BRIEF GUIDE FOR AGENT-BASED MODELLING

Viliam KOVÁČ*, Ján PASTIRČÁK**, Lukáš FRIGA***
*Department of Finance, Faculty of Economics, Technical University of Košice, Němcovej 32, 040 01 Košice, Slovak Republic, tel.: +421 55 602 3262, e-mail:viliam.kovac@tuke.sk
**Department of Electronics and Multimedia Communications, Faculty of Electrical Engineering and Informatics, Technical University of Košice, Park Komenského 13, 041 20 Košice, Slovak Republic, tel.: +421 55 602 4180, e-mail:jan.pastircak@tuke.sk
***Department of Finance, Faculty of Economics, Technical University of Košice, Němcovej 32, 040 01 Košice, Slovak Republic, e-mail:lukas.friga@student.tuke.sk

ABSTRACT

This paper is devoted to agent-based modelling as a form of simulation process. It provides a brief introduction to agent-based modelling process. This article introduces a new modern approach to compose complex systems. It also deals with a role of agent as the substantial part of agent-based system. Successively, types of usage of agent-based systems are mentioned. General goals of common modelling process are provided in the next part. It offers basic description of agent-based modelling process too. Main attention is paid on a introduction to the software solutions for agent-based modelling, where one of the most applied pieces of software are described with their advantages and disadvantages. Various agent-based modelling software platforms have been developed in recent period. Many of them are aimed at particular activities. The separate chapter is devoted to the NetLogo software with focus laid on brief characteristics of this software followed by description of modelling process within its graphic user interface. The conclusion involves summary of this paper.

Keywords: *agent, model, simulation, NetLogo*

1. INTRODUCTION

Computer science has been developed very significantly in recent years. Time-consuming methods have become more efficient to compute nowadays. Computers play important role in this process. Field of agent-based modelling has arisen as one of new possibilities how to smartly model processes that are difficultly figured and visualized in common ways. This method enables modelling and simulating any type of process comprising one or more entities that behave according to own decisions.

Agent-based modelling is the kind of simulation modelling process. Main substance is created by work of agents - the system participants that generate various actions and mutual interactions. The game theory plays very important role in field of modelling. Agent-based model presents computational instance prepared to undergo simulation.

With the ascend of agent-based modelling usage, various agent-based platforms were developed. Perhaps one of the most unbearable barriers is software development. It is not only about software engineering. The significant part of construction is done in area of determination and choice of parameters. Basement for good agent-based simulation software is creation of suitable modelling universe. Inclusion of agents into the system is also substantial point. [24]

Agent-based modelling is new modern way of research. Interacting agents create the environment suitable for doing experiments. In the real world precise exploration is not performable everywhere, every time and in every requested manner. The easier system is observed, the more apparent interactions or whatever type of activity can be seen, recognized and finally identified entirely. Successively, the origin of these actions is discoverable with the higher probability in prearranged agent-based system than in the real world. More advisable practice is to explore slightly controlled performance of participating agents than to decompose whole complex system with many possible variables influencing output.

One of the first initial points of agent-based modelling process is construction of the set of assumptions about involved agents. Whilst agents represent the real instances acting in the live world, they have to behave in this way. Nowadays computer power provides for the simulation of comprehensive system including many sensitive variables. Agent-based modelling process enables to investigate wide range of activities from microscopic events through common actions to large-scale effects that are not carried out easily. Such approach reveals strengths of the agent-based modelling. [6]

One of the main advantages of agent-based modelling is appearance of unclear and unexpected consequences. This happens also in cases when initial conditions do not show any uncommon signs outwardly. To be precise, agent-based modelling introduces capability to generate complex agent systems with network relations and all the similar components of the complexity theory. This makes agent-based models generally unpredictable by ordinary mathematical techniques. With the rise of computational strength of new technologies it is simpler to simulate huge complex systems.

2. AGENTS

The whole system established on agent-based modelling is complex autonomous decision-making individuals which are called agents. Every agent behaves in its own way. This means it has been assigned some attributes and some rules of behaviour. The agents attributes represent its attitude to the whole modelled system - that is anything describable by any arbitrary characteristics. They are able to run various behavioural actions. On the other hand its

behaviour follows just right these characteristics according to defined rules. Usually the system consists of intelligent agents that are able to evolve at any time throughout the simulation. It means they can independently act based on their goals, environment and other entities. Their individual behaviour is practically nonlinear. Characteristic feature is that thresholds appear in some moments; they can be invoked by coincidence of more actions done in the same time.

