VYTAUTAS MAGNUS UNIVERSITY
INSTITUTE OF MATHEMATICS AND INFORMATICS

Sergėjus IVANIKOVAS

THE PROBLEMS OF PARALLEL COMPUTING IN MULTIDIMENSIONAL DATA VISUALIZATION

Summary of Doctoral Dissertation

Physical Sciences (P 000)
Informatics (09P)
Informatics, Systems Theory (P 175)

Vilnius, 2009

Doctoral dissertation was prepared at the Institute of Mathematics and Informatics in 2005–2009.

Scientific Supervisor

Prof Dr Habil Gintautas DZEMYDA (Institute of Mathematics and Informatics, Physical Sciences, Informatics – 09 P).

The dissertation is being defended at the Council of Scientific Field of Informatics at Vytautas Magnus University:

Chairman

Prof Dr Habil Vytautas KAMINSKAS (Vytautas Magnus University, Physical Sciences, Informatics – 09 P).

Members:

Prof Dr Romas BARONAS (Vilnius University, Physical Sciences, Informatics – 09 P),

Prof Dr Habil Algimantas KAJACKAS (Vilnius Gediminas Technical University, Technological Sciences, electricity and electronics engineering – 01 T),

Prof Dr Habil Kazys KAZLAUSKAS (Institute of Mathematics and Informatics, Physical Sciences, Informatics – 09 P),

Prof Dr Habil Antanas ŽILINSKAS (Institute of Mathematics and Informatics, Physical Sciences, Informatics – 09 P).

Opponents:

Dr Olga KURASOVA (Institute of Mathematics and Informatics, Physical Sciences, Informatics – 09 P),

Prof Dr Dalius NAVAKAUSKAS (Vilnius Gediminas Technical University, Technological Sciences, Informatics Engineering – 07 T).

The dissertation will be defended at the public meeting of the Council of Scientific Field of Informatics in the auditorium number 203 of the Institute of Mathematics and Informatics at 1 p. m. on January 20 2010.
Address: Akademijos str. 4, LT-08663 Vilnius, Lithuania.

The summary of the dissertation was distributed on 20 December, 2009.
A copy of the doctoral dissertation is available for review at the M. Mažvydas National Library of Lithuania, the Library of the Vytautas Magnus University and at the Library of Institute of Mathematics and Informatics

VYTAUTO DIDŽIOJO UNIVERSITETAS
MATEMATIKOS IR INFORMATIKOS INSTITUTAS

Sergėjus IVANIKOVAS

LYGIAGREČIŲ SKAIČIAVIMŲ TAIKYMO DAUGIAMAČIAMS DUOMENIMS VIZUALIZUOTI PROBLEMOS

Daktaro disertacijos santrauka

Fiziniai mokslai (P 000)
Informatika (09 P)
Informatika, sistemų teorija (P 175)

Vilnius, 2009

Disertacija rengta 2005–2009 metais Matematikos ir informatikos institute.

Mokslinis vadovas

prof. habil. dr. Gintautas DZEMYDA (Matematikos ir informatikos institutas, fiziniai mokslai, informatika – 09 P).

Disertacija ginama Vytauto Didžiojo universiteto Informatikos mokslo krypties taryboje:

Pirmininkas

prof. habil. dr. Vytautas KAMINSKAS (Vytauto Didžiojo universitetas, fiziniai mokslai, informatika – 09 P).

Nariai:

prof. dr. Romas BARONAS (Vilniaus universitetas, fiziniai mokslai, informatika – 09 P),

doc. habil. dr. Algimantas KAJACKAS (Vilniaus Gedimino technikos universitetas, technologijos mokslai, elektros ir elektronikos inžinerija – 01 T),

prof. habil. dr. Kazys KAZLAUSKAS (Matematikos ir informatikos institutas, fiziniai mokslai, informatika – 09 P),

prof. habil. dr. Antanas ŽILINSKAS (Matematikos ir informatikos institutas, fiziniai mokslai, informatika – 09 P).

Oponentai:

dr. Olga KURASOVA (Matematikos ir informatikos institutas, fiziniai mokslai, informatika – 09 P),

prof. dr. Dalius NAVAKAUSKAS (Vilniaus Gedimino technikos universitetas, technologijos mokslai, informatikos inžinerija – 07 T).

Disertacija bus ginama viešame Informatikos mokslo krypties tarybos posėdyje 2010 m. sausio mėn. 20 d. 13 val. Matematikos ir informatikos institute, 203 auditorijoje. Adresas: Akademijos g. 4, LT-08663 Vilnius, Lietuva.

Disertacijos santrauka išsiuntinėta 2009 m. gruodžio 20 d.
Disertaciją galima peržiūrėti M. Mažvydo nacionalinėje bibliotekoje, Vytauto Didžiojo universiteto ir Matematikos ir informatikos instituto bibliotekose.

General Characteristic of the Dissertation

The research area and the topicality of the problem. The research area of this work is the analysis of multidimensional data, the investigation of visualization algorithms and the development of parallel realizations of these algorithms. The real world is represented by plenty of data which must be analysed and evaluated. Multidimensional data are constantly met in medicine, technics, economics and other fields. Nowadays technologies make it possible speed up the processing of multidimensional data using high performance computers and parallel computing.

The data should be presented in a really intelligible form if we want to perceive multidimensional data. One of the ways to present multidimensional data is to visualize them. The main idea of data visualization is to present data in such a form which would allow a consumer to interpret them. The work analyses artificial neural network based algorithms for visualizing multidimensional data.

It is easier to recognize data structure and connections between the data points when the multidimensional data are transformed into two or three dimensional space. However while transforming multidimensional data into smaller dimensional space the visualization distortion and projection errors are unavoidable. Thus the main **problem** investigated in this dissertation is the minimization of the projection error of multidimensional data while using artificial neural networks.

The aim and the tasks of the work. The key aim of the work is to develop neural network based multidimensional data visualization methods which efficiently warrant the multidimensional data projection errors minimization and to speed up the artificial neural network training process. To achieve the aim it was necessary to solve the following tasks: 1) to analyse the methods of multidimensional data visualization; 2) to investigate the possibility of speeding up the neural network training process while visualizing multidimensional data; 3) to investigate the influence of the neuron activation function parameters on the network training process; 4) to investigate the possibility of training the SAMANN network by using a part of the analysed data set or using a specially created reduced training data set; 5) to analyse the principles of the SAMANN algorithm and the possibility to parallelise it; 6) to investigate the technological possibility of parallelisation (clusters, SMP computers, Hyper-Threading technology); 7) to create the parallel realizations of the SAMANN algorithm.

The research **methodology** is based on the development of new strategies for the SAMANN neural network training and their experimental investigations.

