# CONTENT

# Plenary Section.

UDC 004.94

**Valentyn Kuzmych, Mykhailo Novotarskyi**

## SIMULATION OF FLUID MOTION IN CLOSED SURFACES USING A LATTICE BOLTZMANN MODEL

The lattice Boltzmann model is an efficient numerical scheme for modeling fluid flows. In this paper, we investigate nonstationary hydrodynamic processes in closed surfaces using the Boltzmann lattice model.

**Keywords:** lattice Boltzmann model, hydrodynamics.

Fig.: 4. Bibl.: 9.

**Target setting.** Reconstructive surgery on the human digestive tract can cause negative consequences. These effects were manifested in the appearance of unwanted deformations, so-called "blind bags", which arose due to the formation of zones of high pressure after changes in the geometry of hollow objects of the digestive tract during reconstructive surgery. For this reason, the development of a mathematical model of fluid flow in the closed surface has become crucial in recent years.

**Actual scientific research and issues analysis.** The first series of in vitro systems have been developed to analyze human digestion [1, 2] at the beginning of the 1990s. Despite the large existing amount of data on the human and animal digestive tract, conflicting results have been obtained [3]. The main limitation of this method is the difficulty of reproducing the geometry and motility of the digestive tract. Unfortunately, it is very difficult to develop an *in vitro* system capable of accurately reproducing the fluid mechanical forces that promote digestion.

Singh et. al presented an advanced fluid dynamics program that offers a promising technique to characterize the mechanisms promoting digestion [4]. Computational fluid dynamics can be used to numerically model the flow of gastrointestinal contents during digestion using knowledge of the motor response of the digestive tract and the physicochemical properties of luminal contents. Pal et. al conducted some initial attempts to simulate the gastric flow during digestion [5, 6], but the computational effort required to reproduce the geometry and motility of the stomach prevented a good characterization of the system.

Our work is devoted to the application of the lattice Boltzmann model (LBM) for modeling the processes of fluid flow on closed surfaces. This is a novel approach to obtaining acceptable results in reasonable computation time.

**Uninvestigated parts of general matters defining.**

The usage of the lattice Boltzmann model for simulation of fluid flow in closed surfaces like the human digestive tract has not been fully studied yet. Therefore, in this article, we attempted to investigate the possibility of using LBM in fluid flow simulation inside biological objects.

**The research objective.** The purpose of this paper is a study hydrodynamic processes in closed surfaces using the Boltzmann lattice model.

**The statement of basic materials.**

The lattice Boltzmann method is a numerical method to solve the Boltzmann equation on a discrete lattice:

$$v \cdot \nabla_x f + F \cdot \nabla_p f + \frac{\partial f}{\partial t} = \hat{\Omega}(f), \quad (1)$$

where $F$ – an external body force, $\nabla_x$, $\nabla_p$, is the gradient in position and momentum space, and $\hat{\Omega}(f)$ is the collision operator. The Boltzmann equation describes the dynamics of a fluid from a microscopic point of view: particles, each with velocities $v_i$, collide with a certain probability and exchange momentum among each other. For ideal collisions, total momentum and energy are conserved in the collisions. The Boltzmann equation expresses how the probability $f(x, v, t)$ of finding a particle with velocity $v$ at a position $x$ and at time $t$ evolves with time.

Assuming $F = 0$, equation (1) will be next:

$$v \cdot \nabla_x f + \frac{\partial f}{\partial t} = \hat{\Omega}(f) \quad (2)$$

For the sake of simplicity, the collision operator is taken in the most frequently used form:

$$\hat{\Omega}(f) = \frac{1}{\tau}\left(f - f^{(eq)}\right) \quad (3)$$

In (3),$\tau$ is a constant defining the time scale, which is necessary for the establishment of local equilibrium, and $f^{(eq)}$ is the density distribution function (so-called Maxwell—Boltzmann distribution function).

Thus, we get the Bhatnagar-Gross-Krook-model (or BGK-model) [7]:

$$v \cdot \nabla_x f + \frac{\partial f}{\partial t} = \frac{1}{\tau}\left(f - f^{(eq)}\right). \quad (4)$$

We make discretization of this model in the space of velocities on a finite set of vectors $\{v_k\}$ with regard for the conservation laws [8]. As a result, we get the system composed of $Q$ equations:

$$\frac{\partial f_k}{\partial t} + v_k \nabla f_k = \frac{1}{\tau}\left(f_k - f_k^{(eq)}\right), k = 0,1,2,\dots,Q-1, \quad (5)$$

where $f_k(x,t) = f(x, v_k, t)$ is the density distribution function associated with the

direction of a velocity vector$v_k$,$f_k^{(eq)}$ is the equilibrium density distribution function corresponding to the vector$v_k$.

We executed the full discretization of (5) with a time step of $\Delta t$ and a spatial step of $\Delta x_k = v_k \Delta t$ [13], in order to simplify computer realization:

$$\frac{f_k(x_k + v_k\Delta t, t + \Delta t) - f_k(x_k + v_k\Delta t, t)}{\Delta t} + \frac{f_k(x_k + v_k\Delta t, t) - f_k(x_k, t)}{\Delta x_k} = \\ = \frac{-f_k(x_k, t) - f_k^{(eq)}(x_k, t)}{\tau}.$$

Setting$\Delta x_k = \Delta t = 1$, we get the Boltzmann lattice equation

$$f_k(x_k + v_k\Delta t, t + \Delta t) - f_k(x_k, t) = \frac{-1}{\tau}\left(f_k(x_k, t) - f_k^{(eq)}(x_k, t)\right), \quad (6)$$

where$x_k$ is a point in the discretized physical space.

According to the BGK-model, Eq. (6) can be solved with the use of two steps.

1. Collision-related step:

$$\tilde{f_k}(x_k, t + \Delta t) = f_k(x_k, t) - \frac{1}{\tau}\left(f_k(x_k, t) - f_k^{(eq)}(x_k, t)\right). \quad (7)$$

2. Flow-related step:

$$f_k(x_k + v_k\Delta t, t + \Delta t) = \tilde{f_k}(x_k, t + \Delta t). \quad (8)$$

In (7) and (8), the distribution function $\tilde{f_k}$ describes a post-collisional state of the elementary volume of a fluid or the particle of a substance at the point of the discrete space$x_k$. In the BGK model, the collisions are considered as oscillations of elementary volumes of a fluid relative to the positions of local equilibrium.

The values of elements of the set $\{v_k\}$ are determined in view of the dimension of a model and the number of connected nodes forming the lattice basic element.

The mesoscopic and macroscopic levels of the modeling are connected by means of the following formulas:

$$\rho = \int_{-\infty}^{\infty} f(x, v, t)dv = \sum_{k=0}^{8} f_i = \sum_{k=0}^{8} f_k^{(eq)}, \quad (9)$$

$$u = \frac{1}{\rho}\int_{-\infty}^{\infty} v \cdot f(x, v, t)dv = \frac{1}{\rho}\sum_{k=0}^{8} v_k f_k = \frac{1}{\rho}\sum_{k=0}^{8} v_k f_k^{(eq)}, \quad (10)$$

where $u$ is the velocity vector of a flow in the fluid, and$\rho$ is the mass density of a flow in the fluid.

## Experiments.

The described method is used in modeling the distribution of pressure in the human stomach. We modeled the stomach in 2 states – a normal state and an anastomosis state. They displayed on Fig.1 – black region denotes cavity, white region – obstacle.

To apply LBM we discretized each model into a square mesh with the size of $256 \times 256$, with both width and height equal to 0.45. Parameters of LBM itself are the following: $\Re = 1000$, $\rho = 1000$. We introduced boundary value in the top as a constant flow directed to the bottom, with a velocity equal to 0.05 m/s.

All experiments were performed on PC with Ryzen 7 5800X CPU and 32 GB RAM, using Pylbm python library [9].

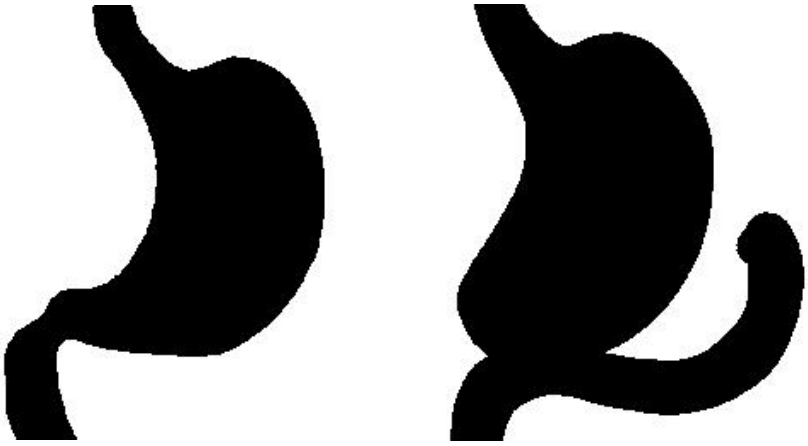


***Fig. 1.*** Left –normal stomach, right – anastomosis

We measured pressure field distribution at modelling times t = 2.46 sec and t = 5 sec. Fig. 2 shows distribution in the case of anastomosis, fig. 3 shows normal state of human stomach.