3. AGENT-BASED MODELLING USAGE

There are four separated fields of real applications where agent-based modelling has been introduced to practice:

- flow simulation;
- organizational simulation;
- market simulation;
- diffusion simulation. [10]

4. MODELLING PROCESS

The beginning of modelling is initiated with assumptions about agents and their interactions. Dynamic consequences of these assumptions are generated by simulation.

In general simulation process is considered to be the third way of getting solution besides deduction and induction methods. Deduction, meaning derivation theorem from assumptions, and induction, implicating discovering patterns in empirical data, are quite similar from certain aspects to simulation. The common point with deduction involves initiating set of assumptions, but the result is not the wide-ranging proof of any theoretical hypothesis as in deduction is. Relation to induction is laid in production of data suitable to undergone examination by this method. But these information, which simulated creating output, are originated in experimental environment, whilst induction uses real obtained data from practical experiments generally.

Agent-based modelling covers several parts of economics science. One of them is specialized to computational economic modelling and is called agent-based computational economics.

Each objective of agent-based modelling can be characterized by one of four successive forms:

- empirical aim;
- normative aim;
- heuristic aim;
- methodological aim.

Empirical aim should be explained by repeated activities and interactions of agents. The observed result is evidenced by agents' actions that are repetitive. Normative type of goal looks after exhaustive usage of well-known rules in agent-based system. This means the simulated system is seen essentially steady during its whole runtime and behaves expectable according to basic normative guidelines. Heuristic approach reveals possible hidden phenomenon in the system. Usually this spectacle outgrows basic interaction between two agents and hence influences the whole network. Methodological aim is elementary objective that can be reachable by rudimentary principles.

5. ADVANTAGES OF AGENT-BASED MODELLING

Three elementary statements about agent-based modelling say that such modelling method:

- captures emergent phenomena;
- provides natural system description;
- abounds flexibility.

Very important fact is that agent-based modelling is capable of dealing with emergent phenomena. Simple definition declares emergent phenomenom results from mutual interactions of individual entities. According to definition emergent phenomenom cannot be assigned to whichever partition of the whole system, because then unexpectability and partially also unpredictability fades out. [9]

Natural description of the particular system is offered by the fact that agent-based system describes usual behaviour of involved entities - such system is consisted of so-called behavioural entities. Agent-based modelling makes created model more similar to reality. It is needed to notice that stochasticity is present in model and is applied to agents' behaviour.

Flexibility of agent-based modelling lies in work with agents. They are manageable in any way user wants. Every one aspect of agents' life is controllable by attributes assigned to them. It is applied also on their mutual interactivity and all the communication or relations they together create. [5]

6. AGENT-BASED MODELLING SOFTWARE

There is a lot of software aimed to simulate various systems. In this chapter the alphabetical list of agent-based modelling software is following. Many of these below mentioned pieces of software are considered to be middleware. It means they provide appropriate relation between two or more applications. We have described ten of the very often handled applications that are used to model agent-based system - ordered in ascending alphabetical order.

6.1. Adaptive Modeler

Adaptive Modeler is multiple window software aimed for agent-based financial market simulation models to forecast prices. It uses semiautomatic genetic programming engine where user chooses, what attributes of agents in role of investors, are important for each individual. Then it is needed to describe the levels of the attributes. The foremost weakness is maximal number of agents that can reach only

up to three. On the other hand possibility called population partially substitutes low number of agents. Real market data serves as input for the technical trading rules that are developed by adaptive form of genetic programming. The forecasts are based on the behaviour of the entire virtual market instead of only the best performing agent. This results in robustness of the model and its ability to adapt to changing market environment. Output can be visualized in real-time manner using two-dimensional graphics - diagrams and charts. It is distributed in free evaluation version with some restrictions against full proprietary version for research. [3]

