Computing Utilization via Computer Networks

N. Pham[1], B.M. Wilamowski[1], and A. Malinowski[2]

[1] Electrical and Computer Engineering, Auburn University, Alabama, USA
{nguyehu,wilambm}@auburn.edu
[2] Electrical and Computer Engineering, Bradley University, Peoria Illinois, USA
olekmali@ieee.org

Abstract. The dramatic growth of Internet and network technologies, etc leads to different perspectives of computing methodologies as well as changes of software business model. If the traditional business model for software is one-time payment for a license for one machine with unlimited use, the development of Internet and network technologies, etc makes it possible for users to pay on their consumption as they pay for water, gas and electricity. With advanced technology all computing and storing process can be centralized on the infrastructure of service providers. With this new model, users don't have to concern about deploying their infrastructure, security, etc which will be responsible by service providers. This new trend grows extremely fast in last couple years and attracts a lot of researches from scholars such as Grid Computing model, Client Server model and especially Cloud Computing model with its scalability. In this paper we do not analyze differences between these utility computing models and what model will be the main field in the future. Instead we present how to use computer networks as a mean of computing and simulation and how computer networks are considered as a solution to boost technology development. Two software applications through computer networks were developed and applied successfully in teaching and learning courses in Auburn University and Bradley University are presented in this paper. It is a typical example of enhanced interaction between human and CAD tools while computer networks play a role as a human system interface.

1 Introduction

Since Internet has played an important role in communicating and exchanging information in the world [Wilamowski and Malinowski 2001; Manic et al. 2002], there are many applications developed and deployed on its basis from companies to academic institutions [Wilamowski et al. 1998; Malinowski and Wilamowski 2000; Wilamowski et al. 2000]. Up to now, Internet has been used as an effective means of computing and simulation. Explosion of network technologies and multi-core processor technologies with faster speed makes network computing become an interesting realistic and economic model. Network computing really changes topologies to design software applications [Arano et al. 1996]. Computer networks have been changing not only the engineering view but also the business view as well. Nowadays, many companies mostly rely on computer systems and networks for functions such as order entry, order processing, customer support,

Z.S. Hippe et al. (Eds.): Human - Computer Systems Interaction, AISC 98, Part I, pp. 239–252.
springerlink.com

supply chain management, internal communication, and employee administration, etc. In other words, computer networks become a backbone to keep business running. Because of this importance, the reliability and availability of such systems and networks have to be concerned and they are really critical factors of system level management [Juan et al. 2007].

The advance of World Wide Web technologies plus the improvement of network security and network speed, etc make Internet more dynamically interactive with human. It is not just a tool to display and exchange information, it can be used as calculating means for complex problems which can't be solved by a single desktop or laptop. Grid Computing is a clear justification for the power of network computing which was started in the mid 1990s. Grid Computing takes advantage from the existing infrastructure with limited resources of each academic institution and uses computer networks to combine all these institutions together to create a Computing Grid analogous to an electric power grid. This architecture of Grid Computing can be used to solve the biggest and the most complicated computations which may be impossible to be solved or it will takes years and years to finish by a single computer or by a single institution. In other words, a complicated computation can be divided into simpler computations which can be done parallelly by different computers on Grid. Obviously, the technical advances in Computer and Network technologies lead to a new trend of software development in the telecommunication network management industry. The rapid growth in the field of embedded computing as mobile devices creates substantial opportunities for network computing. Over 90% of all processors are sold for embedded use. Mobile devices with limits of computing power, memory capacity or battery capacity are unable to do complex computing. However mobile devices can be connected to networks, then it can be connected to computing utilities through computer networks [Logethran et al. 1998]. Running software through computer networks is a typical software application. Users can use any software via a graphical user interface without installing. This approach has several benefits:

- *Universal user interface on every system*: only one graphical user interface can be accessed to software via Web browsers for all systems.
- *Portability*: all computations are done on the centralized facilities which are often multi-core servers therefore multiple users can remotely access to software at the same time from anywhere by Web browsers.
- *Intellectual protection*: software is a form of intellectual property need to be protected. Copyright violation becomes one of the biggest issues of software companies in recent years. This situation creates a hurdle for development of many software companies. Network computing is one of the choices to solve this problem by allowing users to use software but not own any software version. Therefore, it limits Copyright violation.
- *Legacy software*: old software which can't be compatible with a new platform can be reused by running it in a dedicated environment on the server while its

new graphical user interface is used as a tool to interact with users on new systems for which the particular application is not available.

