UDC 004.4

Ph.D. **Haisha O.O.** (ICM CSIC, Barcelona)
Doc. Tech. Sci., Prof. **Lienkov S.V.** (MI KNU, Kyiv)
Haisha O.O. (MKU, Mikolaev)

DOI: http://doi.org/10.17721/2519-481X/2025/87-08

ALGORITHM AND SOFTWARE TOOL FOR SOLVING THE PROBLEM OF LOCAL PASSENGER TRAFFIC

In the context of contemporary urbanization and global environmental challenges, the optimization of passenger transportation systems has emerged as a significant area of research, particularly in the domain of energy conservation and ecological impact reduction. This study addresses a specific aspect of this broad issue: the efficient utilization of empty seats in private vehicles to increase transportation throughput without expanding road infrastructure. The central idea lies in the redistribution of passengers who share similar or overlapping routes with private car owners, leveraging modern information technologies to facilitate real-time matching and communication.

The work begins by outlining the conceptual foundation of the problem, emphasizing that numerous urban residents travel considerable distances daily, while the available transportation infrastructure remains constrained. In light of this, substantial transport efficiency gains can be achieved by utilizing vacant seats in private vehicles. While socio-economic and motivational factors such as environmental awareness, financial incentives, or altruistic behavior may drive such initiatives, the paper focuses solely on the technical and algorithmic aspects of the solution, deliberately excluding the subjective dimension of driver motivation.

A formal problem statement is constructed, modeling the city road network as a mathematical graph where intersections are represented as nodes and road segments (quarters) as edges. Each driver's and passenger's route is defined as a sequence of intersections, and the core condition for assigning a passenger to a driver is the full inclusion of the passenger's route within the driver's route. This constraint, referred to as the "Route Rigidity Rule" (3R), ensures that neither the driver alters his route nor the passenger deviates from his intended path.

The algorithm's architecture is centered on string processing techniques, where route sequences are encoded as strings and inclusion is checked using substring operations. A key function, I(d, p), is introduced to quantify the number of overlapping quarters between a driver's route d and a passenger's route p. This function serves as the basis for verifying route compatibility and ultimately calculating the effectiveness of a particular distribution.

The effectiveness of the system is defined through an objective function E, representing the total number of passenger-quarters served. The optimization goal is to maximize E by efficiently assigning passenger requests to appropriate driver routes. The algorithm iteratively considers driver routes and selects the best-fitting passenger routes based on maximum inclusion, ensuring each route is used only once. An empirical example illustrates how different assignment strategies yield varying levels of effectiveness, measured in the total number of quarters fulfilled.

To operationalize this model, a software solution was developed featuring a user interface for inputting and processing route data. Furthermore, a methodology is proposed to estimate environmental benefits through fuel savings, translating the system's output into quantifiable ecological impact. Experimental results confirm the algorithm's stability and efficiency in handling thousands of routes, with the software consistently distributing passengers in accordance with defined constraints and maximizing the effectiveness metric.

Keywords: urban transportation optimization, ride-sharing algorithm, route matching, intelligent transport systems.

Introduction

In our globalized modern world newish challenges arise in front of the scientific society. One of the most important directions is saving of natural resources, protection of environment, and so on

– ecological researches in short. Ecological problems may be solved in different ways: emission cleaning, recycling, targeted technologies for the protection of certain species, etc. Important place is taken by such branches as energy and resource saving. One can economize everywhere, during any action and as the result it will help to save our nature and the entire planet in the end. The simplest way of economizing is to implement some organizing or legislative measures (basically, they differ only that in the first case their compliance is only recommended, and in the second one it is mandatory).

Coming closer to the concrete topic of this research it is possible to state hypothetic rule: “if you have empty places in your private car you MUST take some passengers with the likely routes to fulfill them”. Of course such claim violates driver’s rights and is completely unacceptable in democratic societies (although, in Cuba, for example, drivers of ANY state cars with empty places MUST drive any passenger, who asked it by the raised hand). In any case, it is possible to ask private drivers to do so, or just to propose such possibility, especially if the trip would be payable.

Let’s note that totally unimportant for us does passenger pay money for the trip or not. We will take into account the problem from the technical point of view, but the motivation of the driver is not important, we only know that some person wants to find passengers for the specified route.