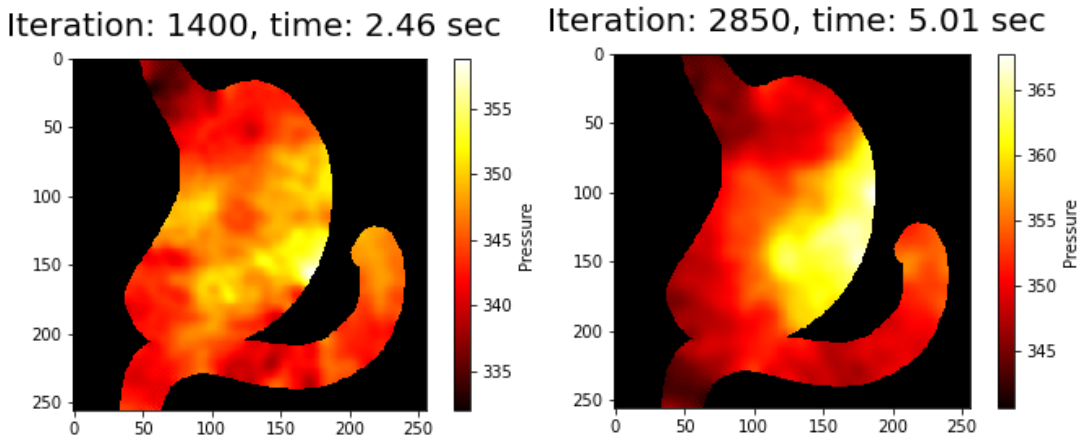



***Fig 2.*** Pressure field distribution in anastomosis model

Results demonstrated higher density near the right wall of the stomach, in case of anastomosis than in the normal state. Also, anastomosis model shows high pressure in “blind bag” under stomach. In real situations it can cause development of negative consequences.

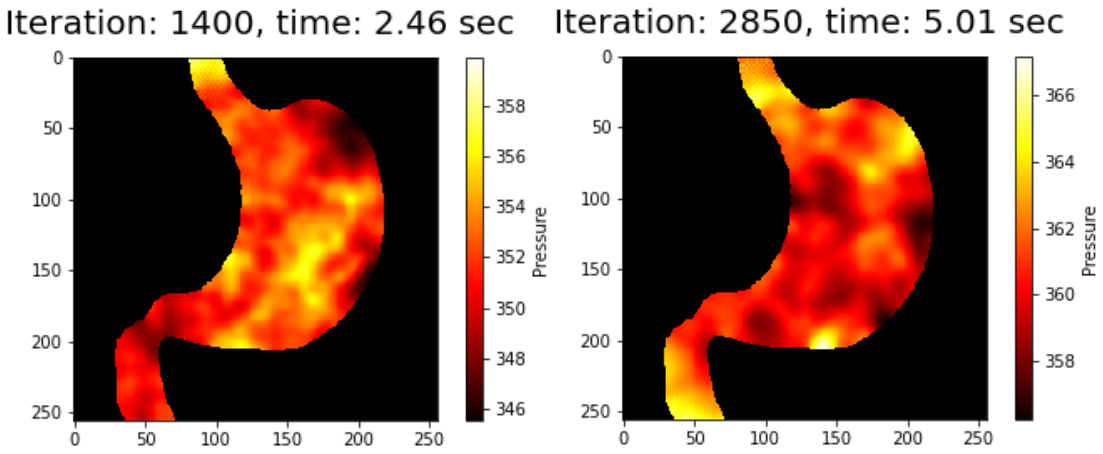



***Fig 3.*** Pressure field distribution in normal state

Another point is range of pressure values in both states are also different. At the modelling time 2.46 sec, in anastomosis state pressure field values fall in range from 330 to 360, in normal state – from 346 to 359. At the modelling time 5.01 sec, in anastomosis state pressure field values fall in range from 341 to 366, in normal state – from 357 to 366. We investigated relationship between average pressure inside stomach area and modelling time in aforementioned states. Fig.4 shows this relationship. During all period of modelling, average pressure in normal state is higher, than in anastomosis. Due to this outcome and previously mentioned results, we can conclude that pressure field in anastomosis state irregular in comparison to normal state of stomach.

## Conclusions.

This paper investigates the application of the lattice Boltzmann model in the simulation of fluid motion on closed surfaces. The human digestive tract was chosen as an appropriate example of a closed surface, due to the practical significance of this model. Conducted experiments show the clear distinction of modeled behavior between the normal state of the stomach and the anastomosis state. This result indicates the practical significance of our work.

Our paper clearly has some limitations. We investigated only the 2D domain, which cannot provide perfect accuracy. Despite this, we believe our work could be the basis for other improvements – handling the 3D domain and more sophisticated boundary conditions, combined with machine learning approaches.

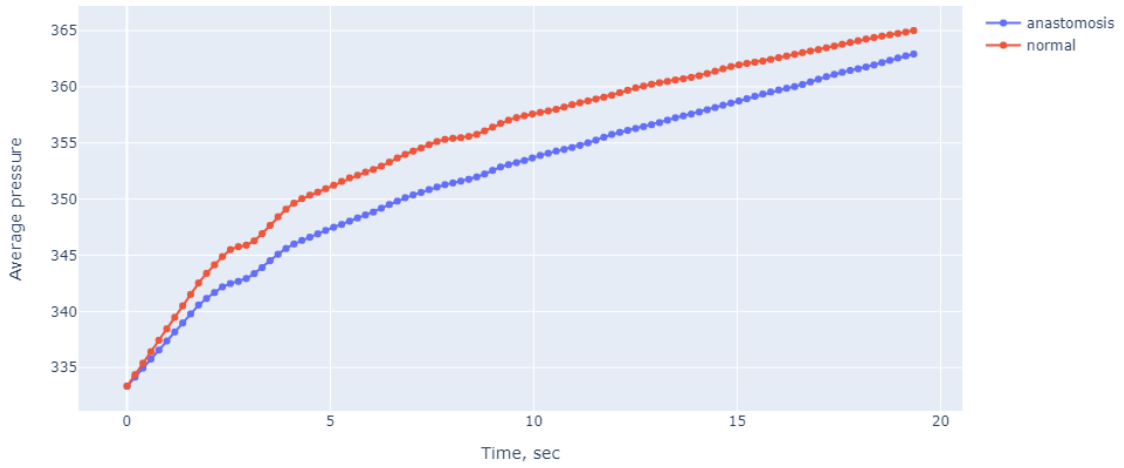



***Fig.4*** Average pressure during modelling

## References

1. Aoki S. *Evaluation of the correlation between in vivo and in vitro release of phenylpropanolamine HCl from controlled-release tablets* /. Aoki S, Uesugi K, Ozawa H, Kayano M. // Int J Pharm, 1992 – Vol 85 – P. 65–73

2. Molly K. *Development of a 5-step multi-chamber reactor as a simulation of the human intestinal microbial ecosystem* / Molly K, Vandewoestyjne M, Verstraete W. // Appl Microbiol Biotechnol, 1993

3. Yoo JY. *GIT physicochemical modeling-a critical review* / Yoo JY, Chen XD // Int J Food Engr, 2006 - 2(4), Art. 4

4. Singh SK. *Fluid flow and disintegration of food in human stomach* // University of California, Davis, CA: Biological Systems Engineering, 2007.

5. Pal A. *Gastric flow and mixing studied using computer simulation* / Pal A, Indireshkumar K, Schwizer W, Abrahamsson B, Fried M, Brasseur JG. // Proc R Soc Lond B, 2004

6. Pal A. *A stomach road or ‘‘Magenstrasse’’ for gastric emptying*. / Pal A, Brasseur JG, Abrahamsson B. // J Biomech, 2007

7. Bhatnagar P.L. *A model for collision processes in gases. I: Small amplitude processes in charged and neutral one-component system* / Bhatnagar P.L., Gross E.P., Krook M. // Physical Review.– 1954. – Vol.94, №3 – P.511–525.

8. He X. *Theory of the lattice Boltzmann equation: from Boltzmann equation to lattice Boltzmann equation* / He X., Luo L-S. // Physical Review E.– 1997.– Vol. 56, №6.– P.6811–6817.

9. https://github.com/pylbm/pylbm

## AUTHORS

Kuzmych Valentyn – PhD student, Department of Computer Engineering, National Technical University of Ukraine “Igor Sikorsky Kyiv Polytechnic Institute”.

E-mail: valentine.kuzmich@gmail.com

Novotarskyi Mykhailo – full professor, Department of Computer Engineering, National Technical University of Ukraine “Igor Sikorsky Kyiv Polytechnic Institute”.

E-mail: novot@ukr.net

## **Parallel Section SEC.** Security, Fault Tolerance.

UDC 681.327

**Victor Porev**

# FEATURES OF RUN-LENGTH ENCODING FOR TRUE COLOR RASTER IMAGE FORMAT

The paper considers the approach to improving the compression method based on run-length encoding. Describes a modification of the compression scheme RLE-БП to provide the ability to compress lossless bitmaps of True Color format. The main goal is to achieve a high decompression rate with a competitive degree of compression due to the variability of the construction of code sequences according to the RLE-БП.

**Key words**: compression, lossless, run length encoding, RLE-БП.

Fig.: 1. Bibl.: 8.

**Relevance of the research topic.** An important role in the digital era is played by the search and implementation of effective information coding methods, in particular data compression methods.

**Formulation of the problem.** For many applications, there are requirements to provide the highest possible degree of compression for data storage and to obtain the fastest possible decompression when reading such data. To a large extent, such requirements are contradictory, so it is necessary to find some compromise. The balance between the degree of compression and the speed of decompression can change as a result of improving the encoding of information.

**Actual scientific researches and issues analysis.** Research on information compression has been going on for many decades. One of the first known compression methods is the Huffman method, according to which the most popular symbols are encoded with a shorter prefix code, and the less popular ones with a long one [1]. Also, the run length encoding (RLE) method was proposed quite a long time ago, according to which chains of the same symbols are coded by the number of repetitions-character code pair [2]. For text compression, this encoding is ineffective, but the RLE method proved to be useful for encoding bitmap images with a limited number of colors - no more than 256. Later, dictionary LZ encoding methods [3, 4] were used to compress bitmap images. These methods have a significantly higher degree of compression compared to RLE and have been used for such graphic formats

as GIF, PNG. The above methods belong to the category of lossless compression - encoding does not change the original data in any way. Such methods are mainly focused on the 256 color format, although PNG allows you to store True Color images as well.