- *Scalability*: Computing Networks can be deployed and scaled very quickly which is one of the key factor for success of business and development of technology.
- *Computing power*: software applications are located on a server which is a model of super computers, therefore it will improve computing speed. Once a multi-core desktop with a hundred of processors has not become realistic yet, network computing is a good choice to save resources.
- *Compatibility*: software can interact with any platform. It is independent of the operation systems, users do not have to set up or configure software unless it is implemented on the server in the form of stored user profile.

For this particular application in this paper, one of the big issues need to be addressed is how users can control the simulation process and how multiple users can use it at the same time without overloading the system.

2 Overview of Network Programming Technologies

HTML alone does not provide the functionality needed for a dynamic, interactive environment. Additional technologies are used to implement the dynamic behavior both on the client side (inside a Web browser) and on the server side to render a Web page dynamically. Most common network programming tools used for developing dynamic websites on the client side are JavaScript, Ajax extension to JavaScript, and Java or ActiveX applets. Most common tools on the server side are PHP, Sun Microsystems' Java Server Pages, Java Servlets, Microsoft Active Server Pages (APS) technology, and Common Gateway Interface (CGI) scripts using scripting languages such as PERL, server-side JavaScript, ActiveX and Python, or precompiled binary programs [Malinowski and Wilamowski 2001].

The internet bandwidth is significantly improved with time and already adequate for network computing. However, in order to make applications more robust, the data flow should be designed effectively. The key issue is to solve problems associated with a new way of software development so that software application will be possible through the Internet and Intranet. Therefore task partitioning is very important. Which parts of software should be done on the client machine or the server machine. To get this goal needs to satisfy some requirements as:

- Minimization of the amount of data exchange through computer networks to reduce network traffic
- Task partitioning between the server and the client needs to be effective
- Selection of suitable programming languages used for various tasks
- User interfaces have to be friendly

- Use of multiple servers distributed around the world and job sharing among them to avoid overload for one server
- Security and account have to be safe
- Portability of software used on the servers and the clients
- Other

The next section will discuss two examples using computer networks as an interface to enhance interaction between the client and the server.

3 Examples of Computations through Computer Networks

3.1 Neural Network Trainer

The artificial neural network (ANN) applications are gradually increasing in last couple years. The ANNs are widely applied in the fields of VLSI [Cameron and Murray 2008] [Indiveri et al. 2006], image processing, control systems, prediction, etc. ANNs showed their potential power for a lot of real applications but it is so frustrating to train ANNs successfully. The challenges of this success are how to design a good architecture and a suitable algorithm to train ANNs. Because of this purpose many training algorithms are introduced for ANNs in order to attain faster speed as well as increase success rate. Even though Error Back Propagation (EBP) is considered as the most popular training algorithm of ANNs [Ruhmelhart et al. 1986], it is not an efficient training algorithm because of its slow convergence and inability to handle complicated problems. Many advanced algorithms have been developed lately as gradient descent, conjugate gradient descent, Levenberg Marquardt, Neuron by Neuron (NBN), etc and gave better results. For example, NBN can train ANNs 100 times faster than EBP. With big networks this NBN algorithm has its limitation and its advantages diminish because NBN algorithm requires more computations in each iteration. For all these algorithms storage and computational requirements are different, some are good for this application but not good for the others. It means that it is difficult to find a particular training algorithm that can be best for all applications under all conditions. This paper does not attempt to analyze differences as well as advantages or disadvantages of algorithms. Instead it will introduce a new ANNs training tool which includes in both first order and second order methods and also handles arbitrarily connected neural networks that are not found in the existing trainer as MATLAB Neural Network Toolbox or Stuggart Neural Network Simulators (SNNS) [WWW 2002].

Neural network trainer NBN 2.0 is developed based on Visual Studio 6.0 using C++ language hosting on the server and communicating with the clients through PHP scripts [Hao and Wilamowski 2009]. Its main interface is shown in Fig. 1.