Research subject. The research subject of the dissertation is artificial neural networks for multidimensional data projection. The main topics closely related to this subject are: 1) multidimensional data visualization; 2) dimensionality reduction (projection) algorithms; 3) feed-forward neural networks; 4) self-organising maps; 5) clustering algorithms; 6) multidimensional data projection errors; 7) projection of the new data; 8) parallel and distributed computing.

The scientific novelty and the defence propositions. A parallel realization of the SAMANN algorithm for multidimensional data projection has been created. This algorithm allows to visualize larger data sets and to perform the process of visualization of multidimensional data quicker than the serial algorithm.

Hyper-Threading technology allows using the computer hardware more effectively. However this technology is not effective while using it for parallel SAMANN algorithm.

It has been experimentally established how to select the value of the SAMANN neural network neuron activation function sloop parameter so that the algorithm would work efficiently.

The work has proposed the strategy for creating a reduced neural network training data set. The use of such a training data set makes the neural network training process faster and the obtained projection errors are not worse than the ones that are obtained while training a neural network using all the analysed data set during the same time.

The approbation and the publications of the research. The main results of this dissertation were published in 6 scientific publications: 1 article in the periodical scientific issue from the ISI Web of Science list; 2 articles in the periodical scientific issues from the ISI Proceedings list; 1 chapter in the reviewed book (Springer) and 2 articles in the proceedings of scientific conferences included into the ISI Proceedings list.

The main results of the work have been presented and discussed at 4 international and 1 national conferences.

The scope of the scientific work*.* The work is written in Lithuanian. It consists of 6 chapters, and the list of references. There are 104 pages of the text, 49 figures, 7 tables and 98 bibliographical sources.

1. Introduction

This chapter describes the research subject, the relevance of the problem, the scientific novelty of the results. Also the objectives and tasks of the work are formulated in this chapter.

2. Multidimensional Data Projection Methods

The chapter is devoted to the review of the various multidimensional data projection methods. Multidimensional data, it means the data that require more than two or three dimensions to represent, can be difficult to interpret. One of the main strategies used to handle very high dimensional data is its dimensionality reduction. The task is to reduce the dimensionality of the data to two or three for data visualization. A large number of different projection methods have been developed for this task. There are linear and nonlinear projection methods. The principal component analysis, the projection pursuit present the linear projection methods; while the multidimensional scaling, the principal curves, the triangulation, the ISOMAP present the nonlinear ones. The use of nonlinear projection methods allows preserving a more precise data structure. Nevertheless, the projection errors are inevitable. So it is necessary to look for the ways of minimizing these projection errors.

One of multidimensional scaling methods to map a high-dimensional space onto a space of lower dimensionality is Sammon mapping. Suppose that we have *m* data points, $X_i = (x_{i1}, x_{i2} \ldots, x_{im})$, $i = 1, \ldots, n$, in a *n*-dimensional space and, respectively, we define *m* points, $Y_i = (y_{i1}, y_{i2} \ldots, y_{im})$, $i = 1, \ldots, d$, in a *d*-dimensional space (*d*<*n*). The pending problem is to visualize these *n*-dimensional points $X_i, i = 1,...,n$ onto the plane R^2. Let d_{ij}^* denote the distance between X_i and X_j in the input space, and d_{ij} denote the distance between the corresponding points Y_i and Y_j in the projected space. The Euclidean distance is frequently used. The projection error measure *E* (so-called Sammon's stress) is as follows: $E = \frac{1}{\sum\limits_{i,j=1;i<j}^{m} d_{ij}^*} \sum\limits_{\substack{i,j=1 \\ i<j}}^{m} \frac{(d_{ij}^* - d_{ij})^2}{d_{ij}^*}$. This is the measure of how well the distances are preserved when the patterns are projected to a lower-dimensional space.

3. The Application of Artificial Neural Networks for Multidimensional Data Visualization

Artificial Neural Networks (ANN) are inspired with biology. The idea is to build systems that reproduce the structure and functioning of the brain neurons. The research in this field began in the 1940s, with the works of McCullogh and Pitts, followed by Hebb, Rosenblatt, Widrow. An Artificial Neural Network can be described as a set of interconnected adaptive units generally organized in a layered structure.

ANN are capable to visualize the multidimensional data. This chapter presents self-organizing neural networks, curvilinear component analysis implemented by neural networks and NeuroScale methods.

Mao and Jain (J.Mao, A.K.Jain, Artificial neural networks for feature extraction and multivariate data projection, *IEEE Transactions on Neural Networks*, Vol. 6, No. 2, 1995, p. 296–317) have derived a weight updating rule for the multilayer feed-forward neural network (SAMANN) that minimizes Sammon's stress using the gradient descent method. Sammon algorithm preserves all interpattern distances as well as possible. But the Sammon mapping has a drawback. It lacks generalization, which means that new points cannot be added to the obtained map without recalculating it. The SAMANN network offers the ability of generalization. New data can be projected to the map without the recalculation. The SAMANN method is investigated in this work. The SAMANN uses a feed-forward neural network where the number of input units is set to be the feature space dimension *n*, and the number of output units is specified as the extracted feature space dimension *d* (Fig. 1).

The SAMANN unsupervised backpropagation algorithm is as follows:

1. Initialize the weights randomly in the SAMANN network.
2. Select a pair of patterns randomly, present them to the network one at a time, and evaluate the network in a feed-forward fashion.
3. Update the weights in the backpropagation fashion starting from the output layer.
4. Repeat steps 2–3 a number of times.
5. Present all the patterns and evaluate the outputs of the network; compute Sammon's stress; if the value of Sammon's stress is below a predefined threshold or the number of iterations (from steps 2–5) exceeds the predefined maximum number, then stop; otherwise, go to step 2.

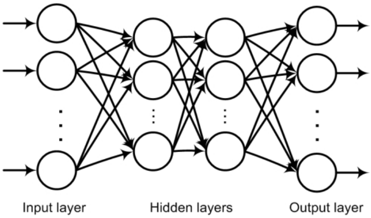

Fig. 1. Feed-forward network for Sammon's projection (SAMANN)

4. The Peculiarities of the SAMANN Neural Network Training

When projecting multidimensional data, it is very important to achieve good results in a short time interval. In the consideration of the SAMANN network, it has been observed that the projection error depends on different parameters. One of these parameters is the neuron activation function sloop parameter. Usually the sigmoid activation function *g(x)* with the range (0; 1) is used for each neuron output:

$$g(x) = \frac{1}{\left(1 + e^{-kx}\right)},$$

k is the slope parameter of the sigmoid function. By varying the parameter *k*, we obtain sigmoid functions of different slopes, as illustrated in Fig. 2. In the limit, as the slope parameter approaches infinity, the sigmoid function becomes simply a threshold function. The sigmoid function is differentiable, whereas the threshold function is not.

The sigmoid function is by far the most common form of activation function used in the construction of artificial neural networks. It is defined as a strictly increasing function that exhibits a graceful balance between linear and nonlinear behaviour. The sigmoid function features a remarkably simple derivative of the output with the respect to the input.

