Applied Realization of Neural Network and Neurolike Parallel-Hierarchical System Based on GPGPU

Andriy YAROVYY
Vinnytsia National Technical University
95, Khmelnytske Shosse, Vinnytsia, 21021, Ukraine
axa@vinnitsa.com

***Abstract* — The paper considers main aspects and features of applied realization of neural network and neurolike parallel-hierarchical systems based on GPGPU technologies. The research performs the analysis of the architecture of modern GPU and models of GPU parallel programming, as well as considers ways of GPU programs optimization for parallel computing in general-purpose parallel-hierarchical systems, proposed as a networking model of neurolike scheme of information processing. On the basis of research performed and results obtained software modules are offered for realization at CPU and GPU of large-scale neural and neurolike parallel-hierarchical networks of different topologies.**

***Index Terms* — parallel-hierarchical neurolike systems, neural networks, parallel computing, GPGPU, video card programming, image processing, forecast.**

I. INTRODUCTION

Nowadays the demand for computerized systems of super high computing power constantly grows. This is connected not only with the constant growth of huge volumes of information of different nature, but also with the increase of complexity of the calculations. Operational response and decision-making in real-time, processing of video data with great sharing capacity, classification and prediction of rapidly changing dynamic data are some of the most urgent problems of today, that require application of very productive intelligent computing systems. Similar systems can be realized on a variety of methodological bases and different element base, but parallel technologies, including parallel neurolike network devices become more and more popular nowadays [1-3].

One of the most efficient ways of processing of large data arrays is their parallel processing based on specialized system solutions, including neurolike parallel-hierarchical systems. However, at the same time, there appears the problem of software-hardware implementation of such type of systems, namely, selection of adequate hardware and software platform for rapid and efficient parallel processing of large data arrays [1-5].

The example of parallel-hierarchical pyramidal structures is the class of multi-SIMD-systems, where several levels of identical processor elements (PE) operate autonomously in SIMD-mode. Each PE has corresponding image element at appropriate permission level. This group includes the following systems: PCLIP, PAPIA, GAM, SPHINX [6]. In other cases, several powerful identical processing units are connected in a hierarchical pyramidal structure. Each processing unit is responsible for the part of processed image. Systems Uhr's Array / Net and EGPA belong to this family [6].

Performance of classic single-processor computer systems together with the existing sequential algorithms does not allow to carry out the processing of large arrays of data in real time. As a result, parallel algorithms for orthogonal transformations [7], allowing efficient parallelization have been constructed. Here, linear transforms are performed using parallel systems such as SIMD and MISD with random predefined number of branches. For such parallel orthogonal transforms, methods of synthesis of orthogonal basis systems transformations in which have more rapid computational algorithms are elaborated.

In previous research it was proved that a successful solution of such problem is the usage of video cards for general purpose computation (GPGPU) [5,6].

II. ANALYSIS OF PARALLEL COMPUTATION ON GPU

Research are concentrated on development of neuro-emulator – system, constructed on the basis of the cascade-connection of universal SISD-, SIMD-or MISD-processors (e.g., Intel, AMD, Sparc, Alpha, Power PC and others), that realizes typical neuro-operations (weighted summation and nonlinear transform) at the program level. Solution of this problem is possible only on condition of correct and substantiated choice of neuro-accelerator – neurocomputer realized as a card or module with parallelization of operations at the hardware level or structurally-autonomous system. Neurocomputers manufactured in the form of cards (virtual neurocomputers), as a rule, are designed for installation in the expansion slot of a computer system (standard PC).

On the basis of research carried out [5,7], it is proposed as neuro-accelerator, as hardware platform for realization of large-scale neural and neurolike parallel-hierarchical networks to choose GPGPU technology – General-Purpose computation on Graphic Processing Units. GPGPU – is the application of powerful video card for performing specialized, including parallel, computations, which are usually performed by the CPU in your PC. Since modern technologies of video cards manufacturing allow to use 128-nuclei specialized processors, in comparison with modern 4-nuclei multimedia CPU, then their application for neuro-

emulation of different topologies of scale neural and neurolike parallel-hierarchical networks is actual and promising.

