonetheless, a heuristic approach can be made to accelerate the search[6,7]. Tulp and Siklóssy decided to incorporate heuristics on a timed table network to speed up the fastest path search [8].

Now, we will discuss how this problem has been concretely addressed to public transport and different solutions that have been proposed.

Google has made lots of research in the area; they have developed navigation systems for walking, car and public transport. Nevertheless, Google maps public transport navigation systems is only available in a small amount of highly developed cities. This is because they find public transport routes using real time data that is uploaded by transit companies in a standardized format called General Transit Feed Specification (GTFS). The GTFS format requires many parameters; some of the most important are stops location, trip schedules and transfers. This information is, in many occasions, made public by the transit agencies; so many developers use it to create accurate routing algorithms.

K. Nachtigal modeled railroads transport network in Germany as a discrete time dependent network and implemented an algorithm that searched using Artificial Intelligence (AI) heuristic techniques and a label correcting technique to find fastest paths between two stations, in his work he shows that AI search techniques are more efficient than algebraic search like Dijkstra for one to one search methods [9].

To speed up computer processing Jerald Jariyasunant proposed in his work pre-generating all possible paths from station X to Y, and after that, looking for the optimal path in the result set instead of calculating the optimal paths each time the user requires it. This method has been already used in some train navigation systems in Germany using real time data [10].

Although many of these solutions are widely proven and efficient, they cannot be applied to Mexico City public transport system. This is because the transport system lines are not properly regulated and there is no source of real time data of any of them. To this problem we must add that the transports speed is highly variable and arrival and departure times are not scheduled.

3. Public transport navigation system

The PTNS designed in this work generates the fastest path between input stations A and B using knowledge of expert public transport users and displays the path on a mobile device This navigation system is capable of finding routes in graphs conformed by over 19,000 stations in less than 10 seconds . Three layers compose the PTNS: mobile I/O layer, route search layer, and transport database layer. The architecture of the dataflow can be seen on Figure 1.

