ISSN 2712-0554
Original Research

Heritage and Sustainable Development
Vol. 4, No. 2, December 2022, pp.185-190
https://doi.org/10.37868/hsd.v4i2.149

Simulation model of traffic flow based on agent-based modeling

Mikhail Gorodnichev*
Moscow Technical University of Communication and Informatics, Russian Federation

*Corresponding author E-mail: gorodnjchev@yande.ru

Received Aug. 24, 2022
Revised Dec. 12, 2022
Accepted Dec. 30, 2022

Abstract
The work is devoted to traffic flow simulation using agent-based modeling using distributed computing. The role of modeling traffic flows in the modern world is considered. The types of traffic flow modeling are considered. The developed software makes it possible to simulate the movement and interaction of vehicles.

Keywords: Traffic flow modeling, Simulation modeling, Agent-based modeling, Software developing, Client-server architecture

1. Introduction

In the modern world, where every year there is a constant growth of transport communication, the problem of traffic management is acute, in large cities. The increase in transport, both personal and public, has led to road congestion, hours of traffic jams, an increase in the number of accidents, etc. This all presents a complex dynamic problem. It includes many complex processes, elements that interact with each other, such as cars, driver behavior, road geometry, parking spaces, road signs, pedestrian zones.

Traffic flow is a physical phenomenon that is extremely difficult to model mathematically. The simplest traffic scenario is a one-dimensional one-way pipeline road. But in reality, such a variant of movement is impossible, since in modern realities the roads have a complex system with many lanes and interchanges. Design improvements to the traffic management system are extremely important in our time. In such a difficult situation, computer modeling is an effective solution to the problem, since it provides estimates for various motion conditions. It can help understand and analyze traffic, assess current problems, and propose effective solutions. Traffic simulation can support transport planning and traffic management decision making for long-term sustainable urban development. All this creates the need for the research presented in this paper.

Modeling traffic flows is aimed at solving problems such as:

- increasing the maximum traffic capacity of the road network,
- increasing the efficiency of using the capacity of the road network,
- regulation of the volume and structure of transport demand.

The road system and transit systems are presented as networks for computer analysis. Networks are made up of links representing segments of highways or public transport lines, and nodes representing intersections and other points in the network. Channel data includes channel travel time, average speed, throughput, and heading. Node data is more limited to information about which links connect to the node and the node's location (coordinates). The node data can also include intersection data to help calculate the delay that occurs at intersections. The travel modeling process follows trips as they begin in the trip generation zone, traverse the network of connections and nodes, and end in the trip attraction zone. The modeling process is known as a four-step process and includes [1] trip generation, trip allocation, mode separation, and traffic allocation.

2. Research method

To develop a traffic flow model based on agent-based modeling using distributed computing, it is necessary to draw up a general methodology for constructing and working with traffic flows, then develop recommendations for its construction [2]. The general methodology for building and working with transport models, the methodology includes the following steps:

- preliminary analysis and selection of simulation software
- collection and preparation of initial data for building a model,
- input of the received data into the model,
- testing the model (checking the correctness of the entered data),
- performing experiments and analysing the results,
- formation of reporting materials.

The presented model is more relevant for simulation models of any type. The use of analytical models assumes direct calculations for one or several previously known dependencies. Particular attention is paid to the collection of initial data. The purpose of the data collection phase is to prepare a complete dataset of the raw data needed to build the model. The input data for this stage are:

- choice of topology,
- the required types of input data,
- selected technologies.

3. Results and discussion

As a result, a complete set of initial data should be obtained, collected in compliance with all the requirements for the specified accuracy. A cellular automaton model is chosen. This is one of the microscopic traffic patterns. In this model, the road is composed of cells. Particles move from one cell to another. The first cellular automaton models for modeling traffic flows were presented by Nagel and Schreckenberg (1992). In the cellular automaton model, the street is divided into cells in a typical space, which is the space occupied by vehicles in dense traffic. The space depends on the length of the vehicle and the distance to the previous vehicle [2], [3]. Each cell can be occupied by no more than one vehicle or empty Figure 1.