6.2. AgentBuilder

AgentBuilder is the integrated software tool aimed at creation of general purpose multi-agent systems. The author describes it as entry-level tool in field of modelling agent-based systems. There are more versions to download - pro, lite and also evaluation version that is restricted to run only chosen sample agent simulations. All of them use proprietary license agreement. Pro version offers in addition to lite version the agency manager for individual control of agents and management of data flow among them. AgentBuilder consists of two major components - the AgentBuilder Toolkit and the Run-Time System. The first component includes tools to manage the agent-based software development process, to analyze the agent operations, to design and to develop agent communication networks, to define behaviour of individual agents and to debug and to test agent system. The Run-Time System provides engine to execute agent software in generated environment. Agents communicate using the Knowledge Query and Manipulation Language. According to this AgentBuilder allows the developer to define new interagent communication commands that suit his particular need. [1] [2]

6.3. AnyLogic

AnyLogic describes itself as the only multiple-method simulation software that supports discrete event, agent-based and system dynamics simulation. Probably the biggest advantages lie in option to choose three-dimensional graphics and in use of Real Time Unified Modelling Language, what is very familiar to Unified Modelling Language standards. Some parts of source code can be written directly in Java and then imported into the application. This tool is compliant with geographic information systems too. The author provides a few types of license keys, for instance for educational purpose, university research, professional or advanced version which also contains some optimization package. There is also possibility of obtaining time limited trial key. [4]

6.4. Framework for Agent-based Modeling with Java

FAMOJA is acronym standing for Framework for Agent-based Modeling with Java. This software framework contains collection of Java classes that help in the rapid prototyping of agent-based models. The main purpose is simulation of resource flow management in environmental systems. The main convenience is existence of its own application programming interface. Models can be saved into files and then distributed together with their own Java classes and documentation to provide sole easily downloadable file which can be directly loaded right into the FAMOJA application. It is distributed with Lesser General Public License and source code is made public on its website. As it was mentioned above, the tool uses Java programming language. [12]

6.5. Foundation for Intelligent Physical Agents Ontology Service

Foundation for Intelligent Physical Agents Ontology Service, stylized as FIPA-OS, is the component-based toolkit enabling rapid development of agent systems absolutely fulfilling the Foundation for Intelligent Physical Agents standard requirements. There are two versions of this software that are differentiated by their platform - Standard Foundation for Intelligent Physical Agents Ontology Service designed for large operation systems and Micro Foundation for Intelligent Physical Agents Ontology Service suggested for small operation systems primarily implemented in portable devices. They have seven-level hierarchy to meet all the requisite standards. Connection with Jess rule engine is possible and suitable for creating agents. Both two use Java programming language and are accessible as open source under own proprietary license. [14] [15]

6.6. Java Agent Development Framework

Java Agent Development Framework, known also under acronym JADE, is the software framework aimed at making straightforward the implementation of autonomous entities into distributed peer-to-peer applications of multi-agent systems. Involvement set of graphical tools supports the debugging and deployment phases. This framework meets the Foundation for Intelligent Physical Agents standards. The configuration of agent system can be changed even during runtime. The whole framework is implemented in Java programming language. It is distributed as freeware under Lesser General Public License 2. [8] [17]

6.7. Java Agent Services

Java Agent Services is the industry standard specification and the application programming interface for arrangement of agent-based platform infrastructure. It has been developed by the Java Community Process in accordance with the Foundation for Intelligent Physical Agents' Abstract Architecture standard [13] [22]. The first motive to form such norm was intention to design the basis for creating commercial applications based on the Foundation for Intelligent Physical Agents specifications. Otherwise said, it represents the defined set of objects and service interfaces to support the deployment and operation of autonomous interacting communicative agents. It provides also Java Remote Method Invocation to perform remote procedure call from distant system

and the Lightweight Directory Access Protocol to access and maintain distributed directory information over Internet Protocol [23]. The testing and correction tool Technology Compatibility Kit has been introduced to validate constructed applications. [21] Java Agent Services is accessible as open source under Common Public License. [16] [18]

6.8. Mega-Scale Interactive Agent-Based Model Simulations on the GPU

Mega-Scale Interactive Agent-Based Model Simulations on the GPU is the general purpose agent-based modelling system using the graphics processing unit. It uses its own programming language. The author tries to make full use of the computing power and bandwidth. All the computations have to be restricted exclusively to the graphics processing unit with no communication heading for central processing unit. This is done because of slow connection between graphics processing unit and central processing unit. Therefore Mega-Scale Interactive Agent-Based Model Simulations on the GPU is able to use innovative methods for representing and manipulating agent and environment data in graphics processing unit texture memory. The source codes for the project are not available. [19]

