Simulation modeling of robot interaction algorithms

Doryshin Mikhail Olegovich
Department of Economics, Management and Business Informatics
National Research University "Higher School of Economics"
Perm, Russia
modoryshin@edu.hse.ru

Scientific Advisor:

Zamyatina Elena Borisovna
Department of Economics, Management and Business Informatics
National Research University "Higher School of Economics"
Perm, Russia
e_zamyatina@mail.ru

Abstract—**The paper presents a study of robot interaction algorithms. The study of algorithms that control the behavior of robots is relevant due to the development of digital clones of various autonomous machines and their wide application in various fields: in manufacturing, medicine, military. The paper considers the application of nanorobot swarms in medicine. The study is performed using simulation methods. The simulation system NetLogo is used as a toolkit.**

Keywords—Robot, nanorobot. robot interaction algorithm, simulation modeling, agent-based modeling, digital clones.

I. INTRODUCTION

Recently, there has been a widespread use of robots in various fields in production [1,2,3,4], in construction [5], in medicine [6], in military affairs, regular household [10] etc.

The main advantages of automation using robots are:

1. Replacing a person in the performance of tasks that may negatively affect his health;

2. Improving the quality of work and productivity of enterprises;

3. Reduction of staff remuneration costs;

4. Elimination of the human factor from a number of business processes.

In conditions when the development and testing of robots in real conditions require too high financial and time costs, they often resort to the use of digital twins. A digital twin is a digital (virtual) model of any objects, systems, processes or people. It faithfully reproduces the shape and action of the original and is in sync with it.

When it comes to developing digital twinst in certain scenarios, one has to use several or a whole complex (“swarm”) of interacting robots. Robots act according to certain algorithms.

In this case, there is a need to apply certain methods to make sure that the interaction of robots is correct, which together try to solve the problem , to evaluate the effectiveness of robot interaction algorithms.

As such methods, it is advisable to use simulation methods that allow you to simulate the behavior in time of a complex simulated system, which is a swarm of robots (often fundamental mathematical methods cannot be applied in this case, since they require abandoning a number of details of the simulated complex system).

Each of the interacting robots can be represented by an agent acting according to a certain scenario in the agent- based simulation system (ASIM). We will use NetLogo [8] as a tool for ASIM . ASIM NetLogo is free software and has proven itself [9] when used in a number of projects. So, let's consider in more detail those tasks that are solved thanks to the use of robots.

II. EXISTING SOLUTIONS

Currently, robots are used in almost all areas of human activity. Robots are used to solve a huge number of tasks, from complex and dangerous, like transporting heavy loads, to trivial, like counting goods in a store. So the authors of the article "Robotics and automation of production: the state of the art" [1] consider the use of robots and other smart machines in industry to automate a huge range of processes from production to financial and economic. The use of robotics can not only significantly increase productivity in enterprises, but also ensure safety when performing dangerous tasks. Automation of production processes using robots makes it possible: (a) to replace a person in the performance of tasks that may adversely affect his health; (b) improve the quality of work and productivity of enterprises; (c) reduce staff costs; (d) eliminate the negative impact of the human factor in a number of business processes.

The authors of the article “Task selection by autonomous mobile robots in a warehouse using deep reinforcement learning » [4] describe the solution to the problem of increasing the efficiency of a team of autonomous loading machines by improving the construction of routes for each machine, as well as preventing collisions between them.

The algorithm is based on fairly simple rules for the interaction of robots with cargo objects, as well as interaction with each other. The algorithm is designed to train the robot control system in a warehouse. The essence of the algorithm is as follows: agent robots are located in the virtual warehouse, each of the robots must perform a specific task. Agent robots can be in one of two states: (a) the robot is active if it is performing a task at a particular moment in time; the robot is inactive (idle). Each inactive agent robot is assigned one of the pending tasks. This task can be completed in a specific location. As soon as the robot reaches the desired location, the task is marked as completed, and the robot again switches to an inactive state. To reach a specific location, robots need to avoid collisions with obstacles. The algorithm works as long as there are outstanding tasks. In the event that a collision occurs between two agent robots, the algorithm also stops working.

The authors of the article claim that they managed to develop a control software system for constructing routes for loading robots in a warehouse, which allows them to perform work efficiently.