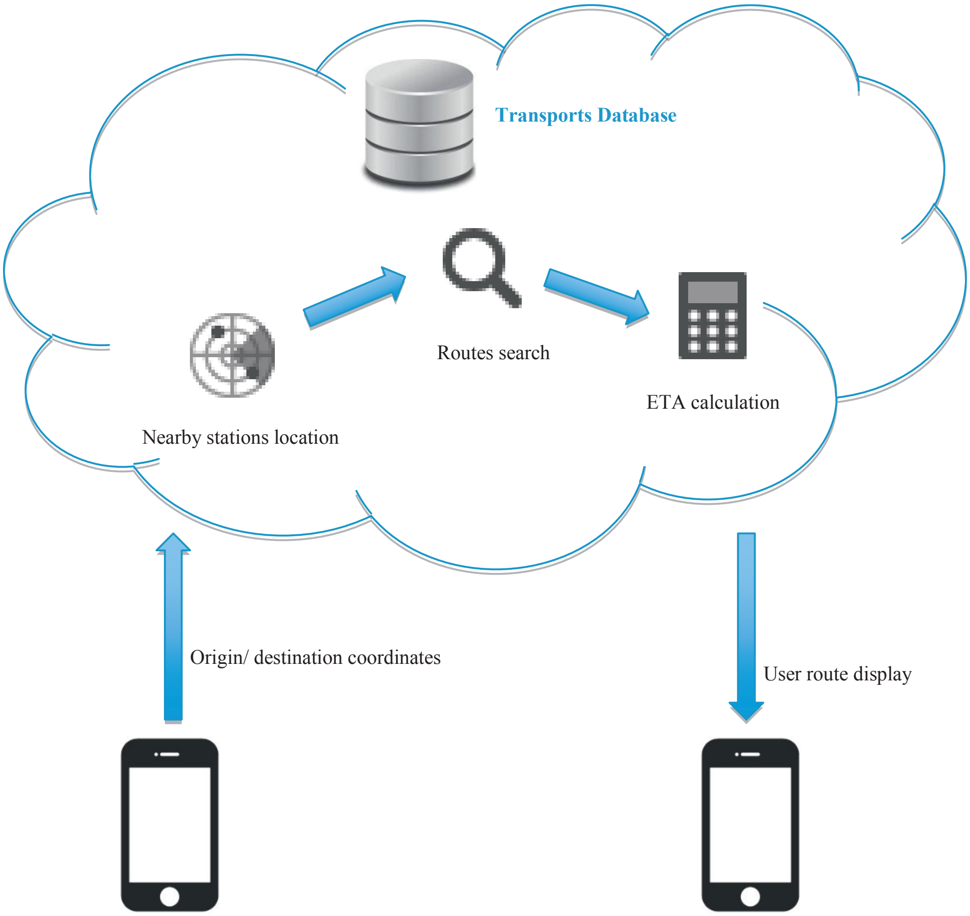

Figure 1: PTNS Architecture and Data Flow

When a user wants to get the fastest route from station A to B, he access to the Mobile I/O layer, in this layer the user's desired origin and destination is retrieved given a landmark or a direction specified using Google maps API. The user also provides the time in which he desires to do the trip. The mobile application sends this information to the route search layer. In that layer a web service uses the provided information to locate in the database all stations that are 500mts round of the destination coordinates. In case there are no stations that close, the system looks for the nearest station, these will be the destiny stations DS. The same process is made with the origin coordinates, generating the origin stations OS.

After this, the Route search layer will look for an optimal route from each station in OS to ach in DS; each of these routes is found by finding the path of minimum delay in a graph where the nodes represent the transport stations, the edges the connections between the station and the weights the time between stations. The system will search for an optimal route to the destination in this graph, using an heuristic driven search, the heuristic used is described in point 3.2.

After finding the fastest routes from the stations in the OS set to the ones in DS, the web service provides the Mobile I/O layer the three fastest routes found, including the stations that form it and the ETA. In this layer, the mobile device used by the user displays these routes in a list so the user chooses the one that best fits him; the chosen route is displayed in a map. A step-by-step guide and ETA are also displayed.

The database used by the system contains 12,000 public transport stations, the estimated waiting time for each transport line, the estimated speed of each transport and the transfer time for stations where transfer is possible. Each station in the database has its coordinates, the name of the service and line it belongs to.

3.1 Expert users and knowledge extraction

A public transport expert user is defined as a person with the following characteristics:

- Uses a public transport at least five times in a week.
- Used the public transport for at least four years.
- Is familiar with all public transport lines of Mexico City.
- Knows the average speed of each transport line.
- Knows the average waiting time for each transport line.
- Knows many public transport station locations.
- Knows average transfer time between many transport lines.

Due to their experience, expert users knowledge is a key feature in creating a heuristic that can be used in a search algorithm that imitates their process of fast public transport routes calculation.

The knowledge extracted from expert users for the creation of the PTNS and expert users heuristic was the following:

- Line waiting time
- Line average speed
- Transfer time

To extract this knowledge from expert users, different surveys were made for each transport line, 200 expert users were surveyed about regulated transport line (subway, trans, train, regulated bus lines) and 400 were surveyed about chaotic bus lines. A total of 600 users were surveyed.

3.2 PTNS Expert users heuristic

The Expert Users Heuristic (EUH) imitates the behavior of expert Mexico City public transport users; this heuristic is built by the following components explained in 3.2.1 – 3.2.4.