In general, the True Color format, such as 24 bits per pixel, is very commonly used for photo-type images. But compression methods, even as effective as dictionary LZ, are not used for digital photos, so in many cases they do not provide any compression. Digital photographs use lossy compression techniques such as JPEG [5] or variations of wavelet encoding such as JPEG2000 [6]. Lossy compression methods make it possible to compress the image of digital photos significantly - tens of times - without a noticeable deterioration in the perception of the image by a person. However, such methods cannot be used for some applications where complete preservation of the original data is required.

Among the lossless compression methods, dictionary LZ-like methods seemed to have the absolute advantage due to the highest degree of compression of repetitive data. In any case, compared to the Huffman and RLE methods. But it is not quite so. In particular, the RLE method provides a much higher operating speed and does not require additional memory for the dictionary. In addition, RLE is generally convenient to use for encoding individual rows (or columns) of a raster, and dictionary LZ-like methods lose their effectiveness here, since the contents of the dictionary are created based on all the data. Another advantage of RLE is the possibility of parallel (or multithreaded) encoding and decoding of different raster fragments.

Therefore, the RLE method was chosen as the basis for solving tasks related to providing fast direct access to individual fragments of large rasters. Around 2004, to support the geoinformation system, methods were invented to modify the RLE method to significantly increase the degree of compression while maintaining the speed of decompression and providing direct access to raster fragments. These methods were named RLE-БП according the first letters of the authors' Ukrainian surnames - Блінова, Порєв. Later, in 2008, basic information about RLE-БП was published in particular in [7] and some other publications.

The main idea of RLE-БП consists, firstly, in improving and significantly complicating code sequences, and secondly, that for each line of the raster, the software coder looks for such values of the code parameters that provide the minimum total amount of code for each line. Thanks to this, it was possible to significantly

increase the degree of compression compared to simple RLE and in many cases for images of business graphics to approach the level of compression, for example, the LZW method [6, 8].

Tatyana Blinova

Victor Porev

***Fig. 1.*** The authors of RLE-БП

**Uninvestigated parts of general matters defining.** The question of encoding chains of repeating pixels in the True Color color format is insufficiently researched.

**Setting objectives.** The main tasks are to invent such methods of encoding chains of repeating pixels that are capable of providing high decompression speed and organizing direct access to compressed data in the form of True Color format images.

**The statement of basic materials.** To support True Color bitmap compression capabilities, some encoding variants have been added to the RLE-BP set. These variations for the 24-bit format are named methods 2-4 and are described as follows.

Encoding method 2. Three types of code sequences are used:

0 c..c - single pixel of any color (c..c) - 1+24 bits

10 n..n m..m - main color pixel chain (m..m) - 2+N1+M bits

n..n - code of N1 bits to represent (n-1), where n is the length of the string

m..m - M bits code of the main color

11 n..n c..c - chain of pixels of any color (c..c) - 2+N2+24 bits

n..n - code of N2 bits to represent (n-2), where n is the length of the string

where

M = 1...8 – number of bits to represent the main color index

N1 = 0...10 - the number of bits to represent the length of the main color chain

N2 = 0...5 - the number of bits to represent the length of a string of any color

Encoding method 3. Three types of code sequences are used:

0 m..m - single pixel of the main color (m..m) - 1+M bits

10 n..n m..m - main color pixel chain (m..m) - 2+N1+M bits

n..n - code of N1 bits to represent (n-2), where n is the length of the string
m..m - M bits code of the main color
11 n..n c..c - a single pixel or a chain of pixels of any color (c..c) - 2+N2+24 bits
n..n - code of N2 bits to represent (n-1), where n is the length of the string
where
M = 1...8 – number of bits to represent the main color index
N1 = 0...10 - the number of bits to represent the length of the main color chain
N2 = 0...5 - the number of bits to represent the length of a string of any color

Encoding method 4. Two types of code sequences are used:
0 n..n m..m - single pixel or chain of pixels of the main color (m..m) - 1+N1+M bits
n..n - code of N1 bits for the number of (n+1) pixels
m..m - M bits code of the main color
1 n..n c..c - a single pixel or a chain of pixels of any color (c..c) - 1+N2+24 bits
n..n - code of N2 bits for the number of (n+1) pixels
where
M = 1...8 – number of bits to represent the main color index
N1 = 0...10 - the number of bits to represent the length of the main color chain
N2 = 0...5 - the number of bits to represent the length of a string of any color

These methods are used in the software of the geographic information system and in applications for distance learning.

**Conclusions.** The modified RLE-БП run length encoding method, which allows to achieve a competitive compression ratio at a high decompression speed in the mode of direct access to raster fragments, is highlighted. A modification of the RLE method for compression of images in True Color format is proposed.

## References

1. D.A. Huffman. A Method for the Construction of Minimum-Redundancy Codes // Proceedings of the IRE. 40 (1952): 1098–1101. doi:10.1109/JRPROC.1952. 273898.

2. S.W. Golomb. Run-Length Encodings // IEEE Trans. Information Theory, 12:3 (1966) pp. 399-401.

3. Ziv J., Lempel A. Compression of Individual Sequences via Variable-Rate Coding // IEEE Trans. Inform. Theory, 1978, V. 24 (5), pp. 530–536

4. Welch T. A Technique for High-Performance Data Compression // Computer, 1984, V. 17 (6), pp. 8–-19.

5. Gregory K. Wallace. The JPEG Still Picture Compression Standard // IEEE Transactions on Consumer Electronics. Vol. 38, No 1, Feb 1992, pp. xviii-xxxiv

6. High Throughput JPEG 2000 (HTJ2K) and the JPH file format: a primer // ISO/IEC JTC 1/SC 29/WG1 | Document N87018. URL: http://ds.jpeg.org/whitepapers/ jpeg-htj2k-whitepaper.pdf

7. Blinova T., Porev V. Some Methods Of The Raster Encoding In Geographic Information Systems // Proc. int. conf. CODATA`21, Kyiv, 2008. – p.153.

8. Victor Poriev. Improving the method of run length encoding // Proceeding of International Conference on Security, Fault Tolerance, Intelligence" (ICSFTI2019), 14-15 may 2019, Kyiv .- pp.165-169.

## AUTHORS

**Victor Porev** – associate professor, Department of Computer Engineering, National Technical University of Ukraine "Igor Sikorsky Kyiv Polytechnic Institute".

E-mail: v_porev@ukr.net

## EXTENDED SUMMARY

**Victor Porev**

# FEATURES OF RUN-LENGTH ENCODING FOR TRUE COLOR RASTER IMAGE FORMAT

**Relevance of research topic.** An important role in the digital era is played by the search and implementation of effective information coding methods, in particular data compression methods.

**Formulation of the problem.** For many applications, there are requirements to provide the highest possible degree of compression for data storage and to obtain the fastest possible decompression when reading such data. To a large extent, such requirements are contradictory, so it is necessary to find some compromise. The balance between the degree of compression and the speed of decompression can change as a result of improving the encoding of information.

**Analysis of recent research and publications.** Research on information compression has been going on for many decades. The RLE method was chosen as the basis for solving tasks related to providing fast direct access to individual fragments of large rasters. At one time, methods of modifying the RLE method were invented to significantly increase the degree of compression while maintaining the speed of decompression and providing direct access to raster fragments. These methods were called RLE-БП.

**Selection of unexplored parts of the general problem.** The question of encoding chains of repeating pixels in the True Color color format is insufficiently researched.

**Setting objectives.** The main tasks are to invent such methods of encoding chains of repeating pixels that are capable of providing high decompression speed and organizing direct access to compressed data in the form of True Color format images.

**Presentation of the main material.** To support True Color bitmap compression capabilities, some encoding variants have been added to the RLE-БП set, which are designated as methods 2-4 for the 24-bit format.

These methods are used in the software of the geographic information system and in applications for distance learning.

**Conclusions.** The modified RLE-БП run length encoding method, which allows to achieve a competitive compression ratio at a high decompression speed in the mode of direct access to raster fragments, is highlighted. A modification of the RLE method for compression of images in True Color format is proposed.

**Key words**: compression, lossless, run length encoding, RLE-БП.
UDC 378.1

**Victor Porev**

# PATTERN OF OWNEDRAW GUI FOR MULTI-MODE SOFTWARE APPLICATIONS

The article considers some aspects of building graphical interfaces with non-standard elements for multi-mode software applications. The onDraw-onTouch pattern is proposed and analyzed.

**Key words**: application, GUI, multi-mode, pattern.

Fig.: 5. Bibl.: 6.

**Relevance of the research topic.** Design patterns play an important role in software engineering by providing sample solutions for a particular class of software applications.

**Formulation of the problem.** A convenient and adequate graphical user interface (GUI) is a necessary component for a wide list of different software applications. This is especially true for multimode applications, such as those that provide extensive functionality for entering, editing, and displaying information. Multimode of such applications can be thought of as the ability to transition from one state to another, with context-sensitive graphical controls for each state. Application development environments typically provide programmers with some set of standard controls. Such standard elements are supported by corresponding API classes and functions. But if a developer wants to diversify the user interface by adding his own custom controls, he has to write a lot of code himself. This can be simplified by describing typical structures in the form of an architectural pattern. The pattern largely unifies the solution, which makes it easier to build assets and ensure their reuse.

**Analysis of recent research and publications.** The first significant advance in the classification of patterns is the Design Patterns book [1]. Since then, more than 2 decades have passed, but the relevance of patterns does not decrease, as they make it easier for programmers to implement effective architectural solutions that have already been developed.

An important step in the development of the science of patterns was the work of Martin Fowler, in particular, the description of the dependency injection pattern [2]. Dependency injection makes it easier to design software systems with extensibility, in particular, with a developed graphical user interface.

In general, an event-driven approach is mainly used for GUI implementation. This approach was described as an Observer pattern in the Design Patterns book.