The paper [5] discusses the results of studies on the use of robots for building the walls of brick buildings. Robots are used to automate the process of laying bricks. The robot is also either active or inactive. If the robot is active, then it performs the following sequence of operations: (a) moves to the brick storage; (b) picks up bricks; (c) moves to the location of the bricks; (d) laying bricks at the indicated position.

The conducted studies have shown that the simulation of the interaction of robots allows avoiding collisions due to the clear formation of schedules for the stages of work performed, blocking the workspace during the laying of bricks, setting the exact duration of the stages of work (determining the speed of movement of the robot, the exact time of capturing and installing bricks), taking into account external factors . Similar studies are considered in [6,7], etc.

Let's take a closer look at the use of robotics in medicine, build a simulation model of the algorithm and perform a series of experiments that will determine, in particular, the number of nanorobots needed to destroy cancer cells.

III. Algorithm of operation and interaction of nanorobots

"On the scalability of agent-based modeling for medical nanorobotics » [6] considers the use of nanorobots in the field of medicine, and more precisely, their application in the diagnosis and treatment of oncological diseases.

The task of nanorobots is to detect "bad" cells and neutralize them by introducing the appropriate drug. Each of the nanobots has two sensors with different ranges (aura-signal and aura-detector). The nanorobot moves randomly in space. If a “bad” cell falls within the radius of action, the nanorobot detects it, immediately comes into contact with it, injects a drug and sends a signal to other nanorobots. If the nanorobot receives a signal from another robot, then it starts moving towards the robot that sent the signal and also starts injecting the drug into the “bad” cell.

It should be noted that when moving in space, nanoworks may encounter obstacles on the way. They must be bypassed in order to continue the search for "bad" cells. Cells also move randomly.

The article also proposes the idea of forming an experiment roster aimed at conducting examinations of different behavioral patterns for the nanorobots, thus allowing investigation of several different robot interaction cases.

IV. NetLogo simulation system

NetLogo discrete simulation system was created in 2002, the latest version of NetLogo 6.2.0 was released in September 2019. NetLogo can run on all underlying platforms: IOS, Windows, Linux. NetLogo provides many specialized libraries with which the user can create new models . In addition, the system contains a large number of ready-made models open for modifications. Users can also add their own models to the SIM. It should be noted that the Netlogo programming language of the same name is used to describe the models .

NetLogo implements agent-based modeling . Agents perform their assigned functions and act in parallel in time. Agents can be divided into the following types: (a) turtles (represent an elementary object of any model, have properties such as number, color and coordinates, custom properties can also be added to turtles); (b) links (link) (describe links between two turtles, links can be directional or non-directional); (c) patches (patch) (represent elements of the modeling area that turtles can move on, patches have specific coordinates). The development of simulation models in the NetLogo environment involves the creation of procedures that describe the interaction between agents.

V. Building a Model and Description of Experiments

We will develop a simulation model to study the interaction of nanorobots designed to fight cancers.

The model will realize the interaction between multiple types of NetLogo agents. First we need to identify the main elements of the algorithm that should be present in the model. These elements are:

1. Nanorobot;
2. Cancer cell;
3. The area of movement of the nanorobot / cell;
4. Aura is the signal of the nanorobot;
5. Aura is a nanorobot detector;

Now that the main elements are designated, we can designate a NetLogo agent for each of them:

1. The nanorobot (turtle). The nanorobot is the main object in this model. He must constantly move around the virtual environment in search of cancer cells, so he must be presented as a turtle agent;
2. The cell (turtle). Like the nanorobot, the cell is one of the main objects in the model. She also has to move around the virtual environment, so the most suitable agent for her is turtle;
3. The area of movement (patch). In NetLogo models, all turtles move around patches. The movement area is an empty area in a virtual environment where cells and nanorobots can move around;
4. The connection between nanorobots, received from the aura signal (link). When the nanorobot begins to interact with the cell, it sends a signal for help to other robots. Those who receive the signal begin to move in the direction of the source to take part in the elimination of the cell. Thus, a

connection appears between two robots (source and receiver), which can be represented by the corresponding agent;

5. The connection between nanorobots and cells (link). In order to program the process of interaction between the robot and the cell, an element of communication between them is necessary. An appropriate agent can be used for this. Thus, when more connections appear between one cell and several nanorobots, the elimination process can be accelerated.