6.9. Repast

Repast is acronym standing for Recursive Porous Agent Simulation Toolkit. It is the group of platforms aimed at agent-based modelling and simulation for general purposes. There are two main versions - Repast Simphony which is based on Java programming language and Repast for High Performance Computing that is based on C++ programming language. The first one is suitable for most of users, while the second one is able to run very huge number of models on supercomputer. [7] Repast is considered to be one of the fastest platforms in field of runtime of complex comprehensive models. It provides explicit methods for scheduling actions in both ways - in fixed time steps and dynamically too. Repast is deliberated to be Java version of Swarm - it should offer all the advantages of Java programming languages with complexity of Swarm. It is spread under several third party licenses - notably under Eclipse Public License, Common Public license, Library General Public License and Apache License. [11]

7. NETLOGO

NetLogo is the multi-agent modelling environment originally aimed at teaching and educational purposes. It is suitable for modelling complex system. Users are able to give instructions to huge number agents which all of them operate independently from. This is very useful function, because it makes possible to explore the connection between the microscopic level behaviour of individuals and the macroscopic level patterns that appear from their interaction. [20] On the one hand NetLogo is as easy as appropriate to play for students for educational purposes or to show interesting fields of life through modelling and simulation. On the other hand it offers powerful tools to manage whole large systems.

7.1. NetLogo History

The newest version of NetLogo is marked as 5.0.4. The previous stable distributed version were 1.3.1, 2.1, 3.1.5, 4.0.5 and 4.1.3. Built models constructed by older versions are not always compatible with successive versions. This incompatibility is solved by the Transition Guide, which offers ways how to reconfigure already built models for run in newer version.

7.2. NetLogo Advantages

One of the main advantages of NetLogo is capability of visualization. There is no need to write own source codes for visual display of agent-based system. Simulation itself is demonstrated in separate window. Standard way is two-dimensional animation of running simulation. Besides this, NetLogo is able to demonstrate simulation in three-dimensional way. This visualization can be rotated in two axes to get better angle of view during observing.

7.3. Network Use of NetLogo

There is network tool to add more computers to participate on one project called HubNet. It provides management of modelling and simulation of shared project. Primarily it has been developed to run participatory simulations in the classrooms for schools where teacher initiates the particular model and eventually runs the simulation. Successively students in classroom are able to set the shared model parameters on their own.

7.4. NetLogo Distribution and Licensing

Accessibility to the software itself, to the entire documentation and all the sample models is done through the NetLogo official website `http://ccl.northwestern.edu/netlogo/`. The whole system is made public under Distribution of NetLogo and its models is done through its website under the General Public License 2 as open source. There are several more applied licenses that are related to the third party contributions where the licenses for ASM, Colt, Java Open Graphics Library, JHotDraw, JpegImagesToMovie, Log4j, Macintosh Runtime for Java Adapter, Matrix3D, MersenneTwisterFast, MovieEncoder, PicoContainer, Parboiled, Pegdown, Quaqua Look and Feel and for Scala belong.

ASM is the library aimed at Java bytecode generation - the code involving proper instruction set for Java virtual machine execution. The certain parts of NetLogo source code comes from the Colt project - namely the random-gamma primitive. Java Open Graphics Library is used by NetLogo to render three-dimensional graphics. The Open Graphics Library is known as OpenGL and the particular OpenGL for Java under acronym JOGL. It represents Java application programming interface for the Open Graphics Library. JHotDraw is the component part of

the modelling partition in NetLogo. JpegImagesToMovie embodies the Java source code for creating movies. Log4j is the library used for logging. Macintosh Runtime for Java Adapter, known under acronym MRJ Adapter or by its newer name Mac OS Runtime for Java, is the library to run virtual machine for Java application in Mac OS graphical interfaces for Macintosh operating systems until Mac OS X version has come. Matrix3D is the Java class aimed to process three-dimensional matrix operations. MersenneTwisterFast represents the Java class for random number generation. MovieEncoder is the part of source code adapted to NetLogo from the external Java class aimed at creating movies from simulations. PicoContainer embodies the library that makes source code friendlier to read and to understand. It serves for dependency injection to abandon hard coding. Parboiled and Pegdown are the libraries used for the information tab construction. Quaqua Look and Feel represents the library aimed at design proposing for Mac OS graphical interfaces of Macintosh operating systems. Finally Scala is the programming language which NetLogo language is mostly created by.