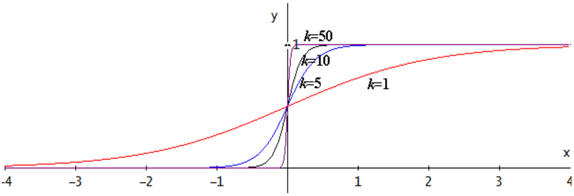

Fig. 2. Sigmoid function for varying slope parameter *k*

The general weight updating rule for all the hidden layers, $l = 1, \ldots, L-1$ and for the output layer ($l = L$) of the perceptron (J.Mao, A.K.Jain, Artificial neural networks for feature extraction and multivariate data projection, *IEEE Transactions on Neural Networks*, Vol. 6, No. 2, 1995, p. 296–317) is:

$$\Delta w_{jt}^{(l)} = -h \frac{\partial E_S(\mu, \nu)}{\partial w_{jt}^{(l)}} = -hk(\Delta_{jt}^{(l)}(\mu) y_j^{(l-1)}(\mu) - \Delta_{jt}^{(l)}(\nu) y_j^{(l-1)}(\nu)), \quad (1)$$

where $w_{jt}^{(l)}$ is the weight between the unit *j* in the layer $l-1$ and the unit *t* in the layer *l*, *h* is the learning rate, *k* is the slope parameter of the sigmoid function, $y_j^{(l)}$ is the output of the *j*th unit in the layer *l,* and μ and ν are two points from the analysed data set $\{X_1, X_2, \ldots, X_m\}$. The $\Delta_{jt}^{(l)}$ are the errors accumulated in each layer and backpropagated to a preceding layer, similarly to the standard backpropagation.

The investigations have revealed that, in order to achieve good results in training the SAMANN network, one needs to select correctly the neuron activation sloop parameter *k.* To investigate the dependence of data visualization on the neural network activation function parameters two real data sets were used: Pima Indians Diabetes data set (768 8-dimensional points from two classes) and Statlog data set (690 14-dimensional points, classes are two).

During the experiments were used different values of slope parameter *k.* The variation of the slope parameter changes the shape of the sigmoid activation function and also changes the learning rate as the learning parameter (both *h* and *k* parameters are multipliers in the formula 1). During the experiments the value of the learning rate parameter *h* was set to 5 and several values of the slope parameter were tested (*k* = 5, 10, 30). Experiments using bigger *k* values were also performed, but the results are not as good as the presented ones.

The results of the experiments on the slope parameter of the neuron activation function are presented in Fig. 3.

The experiments with both data sets showed that slope parameter enables us to increase the speed of the neural network training process. Setting the proper value of the neuron activation function sloop parameter affects the neural network training process in a better way than adjusting the learning rate parameter.

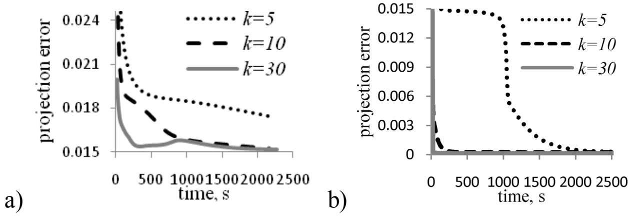

Fig. 3. The dependence of the projection error on the computation time for the (a) Pima Indians Diabetes data set and (b) Statlog data set, while using different slope parameters *k* for the neuron activation function

Another topic of this chapter is the investigation of the proper neural network training subset construction methods. During the first experiments SAMANN network was trained by a subset (i.e., a part of all points) of the analyzed data set. A fixed amount of random points from the original data set were selected for the neural network training.

Two data sets were used in the experiments: Artificial data set (966 10-dimensional points from four classes), Page blocks classification data set (5473 10-dimensional points from five classes).

The projection errors were calculated and the calculation time was measured using different subsets (approx. 50% and 30% of the initial data set) and the whole analyzed data set for the SAMANN network training. The results of the experiments are presented in Fig. 4. The obtained projection errors were smaller and training process was quicker for all tested data sets while using 50% of initial data set for training SAMANN. Training process is speeded-up even more while using 30% of the analyzed data set for training, but the obtained projection errors are not always good enough.

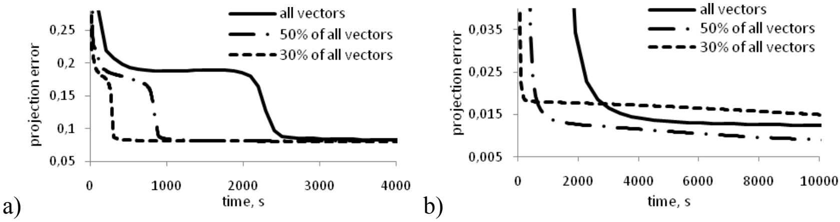

Fig. 4. Dependence of the projection error on the computation time for the Artificial data set (a) and Page blocks classification data set (b) using different number of training points

Another way to create the training subset is clustering of the analyzed data set. In this case the data visualization process, using neural network, is divided into three stages: clustering of the analyzed data set, the neural network training using only the clustered data, the visualization of the analyzed data set.

Two different clustering methods have been used for the analyzed data set reduction: *k*-means and SOM.

Using clustering for the analyzed data set reduction, the obtained training subset for SAMANN network consists of new points. Using *k*-means method, training subset is constrained by clustering the initial data set into *k* clusters. Cluster centroids are used to train the network and projection error is calculated by using the initial data set. Using SOM for the training subset construction the size of the map is defined and SOM is trained by using the initial data set. Then the SOM neurons-winners are used for the SAMANN network training. Projection error is calculated using the initial data set as previously.

Four experiments were drawn out working with the Artificial data set using *k*-means clustering method for training set construction. The neural network was trained by the initial data set (all multidimensional points) and the reduced training subsets. The training subsets of different size were used: 450 points (approx. 50% of the initial data set), 250 points (approx. 25% of the initial data set), 100 points (approx. 10% of the initial data set). Five different training sets were analyzed in the case of the Page blocks classification data set. As in the previous experiments the neural network was trained by the initial data set (all multidimensional points) and the reduced training subsets (2500 points (approx. 50% of the initial data set), 1400 points (approx. 25% of the initial data set), 900 points (approx. 15% of the initial data set), 400 points (approx. 7% of the initial data set)). The results of the experiments are presented in Fig. 5. The best results have been obtained using 25% of the initial data set for training subset in the case of Artificial data set and 7% in the case of Page blocks classification data set.

Using SOM clustering method the same tests as in case of *k*-means were performed. The neural network was trained by the initial data set (all multidimensional points) and the reduced training subsets. The reduced training subsets were composed of SOM neurons-winners. The different size of SOM was used. Working with the Artificial data set training subsets were constructed using SOM of such size: 25x25 (after the training we have got 481 neurons-winners, i.e., approx. 50% of the initial data set), 16x16 (after the training we have got 236 neurons-winners, i.e., approx. 25% of the initial data set), 10x10 (after the training we have got 95 neurons-winners, i.e., approx. 10% of the initial data set).