Subsequently, the technology of interaction between system elements, such as message exchange or event processing, was designated as callback. For example, as in the Windows message system [3]. In some systems, the term ‘listener’ is used to denote such mechanisms [4].

Some examples of implementation of non-standard GUI elements in the form of software libraries are known [5, 6].

**Uninvestigated parts of general matters defining.** In the opinion of the author, there is a need for some generalization of the approach to the design of non-standard graphical user interfaces. Such a generalization should be made in the form of appropriate pattern design.

**The research objective.** The purpose of the research is to find some unified template - a pattern for describing the construction of event-driven architecture, focused on the active use of graphic and sensor capabilities of modern systems.

**Presentation of the main material.** Proposed pattern structure. The figure below shows the simplified class diagram of the onDraw-onTouch pattern.

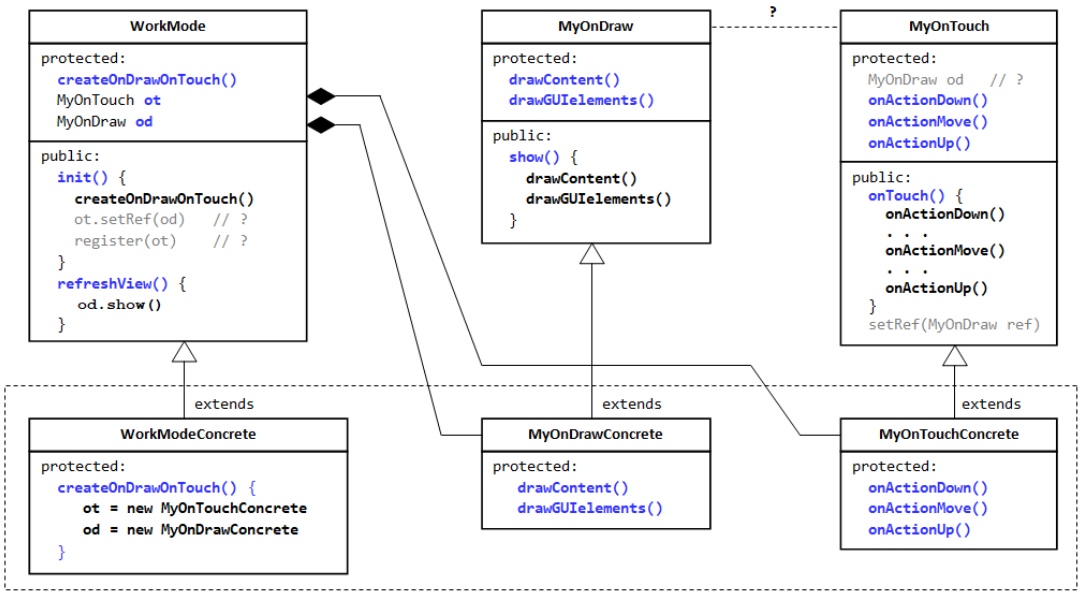



***Fig.1.*** Pattern onDraw–onTouch

Classes of the WorkMode hierarchy describe the states (modes of operation) of the program. The base class may or may not be abstract - it may have default member definitions, such as for the start state of the program. The WorkModeConcrete classes describe the states in each specific work mode.

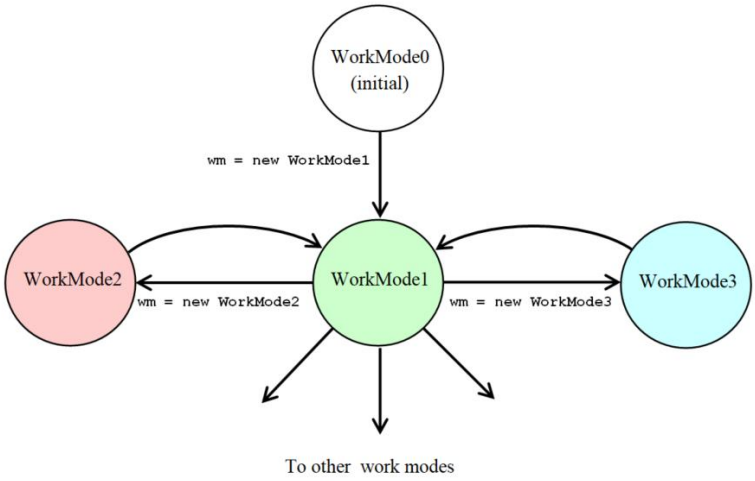



***Fig. 2.*** State transitions in a multi-mode application such as an editor

Thus, it can be imagined that in order to switch to some mode, for example, WorkMode2, it is necessary to create an appropriate state object

```
WorkMode wm = new WorkMode0     //start from some initial state
. . .
wm = new WorkMode2
wm.init()
wm.refresh()
```

During the initialization of an object of the WorkModeConcrete class (in this example, it is WorkMode2), objects of the MyOnDrawConcrete and My On Touch Concrete classes are created. It may be appropriate in some cases to use a constructor instead of the init() method. This is at the discretion of the programmer.

Thus, each object of the WorkMode class creates and encapsulates objects of the MyOnDraw and MyOnTouch classes. In this pattern, it is implicitly assumed that access to objects of the MyOnDraw and MyOnTouch classes from the outside is closed, although this is not necessary. Depending on the implementation platform, you can provide for registering a touch event listener, for example, through the API of the corresponding sensor. This is partially shown in the class diagram in the body of the init() method.

One of the interface methods of the WorkMode class is the refreshView() method, which calls the show() method of the MyOnDraw class. The show() method

displays two things: some background content plus images of active custom GUI elements. As a general rule, active GUI elements should be in the foreground, so the call to drawGUI elements() in the body of the show() method is written last.

Based on platform considerations, it is possible for an implementation to provide a dependency of the MyOnTouch class on the MyOnDraw class. To do this, you can pass a reference (pointer) to an object of the MyOnDraw class to an object of the MyOnTouch class by calling the setRef(MyOnDraw) method. When might it be needed? Imagine that each time you move the cursor, you need to redraw the image in the window. Then, in the body of the onActionMove() method of the MyOnTouch class, you should provide a call to the drawing method of an object of the MyOnDraw class for example od.show().

Thus, for each specific state (mode of operation), the view of the application window is described by the program code for implementing the drawContent(), drawGUIelements() methods, and the logic for handling touch events is described in the onActionDown(), onActionMove(), onActionUp() methods. This is the main essence of this pattern. The list of touch event methods can be extended, for example, to implement multitouch.

When implementing this pattern, it is necessary to provide for the consistency of the display and touch coordinates of all active elements. To do this, you can provide appropriate members in the MyOnDraw class, which would also be visible in the MyOnTouch class.

Pattern onDraw-onTouch can be used to build ownerdraw GUIs for a variety of applications for many operating systems and platforms.

Features of the implementation of the onDraw-onTouch pattern on the Microsoft Windows platform. It is possible to choose several approaches for implementation. The figure below shows an implementation of the onDraw-onTouch pattern based on the Windows API.

You need to program a window callback function that calls the Windows message handlers. In the WorkMode base class, you can define such methods as message handlers:

onPaint() - WM_PAINT handler

onLBdown() - WM_LBUTTONDOWN handler

onMouseMove() - WM_MOUSEMOVE handler

onLBup() - WM_LBUTTONUP handler

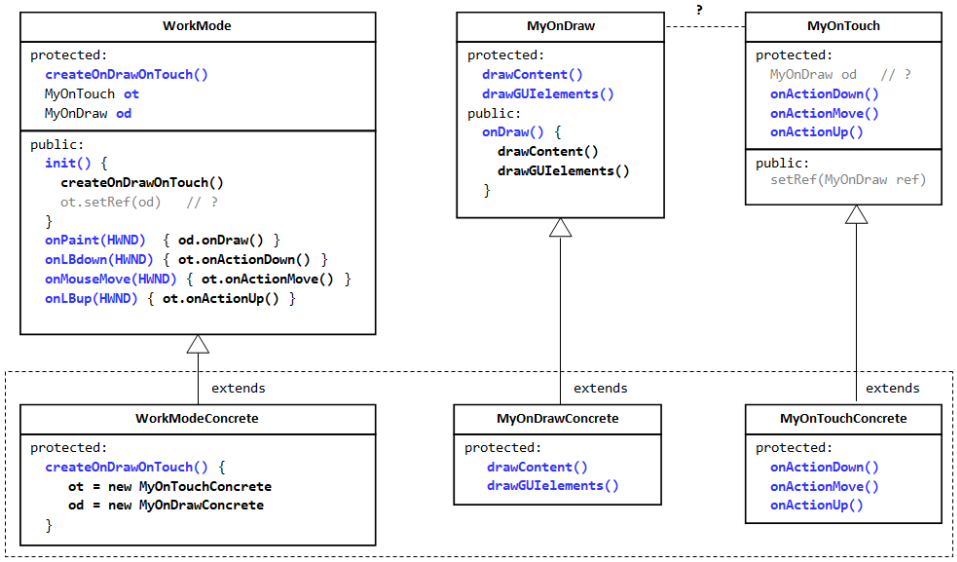



***Fig.3.*** Pattern onDraw–onTouch implementation for Windows API

The refreshView() method from the WorkMode class can be omitted, since the InvalidateRect() Windows API function can be called directly instead.