7.5. NetLogo Support

The official NetLogo development website is located on the GitHub server `https://github.com/NetLogo/NetLogo`. All the materials found there are freely available to any registered user. Everyone can use it, change it in his own way and hence reuse in his model. A lot of samples from this source are only in process of development, eventually they are only the unfinished projects, for instance to establish the environment of agents to use it in the way the further user wants.

The official development discussion can be found on the Google Groups website where NetLogo has its own group `https://groups.google.com/forum/#!forum/netlogo-devel`. Each user is welcomed to contribute with his ideas and thoughts.

7.6. Modelling in NetLogo

The NetLogo website offers several guides to help beginners with modelling in NetLogo. Helping materials can be found in:

- the Interface Guide;
- the Info Tab Guide;
- the Programming Guide;
- the NetLogo Dictionary.

The Interface Guide describes the elementary menu in the NetLogo window. It says how to do basic operations with models. Moreover, it refers to the components that can be used in modelling process.

NetLogo offers nine components - in order as they are displayed in the menu:

- button;
- slider;
- switch;
- chooser;
- input;
- monitor;
- plot;
- output;
- note.

Button has two variants - once option and forever option. Once button executes given source code only one time, whilst forever button executes instructions until it is clicked on again. Slider represents global variable which all agents are possible to reach. It gives any number from given interval by given increment value. Switch shows state of Boolean data type meaning only two possible statements - true or false. Chooser offers drop down menu for user to choose particular value for given variable of any data type. Input lets user to write its own value for specified variable, but this option is restricted only to numbers and strings. Monitor only displays value of chosen reporter. Plot visualizes dependence of two variables and is interactive in meaning of continuous refresh during model runtime. Output is area aimed at presenting simulation results. It is allowed to have only one output in model. Finally, note serves only as label to show any informative text for user.

The Info Tab Guide shows general help not related to particular model.

The Programming Guide offers twenty-nine topics. Each topic contains short description with links to applied procedures and code example. All the code samples are involved in the Models Library.

The NetLogo Dictionary consists of all procedures which can are recognized by the NetLogo programming language. They are listed in alphabetical order. Almost everyone involves code sample to show its usage.

7.7. NetLogo Window

The NetLogo window consists of three tabs:

- the Interface Tab;
- the Info Tab;
- the Code Tab.

The Interface Tab represents basic view of model. There are displayed all the applied components included in the opened model. There is area where the simulation itself is visualized too.

The tab itself can be divided into three parts:

- the ribbon;
- the desktop;
- the command centre.

The upper ribbon is strip with three sections where following buttons:

- the edit button;

– the delete button;

– the add button;

– the component chooser button;

– the speed slider;

– the update checkbox;

– the update chooser;

– the settings button.

The first section is related to management of model components. The second one deals with simulation speed. The third segment comprises only settings button where model world settings are available.

The desktop is the sector where all the components applied in the model are placed. It involves the world with simulation animation too.