Let's imagine nanorobots that are constantly looking for "bad" cells using Netlogo agents (using Turtles). Since the "bad cells" also move, they should also be represented with the help of turtle agents. The area of movement of "bad" cells and nanorobots will be presented in the form of patches (Patch). When interacting with nanorobots (a nanorobot sends a signal to other nanorobots if a "bad" cell is detected), as well as when interacting with a "bad" cell, you should use the NetLogo Link agent (communication).

The nanorobot can be in one of three states: in the state of search, in the state of interaction, in the state of destruction. Search state: the robot moves along a random trajectory, if a “bad” cell is found, it enters the interaction state. Interaction state: signaling to other nanorobots, transition to a state of destruction. Destruction state: destruction of the "bad" cell.

All actions in one state or another are described by procedures in the NetLogo language, let's list them: move-robots (describes the movement of a nanorobot); move-cells (describes cell movement); setup-space (changes the color of patches); setup-cells (used to create cells); find-cell (search for cells), etc.

Now we can move on to experimenting. For the number of cells and the number of robots, you can define a default value of 200 (200 cells and 200 robots), for the aura sensor, the default value is 10 patches, and for the aura detector, the default value is 2 patches. These values were used in the work [6] presented in the article from which the model algorithm was taken. Also, it should be noted that each cancer cell has a base health of 100 units. When interacting with a nanobot, the cell loses 5 health units per tick. A tick is a time unit of the ASIM NetLogo model. Next, we present the details and results of each of the experiments.

A. *Experiment №1*

As part of the experiment, the following input parameters are set: number of robots: 1 robot; number of cells: 200 cells; aura detector radius: 2 patches; aura signal radius: 10 patches. The total running time of the model was 13623 ticks (see Fig 1). It should be noted that the rather long time of the model operation is obvious, since the nanorobot needs to constantly move from one cell to another. In addition, the process of cell destruction also takes a long time, since one nanorobot is able to take only 5 cell health units per tick.

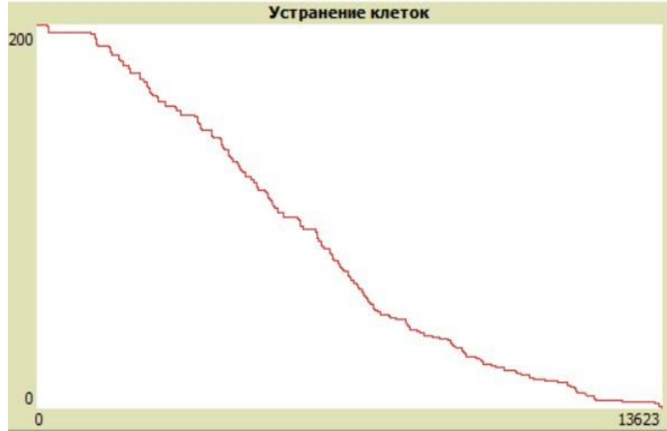

Fig. 1. Graph of the dynamics of changes in the number of cells (the number of robots is 1; the number of cells is 200)

B. *Experiment 2*

Input parameters: number of robots: 1000 robots; number of cells: 200 cells; aura detector radius: 2 patches; aura signal radius: 10 patches. The total running time of the model was 618 ticks (see Fig 2). Despite the fact that the number of nanorobots significantly outnumbers the number of "bad" cells, the process of cell destruction still took quite a long time. This is due to the fact that when receiving a signal for help, each free robot changes direction and starts moving towards the source of the signal, instead of trying to detect a "bad" cell nearby. Thus, it can be stated that the time costs are due to the movement of robots to the source.

