SIMULATION OF WEATHER IN AGENT MODELS

Vitaliy Lytvynov, Artem Zadorozhnii

***Abstract**: The article deals with the agent-based approach to creating simulation models. A classification is proposed for agent models depending on their behavior and ability to move. Depending on the ability to move and affect other objects, we have been proposed to distinguish between active and passive agents, spatial and time series agents. Particular attention in this article is paid to time series agents and their simulation using neural networks. The article also considers training and testing tool for neural networks developed within the research for the possibility of using neural networks for time series simulation.*

***Keywords**: agent, agent simulation, rescue neural network, time series.*

Introduction

Agent [1] is an entity, which has an activity, autonomous behavior may interact with the environment and other agents make decisions based on some set of rules. Agent models are useful for studying systems, the behaviour of which is determined not by global laws, but by contrast, these laws are the result of individual activity of multiple agents.

The transition from the concept of the object to the concept of the agent occurs when at least one of the three types of behaviour is exist for object: reactive, collective or management behaviour. Reactive behavior occurs in the object as a response to external factors such as ambient temperature, rainfall. Agent, based on these factors can change its settings and move in space. Collective behavior occurs when there is a space of at least two objects that can change its state and move in space in relation to each other. We can say about managerial behavior when one or more agents control the behavior management group of agents.

Agent simulation is a convenient way of simulation modeling, which allows describing the behaviour of each individual agent, which in turn allows distinguishing the development and testing of each agent as a separate independent subtask.

Operation is a set of actions of different forces, coherent and related by purpose, tasks, place and time, which are conducted simultaneously and consistently in accordance with a single concept and plan to solve problems in the strategic or operational direction (in a certain zone, area) within a reasonable period of time. The operation is performed in accordance with one of several plans developed in case one of the possible situations encountered.

Types of agents in operation models

Model agents differ from each other by sets of properties, but the behavior of agents is more important for agent simulation. In terms of spatial agent simulation, behaviour is a change by the agent of the location in space, as well as interaction with other agents. Depending on the presence or absence of the ability to move in space, we shall distinguish active and passive agents. Active agents can freely change their location in space, while passive agents lack this capability. A special type of agents is spatial agents, which cannot move in space like passive agents, but actively interact with active agents in the process of their moving in space. For example, when moving the rescuer agent on the map, diverse spatial agents, such as rivers, roads, forests, can affect the velocity property, as well as agent's energy reserve. The effect of spatial agents on the velocity property will lead to a

change in the rate of movement of the active agent in space depending on the spatial agent. For example, the active agent will move across the field or road much faster than across the forest, or when crossing the river. The effect of spatial agents on the energy supply, namely the reduction of the value of this parameter while moving will allow to avoid the active agent's infinite movement in space. Spatial agents allow to create more detailed models of interactions. Parameters of active agents depend on the resources, which are limited, and their number decreases depending on the actions of the active agent in the course of model time. Resources can be destroyed, exhausted or captured by agents of the counteracting plan of operation, resulting in a rapid decrease of parameters of active agents. For example, when a resource "Fuel" is exhausted, the agent "Car" stops its movement, but remains intact, and the exhaustion of the resource "Fuel" for aircrafts leads to their destruction.

Time series agents

Simulation of the behaviour of agents, depending on the factors, which can be represented as a time series, also has a very important meaning. The examples of such factors may include temperature, humidity, rainfall, solar radiation. For the simulation of the factors, which can be represented as a time series, a special type of agents has been introduced – time series agents.

One of the problems to be addressed in the simulation of the movement of active agents is the selection of the optimal route or several routes, depending on practicability and cost of passing along this route, which fits the traveling salesman problem. The traveling salesman problem is to find the most profitable route through the points given. The statement of the problem includes the profitability criterion of the route (shortest, cheapest, cumulative criterion, etc.) and corresponding distance matrices, cost matrices, etc.

Let us consider an example of how a change in external factors may affect the change in the balance of route sections and eventually the change in the route of agricultural machinery. Figure 1 show a map with two agricultural enterprises, which are in different localities (Anisov and Baklanova Muraveyka). Each of agricultural enterprises owns a certain set of agricultural machinery. The enterprises have entered into an agreement that in case of shortage of agricultural machinery at one of the enterprises, the other should help if the vehicles are available.