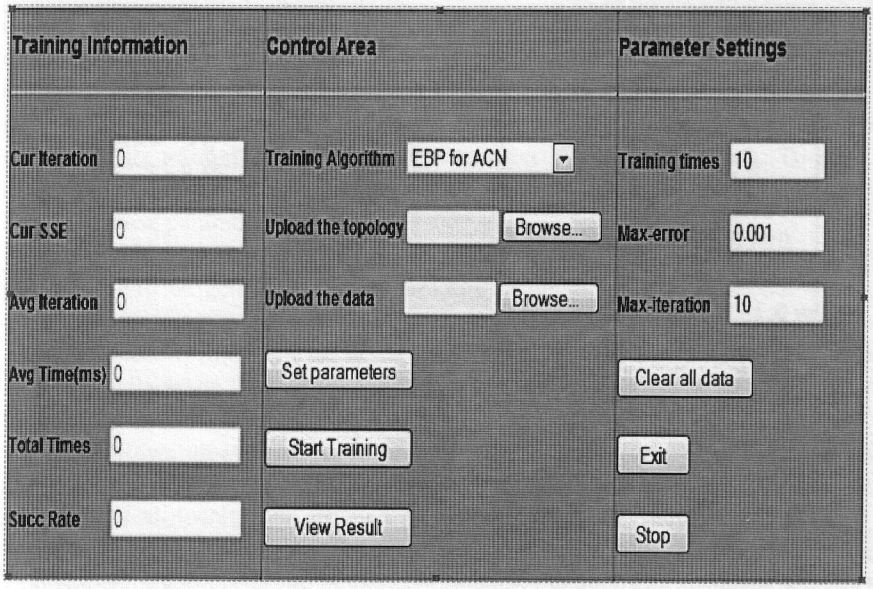

Fig. 1 Neural network trainer interface

NBN 2.0 is developed with four different types of training algorithms:

- Error Back Propagation for arbitrarily connected neuron (EBP for ACN)
- Levenberg Marquardt for multilayer perceptron (LM for MLP)
- Neuron By Neuron (NBN)
- Neuron By Neuron, forward-only (NBN- forward only)

To use this tool users have to upload two files: one topology file and one data file through the neural network interface Fig. 1. As mentioned earlier, this trainer can handle arbitrarily connected networks and it uses the similar solution as Net-list in the SPICE program. These two files have to follow certain syntax so that the training tool can make sense in the correct way.

- Topology file

The topology files are named "*.in". They are mainly used to construct neural network topologies for training. The topology files consist of four parts: topology design, weight initialization (optional), neuron type instruction and training data specification. The topology design is aimed to create ANN structures. Each line in the topology file is "n [b] [neuron type] [a1 a2 ... an]", which means the input neurons indexed with a1, a2,..., an are connected to the output neuron b with a specified neural type (bipolar, unipolar or linear). Fig. 2 presents the topology file for the neural network parity-3.

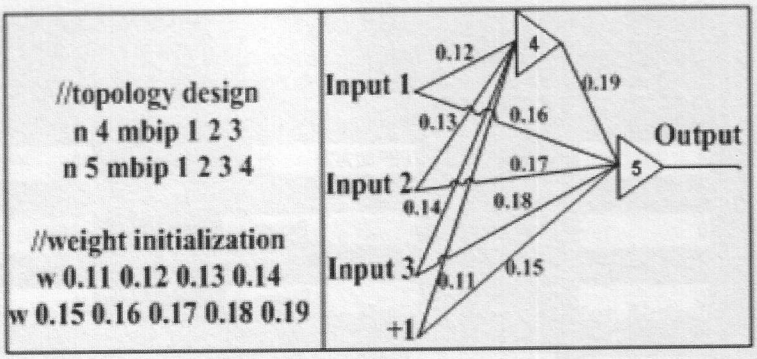

Fig. 2 Weight initialization for parity-3 problem with 2 neurons in FCN network

- Training pattern file

The training pattern files consist of input patterns and related desired outputs. In a training pattern file, the number of rows is equal to the number of patterns, while the number of columns is equal to sum of the number of inputs and outputs for each pattern. However, only with the numerical data in a training pattern file, one can't tell what number of inputs and outputs, so the neural topology should be considered together in order to decide those two parameters (Fig. 3). The training pattern files are specified in the topology files as mentioned above, and they should have the same route as the related topology files.

training data	*topology*	*explanation*
-1 -1 -1 -1 -1 -1 1 1 -1 1 -1 1 -1 1 1 -1	//2 inputs and 2 outputs n3 mbip 1 2 n4 mbip 1 2	The first command line of topology shows that there are 2 inputs, since there are 4 columns in training data, so the number of output is 2.
1 -1 -1 1 1 -1 1 -1 1 1 -1 -1 1 1 1 1	// 3 inputs and 1 output n4 mbip 1 2 3 n5 mbip 1 2 3 4	The first command line of topology shows that there are 3 inputs, since there are 4 columns in training data, so the number of output is 1.