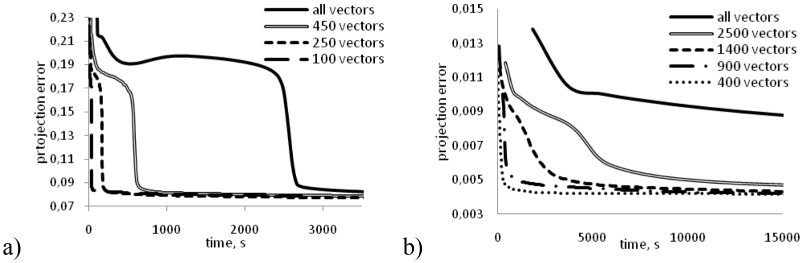

Fig. 5. The dependence of the projection error on the computation time for the Artificial data set (a) and Page blocks classification data set (b) using different size of training subset

In the case of the Page blocks classification data set the following configuration of SOM was used: 57x57 (2140 neurons-winners were obtained from this SOM, i.e., approx. 50% of the initial data set), 40x40 (1313 neurons-winners were obtained from this SOM, i.e., approx. 25% of the initial data set), 33x33 (961 neurons- winners were obtained from this SOM, i.e., approx. 15% of the initial data set), 20x20 (390 neurons- winners were obtained from this SOM, i.e., approx. 7% of the initial data set).

The results of the experiments are presented in Fig. 6. The results of the using of SOM in the case of Artificial data set are not as good as working with *k*-means. While working with Page blocks classification data set using SOM the good results were obtained while using 25% and 15% of the initial data set for the neural network training. The results obtained while using 50% and 7% of the initial data set for the neural network training are worse than the corresponding results obtained working with *k*-means. The best results were obtained using 15% of the initial data set for training subset.

Comparing *k*-means and SOM methods such conclusions can be made: *k*-means performs better working with smaller data sets; while working with large data set both methods give rather similar results.

Investigating the ability to visualize large data set using SAMANN, three real data sets were used: Page blocks classification data set (5473 10-

dimensional points from five classes), Pen-Based recognition of handwritten digits data set (10992 16-dimensional points from ten classes), MAGIC Gamma telescope data set (19020 10-dimensional points from two classes). The SAMANN network was trained by a subset of the analyzed data set to speed up the training process and to improve the precision.

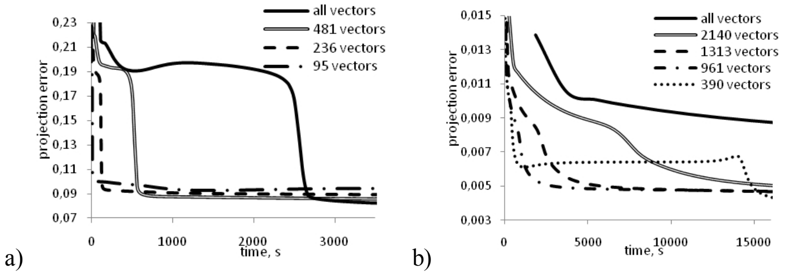

Fig. 6. The dependence of the projection error on the computation time for the Artificial data set (a) and Page blocks classification data set (b) using different size of training subset

The projection errors were calculated and the calculation time was measured using the reduced training subset and the whole analysed data set for the SAMANN network training.

Working with each data set three experiments were drawn out. The neural network was trained by the initial data set (all multidimensional points) and two reduced training data sets. In all the experiments the projection error was calculated using the initial data set. The first reduced data set was created using *k*-means clustering. The second reduced training data set was created using SOM. The size of all the reduced training data sets was approximately 10 times smaller than the size of the original data set. Running all the experiments the run time of the program was limited to approximately 20 hours.

Working with Page blocks classification data set the training process using the full data set took 20.04 hours. 116 training iterations were accomplished. The projection error was 0.1183. Using *k*-means clustering for creating the reduced training subset 7437 training iterations were accomplished during 20 hours. The projection error was 0.0365. Using SOM for creating the reduced training subset 7860 training iterations were accomplished during 20 hours. The projection error was 0.036. The results of the experiments are presented in Fig. 7.

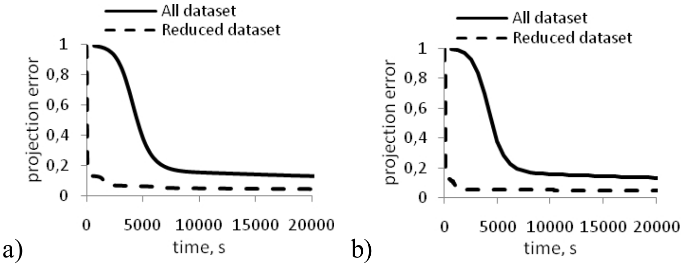

Fig. 7. The dependence of the projection error on the computation time for the Page blocks classification data set, using (a) SOM method and (b) *k*-means method

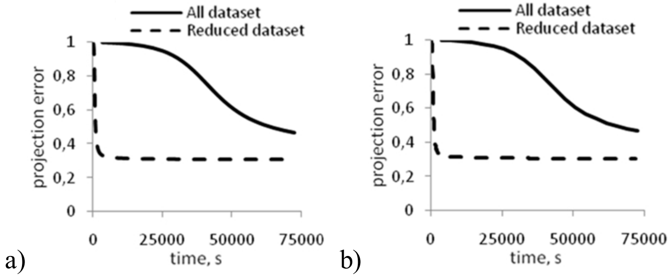

Fig. 8. The dependence of the projection error on the computation time for the Pen-Based recognition of handwritten digits data set, using (a) SOM and (b) *k*-means

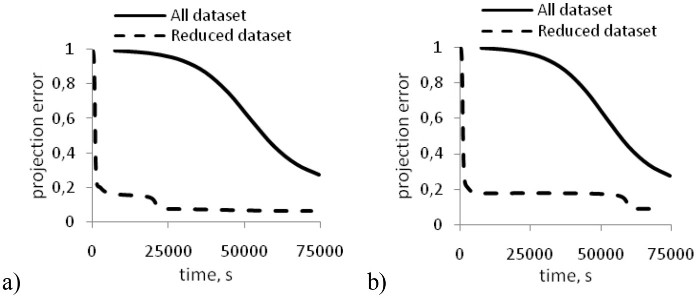

Fig. 9. The dependence of the projection error on the computation time for the MAGIC Gamma telescope data set, using (a) SOM method and (b) *k*-means method

Working with Pen-Based recognition of handwritten digits data set the training process using the full data set took 20.18 hours. Only 20 training iterations were accomplished. The projection error was 0.466. Using *k*-means clustering for creating the reduced training subset 1418 training iterations were accomplished during 20 hours. The projection error was 0.303. Using SOM for creating the reduced training subset 1482 training iterations were accomplished during 20 hours. The projection error was 0.308. The results of the experiments are presented in Fig. 8.