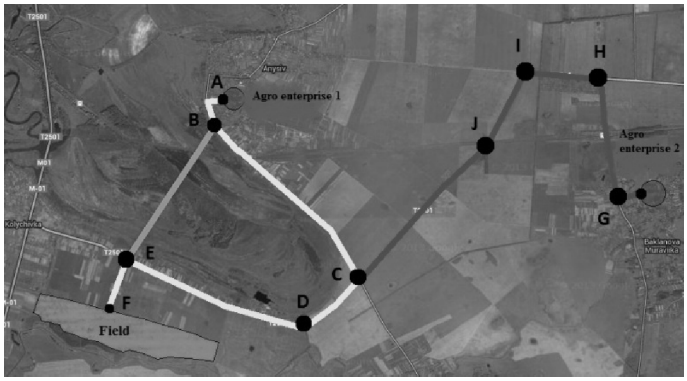

Figure 1. Model of movement of the agricultural machinery

The agricultural enterprise located near the vertex A owns the field located near the vertex F, and, throughout the year, performs a series of agricultural operations in this field, such as planting, processing and collection of crops.

The agricultural machinery from this agricultural enterprise can reach the field by two routes, one of which passes through the wetland (A-B-E-F), and the second one – along the asphalt road (A-B-C-D-E-F). The second route is considerably longer and, therefore, to transfer the agricultural machinery along this route, one need to use a significantly greater amount of fuel and time. In the summertime, the agricultural machinery can travel along the

route segment B-E, but in some years, heavy rains flood the lowlands, through which the above section passes, which prevent the movement of vehicles. Thus, the effect of weather conditions can alter the routes of movement of the active agents.

Simulation of weather factors using neural networks

Since the simulation should be carried out considering weather factors, one of the subtasks to be solved is to simulate a change in weather conditions, for which neural networks are convenient to use. Weather factors are often presented in the form of a time series – an array of pairs <time>-<parameter value>. One of the methods, which can be used for the simulation of time series, is neural networks, which can be trained through the use of weather time series. The network trained based on the weather time series can be used as an element of the time series agent. For the topology, for example, the Ward network and multilayer perceptron, which are shown in Figures 2 and 3, are suitable for simulation of time series.

The multilayer perceptron is a network consisting of several layers of neurons connected in series. At the lowest level of the hierarchy, the input layer is located, consisting of sensor elements, the task of which purpose is only reception and distribution of input information through the network. Then, there are one or, more rarely, several hidden layers.

Each neuron in the hidden layer has a number of inputs connected to the outputs of the neurons of the preceding layer, or directly to the input sensors X_1, …, X_n, and one output. Neuron has a unique weight vector w. The weights of all the neurons of a layer form a matrix, which shall be denoted by V or W.

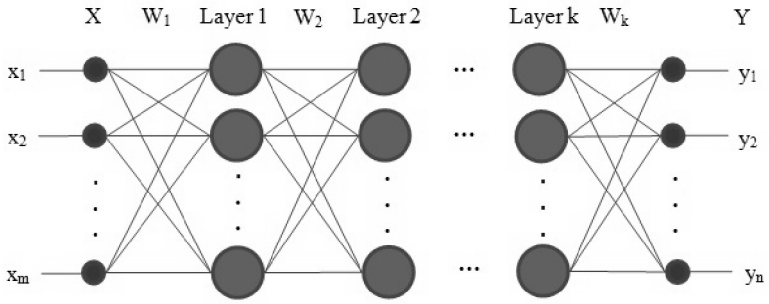

Figure 2. Multilayer perceptron

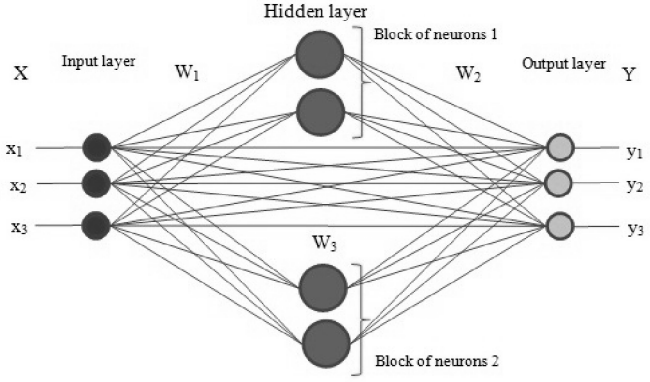

Figure 3. Ward networks

To represent the state of the neural network, the analog logic is used, in which the valid states of synaptic connections are determined by arbitrary real numbers and the degree of neuronal activity – by real numbers

between 0 and 1. Sometimes, models with discrete arithmetic are studied, in which the synapse is characterized by two Boolean variables: activity (0 or 1) and polarity (-1 or +1), which corresponds to the three-valued logic. States of neurons can thus be described by a Boolean variable. This discrete approach makes the configuration space of states of the neural network finite (not to mention the benefits of the hardware implementation).