Organization of all the process is not simple and for its maximal efficiency needs for the most modern technologies of communication and information processing (information technologies, or IT briefly). So, actual problem, which is considered in this work, arises [1]. To solve it, firstly we will develop an appropriate algorithm and then implement it in source codes (of some selected programming language) giving finally working program which will allow to perform investigation (using modeling) of all the system processes [2]. If to complement such module with mobile application that in real-time will collect all the needed input information and then to spread system’s decisions among drivers and passengers the product will have real wide introduction (so is very perspective) [3].

Problem statement

Modern city is the place where millions of people live. Completely natural situation that some people work near their homes, but most must go through tens of kilometers to reach job’s place or by other needs. Due to obvious reasons bandwidth of existing roads cannot be increased much more times, especially at old-planned European cities. So population, and correspondingly people’s flow, increases, but total increasing of moving transport vehicles is impossible.

So it is expedient to search for other ways to increase people’s flows intensity. One of some appropriate possibilities is to use empty seats in the personal private cars going by their own route [4]. Taking into account great number of other people who need transport it is possible to find some passengers those have the same destination (or nearby) as the private car driver [5].

The motivation of car driver may be of different type [6]:

- just financial, cause passengers will pay for the trip;

- including social considerations, cause people with small children or old people (sick or disabled, etc) often forced to spent much time while wait for public transport (and then forced to ride in not so comfortable conditions, like in the private car);

- taking into account ecological problems of our globalized world, in which every empty seat in the moving car bring closer the global warming;

- complicated reasons, other.

Using this way the only task is to find passengers and appropriate drivers and to get contact with them. It is very complicated task if do not using of modern information technologies (IT further), but with them such problem can be solved [7]. It may be broken down into some smaller subtasks [8]:

- passengers who do not have managed cars must inform some entity (some information center) of their needs, i.e. note source and destination points and desired departure time;

- practically the same thing must do car driver: note source and destination points and approximate departure time;

- such actions do many passengers and drivers and then computer algorithm distributes passengers among empty places in the cars.

So, for now we may start development of corresponding algorithm.

Main part

Firstly to develop any algorithm that take into account human interactions the priorities of persons must be established. At this case two questions about involved persons priority arises:

- if some passenger's route is less greater than driver's initial one would he change it and deliver the passenger to his final point?

- under the same conditions would passenger make last small section of route on foot?

To develop concrete algorithm we must answer both of these questions [9] and proposed variant is "no" (negative). So, all the routes are rigid and further we will call this thesis the "route rigidity rule" or "3R". It means that during trip planning all passenger routes must be included into driver ones or in other words appropriate driver's routes must be larger than passenger's. In future advanced versions of developed algorithm there may be taken into account readiness of driver and especially passengers to correct a little their route to satisfy each other, but for the moment, for the first simplest algorithm version we will accept situation described above (no changes to any routes: driver's or passenger's).

Next question is how to identify various routes and at a whole how to note any objects on the map? It is proposed to note anything using crossroads numbers. Firstly all the transport infrastructure of the city or maybe of its significant part is appeared as such mathematical object like graph – Fig.1. Then vertices of the graph are numbered using standard rule: starting from the upper left corner to the right and then going down. So all the intersections are numbered and therefore any route may be presented as consequence of such numbers. Here is another one else assumption of the algorithm that all the routes start from some intersection and finish at another one, so there are no start and finish points between the crossroads, but only exactly at them.