Working with MAGIC Gamma telescope data set the training process using the full data set took 20.65 hours. Only 10 training iterations were accomplished. The projection error was 0.2758. Using *k*-means clustering for creating the reduced training subset 623 training iterations were accomplished during 20 hours. The projection error was 0.086. Using SOM for creating the reduced training subset 618 training iterations were accomplished during 20 hours. The projection error was 0.063. The results of the experiments are presented in Fig. 9.

The results of using training subsets reduced with *k*-means and SOM methods are almost the same. In both cases good projection errors are obtained faster, than while training the neural network using all analysed data set. The advantage of SOM is that this method is faster than *k*-means clustering method. Sometimes the results of training the SAMANN using training set reduced with *k*-means gives better projection error.

6. The parallel Computing and Artificial Neural Networks

This chapter presents the review of the parallel computing. The investigation of the ability to use parallel computing for SAMANN neural network training was made. The Hyper-Treading technology used in the newest Intel processors was examined. Was tested the performance of the multi-threaded programs by using memory intensive tasks. The standard heat conduction problem was chosen to test the SMP system performance. Several criteria were used to measure the program speed-up and the efficiency of computer usage.

Speed-up: $S_p = \frac{T_1 - T_p}{T_1}$. The efficiency of algorithm: $E_p = \frac{T_1}{p \cdot T_p}$.

Where p is the number of threads used, T_1 is the serial program working time and T_p is the multi-threaded program working time using p threads.

The heat conduction type tasks are widely used. The particularity of this type of tasks is a large amount of data to be processed. Tasks like weather forecast, heat spread or heat conduction, can be qualified as the tasks of such type. In this work, was chosen the algorithm of heat conduction while edges of flat rectangular plate are heated and the temperature of the plate is calculated. Heat flows through thermally conductive materials according to the process generally known as "gradient transport".

To solve the chosen heat conduction problem the approximation of finite differences was used. There was defined a discreet net on the plate

$$w_h = \{(x_i, y_j): x_i = ih, y_j = jh, 0 \leq i, j \leq N\},$$

where N is the number of rows and columns in the matrix w_h , $h = 1/N$ is a step of the net. The solution $U_{ij} = U(x_i , y_j)$ is calculated only in nodes of the net. The dimension of the matrix w_h is $(N + 1)^2$.

In each node of the net, fluxion is calculated using the approximation of finite differences. To calculate the value of the node, four neighbouring nodes are used. So we have linear equation system

$$U_{ij} = (U_{i\text{-}1,j} + U_{i+1,j} + U_{i,j\text{-}1} + U_{i,j+1}) /4, \qquad 1\leq i,j \leq N\text{-}1.$$

The values of side nodes are found from side conditions. The linear equation system of $(N - 1)^2$ equations is solved using Jacobi iterational method. First of all, starting evaluation U_{ij}^{0} is chosen. Then iterational process is repeated until two neighbouring evaluations are close enough. The number of iterations is proportional to the amount of nodes $O(N^2)$.

Data parallelism is used in the algorithm. The same computations are performed with all the data. So, we can get the parallel algorithm by dividing the net nodes among processors. Each of the processors gets a part of the matrix and performs the iterative calculations with its local part of the data. After each iteration neighbouring processors synchronize the values of the side nodes to continue the iterative process.

Tests with serial and two threads parallel program were performed using Pentium 4 HT based system (Pentium 4 HT 3.2 GHz, 512 KB L2 cache, 512 MB RAM). These tests present that the use of two threads allows increasing the system performance up to 37% while working with a smaller amount of data and about 30% with a large amount of data using a single processor system with Hyper-Threading technology. The obtained result is close to the announced speed-up of Hyper-Threading technology. So, we can compute faster and use the possibilities of the system more effectively by creating multi-threaded applications to process a large amount of data while working with a single processor system with Hyper-Threading technology.

The tests with serial, two, three and four thread parallel program were performed using Dual Xeon Server (Dual Xeon 3.2 GHz, 1 MB L2 cache, 2GB RAM). As Intel Xeon processors also have the Hyper-Threading technology, there was a possibility to work with four threads on this system. The program working time was measured during the experiments.

The tests showed that the usage of serial program while working with the SMP system is inefficient. The same program with the same data worked faster with a single processor system. The usage of two threads working with the SMP system allows increasing the performance up to 68% (see Fig. 10). So, the efficiency of the system work increases. But even using two threads working with the SMP system and with a single processor system, programs working time is comparable.

The using Hyper-Threading technology makes it possible to work with up to 4 threads in parallel. So, using three threads the speed-up increases up to 72% and working with 4 threads program speed-up reaches up to 84% (see Fig. 10).

The efficiency of the algorithm is bigger than 1 while working with two threads (see Fig. 11). It means that using 2 threads not only the computation resources affect the speed-up, but also the optimization of memory work. Working with smaller amount of data the affect of memory optimization is rather big.

Using more than 2 threads working in parallel there is no more computation resources (there are only two physical processors in the system) but we still have a speed-up of the program. The tests showed that the usage of three threads is less effective than the usage of four threads. The amount of threads used should be multiple of 2 to achieve better results. Such amount of used threads comes because of the Hyper-Threading technology essence. This technology divides each physical processor into two logical processors. So, to optimize the work of all processors and to achieve better results it is better to use the even amount of threads.

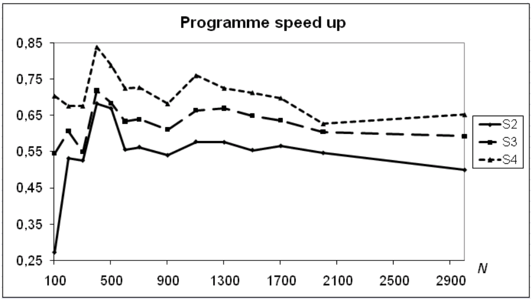

Fig. 10. Program speed-up

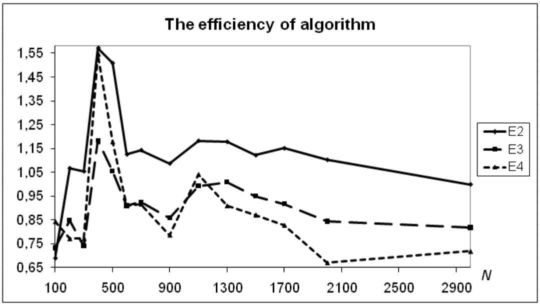

Fig. 11. The efficiency of algorithm

Sometimes the efficiency of the algorithm is bigger using 4 threads than using 3 threads working with a smaller amount of data (especially with *N* = 500). The efficiency of the algorithm using 3 or 4 threads mostly is less than 1 (see Fig. 11), so it is possible to make a conclusion, that in this case the operative memory is used more effectively.