In the context of program realization the work aimed at creation of neuro-package intended for realization of various topologies of scale neural and neurolike parallel-hierarchical networks and the possibility of their computation at the GPU (realization of the processes of parallel-hierarchical processing of information and corresponding methods of teaching) is carry out. Under the term topology we mean a specific set of network layers, correspondingly interconnected. Number of network layers, connections between them, the number of neurons in the layer, function of layer activation, connections between neurons of different layers are specified by the user. One of the key features of the software – is flexibility of topology creation that makes possible the realization of large-scale parallel-hierarchical and the hierarchy-hierarchical neurolike networks [5].

Number of computing units in modern video cards is much more than two (10 – for video cards ATI RadeonHD 48xx and 10 – for video card GeForce 280GTX). Number of cores in each computing block is much higher (80 cores – for a video card ATI RadeonHD 48xx and 24 cores – for a video card GeForce 280GTX). Local memory is used only for general purpose computations to synchronize threads and is realized not in all video cards (e.g. it is not available in ATI RadeonHD to 4th series). Texture blocks, each SIMD-block having up to 4, possess their own cache (intermediate buffer with fast access that contains a copy of the information stored in the memory with less rapid access but with highest probability to be requested) and allow to load data from global memory into private, which belongs only to a specific block of SIMD-core ("managing" the data in its own cache). All read and write operations in global memory are performed using memory controller and thus are stored in the cache of memory controller [8].

Hence principle of parallelism is implemented in modern video cards on many levels. In parallely operating over different tasks (streams of instruction) SIMD-units parallel processing of one task data is performed. In modern video cards from 2 to 8 memory controllers are used, increasing memory capacity.

Generalized model of GPU programming

Prior to program execution, its constants are loaded in "global cache of constants". After this special program manager allocates a certain number of SIMD-blocks on which it will operate. All core units in SIMD block – perform the same operations but over different operands (single instruction multiply data), that is, divide the flow of instructions. Each core can simultaneously process multiple threads, the number of which is limited only by the size of local memory of the core. Simultaneous processing means simultaneous placing of data in the memory and step-by-step execution of instructions from these threads that enables to mask the delays while reading from the global memory.

Fig. 1 shows generalized model of video cards programming. All modern platforms API (Application Programming Interface) for programming of video cards support the possibility of using multiple display adapters within one system (SLI in NVidia, Crossfire / CrossfireX on AMD/ATI). For each video card, you can assign multiple execution contexts, and each context is associated with one stream of operation system. Within the context shaders (programs for each of the stages of graphic production line used in three-dimensional graphics to determine the final parameters of the object or image) are created and memory is allocated for user data (textures in graphical interpretation). At the same time only one shader from a context can be performed on one video card. The processes of call and data transfer are asynchronous [8].

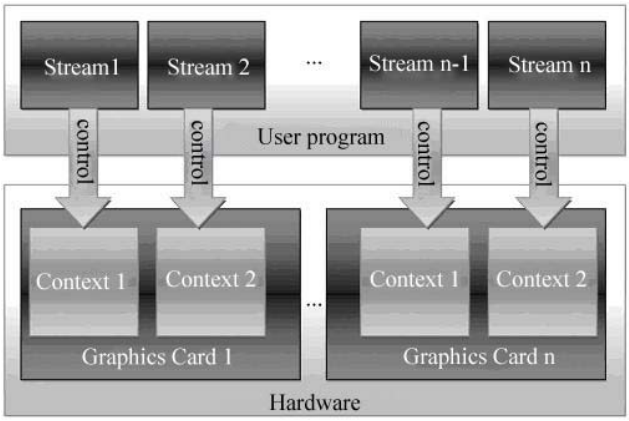

Figure 1. Generalized GPU programming model

III. SIMULATION AND PROGRAM REALIZATION OF LARGE NEURAL AND NEUROLIKE PARALLEL-HIERARCHICAL NETWORKS BASED ON GPGPU TECHNOLOGIES