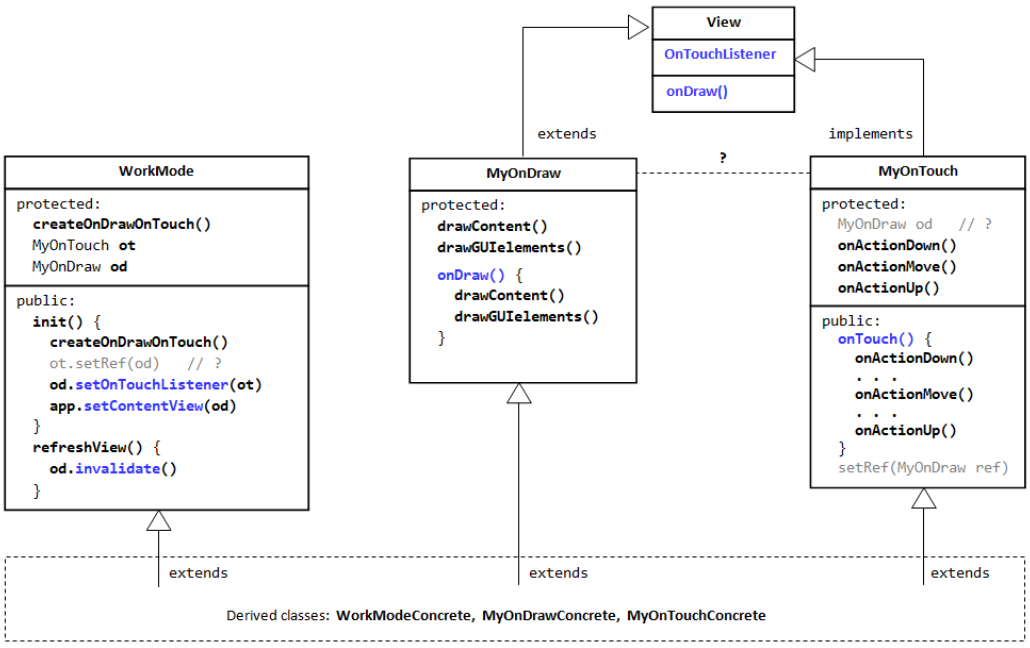



***Fig.4.*** Pattern onDraw–onTouch implementation for Android API

Next, let's look at the features of the implementation of the onDraw-onTouch pattern on the Android platform. To implement this, it is convenient to use the View class from the Java and Kotlin Android API classes. In order to organize the display of graphics, it is enough to override the onDraw() method of the View class in a derived

class. In our case, in the MyOnDraw class. And to access messages from the touch sensor, the View class provides the onTouchListener interface with the onTouch() method, which should be implemented in the user class, for example, MyOnTouch.

To redraw the contents of the window, you can call the invalidate() method of the View class, which leads to a subsequent call to the onDraw() method of the MyOnDraw class. To register the callback methods of the drawing and touch classes, the setContentView() and setOnTouchListener() methods are used.

Below in Fig. 5 illustrates an example of the implementation of this pattern in the Android application MyGIS created and developed by the author of this article.

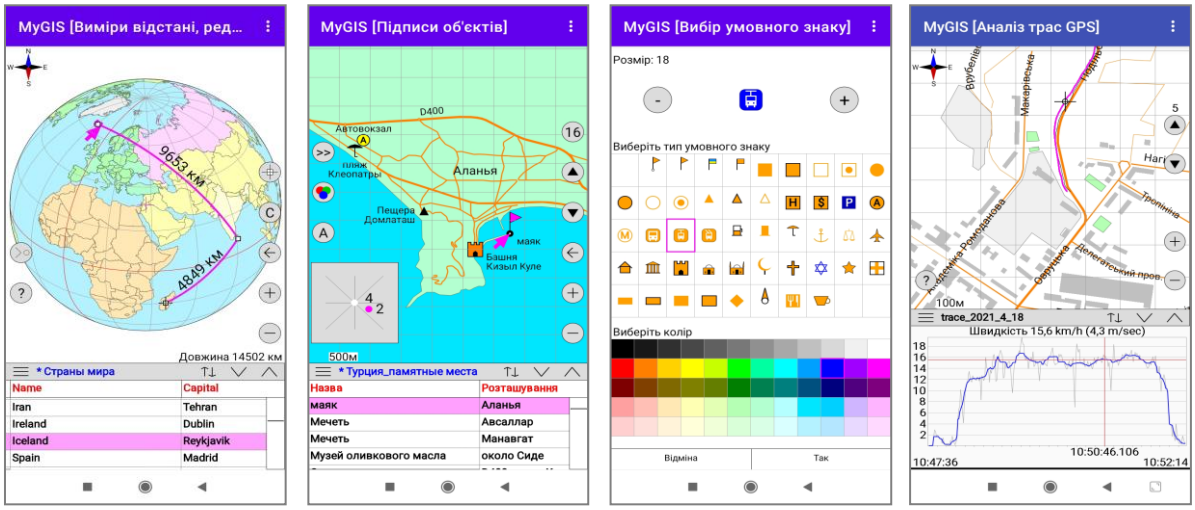



***Fig. 5.*** Examples of custom ownerdraw GUI elements based on the onDraw-onTouch pattern in the MyGIS Android application

The main idea of the GUI is being implemented: What we see is what we can touch.

**Conclusions.** The possibilities of generalizing the description of building user interfaces for multimode applications based on the proposed onDraw-onTouch pattern are considered. Using this pattern can reduce development costs in object-oriented style applications with non-standard GUI elements.

## References

1. Gamma E., Helm R., Johnson R., Vlissides J. Design Patterns: Elements of Reusable Object-Oriented Software. Addison-Wesley, 1994, 395 p.

2. Martin Fowler. Inversion of Control Containers and the Dependency Injection pattern - Forms of Dependency Injection. 23 January 2004.

URL: https://martinfowler.com/articles/injection.html

3. Microsoft. Windows Dev Center. URL: https://developer.microsoft.com/en-us/windows/

4. Google. Documentation for app developers.

URL: https://developer.android.com/docs

5. Fully Skinned UI in wxWidgets (Trying to Emulate OwnerDraw) // wxWidgets Discussion Forum. URL: https://forums.wxwidgets.org/viewtopic.php?t=42924

6. SEGGER. LISTVIEW - Custom (Sample)

URL: https://wiki.segger.com/LISTVIEW_-_Custom_(Sample)

## AUTHORS

**Victor Porev** – associate professor, Department of Computer Engineering, National Technical University of Ukraine “Igor Sikorsky Kyiv Polytechnic Institute”.

E-mail: v_porev@ukr.net

## EXTENDED SUMMARY

**Victor Porev**

# PATTERN ONDRAW–ONTOUCH AND ITS USABILITYFOR OWNERDRAW GUI

**Relevance of research topic.** Design patterns play an important role in software engineering by providing decision patterns for a particular class of software applications.

**Formulation of the problem.** A developed, convenient and adequate graphical user interface is a necessary component for a large number of different software applications. It is possible to simplify the creation of a non-standard GUI for multimode applications by describing the typical constructions in the form of an architectural pattern, which largely unifies the solution, which makes it easier to build assets and ensure their reuse.

**Analysis of recent research and publications.** The first significant advance in the classification of patterns is the Design Patterns book [1]. Since then, more than 2 decades have passed, but the relevance of patterns does not decrease, as they make it easier for programmers to implement effective architectural solutions that have already been developed.

**Uninvestigated parts of general matters defining.** There is a need for some generalization of the approach to the design of non-standard graphical user interfaces. Such a generalization should be made in the form of appropriate pattern design.

**The research objective.** The purpose of the research is to find some unified template - a pattern for describing the construction of event-driven architecture, focused on the active use of graphic and sensor capabilities of modern systems.

**Presentation of the main material.** Programs can work out different modes of operation (states). Each mode is described by a class that is derived from some base class. The onDraw-onTouch pattern is proposed, which describes the relationship between the main display classes of GUI elements and event handlers related to user interaction with these elements. Examples of implementation and use of such a pattern are considered.

**Conclusions.** Possibilities of construction of the graphic user interface on the basis of the offered pattern onDraw-onTouch are considered. This pattern describes a simplified generalized approach to GUI implementation. Instead of combining different API controls, each of which usually has significant features of implementation in the program code, the pattern allows you to unify the construction of program code for the GUI in an object-oriented style. This can reduce the cost of developing programs.

**Key words**: GUI, pattern.

**Artemii Kyrianov, Heorhii Loutskii,**
**Oleksandr Chaikovskyi**

# TRIANGULATION OF MOBILE PHONE LOCATION BY BASE STATIONS

The article discusses the method of triangulation in radar location in the cellular network. The issue of the task of changing the structure in which the triangulation is presented is considered, which may arise, for example, when building a greedy or optimal triangulation. Algorithms for their construction operate only with edges and nodes, and therefore they are forced to use data structures of the type "Nodes with neighbors" or "Nodes and edges". On the other hand, the purpose of triangulation may be to model a surface, which requires a data structure such as Nodes and Triangles. That is why the task of transition from one data structure to another arises. Data comparisons were made, the main gains, losses and prospects were identified.

**Key words:** fault tolerance, excess code, Latin square

Fig.: 11. Tabl.: 1. Bibl.: 4.

**Structures to represent triangulation.** As practice shows, the choice of a structure for representing a triangulation has a significant impact on the theoretical complexity of the algorithms, as well as on the speed of a specific implementation. In addition, the choice of structure may depend on the purpose of further use of triangulation. In triangulation, 3 main types of objects can be distinguished: nodes (points, vertices), edges (segments) and triangles. In the work of many existing algorithms for constructing the Delaunay triangulation and algorithms for its analysis, the following operations often occur with triangulation objects:

1. Triangle → nodes: get for the given triangle the coordinates of the nodes forming it.

2. Triangle → edges: getting a list for a given triangle

the edges that form it.

3. Triangle → Triangles: Retrieve a list of neighboring triangles for a given triangle.

4. Edge → nodes: get for this edge the coordinates of the nodes forming it.

5. Edge → Triangles: Retrieve a list of adjacent triangles for a given edge.

6. Node → edges: get a list of adjacent edges for a given node.

7. Node → Triangles: Get a list of adjacent triangles for a given node.

In some algorithms, some of these operations may not be used. In other algorithms, edge operations may occur infrequently, so the edges can be represented indirectly as one of the sides of some triangle. Consider the most common structures.