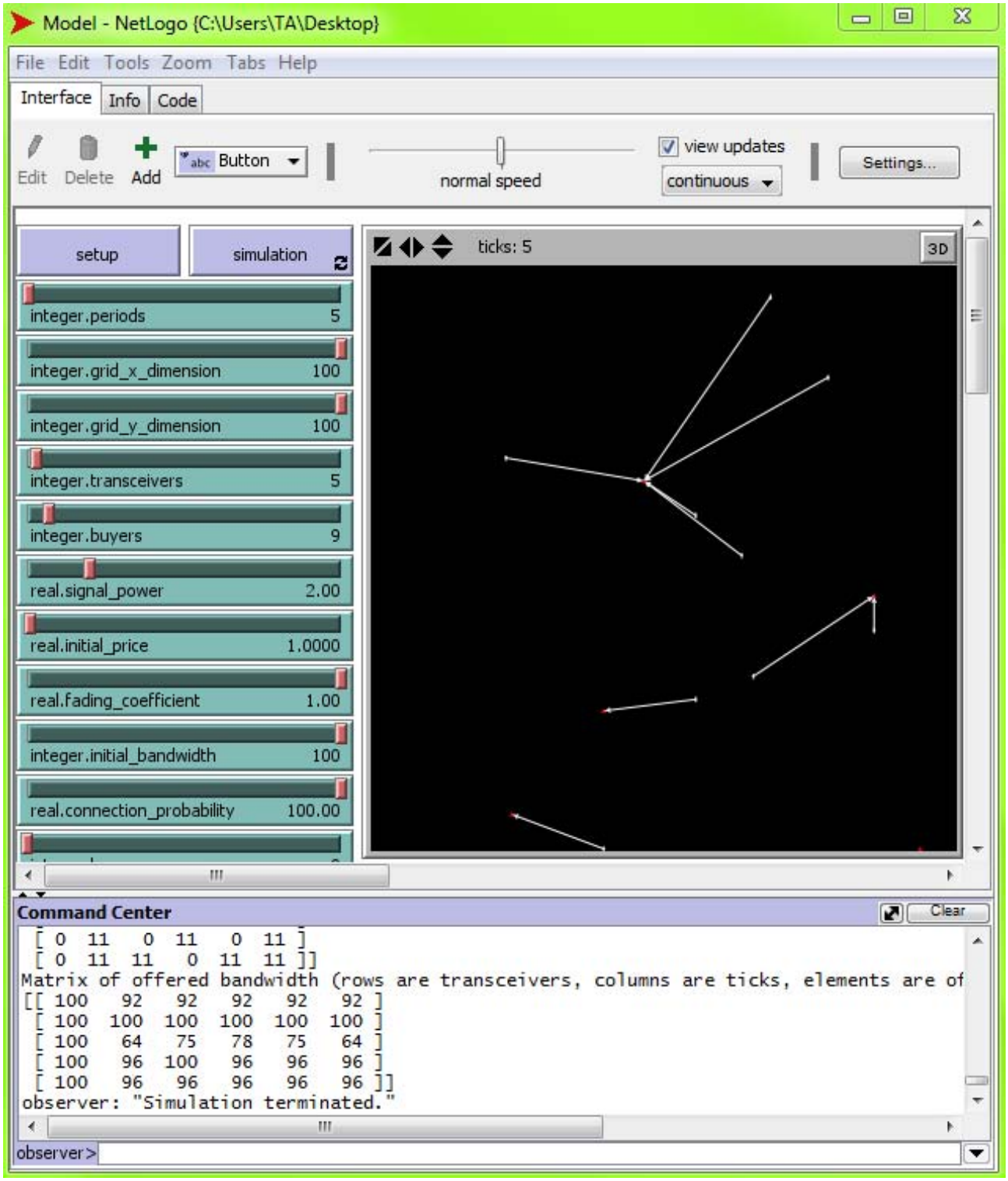

Fig. 1 NetLogo Window - the Interface Tab

The Command centre is the console displaying the entire output from running simulation. It is partially interactive because on the bottom command line is placed. It receives developer's commands and runs them just right after their entering, even during model simulation run. Each sent command has to be assigned to its proper type of entity. Basically majority of commands is run as observer command, but all the other types - from turtles through patches to links commands - are runnable too. The only way to delete the text output history is the button located in header of the command centre.

The Info Tab itself seems to be located little bit illogically between the Interface Tab and the Code Tab in main window. It offers the metadata about the model. The text here is partially preformatted in the HyperText Markup Language, although this part can be entirely changed according to developer's wish.

The upper ribbon includes two buttons:

- the find button;
- the edit button.

The find button looks for any string involved in the metadata. It does search in formatted text, so no HyperText Markup Language tags are findable. The edit button serves to input new metadata.

The Info Tab uses very simple own proprietary interpreter of the HyperText Markup Language in following way:

- line beginning with # mark is formatted as heading type 1;
- line beginning with ## mark is formatted as heading type 2;
- line beginning with ### mark is formatted as heading type 3;
- line beginning with #### mark is formatted as heading type 4.

Standard heading types 5 and 6 are not supported.

The Code Tab serves for input of model source code written in the NetLogo programming language. The upper ribbon divided into three sections and offers following buttons:

- the find button;
- the check button;
- the procedures chooser button;
- the automatical indent check box.

The first section deals with source code as inputted text, the second one serves to choose one particular procedure to display it and the check box in the third section indent separate lines of source code according to their location in its procedure.

7.8. NetLogo Output

There are two options to make output from opened model in NetLogo:

- nlogo file;
- Java applet.