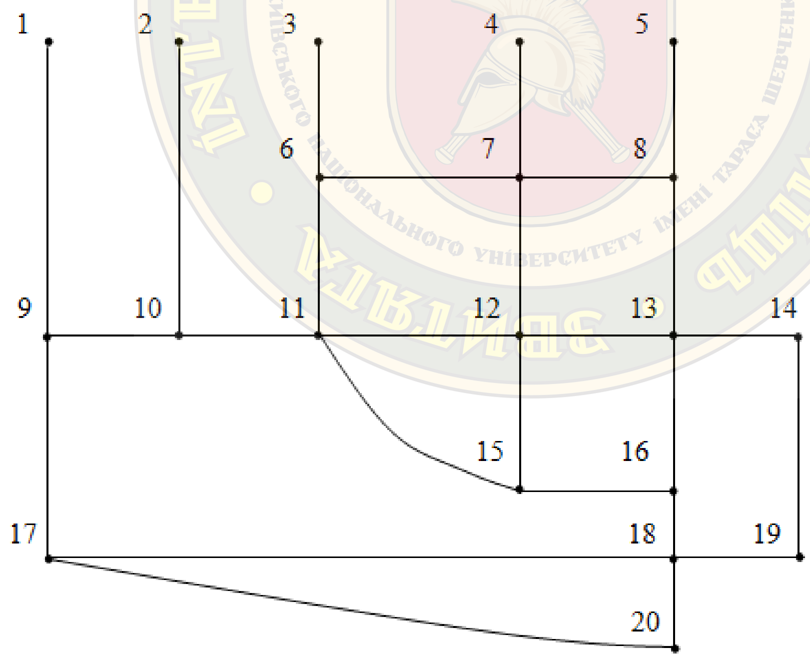

Figure1 - Part of some city transport structure presented as mathematical graph (intersections are numbered)

Logically the sequence of numbers may be presented by simple string, for example some driver's route (numbered by the system for example as i) may have next representation (using crossroads numbers):

$$d_i = \{3\text{-}6\text{-}11\text{-}10\text{-}9\text{-}17\text{-}20\} \quad (1)$$

If some passenger (numbered by the system as j) has the route like

$$p_j = \{6\text{-}11\text{-}10\text{-}9\text{-}17\} \quad (2)$$

then it is possible to say that

$$p_j \subseteq d_i \quad (3)$$

(where $\subseteq$ - is inclusion-or-equality sign) which corresponds to earlier formulated 3R-rule about rigidity of driver and passenger routes and as the result (3) may be named the mathematical form of 3R.

Later we will need the function $I(d, p)$ of two arguments (which are some driver route d and passenger route p) and which is equal to the number of common quarters presented in two these routes.

For example, if to take two routes (1) and (2) then this function would be equal:

$$I(d_i, p_j) = 4.$$

If two routes have no common quarters then:

$$I(d_i, p_j) = 0.$$

Further, if, as it was already told, to represent each route p_j (or d_i) we use simple string STRp_j (or STRd_i correspondingly) then relation between routes (3) causes the same relation (4) between strings:

$$\text{STR}p_j \subseteq \text{STR}d_i \quad (4)$$

Fact (4) means that we have the occurrence of one text string into another and it is simply verifiable with using of strings processing tools of modern programming languages like Java, PHP or C#, so (4) is convenient enough for our general task.

Another one technical (non-principled) question is how the information about planned drivers' routes and desired passenger ones will be delivered to designed information system. The most adequate way is to use specially designed mobile application for Android and/or iPhone with which people could apply for desired passenger routes and from the other side to process passenger requests by drivers offering planned car trips by the city. Also people who cannot or do not want to use mobile application may use personal computer (PC) software for the same purposes (but of course it is much less comfortable than using of mobile versions). So, the source of information for the system it is not the deal of this work, and we will imply that all necessary data already are in the system (and regularly updated).

So in a busy time of a day there will be much information about desired routes of the passengers (hundreds, thousands or maybe tens of thousands of requests) and comparable amount of route propositions from semi-empty cars' drivers. The task of developed algorithm is to distribute most of passengers by cars and to do it maximally effective [10]. So we have the optimization problem which generally may be solved with using different mathematical and/or programming methods and techniques.

But the first step of any optimization is to choose objective function of the system – this is the computable value which we must maximize (if we have maximization problem) or minimize (if this is minimization one). As it is well-known, minimization task elementary can be transformed to maximization one using next mathematical considerations:

$$x \text{ - ? : } f(x) = \min \Leftrightarrow x \text{ - ? : } -f(x) = \max$$

So while it is very simple to convert one type task into another, further we will speak only about one concrete task – the maximization problem. And returning to the question about objective function we may say that it is needed some positive value which may characterize the effectiveness of proposed distribution of passengers by cars: as greater is it, as more effective is the solution.

In this work we propose to use such indicator of effectiveness as total quantity of man-quarters which concrete distribution gives. Of course each quarter in any city has its own length, but if the trip consists of tens of quarters there will be no big error to use mean quarter length for estimation of all the trip distances [11]. That is why such characteristic as quarters quantity is more or less accurate and transparent (means understandable). Let's denote the number of quarters in route p_j (or d_i) by $q(p_j)$ (or $q(d_i)$). So optimization task can be formulated as:

$$E = \sum_{j'} q(p_{j'}) \to \max , \qquad (5)$$

where the sum is calculated by all the passengers which were distributed by cars and got their places.