We can notice that the speed-up coefficient decreases for the larger values of *N*. This effect occurs because of the influence of operative memory. For bigger *N* the influence of the work operative memory is less, so the speed-up coefficient decreases. For large *N* (*N*>5000) the speed-up coefficient stabilizes, the decrement of its value stops. Using 2 threads the efficiency of algorithm is less than 1 with the large matrixes.

The speed-up (as well as the efficiency of algorithm) grows till *N*=500 and then it declines. This effect occurs because of the processors cache. Each processor in the system has 1MB cache memory. Total amount of the cache for two processors is 2MB. Working with real numbers (double format in C language uses 8 bytes of memory) 500x500 elements matrix takes about 2MB of memory. So it is the biggest matrix what fits in to the 2 MB cache memory. Working with such amount of data the usage of RAM should be minimal. The particularities of the cache memory are the reason of other local maximum of curves in the Figures 10 and 11. This maximum occurs with N from 1000 till 1200. 1000 is the multiple of 500 so the usage of cache memory is the possible reason of such variations of the curves.

A drawback of using SAMANN is that the training process is extremely slow and so one of the ways of speeding up the network training process is to use the parallel computing. In this work a parallel realization of the SAMANN was proposed.

Projecting data to the two-dimensional space using SAMANN, it is natural to divide the neural network into two parts (it is also possible to divide the neural network to more than two parts). Each network layer is divided into equal or almost equal parts. Then the learning process of the network can be parallelized. Such a parallel program can be used with shared memory computers. The dual processor or dual or quad core SMP systems are popular and available for most users nowadays. The multi-threaded program can perform faster and more efficiently than the serial one. The usage of this parallel program with clusters or distributed memory systems is complicated because of data synchronization: the program will be inefficient because of the data transfer between the processors. The OpenMP standard was used to create a multi-threaded program.

The proposed parallel algorithm performs network training using two threads. The algorithm divides the SAMANN network training into serial and parallel parts. A part of the computation was performed by one thread (input of the data, normalization of the points, basic weights initialisation, Sammon's error calculation), and the other part of computation (weight renewal, the output of the net calculations) was fulfilled by two threads working in parallel. So, the iterative process of neural network training is parallelised. Each of two processors used in the algorithm performs the training of a part of neural network during each iteration.

While training a neural network, n-dimensional points are presented to the network in pairs $(X_i\ ,\ X_j)$, $i,\ j\ =\ 1,\dots\ m$, $i \neq j$, where m is the number of multidimensional points. In the parallel algorithm, two points are distributed to different processors and the outputs of neural network for these two points are calculated simultaneously. Then the network weights are updated using the error back-propagation algorithm. This process is also performed in parallel. Each processor recalculates the weights of its part of the neural network. Data are synchronized after calculating the network outputs for two points and after updating the weights of each network layer. A neural network is trained using a fixed number of iterations (one iteration is a part of the training process when all the different pairs are presented to the network once).

The usage of the parallel algorithm is not effective with a small neural network or with low-dimensional data sets because of the data synchronization process. Using low-dimensional data sets with a small number of hidden neurons, the amount of computations in each neural network learning step is rather small and the expenses of data synchronization are high.

Working with large data sets the parallel algorithm performs faster than the serial one. For example using a 60-dimensional data set and 500 hidden neurons, we have a large enough amount of computations in each neural network learning step, so the efficiency of the parallel algorithm increases. The parallel algorithm performs ~20% faster than the serial one. By increasing the dimension of the data set or the amount of neurons in the hidden layer, the efficiency of the parallel algorithm also increases.

6. General Conclusions

1. Analysing the SAMANN network it has been noticed that the network that implements Sammon’s algorithm training depends on different parameters. The experiments, fulfilled in this work, show the dependence of the network training on the neuron activation function sloop parameter. Usually the value of this parameter is set to 1. The results of the experiments have shown that the bigger value of the neuron activation function sloop parameter makes possible to obtain a better projection error and the results of visualization.
2. The experimental investigation shows that the optimal value of the neuron activation function sloop parameter is in the interval (10; 30). So by selecting such values of the neuron activation function sloop parameter, a significant economy of the computing time is possible for a fixed number of iterations (up to 3–5 or even more times).

3. The results of the experiments have presented the possibility to find such a subset in the analyzed data set by which training the SAMANN network the lower projection errors are obtained faster than by training with all the points of the set.
4. The construction method of the SAMANN network training data set was proposed. The essence of this method is the clusterization of the analyzed data set. The using of this method makes possible to construct such a training data set that the neural network training process will be much faster (more than 5 times), than while training the neural network by all the analysed data set. The obtained projection error in this case is not worse than the one obtained training the neural network by all the analysed data set.
5. The experiments showed that using the clustering methods for training data set construction makes it possible to visualize significantly larger data sets (more than 10 times larger) than while using all the analysed data set for neural network training.
6. The possibility of using multiprocessor systems and the Hyper-Threading technology for the neural network training were investigated. The Hyper-Threading technology is used in the newest Intel processors. The investigation showed that the Hyper-Threading technology allows using computer hardware more effectively especially while working with multiprocessor systems. But the peculiarities of this technology do not allow using it effectively while working with the parallel SAMANN algorithm.
7. A parallel modification of the SAMANN algorithm has been proposed. This parallel algorithm performs the neural network training dividing the network into several parts. The research has shown that the parallel algorithm allows obtaining better visualization results quicker than using the serial algorithm. The use of parallel algorithm also allows us to visualizing larger data sets.

The List of Published Works on the Topic of the Dissertation

Articles in periodicals

1. IVANIKOVAS S., DZEMYDA. G. Evaluation of the Hyper-Threading Technology for Heat Conduction-Type Problems. *Mathematical Modelling and Analysis,* 2007, Volume 12 Number 3, p. 459–468. ISSN 1392-6292 print, ISSN 1648-3510 online. The ISI Web of Science journal list.