The proprietary file extension of NetLogo model is nlogo. This file type is readable only by the NetLogo application itself. Otherwise, it is only text document containing the pure source code without any marks. Java applet

Besides this exclusive output NetLogo is able to produce applet. It is coded in HyperText Markup Language, so this fulfils requirement of cross-platform run. Such applet involves standard HyperText Markup Language structure - it means the code is divided into head and body parts.

Head includes definition of design styles used in the HyperText Markup Language file itself. It is identical to construction of internal styles using Cascade Style Sheets language.

Then the second part - body - follows. Firstly, it involves notice that the page was generated automatically by NetLogo application and the caution that it is necessary Java 5 or higher version to have been installed and also has to be enabled by browser. Secondly, the brief introduction is situated above the animated applet itself. There is warning the source Java file NetLogoLite.jar with the source package NetLogoLite.jar.pack.gz have to be stored in the same directory as the HyperText Markup Language file with the applet. The package represents standard way of compression archiving of Java files using gzip file compression to have them compiled as the JAR archive file format which stands for Java Archive. Subsequently statement about the applied extensions in the nlogo file is displayed. The whole directories from extensions directory in install folder of NetLogo has to be copied where the applet HyperText Markup Language file is kept. The successive part of introduction says the NetLogo application warns that on some operation systems locally stored applet HyperText Markup Language file must not work properly for unknown reasons. This should be fixed by uploading this file to web server. There is also statement that all of these parts of the HyperText Markup Language file are not compulsory. Developer is allowed to leave them out without breaking the NetLogo application license. It means the sole applet can be in one HyperText Markup Language file. The applet itself is bounded by tags <applet> at its beginning and tags </applet> at its end. The source code for applet is modifiable. For instance, if there are more applets in one HyperText Markup Language file, it is not necessary to store more source files NetLogoLite.jar and packages NetLogoLite.jar.pack.gz. Change of numerical values in archive and value lines in HyperText Markup Language code of the file provides for all the applets are linked the sole file NetLogoLite.jar and

package NetLogoLite.jar.pack.gz. Thirdly, the applet itself is located on the page with notice it is powered by NetLogo and the link to the source nlogo file to be downloadable more easily. Fourthly, legend to the nlogo file follows. It involves metadata about the model. They can be edited by the developer on the Info Tab of the NetLogo application. Fifthly, the NetLogo source code is the final part of the body section of the HyperText Markup Language file. It is shown absolutely in the same way as is coded in the source nlogo file.

8. CONCLUSION

In conclusion, each piece of software has its own dedication. Although all of them represent agent-based modelling means, almost not two of them serve for the exactly identical aim. It is upon each researcher to choose the appropriate application that fulfils the neccessary requirements. This results also in different output they provide. NetLogo and Repast represent absolutely general agent-based modelling applications apart from the other mentioned ones.

To summarise, NetLogo embodies general software devoted to agent-based modelling process. Its use can be found in any field of research. It is suitable for students who begin with agent-based modelling, as well as for skilled researchers. Many attached sample models make introduction to modelling in NetLogo language easier.

ACKNOWLEDGEMENT

This work was supported by the Scientific Grant Agency of the Ministry of Education, Science, Research and Sport of the Slovak Republic under the contract No. 1/0766/14.

REFERENCES

[1] Acronymics: AgentBuilder - An Integrated Toolkit for Constructing Intelligent Software Agents (2004) http://www.agentbuilder.com/Documentation/brochures/ABProBrochure.pdf

[2] Acronymics: Product Description: What is AgentBuilder? http://www.agentbuilder.com/Documentation/product.html

[3] Adaptive Modeler: Overview (2014) http://altreva.com/product.htm

[4] AnyLogic: Overview http://www.anylogic.com/overview

[5] AXELROD, Robert: The Complexity of Cooperation: Agent-Based Models of Competition and Collaboration (1997)

[6] AXELROD, Robert - TESFATSION, Leigh: On-Line Guide for Newcomers to Agent-Based Modeling in the Social Sciences (2014) http://www2.econ.iastate.edu/tesfatsi/abmread.htm

[7] Argonne National Laboratory: The Repast Suite (2013) http://repast.sourceforge.net/

[8] BELLIFEMINE, F.; CAIRE, G.; POGGI, A.; RIMASSA, G.: *JADE - A White Paper*, EXP in Search of Innovation **3**, No. 3 (2003) 6–19 http://jade.tilab.com/papers/2003/WhitePaperJADEEXP.pdf