Main difficulties we may encounter when programming ATI video cards, is the difficulty of writing computational code as the code of computational shaders and call code of ATI Close To Metal (CTM). In particular, shaders are written in ATI Intermediate language (IL), which is assembler-like language with rigid focus on image specificity. To call CTM it is necessary to write a large volume of service code that makes realization of program difficult due to necessity to keep strictly to the sequence of CTM functions call. Alternative to CTM IL is AMD/ATI Brook+ language, which is the extension of C language without the possibility of using object-oriented approach. Programs are developed in C language with additional syntax, resulting program is compiled in C++ code that calls the Brook runtime, which, in its turn, calls CAL runtime. Using this language is not efficient because of redundancy of generated IL-code.

Nowadays open programming standard of various parallel computation facilities, including video cards, called OpenCL is developed [9].

Therefore, the problem how to eliminate some of the above-mentioned drawbacks dealing with video cards programming, in particular, simplification of programmer hardware interface, is very actual. The task of research was to elaborate specialized programming library that would enable the programmer to abstract from specific features of ATI CTM CAL operation and to work with video cards as with a single object. The practical value of the research is to facilitate code writing, as compared with a typical scenario of CAL usage.

In the given research specialized programming libraries,

including "NN-Constructor", were developed. NN-Constructor represents the additional level of abstraction over ATI CTM CAL and is intended for construction of topologies of artificial neural and neurolike networks (in particular, parallel-hierarchical and the hierarchy-hierarchical) and their simulation (fig. 2) [10-12].

Programmer creates only one object of useGPU class, which is further interacted. All that is necessary to define the task for processing to video card is to allocate memory for input / output data and constants (function Allocate), download and compile a program written in IL (function CreateImage), wait for completion of the program (function Execute) and obtain results (function GetResult). Resource deallocation is carried out automatically in the destructor of useGPU class. That is, general procedure of using programming libraries is the following:

Allocate() → CreateImage() → Execute() →
→ GetResult() → ~useGPU().

It should be noted that although the amount of functions of program library is small, but the sequence of calls itself is logically structured.

Since main designation of the suggested programming library is the construction of topologies of artificial neural and neurolike parallel-hierarchical and the hierarchy-hierarchical networks for their simulation, then we consider its basic functional possibilities in the given context. Programming library realizes functions of loading and storing of corresponding description of network topology in text files of specific format, as well as functions for training and processing (signal transfer) in neural or neurolike networks.

In program realization principle of neurolike data processing is chosen; that is, the pulse from neurons, located in the layers, belonging to processing cycle *i* is transferred to neurons of the layers belonging to processing cycle *i+1*. Thus, each value of the input signal I_j can be calculated simultaneously, that is, in parallel. Principle of cycle processing of neural networks is explained by the following figure:

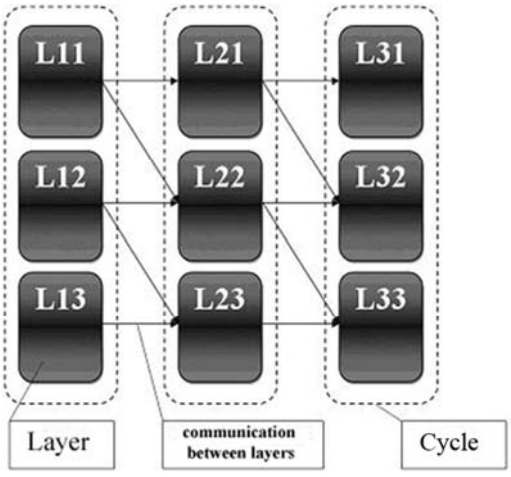

Figure 3. Generalized scheme of the process of cycle processing of neural networks in "NN-Constructor"

Design of neurolike network is carried out by means of joining the layers of neural elements. Layer may contain random number of neural elements, which is limited by maximum size of the array in ".NET framework", and the number of layers is not limited (dynamic list is used). Thus, the number of layers and neural elements are limited only by the volume of RAM installed.