- Distance to destination and trail
- Transfer time
- Line waiting time
- Line speed

3.2.1 Distance to destination and trail factor

The Distance to Destination and Trail Factor (DTF) helps the system search algorithm look for the shortest route from the origin to destination, this factor is measured using the A* search algorithm distance plus cost heuristic. This value consist on calculating for each path from Origin station (O) to station X the length of the trail plus the distance from X to the destination (D)

$$DTF(R_{O,X}) = \sum_{i=0}^{n} d(s_i, s_{i+1}) + d(X, D) \quad (1)$$

$$with:$$

$$DTF(R_{O,X}) = distance\ to\ destination\ and\ trail\ factor\ for\ route\ R_{O,X}$$
$$R_{O,X}\ a\ rout\ from\ the\ origin\ station\ to\ station\ X.$$
$$R_{O,X} = \{s_i, i \in 1\ , 2 \ldots n\}\ \ where\ n\ is\ the\ index\ of\ the\ last\ station\ in\ the\ route\ \ and\ s_i = station\ i\ of\ the\ route$$
$$s_0\ =\ origin\ station\ of\ the\ route.$$

$$s_n = X \text{ the last staion of the route analized.}$$
$$d(s_i, s_{i+1}) = \text{Distance between stations } s_i \text{ and } s_{i+1}\text{, calculated using the harvesine formula}$$

(2)

$$d(s_i, s_{i+1}) =$$
$$2R * asin\left(\sqrt{\left[\left(\frac{\sin\left([lat(s_{i+1}) - lat(s_i)] * \frac{\pi}{180}\right)}{2}\right)^2 + \left(\frac{\sin\left([lon(s_{i+1}) - lon(s_i)] * \frac{\pi}{180}\right)}{2}\right)^2\right] * \cos\left(lat(s_i) * \frac{\pi}{180}\right) * \cos\left(lat(s_{i+1}) * \frac{\pi}{180}\right)}\right)$$

$$\text{with:}$$
$$lat(s_i) \text{ latitude of satation } s_i$$
$$lon(s_i) \text{longitude of station } s_i$$
$$R\text{: Earth radius}$$

The DTF is included in the PTNS to resemble the expert user knowledge of shortest routes, though the system will have the advantage of being able to quickly compute the shortest path between any two stations. Nevertheless, the shortest path is not the fastest one. Therefore more factors taken into account by expert users must be analyzed.

3.2.2 Line waiting time factor

The Line Waiting Time Factor (LWTF) is used to model the time a person has to wait for a transport in a specific line to arrive. The time between transports in Mexico City is highly variable; it depends on the hour, day and type of transport. Expert users know the estimated mean waiting time for each transport and therefore they know which are better to use depending on time and day.

To measure the mean waiting time for each transport line, a sample of times between transports was made for each transport line; each sample consisted of the hour, day, transport line, and time between transports in minutes. After carefully analyzing the data, it was clustered in three main categories for each transport line: high, low and standard waiting time.

This factor is also used to check if transports are available at a certain hour, if a user wants to travel at 1:00 AM to certain destination many transports will not be available, to model this the waiting time for the unavailable transports in near to infinite, this will force the fastest path search algorithm to avoid paths that include unavailable transports.

In Metrobus transport for example the clustered time table obtained for waiting time can be seen on Table 1.

Table 1: Clustered waiting time table

Waiting Time	Days
Long	Mon-Fri 19:00-23:30 Sat-Sun 6:00-23:30
Standard	Mon-Fri 6:00-11:00
Short	Mon-Fri 11:00-19:00
Not Available	23:30-6:00

3.2.3 Transfer factor

Due to the huge extent of Mexico´s City transport system it is probable that users cannot arrive to their destinations using only one mean of transport, therefore they frequently need to make multiple transfers. Expert users know that a fast route generally implies doing a small amount of transfers because stations are generally distant and doing a transfer takes a lot of time. The Transfer Factor (TF) imitates this expert user's behavior by calculating the time it takes to perform a transfer from one service to another. This factor is calculated by measuring the average time a person takes to walk from one line station (A), to another (B) and adding the LWTF of the new line:

$$TF(A,B) = WT(A,B) + LWTF(B) \tag{3}$$

$$with:$$
$$TF(A,B) = transfer\ factor\ from\ station\ A\ to\ B$$
$$WT(A,B) = walking\ time\ from\ station\ A\ and\ B$$
$$LWTF(B) = line\ B\ waiting\ time\ factor$$

The walking time was defined by the results in the surveys made to expert users and by measuring the average time it takes to perform transfers between services. The results were stored into the web service database. This factor is used when the search algorithm is generating possible routes, when it checks if it is good to move from station x to station y, in case station y has a different line or service than station x, the TF would be calculated.