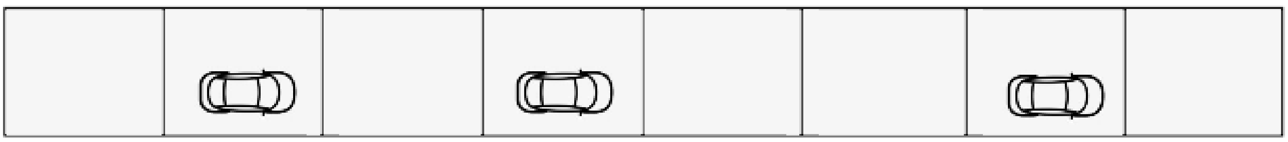

Figure 1. Cellular automata model

The simplest model of cellular traffic automata was developed by Wolfram (1986, 1994) and Bihametal (1992). This model is described as an asymmetric simple elimination model on a one-dimensional road surface. The formula shown in equation (1):

$$x_i(t+1) = x_i(t) + \min\,(1, x_{i+1}(t) - x_i(t) - 1) \tag{1}$$

In this model, vehicles are moved to the front cell if the cell is free. Then, 1 is added to the speed of all vehicles at the same time. The average speed of stochastic movement along a stationary strip with the number of cells equal to n and the number of particles equal to m, and the probability of implementation of the attempt p [2] as shown in equation (2),

$$v(n,m) = \frac{n}{m}\sum_{k=1}^{\min(m,n-m)} \frac{Cp}{(1-p)^{k-1}} C_{m-1}^{k-1} C_{n-m-1}^{k-1}, \quad (2)$$

where,

$$\mathrm{C} = (\sum_{k=1}^{\min(m,n-m)} \frac{n}{k} * C_{m-1}^{k-1} C_{n-m-1}^{k-1} \frac{1}{(1-p)^{k-1}})^{-1} \quad (3)$$

The cellular automata model can be represented as a grid or a circle. Any transport network can be represented as any of the topologies. Let's consider each of the topologies in more detail. We will consider a model of particle movement (grid) along a fragment of a road with three lanes as shown in Figure 2.

Figure 2. Cellular field in three stripes

The multi-band system is similar to the single-band model. We introduce a parameter that describes the likelihood that the car will change lanes, if allowed. A vehicle is allowed to change lanes if at the moment it is on the sections along which it will move freely [3]. Movement options are shown in Figure 3.

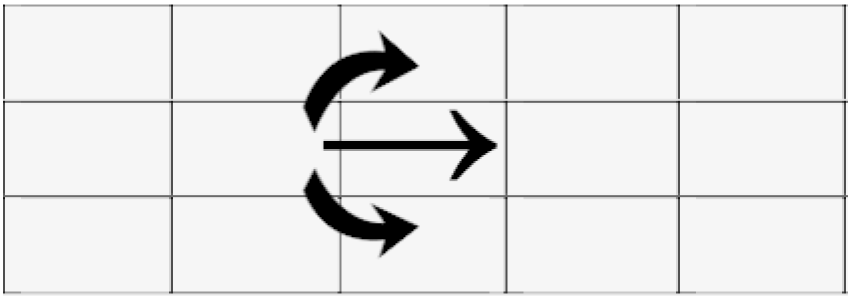

Figure 3. Permissible displacements in case of free movement

For each strip, we introduce the value of the regularity of the flow r, which is the ratio of the number of occupied cells to their total number, where r [0; 1]. If r = 1, we will assume that this is a steady flow, where vehicles move at a speed v = const. If r <1, then individual movements of mc from one cell to another, located diagonally with a change of lane or in front, are possible, provided that the cells are free. Thus, if the cell in front of the particle is free, then the particle goes to this cell. If the cell in front is occupied and the cell on the other lane is free, the particle moves to the adjacent cell. If both neighboring cells are free, then the particle goes to one of them with probability ½. We will consider a circular model as a closed sequence of cells. Each cell at any time can contain only one particle, or the cell must be free. Suppose n is the number of cells, m is the number of particles. At each discrete moment, the particles occupy one of the cells. If we denote a particle by i, then this particle in the direction of motion will be followed by a particle i + 1. Consider a sequence of N = 4n cells [4]. Here each cell can be either empty or contain a particle. For the sequence, the direction of movement is determined, since for each cell the next (k + 1) and the previous (k -1) are set. Here the numbering of cells is 1, n + 1, 2n +1, 3n + 1.