**Data structure "Nodes with neighbors".** In the structure "Nodes with neighbors" for each triangulation node, its coordinates on the plane and a list of numbers (or pointers) of adjacent (neighbors with which there are common edges) nodes are stored (Fig. 1):

Node = record
X: number; ← X coordinate
Y: number; ← Y-coordinate
Count: integer; ← number of adjacent nodes
Nodes: array [1..Count] of NodeNumber; ← list of adjacent nodes
end;

The order of adjacent nodes in the list is usually not important, but in some tasks it is sometimes required that this list of nodes be sorted clockwise or counterclockwise (Figure 1).

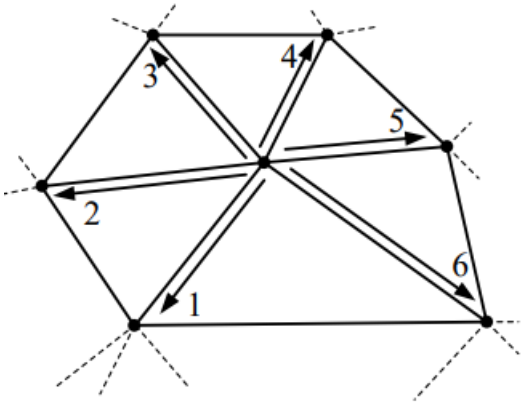



***Fig. 1.*** Connections of nodes of the structure "Nodes with neighbors"

Essentially, the list of neighbors implicitly defines the triangulation edges. Triangles are not represented at all, which is usually a significant obstacle to the further use of triangulation. In addition, the disadvantage is the variable size of the node structure, which often leads to wasteful memory consumption when building a triangulation. The average number of adjacent nodes in a Delaunay triangulation is 6 (this is proved by induction or from Euler's planar graph theorem), so with an 8-byte coordinate representation, 4-byte integers, and 4-byte pointers, the total amount of memory occupied by this triangulation structure is 44 * N bytes.

**Nodes and Edges Data Structure.** In the "Nodes and edges" structure, nodes and edges are explicitly specified. There are no triangles in the structure. For each edge, pointers to two end nodes are stored. For triangles, pointers to the three edges forming the triangle are stored (Figure 2):

Node = record
X: number; ← X coordinate
Y: number; ← Y-coordinate
end;
Edge = record
Nodes: array [1..2] of NodeNumber; ← list of end nodes
end;

This structure is often used in cases where it is required to explicitly represent the edges of a triangulation, but there is no need to work with triangles. In particular, this structure is best suited for constructing greedy and optimal triangulations. This structure consumes quite a bit of memory: with an 8-byte representation of coordinates and 4-byte pointers, there are about 40 * N bytes

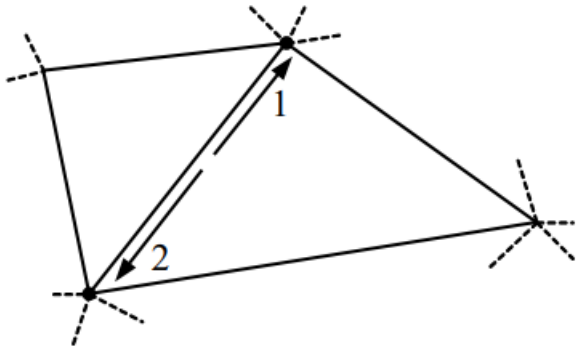



***Fig. 2.*** Connections of edges of the structure "Knots and edges"

**Data Structure "Double Edges".**In the "Double Edges" structure, the basis of triangulation is a list of directed edges. In this case, each edge enters the triangulation structure twice, but directed in opposite directions:

Node = record
X: number; ← X coordinate
Y: number; ← Y-coordinate
end;
Edge = record
Node: Node_number; ← end node of the rib

Next: Edge_number; ← next clockwise in the triangle on the right

Twin: Edge_number; ← twin edge pointing the other way

Triangle: Triangle_number; ← pointer to right triangle

end;

Triangle = record ← There are no required fields in the record

end;

The following pointers are stored for each edge (Figure 3):

1) on the end node of the rib;

2) to the next clockwise edge in the triangle to the right of this edge;

3) to the "twin edge", connecting the same triangulation nodes as the given one, but directed in the opposite direction;

4) to the triangle located to the right of the edge.

The last pointer is not needed to build a triangulation, and therefore its presence should be determined depending on the purpose of the further application of triangulation.

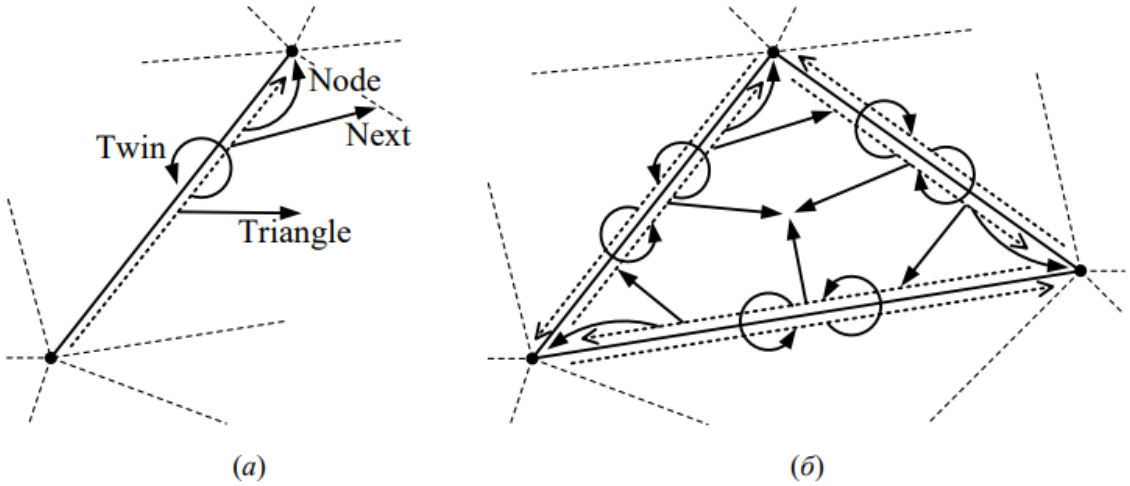



***Fig. 3.*** Edge connections (a) and implicit definition of triangles (b) in the “Double edges” structure

The disadvantages of this structure are the representation of triangles in an implicit form, as well as a large memory consumption, which, with an 8-byte representation of coordinates and 4-byte pointers, is at least 64 * N bytes (not taking into account the memory consumption for representing additional data in triangles).

**Nodes and Triangles Data Structure.** In the "Nodes and Triangles" structure, for each triangle, three pointers to the nodes forming it and three pointers to adjacent

triangles are stored (Figure 4):

Node = record
X: number; ← X coordinate
Y: number; ← Y-coordinate
end;
Triangle = record
Nodes: array [1..3] of NodeNumber; ← generating nodes
Triangles: array [1..3] of TriangleNumber; ← neighboring triangles
end;

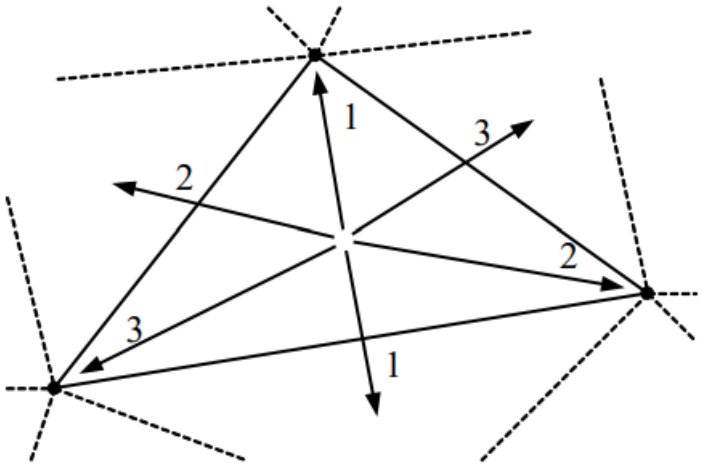



***Fig. 4.*** Connections of triangles of the "Knots and Triangles" structure

Points and neighboring triangles are numbered in clockwise order, while opposite the point with the number i{1, 2, 3} there is an edge corresponding to the neighboring triangle with the same number (Figure 4). The edges in this triangulation are not explicitly stored. If necessary, they are usually represented as a pointer to a triangle and the number of an edge inside it. With an 8-byte coordinate representation and 4-byte pointers, this triangulation structure requires approximately 64 * N bytes.∈ Despite the fact that this structure is inferior to Nodes with Neighbors, it is most often used in practice due to its relative simplicity and ease of programming algorithms based on it.