Fig. 3 Get the number of inputs and the number of outputs from the data file and topology

Besides these two files, there are still couple parameters need to be input from users as *training times* which defines how many times ANNs need to be trained, *max-iteration* which is how many times the same process need to be repeated to update ANNs weights for one training time and *max-error* is an acceptable error limit which is supposed that ANNs will perform well compared with desired outputs. Along with these parameters, there are some other parameters which is

defined as tuning parameters for each algorithm such as *"combination coefficient"*, *"scale constant"*, *"momentum constant"*, *"learning constant"*, *"alpha constant"*, *"beta constant"*, *"gamma constant"*. In order to train ANNs successfully or speed up training process, users are required to tune these parameters to make sure its training process converge and get higher success rate. To reduce data exchange between the client and the server, the neural network interface will check all these parameters on the client side.

During training process the output results will be updated. After training, one result file will be generated which contains all detail information about training algorithm, training pattern file, topology, parameters, initial weights, resultant weights. These results will be saved automatically in database system. Training ANNs usually takes long time, with some big networks it can take thousands of training times or thousands of iterations, therefore this tool is designed in such a way that allows users to stay offline while the training process is running. This approach is very effective when users try to train ANNs by their mobile devices with limited battery capacity.

Another issue of this tool is how to create a friendly graphical user interface with control functions as stop/continue simulation when using software over computer networks. With installed software, these functions are strongly supported by operating systems so it is not a big issue. Using software over Internet is different and is only supported by interactions between the client and server. Because of this reason, software is designed in a different way. Software should have extra control function which has ability to receive requests from the client. JavaScript has certain limitations due to the security model of its implementation by a Web browser. One of those limitations is the inability to retrieve data on demand dynamically from the server. Ajax technology is a solution that allows a JavaScript program embedded inside a Web page to retrieve additional documents or Web pages from the server, store them as local variables, and parse them in order to retrieve data and use it for dynamic alteration of the Web page where the JavaScript is embedded. In this trainer, two buttons "***Stop***" "***Exit***" are used to stop training. The way it works is when a client sends a message to the trainer through the user interface, the trainer will recognize this authoritative message, stop training and send results up to that point back to a client.

In future when Cloud Computing becomes 4th paradigm, software applications can be developed directly on Cloud Operating system as Windows Azure. The approach to design software that is described here may be unnecessary. The Cloud operating system will support all these control functions as any operating systems that are being used.