Language of CPU-version programming library realization is C# for the platform "MS .NET 2.0". The functions of the library is correctly work with different operating systems: MS Windows XP (of "MS .NET 2.0" are installed), MS Windows Vista, Linux (if platform "Mono 1.0" is installed). To use the software, program library should be compiled using "MS Visual C# Express Edition 2005". Activation of the program – adding links to a collection of "NN-Constructor.dll" in the project and further compilation in MS Visual C# Express Edition 2005.

Language of GPU-version of programming library is C++ using the software platforms AMD StreamComputing SDK. The functions of the library work correctly with different operation systems for video cards ATI Radeon HD (series 2000 and higher).

Main steps of program operation: loading from the file (or creation by the user by means of appropriate functions) the number of layers and links between them, number of neural elements in the layer, links between the neural elements in different layers of neural or neurolike network; processing of input information, training of the networks; storing of network topology and simulation results.

To run the program library we need the installed operation system MS Windows XP / Vista (Linux); available MS .NET 2.0 (Mono 1.0); clock speed of at least 400 MHz, the memory capacity at least 128 Mb. File of programming library occupies 1 Mb of disk space.

Let us consider some of the features of the algorithm of neural networks modeling taking into account the specific features of parallel devices programming. To avoid the need to realize the mechanism of synchronization of parallel threads, used algorithm (Fig. 3) requires conversion of input data format. Pulse transfer is performed between cycles (cycle – a set of neurons layers, from which the signal is transmitted simultaneously), in the following way:

1. For each neural element one-dimensional table is formed, each element of the table is the structure "number of connected neuron number in the previous layer – weight of interneural connection".

2. Thus, for each layer of the current cycle a set of tables is obtained by the number of neurons in layer characterizing interneural connections.

3. In addition, for each layer additional one-dimensional table is formed that contains levels of activation of the neurons of the layer.

Such conversion allows to store data required for pulse transfer between cycles in a single array and load them into memory of video card during one cycle of data transfer. It should be noted that such implementation allows to avoid the use of operation of random writing in the memory which is not supported by video cards ATI based on chips younger that series R670,and the need to implement the mechanism of synchronization between parallel streams (which reduces the rate of the program execution).

IV. EXPERIMENTAL RESULTS OF NEUROLIKE AND ARTIFICIAL NEURAL NETWORKS SIMULATION FOR THE PROBLEMS OF STATISTICAL SERIES OF EXCHANGE RATES FORECAST

In the research carried out, real statistic series taken from the open sources of Forex market, which reflects the dynamics of hourly fluctuations between euro – dollar rates, dimension of 4137 results (12.10.2008).The task of the experiment was to obtain the forecast value of the rate change fluctuation with the horizon of forecast – 1 step [13].

To forecast the given task several structures of neural network topologies, including Ward network with the structure 100-100-100-1, 100-25-25-1, 9-8-5-1, and multilayer perceptron with different variations of topologies were selected. As an experimental test case neural network – multilayer perceptron with 8-3-1 topology and method of teaching – back propagation of errors were selected. The task of forecasting has been realized using the developed programming library for design and modeling of topologies of artificial neural networks and neurolike networks "NN-Constructor" (with possibility of processing by CPU and GPU).

Fig. 4 shows the results of the teaching of the given neural network, implemented using neural constructor "NN-Constructor". As it is seen from the figure, as a result of teaching neural network correctly reproduces the dynamics of exchange rate values variations, in particular, the average prediction error is 0.004476721, that is acceptable for the proposed economic problem. Step of forecast in the program is defined as follows: from the input series 8 elements were selected and 1 element of input values series (forecast for 9^{th} element of input series as the forecast horizon is chosen 1).