3.2.4 Line speed factor

When an expert user chooses a transport, one of the main aspects taken into account is its speed, if he wants to arrive to his destination in the shortest time he will use the fastest transports available. Expert users calculate the transports speed based on their experience, knowing that at certain hours, in the case of buses, traffic decreases speed drastically. In the case of subways, during midnight the service is slow.

The transport line speed (LSF) can imitate this expert users behavior by estimating the speed according to the time and day. This data was calculated with help of the survey results and by measuring the time different transport lines take to do round trips, and given the length of the route calculate the speed. Following the same procedure of LWTF we can categorize the speeds of each transport depending on time and day.

4. The fastest route search method

The EUH is used in a search algorithm that expands nodes (stations) of the transport network and compares them to see which one is the best option as a next stop in the route that is being calculated. Given the origin station O, The EUH is calculated for all available sub paths from O to X, where X is a station connected directly to O. The EUH is calculated as follows:

$$EUH(X,X) = \frac{d(X,O)}{WS} + \frac{d(X,D)}{LS(lineX)} + LWTF(X) \tag{4}$$

$where:$

$EUH(X,X)\ is\ the\ experts\ user\ heuristic\ value\ for\ the\ station\ X\ and\ route\ form\ origin\ to\ X$

$$lineX = transport\ line\ of\ station\ X.$$

$$LS(lineX) = line\ speed\ factor\ of\ line\ in\ station\ X.$$

$$LWTF(X) = Line\ waiting\ time\ factor\ for\ the\ transport\ of\ station\ X.$$

$$WS = Walking\ Speed.$$

Once the EUH is calculated for each station X the algorithm stores the pairs of X and their EUH score in the set **Ω**, afterwards the algorithm looks for the $X \in \Omega$ with minimum EUH score obtaining the tuple X_{min} and $Sc(X_{min})$, where X_{min} is the station $X \in \Omega$ with minimum EUH and $Sc(X_{min})$ is the X_{min} EUH score. The station X_{min} is trespassed from set **Ω** to set Ĉ and it is expanded, calculating the EUH value for every station Y directly connected to station X_{min}. The process is as follows:

(5)

$$(5)\ EUH(X,Y) = Sc(X) - \frac{d(X,D)}{LS(line(X))} + \frac{(1 - It_{X,Y}) * d(X,Y)}{LS(line(Y))} + \frac{PD(d(Y,D)) * d(Y,D)}{LS(line(Y))} + It_{X,Y} * FT(X,Y)$$

$$Where$$

$$It_{X,Y} = \begin{cases} 1\ if\ there\ is\ a\ transfer\ from\ X\ to\ Y \\ 0\ if\ there\ is\ no\ transfer\ from\ X\ to\ Y \end{cases}$$

$$Y = station\ directly\ connected\ to\ station\ X$$
$$X = station\ with\ minimum\ EUH\ score\ in\ \Omega\ set$$
$$R_{X_{min},Y} = Route\ from\ station\ X_{min}\ to\ station\ Y$$
$$D = destination$$
$$EUH(Y,Y) = experts\ user\ heuristic\ value\ for\ the\ station\ X\ to\ station\ Y$$
$$Sc(X) = X\ EUH\ score$$

The results are stored in **Ω**, the process is repeated until the destination station S is reached or there are no more nodes in **Ω** to expand, in case there are no nodes this means there is no path form station O to station D. When the station D is reached a backtracking algorithm will rebuild the fastest route using the information of pairs stored in Ĉ.