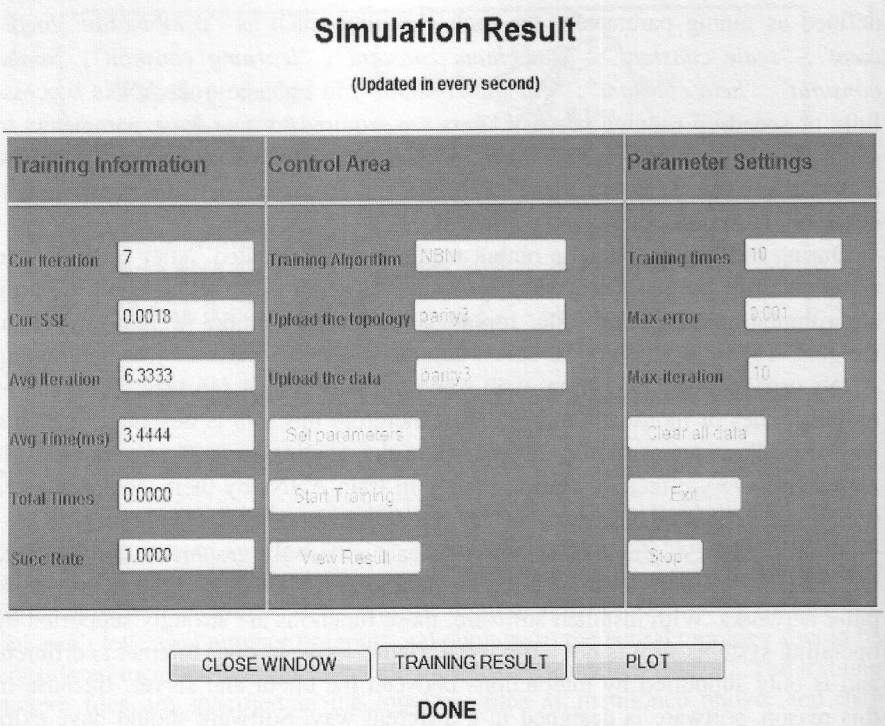

Fig. 4 Training result interface

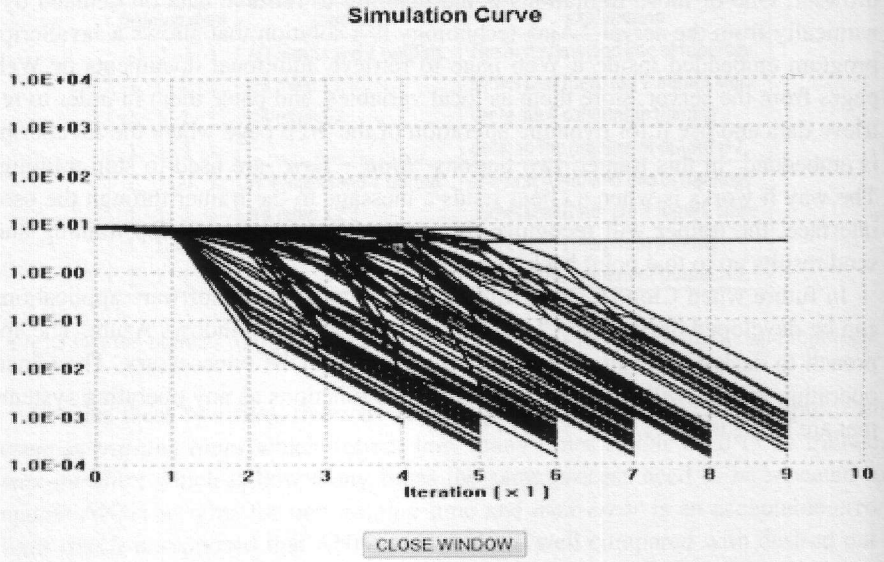

Fig. 5 Output figure

```
Parameters
NBN mu = 0.01000000 scale = 10.00000000
Data File: parity3.in
Topology
3  1  2
4  1  2  3
5  1  2  3  4
Neurons
biplor gain=1.00, der=0.01
biplor gain=1.00, der=0.01
linear gain=1.00, der=0.05
Initial Weights
-0.28000000  0.58000000  0.16000000
0.92000000  0.46000000  -0.08000000  0.28000000
-0.82000000  -0.46000000  0.94000000  -0.62000000  -0.34000000
Results Weights
-0.21352129  0.08990222  0.26889254
-8.09987819  3.40040394  11.61576715  -85.00550976
3.57991511  -1.46565558  -8.29415476  58.78724502  5.95297590
Training Results
Total iteration: 300  Total error: 0.42256835  Training Time: 766
```

Fig. 6 Training result file

3.2 SIP Program

The Spice program implemented through computer network is another example of network computing. The Spice program is the popular software which is widely used to simulate the Integrated Circuits in electronic courses. This is the licensed software which is only available for some machines on campus. It means that students who don't live on campus have to depend on these available machines. Students can have the free version of Spice program but this version can only simulate circuits with limited number of transistors which is often not good enough to run simulation of the Integrated Circuits in electronic courses. In order to make it possible for students to learn electronic courses, the SIP program was developed for this purpose (Fig. 7). The SIP uses Spice3f5 from Berkeley and can simulate the Integrated Circuits with unlimited number of transistors. Users can upload and edit circuit files which have similar formats as capture of the Spice program from MicroSim. After simulation, the SIP program will display results and analysis of simulated data in the form of images or texts [Wilamowski et al. 1998] [Wilamowski et al. 2000]. A unique feature of the SIP versus other Spice simulators is that it is operating system independent. Anyone can access and run a simulation and view results graphically from anywhere with a Web browser via Internet or Intranet.