The rate of data processing in neural network was defined; it is equal to the sum of rate of network teaching and testing rate. In the proposed version the rate of data processing in neural network was 14 seconds.

To confirm the adequacy of the proposed software product and validity of the results obtained computer simulations using one of the professional and recognized in the field of neural processing software package – Statistica Neural Network, Company StatSoft was carried out [14].

In particular, Fig. 5 shows the results of computer simulation of the given neural network (multilayer perceptron with 8-3-1 topology and teaching method – back propagation of error) in software package Statistica Neural Network.

As a result of teaching (Fig. 6) neural network correctly reproduces the dynamics of exchange rate values variation, in particular, the average prediction error is 0.00127200.

For Statistica Neural Network for the proposed variant the rate of data processing in neural network was evaluated. The rate was significantly higher (more than 10 minutes) than in variant using "NN-Constructor".

V. CONCLUSION

Thus, the paper considers main aspects and features of neural networks and applied neurolike parallel-hierarchical systems based on GPGPU technology. The research performed the analysis of architecture of modern GPU and model of GPU parallel programming. On the basis of research and results obtained software modules are offered for realization on CPU and GPU of large-scale neural and neurolike parallel-hierarchical networks of different topologies.

Our investigations show prospects of GPGPU technologies and video cards applications as graphic neural accelerators for improvement of the efficiency of processing of super large amounts of data in various applied problems, including pattern recognition and image processing, forecast of performance of time series, especially with application of new approaches to programming of modern generation of video cards, that allows to abstract from graphically-oriented nature of video cards operation to the usage of GPU as a system of SIMD-processors with high degree of parallelism.

Especially perspective seems the idea of parallel hierarchical structures realization based on opto-electronic components and devices with dynamic multitasking [2]. It is due to combination of the ideas of parallel-hierarchical algorithmics and optoelectronic alternative circuitry it is possible to create efficient (including neurolike) computing and information structures, where extension of functional and intelligent possibilities of hardware and algorithmic tools at different levels of the hierarchy, outpaces the growth of their complexity [2].

REFERENCES

[1] V.V. Voevodin "Parallel computing : Tutorial [for students of higher educational institutions.]" / V.V. Voevodin, Vl.V. Voevodin – Sankt Petersburg: BHV- Petersburg, 2002. – 608 p. – ISBN 5-94157-160-7.

[2] "Parallel-hierarchical model of systemic transformation as optoelectronic means of artificial intelligence: [Monograph.]" / V.P. Kozhemyako, Yu.F. Kutaev, S.V. Svechnikov, L.I. Timchenko, A.A. Yarovyy – Vinnytsia: Universum-Vinnytsia, 2003. – 324 p. – ISBN 966-641-072-9.

[3] "Hardware implementation of parallel-hierarchical network based on DSP" / V.P. Kozhemyako, L.I. Timchenko, A.A. Yarovyy, S. Remezyuk / Book of abstracts of Third International Scientific Conference [Optoelectronic Information Technologies "Photonics ODS–2005"], (Vinnitsa, 27-28 April 2005) - Vinnytsia: Universum-Vinnytsia, 2005. - P. 43.

[4] Krug P.G. "Neural networks and Neurocomputers : Tutorial [for students of higher educational institutions the course "Microprocessors"]" / Krug P.G. – Moskow: Publishing MEI, 2002. - 176 p. - ISBN 5-7046-0832-9.

[5] "Selecting the hardware platform for implementing large-scale neural and neurolike parallel-hierarchical networks [Electronic resource]: (IX International Conference on Control and management of complex systems (CMCS 2008)", Vinnytsia, October, 21-24, 2008) / A.A. Yarovyy, Yu.S. Bogomolov, K.Yu. Voznesensky. – Mode of access: http://www.vstu.vinnica.ua/mccs2008/materials/subsection_2.2.pdf.