The Ward network also comprises several layers of neurons connected in series. The partition of hidden layers into blocks allows using different transfer functions for different blocks of the hidden layer. Thus, same signals received from the input layer are weighed and processed in parallel using several methods, and the result is further processed by the neurons of the output layer. For the input layer neurons, a linear activation function is usually set, and for neurons from the blocks of the hidden and output layer, it is determined experimentally. Figure 3 shows the structure of the Ward network consisting of one input, one output and one hidden layer, and the hidden layer is divided into two blocks of neurons, each of which implements its own activation function.

Before creating a neural network, one shall choose its parameters, such as the size of the input vector and the size of the training sample. The size of the input vector in the prediction of time series is the number of samples of the time series, which is simultaneously applied to the inputs of the neural network at each step of training. The size of the input vector equals to the number of inputs of the neural network. Figure 4 shows the size of the input vector of the neural network during the simulation of time series.

The size of the training sample is the number of samples of the time series, which is applied to the inputs of the neural network in the process of training the neural network. For an adequate time series model, a sample shall be applied covering as much states of the object studied as possible. The size of the training sample often equals to the period of the time series. The correct choice of the size of the input vector affects the imprecision of the model and the time of training the neural network, as well as the choice of the depth of the training sample during training the neural network (the higher the sampling depth is, the more time is required to spend on training the neural network).

Training of the neural network for modeling of weather factors

Within the study of the possibility of using neural networks for the time series simulation, a distributed training and testing tool for neural networks has been developed. The use of distributed architecture was due to the fact that the neural network training takes a lot of time and resources.

The architecture of the distributed training tool is presented in Figure 4. The RMI protocol has been selected a communication protocol between the manager and teaching and testing neural network servers. The use of this protocol allowed transmitting messages between the manager and servers as serialized objects. The neural network creation module allows creating sets of neural networks with various parameters such as the size of the input vector, the number of hidden layers, and the number of neurons in hidden layers. The neural network parameter selection module helps to select the above parameters. To select the neural network parameters, the methods of statistical analysis are used.

The tool user can create a set of untrained neural networks using the neural network creation module. Teaching and testing neural network servers establish a connection with the manager of testing and training servers, and the manager adds them to the list of available servers. Based on the list of untrained neural networks and the list of teaching and testing servers, the user can create a set of tasks for teaching and testing servers.

The user can perform a list of tasks for training neural networks; multiple neural networks can be assigned to one server. In this case, the server will perform all the tasks sequentially. All the servers perform tasks simultaneously and in a distributed manner. Upon completion of the task, the server transmits the trained neural network or

network testing results through the RMI protocol to the server management dispatcher. The trained neural network is stored in a database, which allows saving the state of the trained neural network when exiting the tool.

The neural network training occurs at the teaching and testing neural network server, the interface of which is shown in Figure 5. The teaching and testing server can connect to the Server Manager, and the manager adds the server to the list of teaching and testing servers. The indicator at the bottom of the interface shows the completion percentage of the task sent to the server.

The tool allows to check the results of the neural network training using a set of tools substituted at the Experiments management tab. Figure 6 shows the result of checking the trained neural network. Training the neural network was conducted using a temperature time series. The neural network was trained using the time series, and after the neural network training, the training result has been tested using the means of the Experiments management tab.