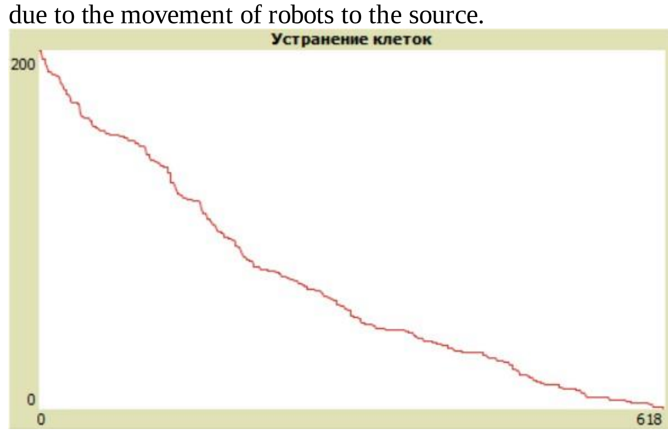

Fig. 2. Graph of the dynamics of changes in the number of cells (the number of robots is 1000; the number of cells is 200)

C. *Experiment №3*

Input parameters: number of robots: 200 robots; number of cells: 1; aura detector radius: 2 patches; aura signal radius: 10 patches. The total running time of the model was 4 ticks (see Fig 3). Several robots managed to quickly deal with one cell. Most of the robots heading for the signal did not even have time to get to the cage.

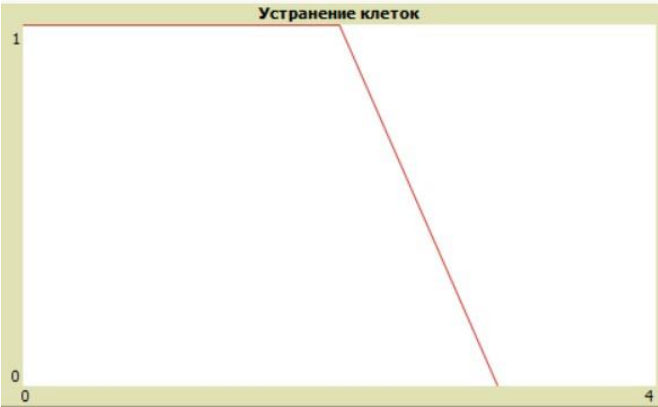

Fig. 3. Graph of the dynamics of changes in the number of cells (the number of robots is 200; the number of cells is 1)

D. Experiment №4

Input parameters: number of robots-200, number of cells-1000, aura-detector radius-2 patches, aura-signal radius-10 patches. The total running time of the model was 2038 ticks (see Fig 4). It should be noted that, despite the large difference between the number of cells and the number of robots, the task of destroying cells was solved quite quickly. Nanorobots were quite effectively distributed among the targets.

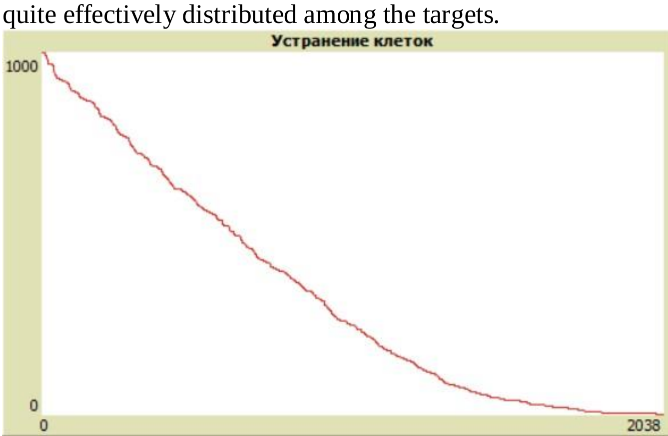

Fig. 4. Graph of the dynamics of changes in the number of cells (the number of robots is 200; the number of cells is 1000)

E. Experiment №5

Input parameters: number of robots: 200, number of cells: 300, aura detector radius: 2 patches, aura signal radius: 10 patches. The total running time of the model was 1175 ticks (see Fig. 5). Nanorobots quickly coped with the task, in fact, the number of cells exceeds the number of nanorobots by a small amount.

Let's try to increase the radius of the aura-signal in order to notify as many nanobots as possible when a "bad" cell is detected.

Fig. 5. Graph of the dynamics of changes in the number of cells (the number of robots is 200; the number of cells is 300, the radius of the aura-signal is 10 patches)

F. Experiment №6

Input parameters: number of robots: 200, number of cells: 200, aura detector radius: 2 patches, aura signal radius: 41 patches. The total running time of the model was 1186 ticks (see Fig.6). Since the nanorobot that found the "bad" cell sends a "powerful" signal to all other nanorobots, they rush to the found cell, ignoring the cells that are in close proximity to them.