To start SIP program, users have to select "***View/Edit***" button to enter a circuit file and then save this file by pressing "***SAVE CHANGES***" button. For example after simulating NPN-PNP amplifier circuit Fig.8 the output window will pop up as Fig.9. SIP program can analyze the Integrated Circuits in three different modes: Transient analysis, DC analysis and AC analysis and has some options to display the node analysis as in Spice version of MicroSim.

With the recent technological advances in network speed and network programming, any computation can be done on the server side or the client side only. In order to develop an effective network application, some issues need to be stressed.

- Reduce data exchange between the client side and the server side by deciding which tasks should be done on the client side or the server side.
- Save bandwidth of computer networks to avoid overloading. Because server is still limited by the number of login users at the same time and scalability is still a solving issue, so all computations need to be robust and effective.
- Select the right technology to maintain the ownership of intellectual property when Copyright violation becomes more concerned.

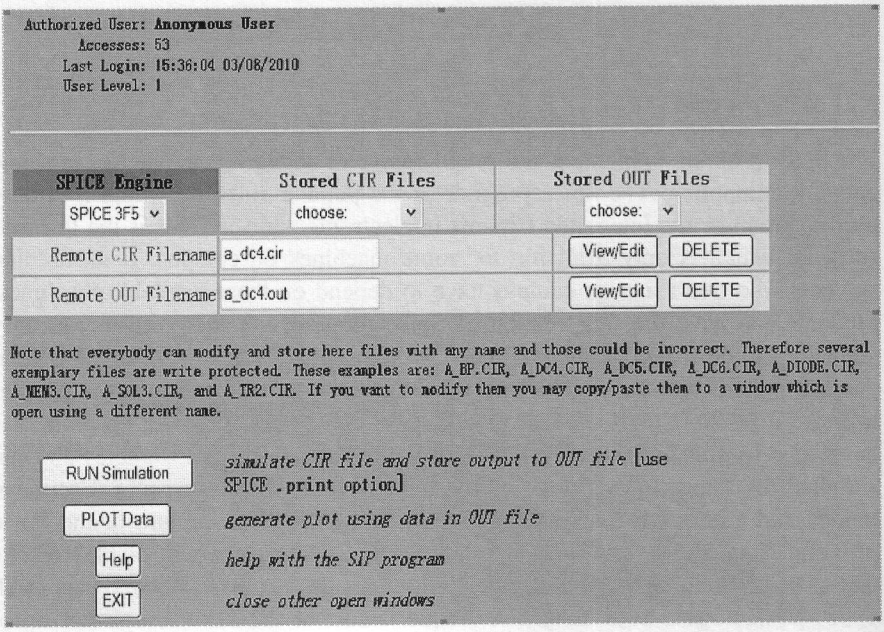

Fig. 7 SIP main interface

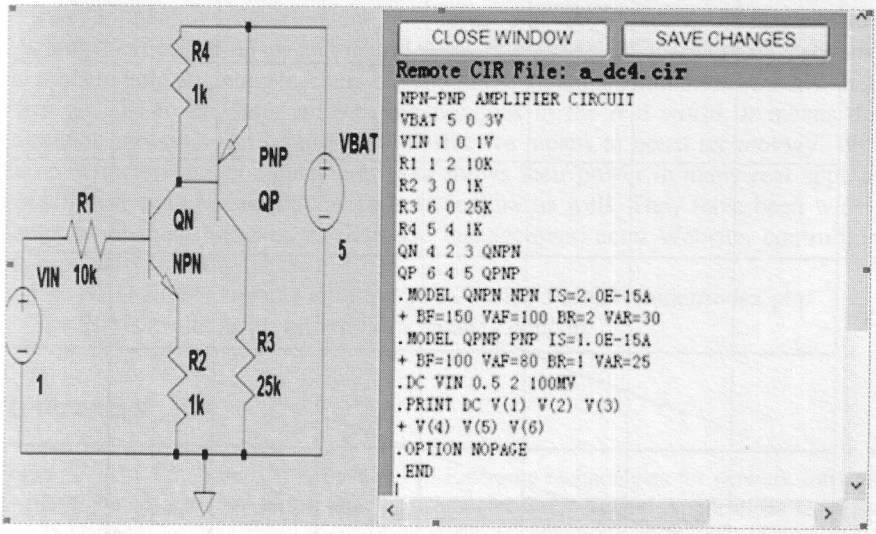

Fig. 8 Schematic and circuit file

NPN-PNP AMPLIFIER CIRCUIT
DC transfer characteristic Mon Mar 22 13:03:18 2010

Index	sweep	v(4)	v(5)	v(6)
0	5.000000e-001	2.999509e+000	3.000000e+000	1.528497e-007
1	6.000000e-001	2.986426e+000	3.000000e+000	1.524884e-007
2	7.000000e-001	2.932099e+000	3.000000e+000	1.512295e-007