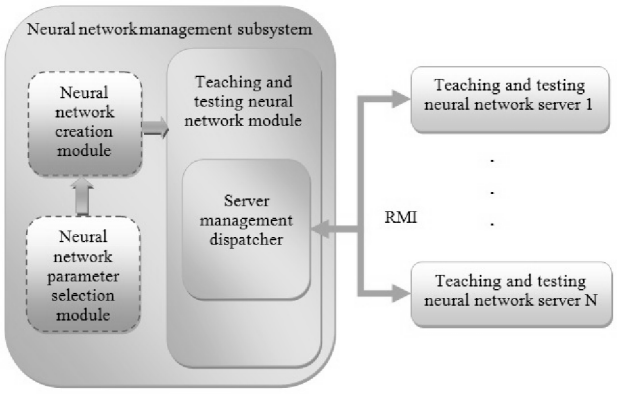

Figure 4. Architecture of the neural network training tool

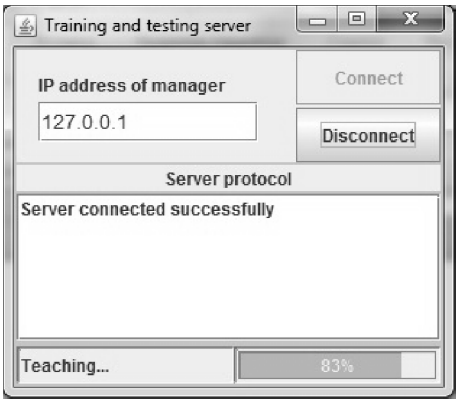

Figure 5. Neural network training window

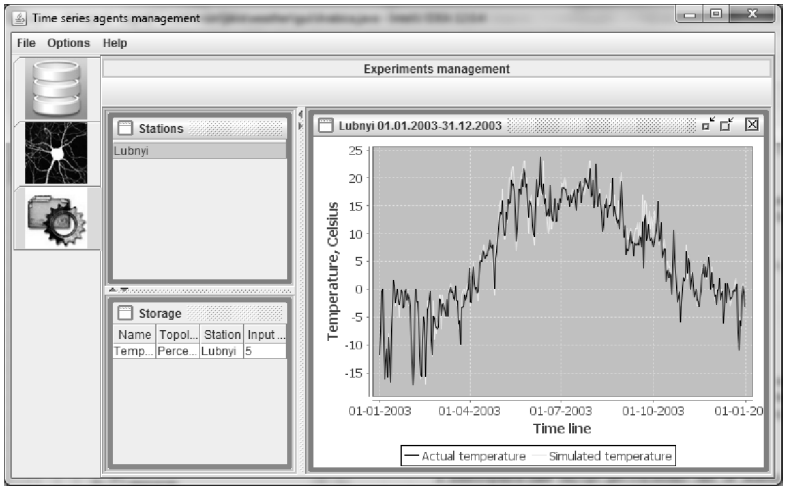

Figure 6. Neural network testing window

Model of the active agent's movement over the time series agent

Figure 6 shows the model of the active agent's movement over the time series agent. The time series agent was trained using a temperature time series. The active agent starts its movement from the point with coordinates specified during the model initialization. The active agent's movement occurs from the point A to point B. The agent's route is over the time series agent. During the time series agent's movement outside the agent time

series, its parameters do not depend on the time series agent's parameters. During the active agent's movement over the time series agent, the interaction between the active agent and the time series agent occurs, and they both can affect each other. For example, the time series agent can reduce the number the active agent's movement velocity or reduce the active agent's reserve forces. In its turn, the active agent can change the time series agent's parameters. Time series agents can also affect passive agents and spatial agents if they are within its range. For example, when modeling agricultural operations, the time series agent, trained on the rainfall time series can change the parameters of the spatial field agent, which the possibility of applying a certain agricultural operation depends on.

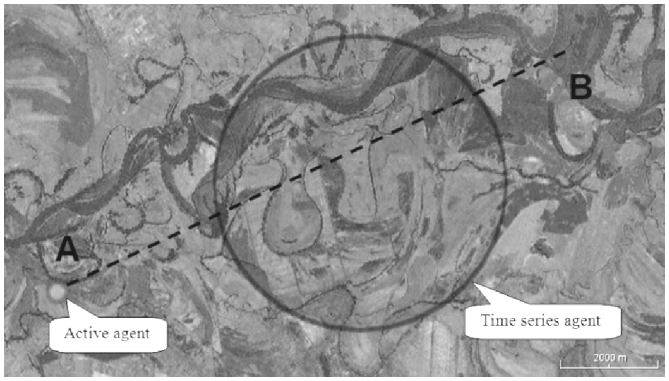

Figure 7. Model of the active agent's movement over the time series agent

Conclusion

Depending on the behavior, active and passive agents have been distinguished, as well as a special type of agents: spatial agents. For modeling the factors, which can be represented as a time series, a special type of agents has been introduced – time series agent. A neural network has been selected a simulation means. Within the study of the possibility of using neural networks for time series simulation, a distributed teaching and testing neural network tool has been developed.

Bibliography

1. Yu.G. Karpov. System simulation modeling. Introduction to simulation with AnyLogic 5/Yu.G. Karpov – St. Petersburg: BKHV-Petersburg, 2005. – 400 p.

Authors' Information

Artem Zadorozhnii *– Postgraduate, the assistant lecturer, Chernihiv National Technological University, 95, Shevchenko street, Chernihiv-27, Ukraine, 14027; e-mail: zaotroy@gamil.com*

Major Fields of Scientific Research: modeling of complicated systems, object oriented programming, distributed systems

Vitaliy Lytvynov *– Dr. Sc. Prof. Chernihiv National Technological University, 95, Shevchenko street, Chernihiv-27, Ukraine, 14027; e-mail: vlitvin@ukrsoft.ua*

Major Fields of Scientific Research: modeling of complicated systems, computer-aided management systems, decision support systems