Using functions $q(p_j)$ and $I(d, p)$ it is possible to express formula (3) as follows:

$$I(d_i, p_j) = q(p_j), \qquad (6)$$

or in other words: number of all quarters in some p_j is equal to the number of common quarters in p_j and some d_i – this is equivalent to thesis (3) that d_i includes (or equal to) p_j.

Relation (6) is the another form of 3R-rule, because all the quarters of passenger route p_j are common with driver route d_i, so p_j is included to d_i – (3).

Equality of two routes may be expressed as:

$$I(d_i, p_j) = q(p_j) = q(d_j). \qquad (7)$$

Relation (7) is the condition of optimal distribution of passenger route p_j onto driver route d_j (complete match).

Natural situation in any transport system is that some passengers stayed without service and in our case they may use public transport, or go by feet (maybe change destination point or own plans and so on); it means that initial j is not equal to final j', but they are close to each other (it realizes if the system is really efficient, and as closer as higher is its efficiency; ideally all the passengers are transported and $j = j'$) [12].

So let's take into account next small example. Let we have in the system 3 passenger routes and 2 planned driver's routes with 1 free place each (we imagine the simplest case for the initial example):

d_1 ={3-6-11-10-9-17-20}
d_2 ={19-18-17-9-10-2}
p_1 = {6-11-10-9-17}
p_2 = {18-17-9-10-2}
p_3 = {9-17-20-21}

It is possible to carry out two next variants of the passengers distribution among cars:

1) $p_1 - d_1$, $p_2 - d_2$
2) $p_3 - d_1$, $p_2 - d_2$

The efficiency (total number of quarters, which were passed by all the satisfied passengers) in the first case is:

$$E_1 = \sum_{j'} q(p_{j'}) = q(p_1) + q(p_2) = 4 + 4 = 8 .$$

In the second one:

$$E_2 = \sum_{j'} q(p_{j'}) = q(p_2) + q(p_3) = 4 + 3 = 7.$$

So first proposed variant is (8 – 7)/8 = 12,5 % more effective than the second one.

If the number of passengers and the number of drivers is not so big, making of the distribution is simple and all the variants may be composed by hand and calculated on the sheet of paper. Another situation is if we have at least some dozens of routes: to find the best distribution manually becomes impossible. In this case only automatic calculations can solve the problem.

So it is very actual task to develop algorithm and finally correctly working software to solve the problem of passenger distribution among cars with empty places.

Mathematical formulation of this problem may be carried out using the theory of sets formalism. Let D is the set of all driver's planned routes which are registered at the system for the moment :

$$D = \{d_1, d_2, \ldots d_M\},$$

where M – is total quantity of drivers' routes.

Analogically let P is the set of all passenger's desired routes, which are presented at the system:

$$P = \{p_1, p_2, \dots p_N\},$$

where N – is total quantity of passengers' routes.

The task is to choose maximum number of elements p_j from P, each of which correspondingly to 3R-rule must be included as the part in some element d_i of D (and in the first approximation of our algorithm one d_i can contain only one p_j regardless of the ratio $q(d_i)/q(p_j)$ that shows some "local effectiveness" of concrete distribution of passenger route p_j to driver route d_i), and to achieve maximum value of objective function (effectiveness indicator) E.

As it well known optimization task with high number of degrees of freedom is very complex and it's far from always possible to get the best, optimal solution, so in many cases when solving optimization tasks then are satisfying with just good enough solution (which is not the best but is better than overwhelming majority of other answers).

So to get such "good" solution we may use brute force process, which can be carried out into two ways:

- taking concrete passenger and searching for an appropriate driver's route;
- to the contrary: taking concrete driver's route and searching for appropriate passengers.

Last way to our mind is some times better because if we have M driver's routes than they correspond to $M \cdot n_m$ passengers (which is ideally must be equal to the number N of passengers registered in the system), where n_m – mean number of empty seats in one registered car (we think that this value is close to 3, so in the following calculations $n_m = 3$).

So if we search drivers for each passenger it will take near N iterations and in another case it will be M iterations. Taking into account $N \sim M \cdot n_m$, it becomes obvious that search the passengers for concrete car is more efficient (but, of course, not exactly n_m times efficiently, because the procedure itself for 1 empty seat search is simpler than search for n_m passengers filling the car, but in general we think that less number of search iterations in outer cycle will give some benefits and this case would be better).