2. IVANIKOVAS S., DZEMYDA G., MEDVEDEV V. Parallel Realizations of the SAMANN Algorithm. *Lecture Notes in Computer Science, Adaptive and Natural Computing Algorithms*, Springer, 2007, Vol. 4432, p. 179–188. ISSN 0302-9743.
3. IVANIKOVAS S., DZEMYDA G., MEDVEDEV V. Large Datasets Visualization with Neural Network Using Clustered Training Data. *Lecture Notes in Computer Science, Advances in Databases and Information Systems*, Springer, 2008, Vol. 5207, p. 143–152. ISSN 0302-9743

Chapter in the reviewed book (Springer)

4. IVANIKOVAS S., FILATOVAS E., ŽILINSKAS J. Experimental Investigation of Local Searches for Optimization of Grillage-Type Foundations. *Springer optimization and its applications. Vol. 27, Parallel scientific computing and optimization: advances and applications*, New York, Springer, 2009, p. 103-112. ISBN 978-0387-09-706-0

Articles in the scientific publications from the ISI Proceedings list

5. IVANIKOVAS S., DZEMYDA G., MEDVEDEV V. Neural network-based visualization using clustered data. *EURO Mini Conference „Continuous Optimization and Knowledge-Based Technologies“ EurOPT'2008*, Vilnius, Technika, 2008, p. 335–341. ISBN 978-9955-28-283-9
6. IVANIKOVAS S., DZEMYDA G., MEDVEDEV V. Influence of the neuron activation function on the multidimensional data visualization quality. *The XIII International Conference Applied Stochastic Models and Data Analysis ASMDA–2009*, Vilnius, Technika, 2009, p. 299–303. ISBN 978-9955-28-463-5

Short description about the author of the dissertation

1999–2003 – Studies at the Vilnius Pedagogical University, Faculty of Mathematics and Informatics – Bachelor of Mathematics (summa cum laude).

2003–2005 – Studies at the Vilnius Pedagogical University, Faculty of Mathematics and Informatics – Master of Informatics (summa cum laude).

2005–2009 – PhD studies at the Institute of Mathematics and Informatics, Systems Analysis Department.

e-mail: Ivanikovas@gmail.com.

LYGIAGREČIŲ SKAIČIAVIMŲ TAIKYMO DAUGIAMAČIAMS DUOMENIMS VIZUALIZUOTI PROBLEMOS

Tyrimų sritis ir problemos aktualumas. Šio darbo tyrimų sritis yra daugiamačių duomenų analizė, vizualizavimo algoritmų tyrimas bei tų algoritmų lygiagrečių versijų kūrimas. Realus pasaulis pateikia daugybę duomenų, kuriuos tenka analizuoti ir vertinti. Medicinoje, technikoje, ekonomikoje ir daugelyje kitų sričių nuolat susiduriama su daugiamačiais duomenimis. Šiuolaikinės technologijos leidžia pagreitinti daugiamačių duomenų apdorojimą naudojant didelio našumo skaičiavimo įrangą bei taikant lygiagrečiuosius skaičiavimus.

Duomenų pateikimas žmogui suprantama forma dažnai būna efektyvi priemonė, norint geriau suvokti daugiamačius duomenis. Vienas iš galimų būdų yra grafinis informacijos pateikimas – vizualizavimas. Pagrindinė vizualizavimo idėja – duomenis pateikti tokia forma, kuri leistų vartotojui suprasti duomenis geriau, nei kai jie pateikti lentelės pavidalu. Darbe nagrinėjami dirbtiniais neuroniniais tinklais grindžiami daugiamačių duomenų vizualizavimo algoritmai.

Transformavus daugiamačius duomenis į dvimatę arba trimatę erdvę, darosi daug paprasčiau suvokti duomenų struktūrą ir sąryšius tarp jų. Tačiau duomenis transformuojant į mažesnės dimensijos erdvę, duomenų vizualizavimo iškraipymai, paklaidos yra neišvengiamos. Daugiamačių duomenų projekcijos paklaidų minimizavimas dirbant su neuroniniais tinklais grindžiamais algoritmais yra pagrindinė šioje disertacijoje sprendžiama **problema**.

Darbo tikslas ir uždaviniai. Pagrindinis disertacijos tikslas yra išvystyti neuroniniais tinklais grindžiamus daugiamačių duomenų vizualizavimo metodus užtikrinant efektyvų daugiamačių duomenų projekcijos paklaidos minimizavimą bei pagreitinant neuroninio tinklo mokymą.

Norint pasiekti šį tikslą, reikėjo išspręsti tokius uždavinius: 1) analitiškai apžvelgti daugiamačių duomenų vizualizavimo metodus; 2) ištirti galimybes paspartinti dirbtinių neuroninių tinklų mokymąsi atliekant daugiamačių duomenų vizualizavimą; 3) ištirti neurono aktyvacijos funkcijos parametrų įtaka neuroninio tinklo mokymuisi; 4) ištirti galimybes SAMANN tinklą mokyti analizuojamos duomenų aibės dalimi arba specialiai konstruojama sumažinta mokymo aibe; 5) išanalizuoti SAMANN algoritmo veikimo principus ir šiuo algoritmo lygiagretinimo galimybes; 6) ištirti technologines lygiagretinimo galimybes (klasteriai, SMP kompiuteriai, Hyper-Threading technologija); 7) sukurti ir ištirti SAMANN algoritmo lygiagrečiąją versiją.

Tyrimų **metodikos** pagrindą sudaro naujų SAMANN neuroninio tinklo mokymo strategijų kūrimas ir jų eksperimentinis tyrimas.

Tyrimų objektas. Disertacijos tyrimų objektas yra dirbtiniais neuroniniais tinklais grindžiami daugiamačių duomenų vizualizavimo algoritmai. Su šiuo objektu betarpiškai susiję dalykai: 1) daugiamačių duomenų vizualizavimas; 2) dimensijos mažinimo (projekcijos) algoritmai; 3) tiesioginio sklidimo dirbtiniai neuroniniai tinklai; 4) saviorganizuojantys neuroniniai tinklai; 5) klasterizavimo algoritmai; 6) daugiamačių duomenų projekcijos į mažesnės dimensijos erdvę paklaidos; 7) naujų daugiamačių taškų atvaizdavimas; 8) lygiagretieji ir paskirstyti skaičiavimai.

Mokslinis naujumas ir ginamieji teiginiai. Sukurtas lygiagretusis SAMANN algoritmas, leidžiantis greičiau atlikti didelės dimensijos daugiamačių duomenų vizualizavimą. Šis algoritmas taip pat leidžia vizualizuoti didesnės apimties duomenų aibes.

Darbe nagrinėta Hyper-Threading technologija leidžia efektyviau išnaudoti kompiuterio resursus, tačiau jos naudojimas dirbant su lygiagrečiuoju SAMANN algoritmu yra neefektyvus.

Eksperimentiškai nustatyta, kaip parinkti SAMANN tinklo neuronų aktyvacijos funkcijos nuolydžio parametro reikšmę, kad algoritmas veiktų efektyviai.

Darbe yra pasiūlyta strategija, kuri leidžia sudaryti tokią neuroninio tinklo mokymosi aibę, kad tinklas apsimokytų žymiai greičiau, o gaunamos per tą patį laiką projekcijos paklaidos būtų neblogesnės už projekcijos paklaidas, mokant neuroninį tinklą visais analizuojamos duomenų aibės taškais.

Darbo rezultatų aprobavimas ir publikavimas. Tyrimų rezultatai publikuoti 8 moksliniuose leidiniuose (trys iš jų periodiniuose): 1 straipsnis leidinyje, įtrauktame į Mokslinės informacijos instituto pagrindinį (ISI Web of Science) sąrašą, 7 straipsniai leidiniuose, įtrauktuose į Mokslinės informacijos instituto konferencijos darbų (ISI Proceedings) sąrašą.