At the initial moment of time, there are m particles on the contour. If at the next moment in time there is a free cell in front of the particle, then it passes into the cell with probability p. The average flow rate on a closed circuit is defined as the ratio of the number of particle displacements over a certain time T to the time interval for one particle:

$$v_{lk} = \frac{s_{lk}}{mT} \quad (4)$$

Where s_{lk} - the number of cells passed during the time
T m - contour particles.

Consider a rectangular mesh made up of 4x4 contours [5]. The contours of such a network are in the same conditions, they are geometrically and topologically identical objects. Each of the contours has N neighbors, with which it is connected by one common cell, through which particles can pass from their contour to the neighboring one. A model consisting of four or more contours is called a flat grid. It has the form of an array (LxK), where each contour is assigned coordinates (i, j), which show its position in the network.

The average flow rate on such a network is calculated as:

$$V = \frac{1}{KL} \sum_{i=1}^{L} \sum_{j=1}^{K} v_{lk} \quad (5)$$

In the model, at the initial moment of time (t_0), each contour contains m number of particles. In this case, the common cell for neighboring contours is either free or occupied by one particle of any of the neighboring contours. Each of the particles at the next moment of time (t_0 + 1) moves to the next cell in the direction of movement, if the cell is free. If at the moment of time 2 the particles claim to be a common node of two adjacent contours, then with probability ½ one of the particles goes to the node, and the second remains in place. Any transport network can be represented as any of the topologies. When choosing a topology, it is worth considering the features of the network for more accurate modeling.

The agent management task is divided into several subtasks, which are automated by independent rules of behavior. Thus, the functionality of the agent can be easily extended without any changes to the existing behavior. The rules of conduct used are highly dependent on the agent environment. The program is a module that implements the movement and interaction of vehicles. The application structure consists of 3 parts - server, web server and client. Let's take a closer look at each component next. For the program to work correctly, a connection must first be established. This is done using socket technology.

In the part of the main function, the initial configuration of the server is performed. First, the IP address and port are set and an instance of the socket class is created, indicating the TCP protocol. The socket listens for incoming connections and waits for a web server connection. When the web server connects to the server, a separate thread is allocated for working with the web server: receiving commands and sending a response to them. The Work Helper class implements the work of the main elements of the system: mesh, cars, intersections and roundabouts. Each of them is executed in a separate thread. For each function, a separate thread is created in which it is executed, thereby ensuring the parallel operation of these functions. The car has 2 states: Direction Motion, Direction Motion Next. Direction Motion shows the direction of the car at the current moment, and Direction Motion Next shows the direction that will be after crossing the intersection. Direction Motion Next is needed to understand whether the car should change lanes or if it can stay in its own lane.

Since the project implements 2 topologies, the movement of cars is also different. Let's take a look at each topology separately. On the stack, the movement occurs as follows. Calling the Make Step function increments the delay counter. The delay is needed to simulate the difference in vehicle speeds. At the initial moment of time, each car is randomly assigned the current value of the delay step. The maximum delay is also set. When the maximum delay is reached, the delay counter is updated and

The vehicle starts moving. Depending on the size of the delay step, the speed of resetting the delay counter is different for each car, respectively, the speed of movement of each car will also be different. After resetting the delay counter, a collision check and traffic light color is performed. If the checks are passed successfully, then a check is made whether the car has reached the intersection, if it has reached, then the direction is changed. Otherwise, it generates movement. Similar to the grid, a delay is implemented on the circles to simulate the difference in vehicle speeds. It also checks for collisions.