[9] BERRY, Brian J. L.; KIEL, L. Douglas; ELLIOTT, Euel: *Adaptive agents, intelligence, and emergent human organization: Capturing complexity through agent-based modeling*, Proceedings of the National Academy of Sciences of the United States of America **99**, No. 3 (2002) 7187–7188 http://www.pnas.org/content/99/suppl_3/7187.full.pdf

[10] BONABEAU, Eric: *Agent-based modeling: Methods and techniques for simulating human systems*, Proceedings of the National Academy of Sciences of the United States of America **99**, No. 3 (2002) 7280–7287 http://www2.econ.iastate.edu/tesfatsi/ABMIntro.EricBonabeau.PNAS2002.pdf

[11] COLLIER, Nick; NORTH, MichaelRepast Java Getting Started http://repast.sourceforge.net/docs/RepastJavaGettingStarted.pdf

[12] Current Version of FAMOJA (Version 1.1.0) - Files (2007) http://www-old.usf.uos.de/projects/famoja/versions/1.1.0/?13854126361

[13] Foundation for Intelligent Physical Agents: FIPA Abstract Architecture Specification (2002) http://www.fipa.org/specs/fipa00001/SC00001L.html

[14] Foundation for Intelligent Physical Agents: FIPA Ontology Service Specification (2000) http://www.fipa.org/specs/fipa00086/XC00086C.html

[15] Foundation for Intelligent Physical Agents: FIPA-OS: About http://fipa-os.sourceforge.net/index.htm

[16] Foundation for Intelligent Physical Agents: Publicly Available Agent Platform Services: Java Agent Services API (JAS) (2003) http://fipa.org/resources/livesystems.html#JAS

[17] Java Agent Development Framework is an Open Source platform for peer-to-peer agent based applications http://jade.tilab.com/

[18] Java Agent Services: Description http://sourceforge.net/projects/jas/

[19] Mega-Scale Interactive Agent-Based Model Simulations on the GPU http://www.me.mtu.edu/~rmdsouza/ABM_GPU.html

[20] NetLogo 5.0.5 User Manual http://ccl.northwestern.edu/netlogo/docs/NetLogoUserManual.pdf

[21] OracleCommunity Resources: TCK tools and documentation: Test tools (2014) https://www.jcp.org/en/resources/tdk/

[22] OracleJSR 87: Java Agent Services https://www.jcp.org/en/jsr/detail?id=87

[23] OracleRemote Method Invocation Home: Remote Method Invocation (RMI) http://www.oracle.com/technetwork/java/javase/tech/index-jsp-136424.html

[24] RALLSBACK, Steven F. - LYTINEN, Steven L. - Jackson, Stephen K.: *Agent-based Simulation Platorms: Review and Development Recommendations*, Simulation **82**, No. 9 (2006) 609–623 http://www.sci.brooklyn.cuny.edu/~sklar/teaching/s10/alife/papers/railsback-sim06.pdf

Received May 20, 2014, accepted June 19, 2014

BIOGRAPHY

Viliam Kováč was born on 3. 5. 1987. He graduated at the Faculty of Economics at Technical University of Košice in 2010. He defended his diploma thesis in field of market trading of currency pairs in terms of technical analysis with the title Testing of Forex Market Trading Strategy. Since 2012 he studies as doctoral student at the Department of Finance at the Faculty of Economics at Technical University of Košice. His area of research covers cognitive radio networks in terms of economic relations.

Ján Pastirčák was born on 24. 10. 1989. He graduated at the Faculty of Electrical Engineering and Informatics at Technical University of Košice in 2013. He defended his diploma thesis in field of physical layer in wireless communications with the title Three-Dimensional Baseband Modulation in OFDM Transmission Systems. He is currently a PhD. candidate at the Department of Electronics and Multimedia Communications at Technical University of Košice. The main scope of his research is dynamic spectrum access and spectrum trading in cognitive radio networks.

Lukáš Friga was born on 16. 4. 1988. He graduated at the Faculty of Business Economics at University of Economics in Bratislava in 2012. He defended diploma thesis focused on practical application of independent component analysis for financial time series. Since 2013 he studies as doctoral student at the Department of Finance at the Faculty of Economics at Technical University of Košice. Current area of his research are economic models based on principles of complexity economics.