Tyrimų rezultatai buvo pristatyti ir aptarti 5 nacionalinėse ir tarptautinėse konferencijose.

Darbo apimtis. Disertaciją sudaro šeši skyriai ir literatūros sąrašas. Bendra disertacijos apimtis 104 puslapiai, 49 paveikslai ir 7 lentelės.

Pirmame skyriuje išdėstytas disertacijos temos aktualumas, tyrimų sritis, suformuluotas tyrimo tikslas, pateikti tyrimo uždaviniai, aprašytas tyrimo objektas, darbo naujumas, darbo rezultatų aprobavimas, pateiktas darbo publikacijų sąrašas, pristatyta darbo struktūra.

Antrame skyriuje pateikta daugiamačių duomenų projekcijos metodų apžvalga. Projekcijos metodų tikslas – pateikti daugiamačius duomenis mažesnės dimensijos erdvėje taip, kad būtų kiek galima tiksliau išlaikyta tam tikra duomenų struktūra, ir palengvinti didelės dimensijos duomenų interpretavimą bei apdorojimą. Skyriuje yra nagrinėjami tiesiniai ir netiesiniai projekcijos metodai.

Trečiame skyriuje analizuotos dirbtinių neuroninių tinklų galimybės vizualizuoti daugiamačius duomenis, kadangi klasikiniai vizualizavimo metodai kartais yra nepajėgūs susidoroti su savo užduotimis.

Ketvirtame skyriuje analizuota specifinė „klaidos sklidimo atgal“ mokymo taisyklė, SAMANN, kuri leidžia įprastam tiesioginio sklidimo neuroniniam tinklui realizuoti Sammono projekciją mokymo be mokytojo būdu. Buvo nagrinėjama SAMANN tinklo mokymo proceso priklausomybė nuo neurono aktyvacijos funkcijos nuolydžio parametro reikšmės. Nustatyta optimali neurono aktyvacijos funkcijos nuolydžio parametro reikšmė. Taipogi šiame skyriuje yra pasiūlytas SAMANN neuroninio tinklo mokymosi aibės sudarymo metodas, kuris leidžia žymiai greičiau apmokyti neuroninį tinklą ir gauti geras projekcijos paklaidas.

Penktame skyriuje apžvelgiamos lygiagrečios SAMANN algoritmo versijos sudarymo galimybės. Atliekamas Hyper-Threading technologijos tyrimas. Pasiūlyta lygiagrečioji SAMANN algoritmo realizacija, leidžianti tinklo mokymui vienu metu naudoti keletą procesorių. Pateikiami gauti rezultatai ir išvados.

Šeštame skyriuje pateiktos bendros disertacijos išvados.

Bendrosios išvados

1. Analizuojant SAMANN neuroninio tinklo mokymo be mokytojo „klaidos sklidimo atgal“ algoritmą, nustatyta, kaip projekcijos paklaida ir tinklo apsimokymo konvergavimas priklauso nuo pasirinktų parametrų reikšmių. Vienas iš parametrų, įtakojančių neuroninio tinklo mokymąsi yra neuronų aktyvacijos funkcijos nuolydžio parametras. Dažniausiai šio parametro reikšmė imama lygi 1. Tačiau eksperimentai parodė, kad, naudojant didesnę neuronų aktyvacijos funkcijos nuolydžio parametro reikšmę, pavyksta greičiau pasiekti gerus vizualizavimo rezultatus.
2. Nustatyta, kad optimali SAMANN tinklo neuronų aktyvacijos funkcijos nuolydžio parametro reikšmė yra intervale (10; 30). Pasirenkant tokias aktyvacijos funkcijos nuolydžio parametro reikšmes, galima žymiai

sumažinti skaičiavimų trukmę (iki 3–5 ir net daugiau kartų) ir gauti gerus vizualizavimo rezultatus per trumpesnį laiką, esant fiksuotam mokymo iteracijų skaičiui.

3. Eksperimentai parodė, kad galima rasti tokį analizuojamos duomenų aibės poaibį, kuriuo mokant SAMANN tinklą, mažesnės projekcijos paklaidos gaunamos greičiau, negu tinklo mokymui naudojant visus aibės taškus.
4. Buvo pasiūlytas SAMANN tinklo mokymosi aibės sudarymo metodas, grindžiamas analizuojamos duomenų aibės klasterizavimu. Naudojantis šiuo metodu galima sukonstruoti tokią neuroninio tinklo mokymo aibę, kad tinklas apsimokys žymiai greičiau (daugiau negu 5 kartus greičiau), negu naudojant visą analizuojamą duomenų aibę, o projekcijos paklaida bus neblogesne, už projekcijos paklaidą, gaunamą mokant neuroninį tinklą visa analizuojama duomenų aibe.
5. Atlikti eksperimentai parodė, kad, naudojant pasiūlytą SAMANN tinklo mokymo aibės sudarymo strategiją, galima analizuoti žymiai didesnės (daugiau negu 10 kartų didesnės) apimties duomenų aibes, negu mokant tinklą visa analizuojama duomenų aibe.
6. Ištirtos galimybės tinklo mokymui naudoti daugiaprocesorines sistemas ir Hyper-Threading technologiją. Ši technologija naudojama naujausiuose Intel firmos procesoriuose. Atliktas HT technologijos tyrimas parodė, kad ji leidžia efektyviau išnaudoti kompiuterio resursus, ypač dirbant su daugiaprocesorinėmis sistemomis. Tačiau šios technologijos ypatumai neleido efektyviai panaudoti jos dirbant su lygiagrečiuoju SAMANN algoritmu.
7. Pasiūlyta lygiagrečioji SAMANN neuroninio tinklo mokymo algoritmo modifikacija, kuri atlieka tinklo mokymą, padalinant jį į kelias atskiras dalis. Tyrimai parodė, kad skaičiuojant lygiagrečiuoju algoritmu galima pasiekti geresnius vizualizavimo rezultatus per trumpesnį laiką (lyginant su nuosekliuoju algoritmu) vizualizuojant didelės dimensijos duomenų aibes.

Sergėjus Ivanikovas

THE PROBLEMS OF PARALLEL COMPUTING IN MULTIDIMENSIONAL DATA VISUALIZATION

Summary of Doctoral Dissertation
Physical Sciences (P 000)
Informatics (09P)
Informatics, Systems Theory (P 175)

Sergėjus Ivanikovas

LYGIAGREČIŲ SKAIČIAVIMŲ TAIKYMO DAUGIAMAČIAMS DUOMENIMS VIZUALIZUOTI PROBLEMOS

Daktaro disertacijos santrauka
Fiziniai mokslai (P 000)
Informatika (09 P)
Informatika, sistemų teorija (P 175)