[6] "Applied aspects of software and hardware implementation of parallel-hierarchical neural systems" / A.A. Yarovyy: Collection of scientific works [Scientific Session of MIPhI - 2009], [XI All-Russia Scientific-Technical Conference "Neuroinformatics-2009"], (Moscow, January, 27-30, 2009), [in 2 parts. Part 2.] - Moscow, MIPhI, 2009. - P. 39-48.

[7] "Comparison of performance graphics accelerators and CPU in the calculations for large volumes of processed data" / Skribtsov P.V, Dolgopolov A.V. // Neurocomputers: development, application – Moscow, Publishing House "Radiotechniques", 2007. - № 9. - P. 421-425. – ISSN 0869-5350.

[8] "GPGPU: General Purpose computations on Graphic Processing Unit [Electronic resource]" – Mode of access: http://www.gpgpu.org.

[9] "OpenCL: Open Computing Language [Electronic resource]" – Mode of access: http://en.wikipedia.org/ wiki/OpenCL.

[10] "Certificate of registration of copyright in a research №27755. Computer program "Software library for design and modeling topologies of artificial neural and neurolike networks" ("NN-Constructor")" / A.A. Yarovyy, Yu.S. Bogomolov, K.Yu. Voznesensky. Registered with the State Department of Intellectual Property of Ukraine, 20.02.2009.

[11] "Certificate of registration of copyright in a research №31705. Computer program "Software library for parallel processing of information based on programming video card ATI ("useGPU")" / A.A. Yarovyy, Yu.S. Bogomolov, K.Yu. Voznesensky. Registered with the State Department of Intellectual Property of Ukraine, 21.01.2010.

[12] "Certificate of registration of copyright in a research №31706. Computer program "Software library for building GPU-cluster with distributed information processing-based programming remote video card ATI ("dotGPU")" / A.A. Yarovyy, K.Yu. Voznesensky. Registered with the State Department of Intellectual Property of Ukraine, 21.01.2010.

[13] "Forex Ukraine" - [Electronic resource] - Mode of access: www.forexua.com

[14] "STATISTICA Neural Networks. Technical description". - [Electronic resource] - Mode of access: http://www.statsoft.ru/statportal/ tabID__32/MId__141/ModeID__0/ PageID__11/DesktopDefault.aspx.