3	8.000000e-001	2.856962e+000	3.000000e+000	1.551910e-007
4	9.000000e-001	2.774824e+000	3.000000e+000	3.011039e-007
5	1.000000e+000	2.689688e+000	3.000000e+000	4.263887e-006
6	1.100000e+000	2.602510e+000	3.000000e+000	1.198754e-004
7	1.200000e+000	2.514241e+000	3.000000e+000	3.627945e-003
8	1.300000e+000	2.425360e+000	3.000000e+000	1.120403e-001
9	1.400000e+000	2.339270e+000	3.000000e+000	2.896294e+000
10	1.500000e+000	2.318968e+000	3.000000e+000	2.965063e+000
11	1.600000e+000	2.307525e+000	3.000000e+000	2.973980e+000
12	1.700000e+000	2.298912e+000	3.000000e+000	2.977156e+000
13	1.800000e+000	2.292466e+000	3.000000e+000	2.977441e+000
14	1.900000e+000	2.287252e+000	3.000000e+000	2.977870e+000

elapsed time since last call: 0.000 seconds.

Total elapsed time: 0.000 seconds.

Current dynamic memory usage = 0,
Dynamic memory limit = 0.

Fig. 9 Simulation output

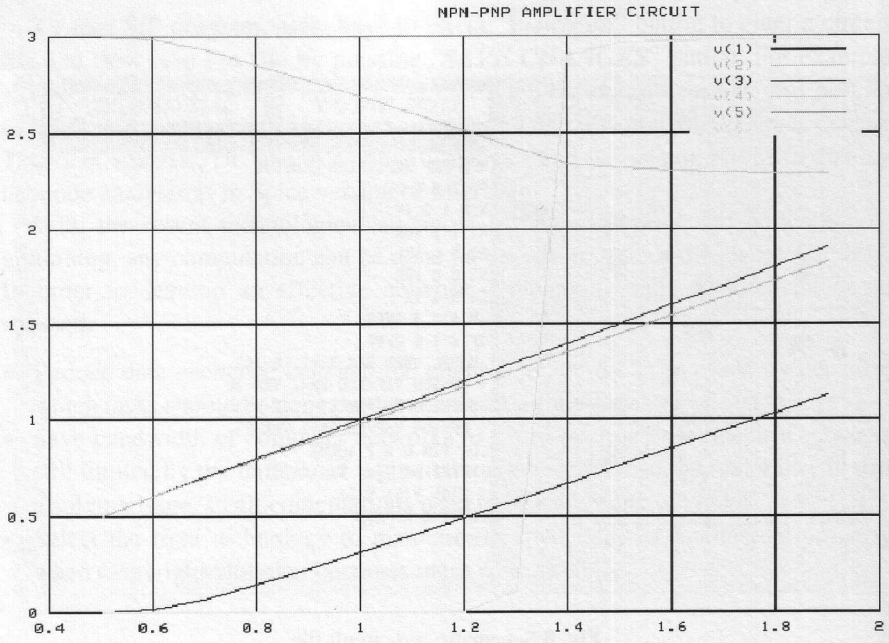

Fig. 10 Output analysis

SIP is a good example to optimize network traffic by task partitioning between the client side and the server side. In case of ANNs trainer users don't have to inspect or analyze data with repeated times, so it is better to generate a graphical image, a result file and send to users. With SIP, users frequently inspect and analyze data many times as changing variables to display voltage nodes or scaling voltage ranges to display, etc Fig.10. In this case there are many requests for different plots of the same data, therefore it could be better to send the data once together with a custom Java applet which could display the same information in many different forms without further communicating with the server.