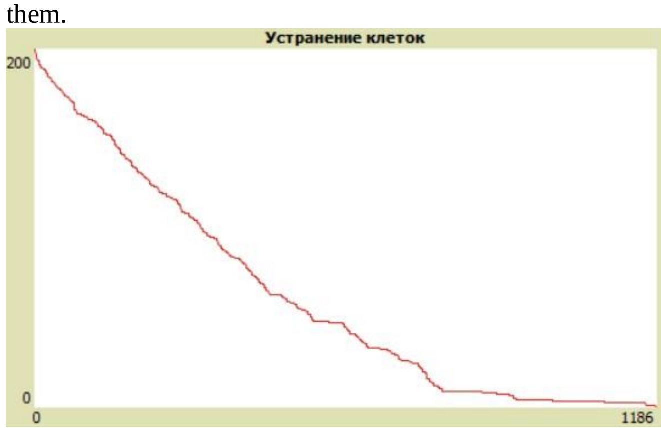

Fig. 6. Graph of the dynamics of the change in the number of cells (the number of robots is 200; the number of cells is 200, the radius of the aura-signal is 41 patches.)

G. Summary

So, studies have shown that by manipulating the values of the radius of the aura-signal and the radius of the aura-detector, as well as the values of the number of nanorobots and cancer cells, it is possible to optimize the work of nanorobots to destroy the said cells. It can also be noted that the robot interaction algorithm is quite sensitive to changes in the aura-signal radius parameter.

VI. Conclusion

The paper presents studies related to the optimization of the algorithm for the interaction of nanorobots to destroy cancer cells. The studies were carried out using simulation methods. The system of agent-based simulation modeling

NetLogo was used as a tool. Nanorobots are represented by agents - IM elements in NetLogo. The experience of using Netlogo can be extended to other examples of the use of robots. Robots can act according to rather complex scenarios; when performing certain procedures, they can use knowledge. In order to perform simulation experiments involving intelligent robots, it is necessary to develop appropriate tools. It is planned to extended this research into realizing intelligent agent aimed at conducting a research on interactions between intelligent machines.

References

[1] Syretsky G. A. Robotics and automation of production: current state / G. A. Syretsky // 2017 Computer and Information Sciences, V.5, No. 2 - pp. 24-29.

[2] Dudenhoeffer , DD Modeling and simulation for exploring human-robot team interaction requirements / DD Dudenhoeffer , DJ Bruemmer , ML Davis // 2001 Winter Simulation Conference. – pp. 730-739.

[3] Viswanathan, J. Using hybrid process simulation to evaluate manufacturing system component choices: integrating a virtual robot with a physical system / J. Viswanathan, WS Harrison, DM Tilbury, F. Gu // 2011 Winter Simulation Conference - pp . 2822-2833.

[4] Li MP Task selection by autonomous mobile robots in a warehouse using deep reinforcement learning / MP Li, P. Sankaran, ME Kuhl, R. Ptucha , A. Ganguly and A. Kwasinski ,/ 2019 Winter Simulation Conference (WSC), 2019 , pp. 680-689.

[5] Mattern , H. Simulation of automated construction using wired robots / H. Mattern , T. Brukmann , A. Spengler, M. König // 2016 Winter Simulation Conference. – pp. 3301-3133.

[6] Liu, ES On the scalability of agent-based modeling for medical nanorobotics / ES Liu // 2015 Winter Simulation Conference. – pp, 1427-1435.

[7] Turnage, DM Simulation results for localization and mapping algorithms / DM Turnage // 2016 Winter Simulation Conference. – pp, 3040-3051.

[8] NetLogo 6.2.0 User Manual: Programming Guide: [Electronic resource]. URL: NetLogo 6.2.0 User Manual: Programming Guide (northwestern.edu).

[9] Ershov N . M . Introduction V distributed modeling V environment NetLogo / N . M . Ershov . – Moscow : DMK Press , 2018. - 264 p.

[10] Lee H. Intelligent scheduling and motion control for household vacuum cleaning robot system using simulation based optimization / H. Lee, A. Banerjee // Proceedings of the 2015 Winter Simulation Conference. – pp 1163-1171.