APPENDIX A

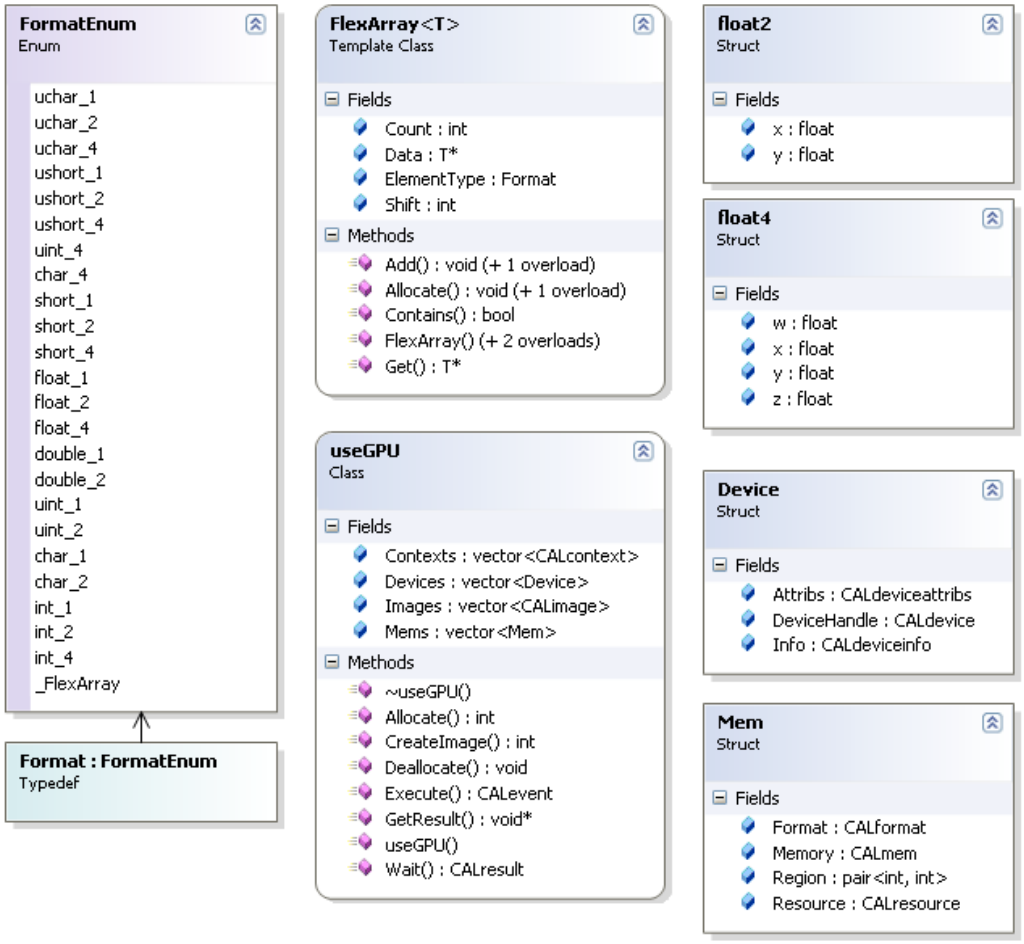

Figure 2. Class diagram of programming library "GPU NN-Constructor".

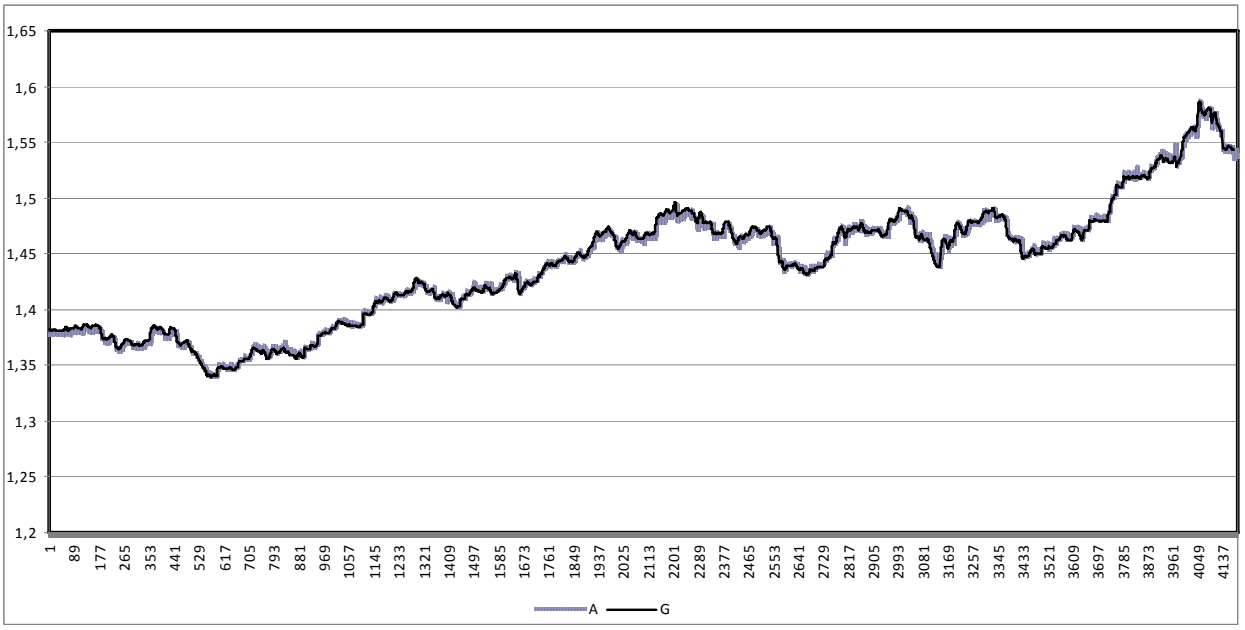

Figure 4. Results of forecast of exchange rates using "NN-Constructor", where series A – original series, series G – forecast series.