Using previously stated information it is possible to develop algorithm of passengers distribution among cars with empty places – Fig. 2.

Moving on to the specifics of constructing a software product that performs the actions considered, we can say that firstly we need some controls to input data about drivers' routes and passengers' routes. Taking into account this would be textual information, let it be two textareas (Fig.3): left for drivers, right – for passengers. At the left textarea except the route itself also is presented the number n_m of empty places in the car (separated by the semicolon). When all the input information is entered into these textareas, some control must be devoted to starting calculation process – let it be button "Go!" at the center of software window. Finally when all the output information is ready, it must be displayed for the user, using some controls. Let it be also two textareas in which routes are displayed structurally (not chaotically): at the left output textarea is displayed driver's route and to the right at the same level are displayed passengers' routes distributed to this driver's route.

Also it will be useful to get some statistical information about the whole process. It may be the value of efficiency indicator E (5), which is equal to the total number of quarters, passed by all the distributed passengers. Also it may be displayed the number of distributed passengers and its ratio to the total number of requests (so, the percent of satisfied passengers).

Using the total number of passed quarters Q it is possible to do some estimation of fuel saved with developed software. Firstly, let one quarter length be 150 m = 0.15 km, then let one public vehicle can carry 90 passengers, and finally let the fuel consumption of such vehicle is 30 l per 100 km. So, the volume of saved fuel is about:

$$Q \cdot 0.15/90/100 \cdot 30 = 0.0005Q \tag{8}$$

So it is simple to estimate the usefulness (in liters of fuel saved) of the entire system taking total number of passed quarters and multiplying it by 0.0005.