5. Algorithm complexity

The complexity of the algorithm showed in this work for the worst case scenario is exponential $O(E)$. This is because in the worst case the algorithm would have to expand all the nodes of the public transport networks until a station y the destination set is reached. The exponential complexity depends on the depth in which the search must be made in order to find a solution and the number of connections each node has (ramification factor).

The average ramification factor in the Mexico City transport network is of 7, therefore in average is possible to go from one station to 7 different stations. This allows us to enunciate the complexity of the algorithm as follows:

(6)

$$O(E) = O(r^l) = O(r^7)$$

$$Where:$$
$$r = ramification\ factor$$
$$l = depth\ of\ the\ solution$$

6. PTNS test and results

The PTNS was tested on 200 persons of ages between 19 and 35. From the 200 persons 140 were expert public transport users and 60 were users unfamiliar with Mexico City public transport system. The group was divided in two subgroups, A and B. Group A was formed by 100 persons that had the PTNS installed in their mobile device, of those, 30 were inexperienced public transport users. Group B was assembled by 100 persons that didn´t have the PTNS installed on their cell phones, of those, 30 were inexperienced public transport users.

The tests were conducted by applying a small survey to two persons to identify their transportation habits and classify them as expert public transport users or inexperienced users and see if at least one of them had a smartphone with the requirements for the PTNS. Afterwards they one of them was assigned to group A, and the other to group B, the PTNS was installed in the phone of the user of group A. They were both assigned a destination, each one departed with ten-minute time difference. Once they arrived to their destination they reported their time of arrival. If the user was of group B he also indicated the ETA predicted by the PTNS. This test was repeated on different pairs of people until it had been tested on 200 persons. The destination and origin in each test case was different and every transport service was used in the tests.

The results obtained from the tests were that persons of group A arrived earlier to their destination 90% of the times. The members of group A had a trip time 20% shorter than those of group B.

In the test pairs where both members were expert users, members of group A arrived earlier 95% of the times. These results clearly show how the EUH used by the PTNS generates routes that are faster than the ones calculated by any user manually in almost all the cases.

In the test pairs where the member of group A was an inexpert public transport user and the member of group B was an expert, the first ones arrived earlier 62% of the times, in average their trip time was 10% shorter. The explanation of this low advantage compared with the first results is because inexpert users are not familiar with Mexico City public transport system, therefore they required more time to find the transport line stations suggested by the PTNS and it is easier they commit mistakes. Nevertheless the fact that inexpert users arrived 62 out of 100 times earlier than an expert user shows how the EUH manages and unifies the knowledge of expert users in such a way that the routes it generates are so superior than those generated by experts users that even an inexpert user can arrive earlier to his destination than someone that has years of experience using public transport.

The ETA generated by the PTNS, when compared with the measured arrival time, showed in average a precision of +/-10 minutes. The trips in which buses were used had a precision of +/- 15 minutes while precision in trips where no buses were used was of +/- 6 minutes. This is because bus waiting time and speed greatly vary depending on driver's skills and traffic. In tests made by inexpert users that had the PTNS the ETA had a precision of -25 min. This is because the system doesn't take into account the expertise of the users and assumes they know how to use the public transport system and will follow efficiently the line proposed when calculating the ETA.

The ETA generated by the PTNS is close enough to the measured arrival time to consider it a good estimator; with this data users can plan their trips more effectively.

During numerous functionality and efficiency tests conducted, the PTNS showed an excellent performance, the system was capable of calculating a set of optimal routes in an average time of 5 seconds. The response time was considered competitive when compared with the response time of other navigation systems like Google Maps and Waze which route calculation time ranges between 5 and 15 seconds.

Tests were made with Dijkstra algorithm implementations to solve the current problem. The results were compared with the ones of the algorithm depicted in this work. The Dijkstra implementation in many cases wasn't able to find an optimal rout in a period of time shorter than 5 minutes. This algorithm was always slower than the heuristic search algorithm proposed in the work. The results of both algorithms were very similar. The routes found by Dijkstra were more optimal than the ones of the heuristic algorithm 10% of the times.