**Nodes, Edges and Triangles Data Structure.** In the "Nodes, edges and triangles" structure, all triangulation objects are explicitly specified: nodes, edges and triangles. For each edge, pointers to two end nodes and two neighboring triangles are stored. For triangles, pointers to the three edges forming the triangle are stored (Figure 5):

Node = record

X: number; ← X coordinate
Y: number; ← Y-coordinate
end;
Edge = record
Nodes: array [1..2] of NodeNumber; ← list of end nodes
Triangles: array [1..2] of TriangleNumber; ← neighboring triangles
end;
Triangle = record
Ribs: array [1..3] of RibNumber; ← generating edges
end;

Points and neighboring triangles are numbered in clockwise order, while opposite the point with the number i{1, 2, 3} there is an edge corresponding to the neighboring triangle with the same number (Figure 5). This structure is often used in practice, especially in problems where it is required to explicitly represent triangulation edges.∈

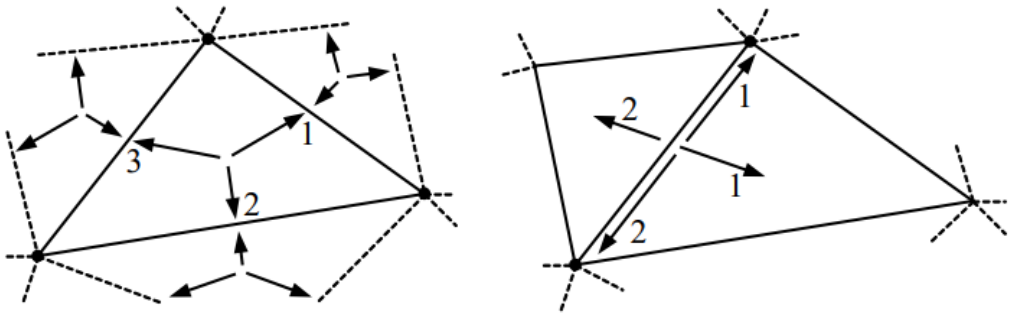



***Fig. 5.*** Connections of triangles (left) and edges (right)
of the structure "Nodes, edges and triangles"

The disadvantage of this structure is a large memory consumption, amounting to approximately 88 * N bytes with an 8-byte representation of coordinates and 4-byte pointers.

**Nodes, simple edges and triangles data structure.** In the "Knots, simple edges and triangles" structure, all triangulation objects are explicitly specified: nodes, edges and triangles. For each edge, pointers to two end nodes and two neighboring triangles are stored. There is no special information for edges. For triangles, pointers are stored to the three nodes and three edges forming the triangle, as well as pointers to three adjacent triangles (Figure 6):

Node = record

X: number; ← X coordinate

Y: number; ← Y-coordinate

end;

Edge = record ← There are no required fields in the record

end;

Triangle = record

Nodes: array [1..3] of NodeNumber; ← generating nodes

Triangles: array [1..3] of TriangleNumber; ← neighboring triangles

Ribs: array [1..3] of RibNumber; ← generating edges

end;

This structure is often used in practice, especially in problems where it is required to explicitly represent triangulation edges.

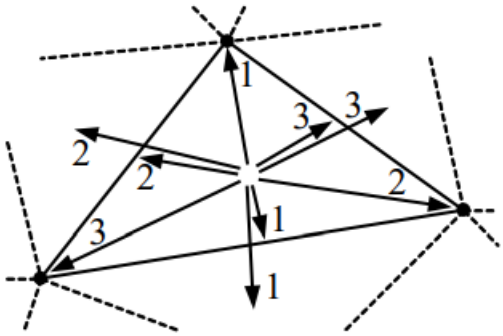



***Fig. 6.*** Triangle relationships in the Nodes, Simple Edges and Triangles data structure

The disadvantage of this structure is a relatively large memory consumption, amounting to about 88 * N bytes with an 8-byte representation of coordinates and 4-byte pointers. To conclude this section, Table 1 summarizes the characteristics of the given data structures, including the memory cost and the degree of representation of the various elements of the triangulation ("–" - the element is absent, "+" - present, "±" - present, but no links to other elements triangulation). In general, it can be noted that the Nodes with Neighbors structure is less convenient than the others, since it does not explicitly represent edges and triangles. Among the rest, the Nodes and Triangles structure is quite convenient in programming. However, some triangulation algorithms require the explicit representation of edges, so the Nodes, Edges, and Triangles structure can be recommended there.

***Table 1.*** **Main characteristics of data structures**

| Data structure name | Memory | Knots | ribs | triangles |
|---|---|---|---|---|
| "Nodes with Neighbors" | 44*N | + | - | - |
| "Knots and Edges" | 40*N | ± | + | - |
| "Double Ribs" | 64*N | ± | + | ± |
| "Knots and Triangles" | 64*N | ± | - | + |
| "Knots, Edges and Triangles" | 88*N | ± | + | + |
| "Knots, simple edges and triangles" | 88*N | ± | ± | + |

**Converting Data Structures.** The problem of changing the structure in which the triangulation is represented can arise, for example, when constructing a greedy or optimal triangulation. The algorithms for their construction operate only with edges and nodes, and therefore they are forced to use data structures like "Nodes with neighbors" or "Nodes and edges". On the other hand, the purpose of building a triangulation may be surface modeling, which requires a data structure, such as "Knots and Triangles". That is why the problem of transition from one data structure to another arises. First, let's consider a fairly simple algorithm for moving from the "Nodes and Edges" structure to the "Nodes with Neighbors" structure. The main goal of this algorithm is to calculate edges adjacent to nodes. Algorithm for converting the data structure "Nodes and edges" into the structure "Nodes with neighbors" Data structures. The initial structures are represented by arrays of Nodes and Ribs. As a result, we should get an array of NewNodes. The algorithm will require a temporary array R of length N to count the number of edges adjacent to the corresponding nodes.

Step 1. Calculate the number of adjacent edges R[i] included in each i-th triangulation node. To do this, we first assign i: R[i]:=0. Then, in a loop over i, we look through all the edges and for each edge Ribs[i] connecting nodes with numbers a=Ribs[i].Nodes[1] and b=Ribs[i].Nodes[2], we increase the counters of edges in nodes : R[a]++, R[b]++.∀

Step 2. Allocate memory for each node of the Nodes with Neighbors structure, using R[i] as the number of nodes in the node's Nodes array. As a result, we get new entries NewNodes[i]. In NewNodes[i] we copy the fields with X,Y coordinates from the corresponding fields of Nodes[i] of the originaldata structures "Nodes and edges". Set other fields to zero for now: NewNodes[i].Count:=0, j: NewNodes[i].Nodes[j]:=0.∀

Step 3. In the loop over i, we look through all the edges and for each edge R[i] connecting nodes with numbers a and b, add to the nodes links to the node adjacent through this edge and increase the counters of adjacent nodes in new nodes:

NewNodes[a].Nodes[NewNodes[a].Count]:=b; NewNodes[a].Count++;

NewNodes[b].Nodes[NewNodes[b].Count]:=b; NewNodes[b].Count++.

***End of the algorithm.*** The complexity of the described algorithm is linear in the number of triangulation nodes. The next algorithm for moving from the "Knots and Edges" structure to the "Knots and Triangles" structure is more complicated. It requires the creation of interconnected triangle structures. The first 3 steps of this algorithm almost coincide with the previous algorithm: in it, a special temporary data structure is created for each node. Algorithm for converting the "Nodes and Edges" data structure into the "Nodes and Triangles" structure. Data structures. The initial structures are represented by arrays of Nodes and Ribs. During the operation of the algorithm, a temporary array R of length N is required to count the number of edges adjacent to the nodes. In addition, we will need a temporary modified data structure "Nodes with neighbors" (extended with a list of adjacent edges and triangles, but without coordinates, since we can get them from the original Nodes array), which will be represented in the TmpNodes array. This temporary structure should have

the following view:

TempNode = record

Count: integer; ← number of adjacent nodes

Nodes: array [1..Count] of NodeNumber; ← list of adjacent nodes

Ribs: array [1..Count] of RibNumber; ← list of adjacent edges

Trns: array [1..Count] of TriangleNumber; ← adjacent triangles

end;

In this structure, the edge TmpNodes[i].Ribs[j] must connect to the node TmpNodes[i].Nodes[j], and the triangle TmpNodes[i].Trns[j] must lie between the edges adjacent to the node, defined by the edges TmpNodes[ i].Ribs[j] and TmpNodes[i].Ribs[j mod TmpNodes[i].Count+1]. The algorithm will need another temporary structure that introduces additional fields for each triangulation edge and which will be provided for the edges in the TmpRibs array:

Temporary Edge = record

IndexInNode: array [1..2] of integer; ← edge numbers in the Ribs lists of edge nodes

Trns: array [1..2] of TriangleNumber; ← adjacent triangles

end;

In this data structure, the triangle TmpRibs[i].Trns[1] must lie on the right side of the vector formed by the nodes Ribs[i].Nodes[1] and Ribs[i].Nodes[2] (Figure 7). The temporary array IndexInNode is designed to quickly determine the next and previous edge in the triangle containing this edge.

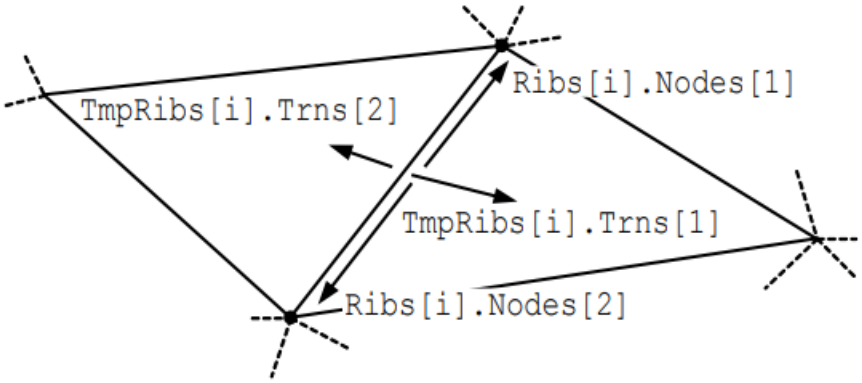



***Fig. 7.*** Temporary connections of edges in the structure transformation algorithm data "Knots and edges" into the structure "Knots and triangles"

**As a result of the algorithm, we should get an array of NewNodes**.

Step 1. Calculate the number of adjacent edges R[i] included in each i-th triangulation node. To do this, we first assign i: R[i]:=0. Then, in a loop over i, we look through all the edges and for each edge Ribs[i] connecting nodes with numbers a=Ribs[i].Nodes[1] and b=Ribs[i].Nodes[2], we increase the counters of edges in nodes : R[a]++, R[b]++.∀

Step 2. Create temporary TmpNodes[i] entries for each node, using R[i] as the lengths of the Nodes, Ribs, and Trns arrays. Other fields are filled with zeros and empty references for now: TmpNodes[i].Count:=0, j: TmpNodes[i].Nodes[j]:=0, j: TmpNodes[i].Ribs[j]:=0, j : TmpNodes[i].Trns[j]:=0.∀∀∀

Step 3. We look through all the edges and for each edge R connecting nodes with numbers a and b, add to the nodes links to R and the node adjacent through this edge R, and increase the counters of adjacent nodes in the new nodes:

TmpNodes[a].Nodes[TmpNodes[a].Count]:=b;

TmpNodes[a].Ribs[TmpNodes[a].Count]:=R; TmpNodes[a].Count++;

TmpNodes[b].Nodes[TmpNodes[b].Count]:=b;

TmpNodes[b].Ribs[TmpNodes[b].Count]:=R; TmpNodes[b].Count++.