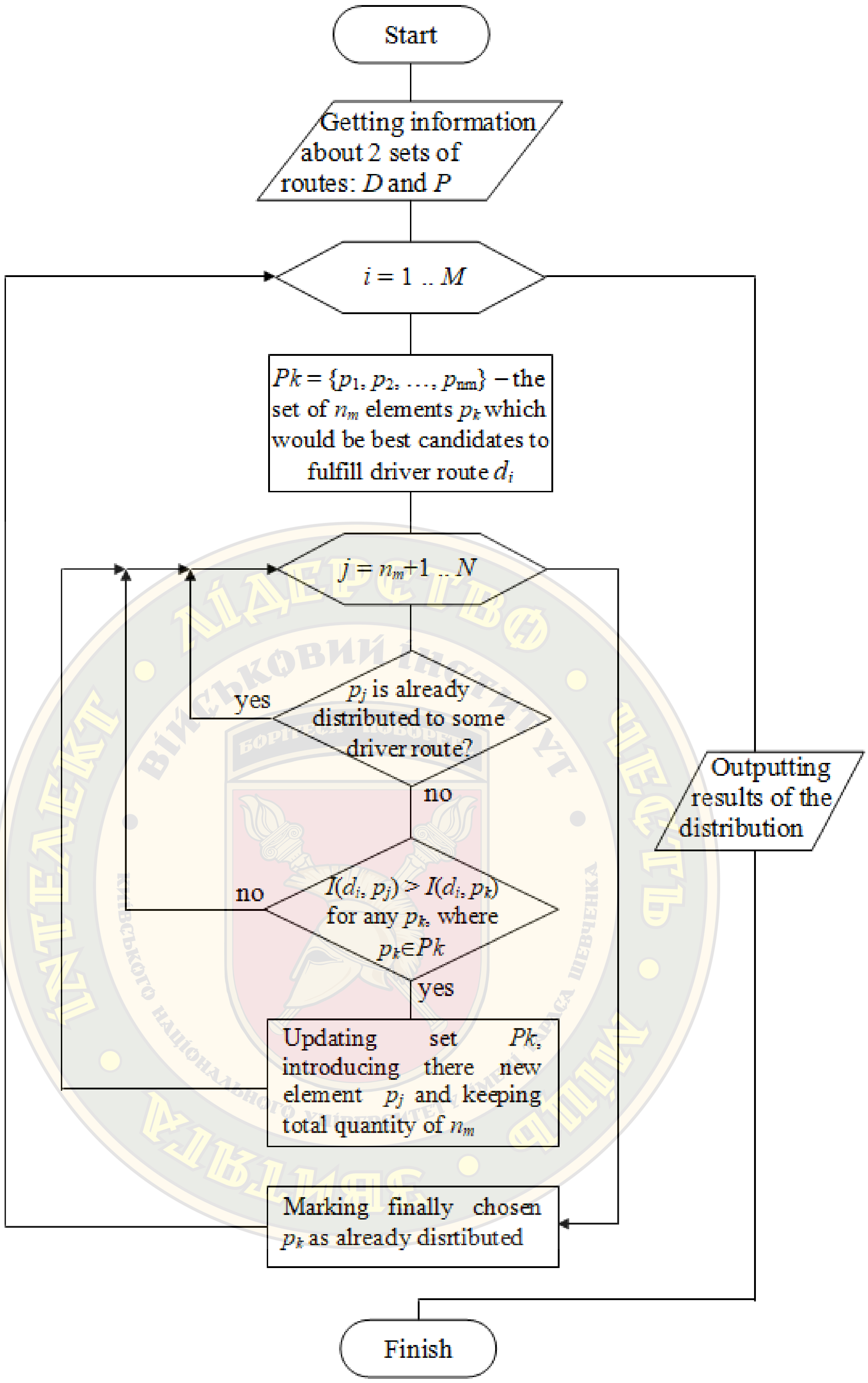

Figure 2 - Scheme of the algorithm to distribute passengers by cars

Next to the user interface design it is possible to take into account the structure of source codes. As object-oriented technology was accepted all the code is located inside the classes. For the complicated software development it may be useful to use classes for the routes:

- abstract class TRoute that is the ancestor of other classes;
- derived class TDRoute which describes driver's route;
- derived class TPRoute which describes passenger's route.

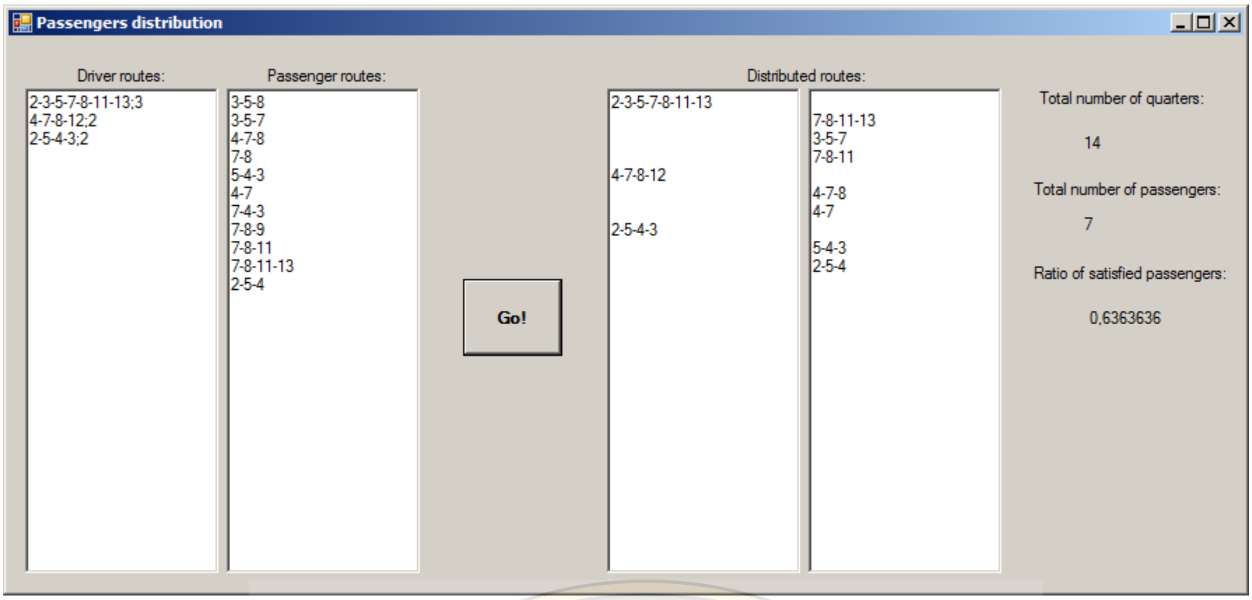

Figure 3 - User interface of developed program and example of its using

Testing was carried out for various transport system graphs and it shows that program works fast with hundreds of routes giving the solution instantly. Also number of passengers distributed by cars always equal to the n_m of the corresponding car. All drivers' and passengers' routes are used only once. If there are many appropriate passenger routes for the current driver route the best undistributed routes are chosen. Everything shows that software is adequate and stable and can be used for further researches.