4 Discussion and Conclusions

This paper shows two examples of how software applications can be implemented through computer networks when network technologies and multi-core processor technologies are advancing. This paper also stresses some reasonable benefits to deploy software applications on computer networks as well as some issues need to be considered to design applications. There are other existing models of computing that are described briefly throughout this paper as Grid Computing, Cloud Computing, etc. They can become the computing models of future because of its computing power, its economic effectiveness and its scalability, etc. Grid Computing models are applied widely in U.S.A Universities to do research about nuclear,

atom and other physics problems. Cloud Computing begins its first steps to deploy Web applications and others.

Two applications described in this paper are very effective in learning and teaching electronic courses and neural network courses. They are not only the useful tools to help students, teachers or scholars to do some simulations but also help them get closer and familiar with technologies in the real world. It means that computer networks can be used as an effective means to boost technology. With these characteristics computer networks shows their power in many real applications in last century and maybe in next century as well. They have been widely deployed in many systems as database management, econ Website, controlling, etc especially in computation.

The NBN 2.0 is available at: http://131.204.128.91/NNTrainer/index.php

The SIP is available at: http:/gdansk.bradley.edu/sip/

References

[Arano et al. 1996] Arano, T., Aoyama, M.: Emerging technologies for network software development: past, present, future. In: Computer Software and Applications Conf., p. 428 (1996)

[Cameron and Murray 2008] Cameron, K., Murray, A.: Minimizing the Effect of Process Mismatch in a Neuromorphic System Using Spike-Timing-Dependent Adaptation. IEEE Trans. on Neural Networks 19(5), 899–913 (2008)

[Hao and Wilamowski 2009] Yu, H., Wilamowski, B.M.: Efficient and Reliable Training of Neural Networks. In: Proc of IEEE Human System Int. Conf., Catania, Italy, pp. 109–115 (2009)

[Indiveri et al. 2006] Indiveri, G., Chicca, E., Douglas, R.: A VLSI array of low-power spiking neurons and bistable synapses with spike-timing dependent plasticity. IEEE Trans. on Neural Networks 17(1), 211–221 (2006)

[Juan et al. 2007] Juan, A.A., Faulin, J., Marques, J.M., Sorroche, M.: -SAEDES: A java-based simulation software to improve reliability and availability of computer systems and networks. In: Simulation Conference, pp. 2285–2292 (2007)

[Logethran et al. 1998] Logethran, A., Pratiwadi, R., Logenthiran, D., Porebski, A., Thomas, D.W.: Software migration of telecommunication network management systems to the Web using CORBA and Java System Sciences. In: Proc. of the Thirty-First Hawaii Int Conf., Kohala Coast, HI, pp. 637–644 (1998)

[Malinowski and Wilamowski 2000] Malinowski, A., Wilamowski, B.M.: Web-based C++ compilers. In: ASEE, Annual Conference, St. Louis, MO, CD-ROM session 2532 (2000)

[Malinowski and Wilamowski 2001] Malinowski, A., Wilamowski, B.M.: Internet technology as a tool for solving engineering problems. In: The 27th Annual Conf. of the IEEE Industrial Electronics Society (tutorial), Denver CO, pp. 1622–1630 (2001)

[Manic et al. 2002] Manic, M., Wilamowski, B.M., Malinowski, A.: Internet based neural network online simulation tool. In: Proc. of the 28th Annual Conf. of the IEEE Industrial Electronics Society, Sevilla, Spain, pp. 2870–2874 (2002)

[Ruhmelhart et al. 1986] Ruhmelhart, D.E., Hinton, G.E., Wiliams, R.J.: Learning representations by back-propagation errors. Nature 323, 533–536 (1986)

[Wilamowski and Malinowski 2001] Wilamowski, B.M., Malinowski, A.: Paper collection and evaluation through the internet. In: Proc. of the 27th Annual Conf. of the IEEE Industrial Electronics Society, Denver CO, pp. 1868–1873 (2001)

[Wilamowski et al. 1998] Wilamowski, B.M., Regnier, J., Malinowski, A.: SIP- spice intranet package. In: IEEE Int. Conf. on Industrial Electronics, 192–195 (1998)

[Wilamowski et al. 2000] Wilamowski, B.M., Malinowski, A., Regnier, J.: SPICE based circuit analysis using web pages. In: ASEE 2000 Annual Conf., St. Louis, MO (CD-ROM session 2520, 2000)

[WWW 2002] Stuttgart Neural Network Simulator (May 2002), `http://www-ra.informatik.uni-tuebingen.de/SNNS/` (accessed April 10, 2010)