The following formulas are used to calculate the coordinates of a vehicle at a roundabout:

$$x_{car} = x_{crossroad} + r * \cos(\alpha) \quad (6)$$

$$y_{car} = y_{crossroad} + r * \sin(\alpha) \quad (7)$$

Where $x_{crossroad}$ and $y_{crossroad}$ intersection center coordinates, r - radius, which is equal to the distance from the center of the intersection to the ring, α is the angle between the starting point of the intersection and the current position of the vehicle.

There are 2 lanes at the roundabout: an outer and an inner lane. Each strip has its own radius. The outer strip has a radius of 40, the inner one has a radius of 30. A socket is created on the web server that establishes a connection to the server. When you click the "connect" button, a connection to the server is established, information is collected from the site and sent to the server to start the simulation. When the Stop button is pressed, a request is sent to the server to stop the simulation.

Let's consider how the movement takes place on a 3x3 grid, with the number of cars equal to 20 and traffic lights. The simulation result is shown in Figure 4.

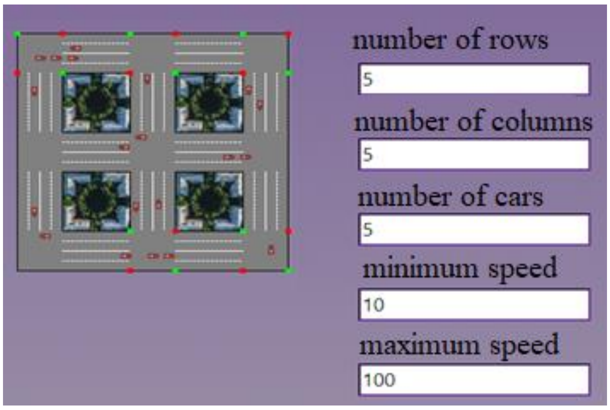

Figure 4. Simulation result

4. Conclusion

Transport modeling plays an important role in enabling spatial development. Models developed by transport modelers help predict, secure, and deploy transport infrastructure in communities. Modeling traffic flows is important as it helps in designing comfortable and safe roads, solving congestion problems, and developing adequate traffic management measures. In the process of research and modeling, the following results and conclusions were obtained:

- A model of traffic flows has been developed and investigated, which combines the property of classical approaches to describing traffic and modern agent-based models.
- A computer algorithm of a microscopic model has been implemented, which implements the interaction of cars on the road system. Based on the developed algorithm with the help of object-oriented programming tools (multithreaded data processing), a transport stream software for networks of various topologies was created.

Declaration of competing interest

The authors declare that they have no known financial or non-financial competing interests in any material discussed in this paper.

Funding information

No funding was received from any financial organization to conduct this research

References

[1] A. Buslaev, A. Tatashev, and M. Yashina, "Flow spectrum of closed/open contours chains," *Comput. Math. Methods*, vol. 3, no. 3, p. e1087, 2021.

[2] J. Hackl and B. T. Adey, "Estimation of traffic flow changes using networks in networks approaches," *Appl. Netw. Sci.*, vol. 4, no. 1, p. 28, 2019.

[3] A. P. Buslaev, A. G. Tatashev, M. J. Fomina, and M. V Yashina, "On spectra of Wolfram cellular automata in Hamming spaces," in *Proceedings of the 6th International Conference on Control, Mechatronics and Automation*, 2018, pp. 123–127.

[4] A. P. Buslaev, M. J. Fomina, A. G. Tatashev, and M. V Yashina, "On discrete flow networks model spectra: statements, simulation, hypotheses," in *Journal of Physics: Conference Series*, 2018, vol. 1053, no. 1, p. 12034.

[5] A. P. Buslaev and A. G. Tatashev, "Exact results for discrete dynamical systems on a pair of contours," *Math. Methods Appl. Sci.*, vol. 41, no. 17, pp. 7283–7294, 2018.