Conclusion

In this work there is taken into account very important problem of transport flows intensification by using private cars of ordinary people (not taxi drivers) which has empty places and going about their own routes. Here was developed simple mathematical model of the process, which is based on the theory of sets. Basic element of any route is the quarter which is situated between two intersections; number of quarters is placed to the head of the corner (in the work widely used the function $q(p)$, which gives the number of quarters in route p) and the number of all the quarters passed by all the satisfied passengers is taken like objective function E of the system (or in other words its efficiency indicator). During passenger routes distribution among driver distributions it is used 3R-rule ("route rigidity rule", which means that driver will not change his route to satisfy any passengers, and the passenger will not go any part of his route by feet; as the result all passenger routes must be included into driver ones). Also it was proposed the function $I(d, p)$ of two arguments (which are some driver route d and passenger route p) and which is equal to the number of common quarters presented in two these routes.

Based on mentioned mathematical grounding it was developed the algorithm of passengers distribution among drivers' ones. Due to developed algorithm scheme it was carried out the implementation of it in software with convenient windowed user interface, which may use not only professionals of IT branch, but any PC user with medium skills.

To give developed software real application there must be improved the mechanism of getting input information for the system work (for now all the input data is typed on the keyboard) and delivering output solution to all the interested parties. It may be realized using an appropriate mobile application (for Android and/or iPhone), development of which is perspective task for future work. For now developed system may be used for modeling of passenger distributions among cars with empty places for scientific-research purposes.

REFERENCES:

1. Leandro do C. Martins, Rocio de la Torre, Canan G. Corlu, Angel A. Juan, Mohamed A. Masmoudi (2021). Optimizing ride-sharing operations in smart sustainable cities: Challenges and the need for agile algorithms. Computers & Industrial Engineering, V.153, 107080.
2. Aydin, O. F., Gokasar, I., & Kalan, O. (2020). Matching algorithm for improving ride-sharing by incorporating route splits and social factors. PloS one, 15(3), e0229674. https://doi.org/10.1371/journal.pone.0229674
3. Ma, S., Zheng, Y., & Wolfson, O. (2013). T-Share: A Large-Scale Dynamic Taxi Ridesharing Service. IEEE 29th International Conference on Data Engineering (ICDE), Brisbane, 410–421.
4. Zhou, Lunwei and Kang, Liujiang and Sun, Huijun and Bao, Yue and Lai, Qingying and Xu, Qianwen and Mashhoodi, Bardia and Ge, Ying-En. (2025) Real-Time Simultaneous Ride-Sharing Matching and Route Planning Based on Adjustable Passenger Pick-Up and Drop-Off Points. Available at SSRN: https://ssrn.com/abstract=5225975 or http://dx.doi.org/10.2139/ssrn.5225975 .
5. Danassis, P., Sakota, M., Filos-Ratsikas, A. et al (2022). Putting ridesharing to the test: efficient and scalable solutions and the power of dynamic vehicle relocation. Artif Intell Rev 55, 5781–5844. https://doi.org/10.1007/s10462-022-10145-0.
6. Cao, Y., Wang, S., & Li, J. (2021). The Optimization Model of Ride-Sharing Route for Ride Hailing Considering Both System Optimization and User Fairness. Sustainability, 13(10), 5610.
7. Connor Riley, Pascal Van Hentenryck, Enpeng Yuan (2020). Real-Time Dispatching of Large-Scale Ride-Sharing Systems: Integrating Optimization, Machine Learning, and Model Predictive Control. INFORMS Journal on Computing, 32(3), 530–547.
8. Md Tawhidur Rahman, Kakan Dey, David R. Martinelli, Sabya Mishra (2021). Modeling and evaluation of a ridesharing matching system from multi-stakeholders’ perspective. IET Intelligent Transport Systems, 15(4), 369–377.
9. Shimamoto, H. (2025). A Matching Model for Ride-Sharing: A Non-Cooperative Game Approach Between Drivers and Riders. *Smart Cities*, *8*(2), 40. https://doi.org/10.3390/smartcities8020040.
10. Hasan, Mohd. (2021). Ride-Sharing Optimization Algorithms for Urban Commuting. Transportation Research Part B: Methodological, 121, 145–161.
11. Guan, L., Pei, J., Liu, X., Zhou, Z., & Pardalos, P. M. (2020). Ridesharing in urban areas: multi-objective optimisation approach for ride-matching and routeing with commuters’ dynamic mode choice. International Journal of Production Research, 60(5), 1439–1457. https://doi.org/10.1080/00207543.2020.1859635.
12. Dastani,Z., Koosha, H., Karimi, H. et al (2024). User preferences in ride-sharing mathematical models for enhanced matching. Sci Rep 14, 27338. https://doi.org/10.1038/s41598-024-78469-1.