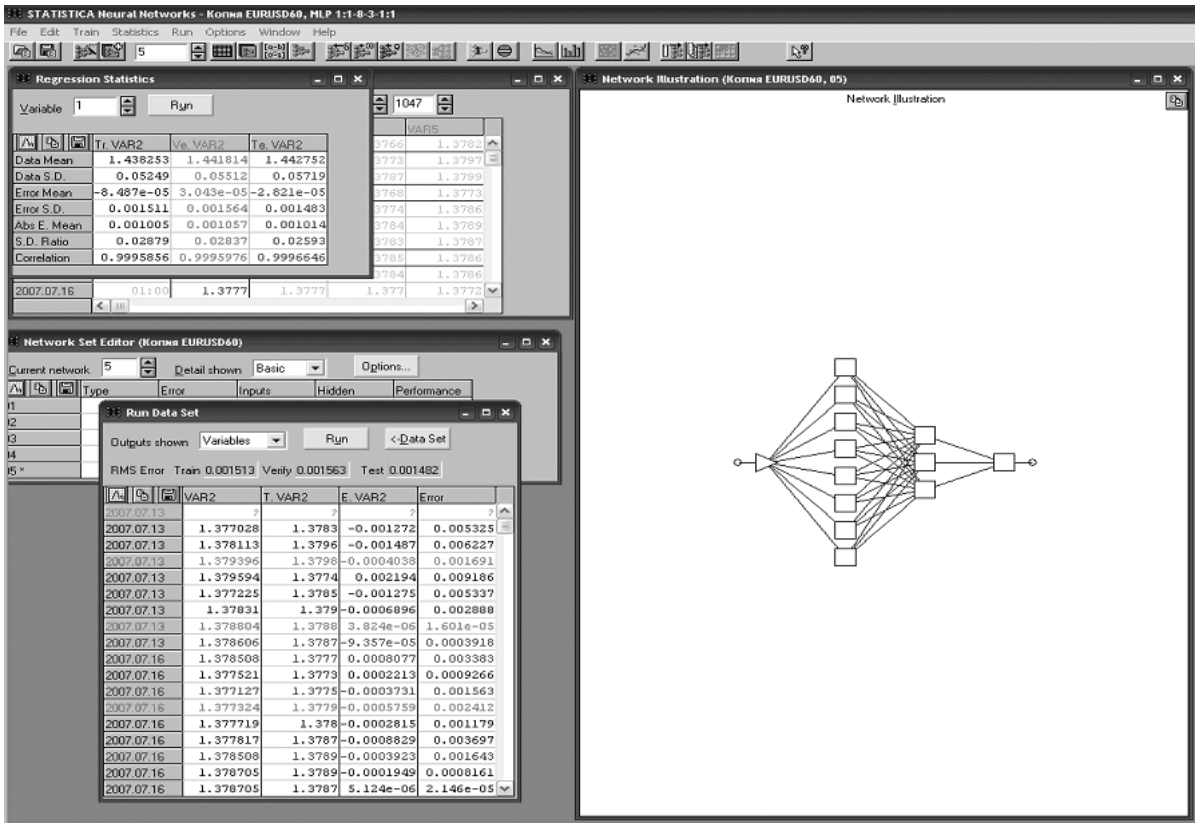

Figure 5. Screenshot of the results of computer simulation in the software package Statistica Neural Network.

Figure 6. Results of forecasting of exchange rates in the software package Statistica Neural Network, where series 1 – original series, series 2 – prediction series.