The possibility of pre-processing the fastest paths between all the stations of the transport network and storing them to avoid search of them on real time was considered. To do this, the best practice would be using a search algorithm similar to Floyd–Warshall or Johnson's. Nonetheless, the complexity of these algorithms is higher than $O(|V^3|)$ where V is the number of vertexes on the graph where the shortest path is searched. The number of stations existent in Mexico is over 19,000. This means the algorithm would consume a very high amount of time in calculating all the optimal routes. Although this operation should apparently be only performed once, the stations availability changes constantly, therefore the process should be repeated frequently making it impossible to show provisional or unexpected changes in the transport system to the PTNS users.

7. Conclusions

The present work shows how the extraction of the knowledge public transport expert users possess its analysis and use in a heuristic search algorithm can help create a PTNS and avoid the transport company's information gap by the estimation of this data. The PTNS presented in this work reduces users travel time and helps them schedule their journeys. This navigation system affronts the chaos and uncertainty in a complex transport dynamic system by grouping the knowledge of its users, which individually might be insufficient but, when grouped, can solve the system efficiently. This solution is presented for Mexico City but as it depends on users experience and not on companies information it can be applied in other cities with similar conditions. In case the transports are better regulated or have real time data available, they can also be included in the PTNS.

Some of the advantages that the PTNS created in this work are the possibility of including transports that have no schedule at all, like public bike stations or including cars in the transport network, thinking of parking lots as their stations. This way the navigation system will be able to offer routes that take into account every single transportation method to their final users. Also, the system finds fast routes in the moment the user requires it, because of this, if there is a change in the transports database, this change will be reflected immediately in the routes generated.

The system allows the possibility of modifying the database, adding provisional transport routes and deleting routes that are temporally unavailable, creating a channel of communication between the transit service providers and the users and enabling the creation of a new kind of transportation where provisional bus routes could be created based on demand.

Additional to these advantages, the fact that the algorithm compensates the scarcity of information with the expert users' knowledge allows it to outperform other algorithms. Finding short paths in environments where other algorithms cannot be implemented because of the lack of information or where they wouldn't be able to find optimal routes.

Future work includes further research of poorly regulated unscheduled bus transport characteristics that can help model better that transport's speed and waiting times so these characteristics can be added to the heuristic proposed, also the implementation of this system in other cities with similar conditions to Mexico.

Acknowledgements

This work has been founded by Asociación Mexicana de Cultura A.C.

References

1. Forstall RL, Greene RP, Pick JB. Which are the largest? Why lists of major urban areas vary so greatly. *Tijdschrift voor economische en sociale geografie* 2009;100: 277-297.
2. Dijkstra E. A note on two problems in connexion with graphs. *Numerische Mathematik* 1959;1:269–271.
3. Bellman R. On a routing problems. *Quarterly of Applied Mathematics* 1958;16:87–90.
4. Orda A , Rom R. Minimum weight paths in time-dependent networks. *Networks* 1991; 21: 295-319.
5. Orda, A, Rom, R. Shortest-path and minimum-delay algorithms in networks with time-dependent edge-length. *ACM Journal* 1990;37: 607-625.
6. Hart PE, Nilsson NJ, Raphael B. A formal basis for the heuristic determination of minimum cost paths. *IEEE Transactions on Systems Science and Cybernetics* 1968; 4:100–107.
7. Chuang TN, Kung JY. The fuzzy shortest path length and the corresponding shortest path in a network. *Computers and Operations Research* 2005; 32:1409-1428.
8. Tulp E, Siklóssy L. Searching time-table networks. *Artificial Intelligence for Engineering, Design, Analysis and Manufacturing* 1991; 5:189-198
9. Jariyasunant J, Mai E, Sengupta R, Algorithm for finding optimal paths in a public transit network with real-time data. *Journal of the Transportation Research Board* 2010; 2256:34-42.
10. Nachtigal K.. Time depending shortest-path problems with applications to railway networks. *European Journal of Operations Research* 1995;83:154–166.