к.т.н. Гайша О.О., д.т.н., проф. Лєнков С.В., Гайша О.О.

АЛГОРИТМ ТА ПРОГРАМНИЙ ЗАСІБ ДЛЯ ВИРІШЕННЯ ПРОБЛЕМИ МІСЦЕВИХ ПАСАЖИРСЬКИХ ПЕРЕВЕЗЕНЬ

У контексті сучасної урбанізації та глобальних екологічних проблем, оптимізація систем пасажирських перевезень стала важливою галуззю досліджень, особливо в галузі енергозбереження та зменшення впливу на навколишнє середовище. Це дослідження розглядає конкретний аспект цієї широкої проблеми: ефективне використання вільних місць у приватних транспортних засобах для збільшення пропускної здатності транспорту без розширення дорожньої інфраструктури. Центральна ідея полягає в перерозподілі пасажирів, які користуються схожими або перекриваючимися маршрутами з власниками приватних автомобілів, використовуючи сучасні інформаційні технології для полегшення зіставлення та комунікації в режимі реального часу.

Робота починається з викладу концептуальної основи проблеми, підкреслюючи, що численні міські жителі щодня долають значні відстані, тоді як доступна транспортна інфраструктура залишається обмеженою. З огляду на це, значного підвищення ефективності транспорту можна досягти, використовуючи вільні місця в приватних транспортних засобах. Хоча соціально-економічні та мотиваційні фактори, такі як екологічна обізнаність, фінансові

стимули або альтруїстична поведінка, можуть стимулювати такі ініціативи, стаття зосереджується виключно на технічних та алгоритмічних аспектах рішення, навмисно виключаючи суб'єктивний вимір мотивації водіїв.

Побудовано формальну постановку проблеми, яка моделює мережу міських доріг як математичний граф, де перехрестя представлені як вузли, а сегменти доріг (квартали) як ребра. Маршрут кожного водія та пасажира визначається як послідовність перехресть, а основною умовою для призначення пасажира водієві є повне включення маршруту пасажира до маршруту водія. Це обмеження, яке називається «Правилом жорсткості маршруту» (3R), гарантує, що ні водій не змінює свій маршрут, ні пасажир не відхиляється від запланованого шляху.

Архітектура алгоритму зосереджена на методах обробки рядків, де послідовності маршрутів кодуються як рядки, а включення перевіряється за допомогою операцій з підрядками. Ключова функція I(d, p) вводиться для кількісної оцінки кількості перекриваючих кварталів між маршрутом водія d та маршрутом пасажира p. Ця функція служить основою для перевірки сумісності маршрутів та, зрештою, розрахунку ефективності певного розподілу.

Ефективність системи визначається цільовою функцією E, яка представляє загальну кількість обслуговуваних пасажирських кварталів. Метою оптимізації є максимізація E шляхом ефективного призначення запитів пасажирів відповідним маршрутам водіїв. Алгоритм ітеративно розглядає маршрути водіїв та вибирає найкращі пасажирські маршрути на основі максимального включення, гарантуючи, що кожен маршрут використовується лише один раз. Емпіричний приклад ілюструє, як різні стратегії призначення забезпечують різний рівень ефективності, що вимірюється загальною кількістю виконаних кварталів.

Для впровадження цієї моделі було розроблено програмне рішення з інтерфейсом користувача для введення та обробки даних про маршрут. Крім того, запропоновано методологію оцінки екологічних переваг за рахунок економії палива, перетворюючи результати системи на кількісно вимірний екологічний вплив. Експериментальні результати підтверджують стабільність та ефективність алгоритму в обробці тисяч маршрутів, при цьому програмне забезпечення послідовно розподіляє пасажирів відповідно до визначених обмежень та максимізує показник ефективності.

Ключові слова: оптимізація міського транспорту, алгоритм спільного використання поїздок, зіставлення маршрутів, інтелектуальні транспортні системи.