Step 4. At each node in the temporary structure TmpNodes, sort the adjacent nodes and edges clockwise (simultaneously in the TmpNodes[i].Nodes and TmpNodes[i].Ribs arrays). For all triangulation edges array

TmpRibs[i].Trns are filled with empty links (zero numbers of triangles): i,j: TmpRibs[i].Trns[j]:=0.∀

Step 5. In the loop over i, for each node, we make a nested loop over j over all adjacent edges. For each edge, determine the number k of the node in the edge and set TmpRibs[TmpNodes[i].Ribs[j]].IndexInNode[k]:=j.

Step 6. In the i loop for each edge, we do a nested j loop through two triangles adjacent to the edge and try to create a new triangle from TmpRibs[i].Trns[j] if this triangle has not yet been created ( if TmpRibs[i] .Trns[j]=0). This triangle will connect the end nodes of the current edge (Ribs[j].Nodes[j], Ribs[j].Nodes[3-j]) and another node, which can be determined using the list of adjacent nodes in node Ribs[j] .Nodes[j] using TmpRibs[i].IndexInNode[j]. In addition, you need to define 3 edges that form a triangle and expose links from these edges to a new triangle.

Step 7. We establish mutual links of triangles to each other. To do this, we make a cycle along i along each internal edge of the triangulation (along all i-th edges with j: TmpRibs[i].Trns[j]≠0) and for it we set mutual links of adjacent triangles TmpRibs[i] .Trns[0] and TmpRibs[i].Trns[1].∀

***End of the algorithm.*** Provided that at step 4 sorting with linear complexity is used (for example, digital), then the complexity of this algorithm is linear with respect to the total number of nodes in the triangulation. In general, even if you use a non-linear sort, the complexity of the algorithm on average will also be linear - O (N). This follows from the fact that the average number of edges adjacent to a node in a triangulation is a constant independent of N, and hence sorting will run O(1) time on average. Conclusion: The above algorithm can be easily modified to obtain other data structures in which triangles are explicitly represented.

**Triangulation of mobile phone location by base stations.** There is a common misconception that the geographic location of any GSM phone can be determined with sufficient accuracy by triangulating over three base stations. This is usually described as follows: for example, if it is possible to determine the distance from the base station to the phone by standard means, then by the distances from three base stations you can get the exact coordinates of the device, and by the distance from two base stations - two points, one of which will be located desired phone. As a rule, popular rumor endows criminal elements

or law enforcement agencies with the ability to use such technology to find the people they need. Part of this statement is true. Standard means can sometimes determine the distance from the phone to one base station. This, perhaps, explains the tenacity of the belief that triangulation is possible. Actually it is not. Before starting a detailed analysis of the triangulation case, it is worth making a significant reservation. It is necessary to draw a clear line between the delusion that the base stations of any GSM network always triangulate the location of the specified phone (variant - all phones in the coverage area) and the possibility of detecting the location of the phone by other means within a single network.

**Location Based Services.** To provide location-based services (LBS), there are many ways based on the availability of additional software and hardware at all base stations of a particular network, and sometimes also in the subscriber's SIM card / phone. An example of such services: show a subscriber a map of the city and his place on it, tell the address of the nearest restaurant or store, tell the location of another subscriber, get directions to a given point. For a number of applications, it is sufficient to approximately know one base station, in the coverage area of which the subscriber is located. This can be done on any GSM network. Result: a circle with a radius of up to 32 km with a center at the installation site of some base station. In urban areas, the radius can be reduced, since the coverage areas of base stations are usually small. It is worth mentioning that information about the “current” base station is updated with every call / SMS or about once an hour, therefore, to improve the accuracy of detection, an SMS is sent to the subscriber immediately before the “measurement” or the subscriber himself is encouraged to send an SMS with a request like “ where am I/where is the nearest restaurant/hotel/subway/...”. This result can be improved with a technique called "time of arrival". All base stations in the network need to be upgraded. Result: a circle with a radius of 100-500 meters centered on the base station installation site. The use of even more advanced methods (their description can be found on the web using the keywords "angle of arrival", "uplink time difference of arrival", "GPS", "assisted GPS") allows you to further reduce the radius of the circle or move its center to real location of the subscriber. The presence of any complex subscriber location detection system in the operator's network is very easy to determine - the operator will sell the relevant services, no one will invest in the creation of the necessary infrastructure just like that. Often a cursory glance at a service's promotional material is sufficient to determine the type of technology an operator is using, simply based on detection accuracy data.

On this excursion into the technology of determining the location of the subscriber can be considered complete and return to the original topic:

1. Is it possible to triangulate the location of a phone in the GSM network by three (four, ...) base stations?
2. Who can carry out this triangulation: the subscriber, the operator, or both parties?

**Triangulation.** Let's start with an analogy. Consider the following statement: "using the ping utility, you can determine the transit time of TCP packets from one computer to another, and therefore estimate the distance between them. Then, by the distances from three computers, knowing their coordinates, you can get the coordinates of the desired computer." If we take four computers, we will connect them with network cables to each other directly, without using intermediate networks, and we will lay the wires strictly in a straight line. Will we be able to determine the coordinates of the central computer in this case, knowing the coordinates of the peripheral ones and using only ping? We can. Does this mean that this method can always be used? Certainly not. Firstly, wires rarely connect two computers directly and strictly in a straight line, and secondly, we, as a rule, do not know the exact coordinates of the “reference” computers. It will be easy to continue this list. Now back to the original statement. Is it possible to determine the distance from the base station to the phone using standard GSM network tools? The short and self-explanatory answer is "you can". Let's ask additional questions:

1. Who is doing the measurements - the base station or the phone?
2. Is such a measurement always possible?
3. Will the shortest distance between them be measured?
4. How accurate will the measurement be?

To understand who can make such a measurement, you need to figure out what the phone and the base station know about each other. It is worth dividing the description into two cases: the phone is in standby mode and the phone is in active mode (they are talking on it, receiving SMS, ...).

**Phone on standby**

Base stations regularly broadcast signals over the air so that phones can know if they are in coverage area. Phones, on the other hand, most of the time do not transmit anything, only receive, in order to save battery power. It is easy to check this in practice by placing the phone next to computer audio speakers and observing the

disturbances induced by the phone, or by buying a simple GSM signal detector key fob. It follows from this that it is impossible to determine the location of a conventional GSM telephone in a conventional GSM network at an arbitrary moment of time simply because the telephone is silent and does not "tell" anyone where it is and where it is being carried. Periodically, the phone notifies the network of where it is in order to facilitate the delivery of incoming calls. This happens:

1. when registering on the network;
2. when a subscriber moves from the coverage area of one group of base stations to another (a group may include several hundred base stations, there may be only a dozen such groups in a city of a million people);
3. periodically - once every half an hour or an hour, depending on the network settings.

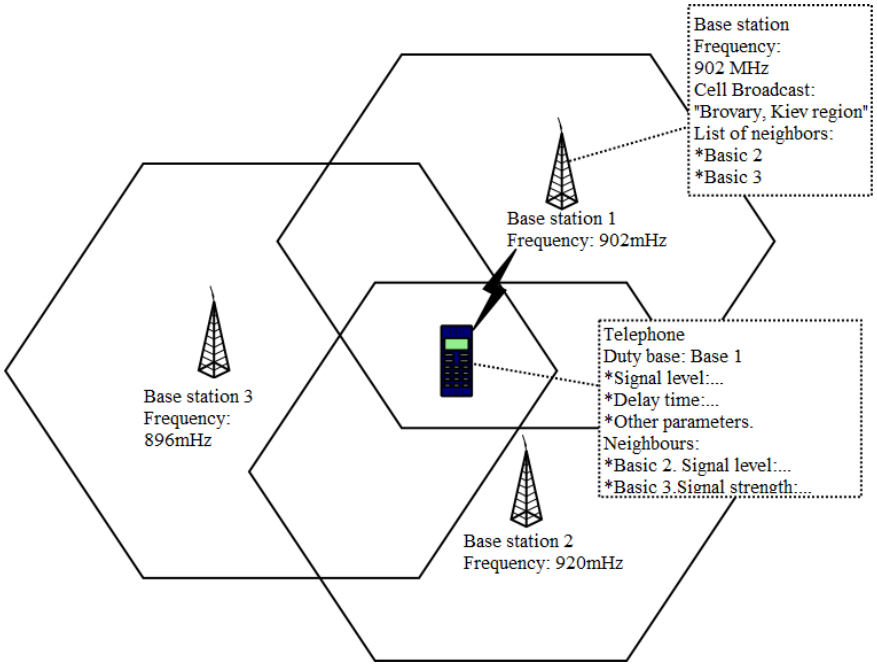



***Fig. 8.*** Base stations

In this case, the phone tells the network only about which base station it “hears” best, without any details like signal strength. Base stations do not keep track of which phones are in their coverage area, this is pointless and technically unfeasible. Accordingly, most of the time, the mobile network has only a very rough idea of where the phone currently lives. Firstly, it is not clear from which base station to measure - since the last update of the location information, the phone could have been carried away for a considerable distance. Secondly, it is unclear what and how to

measure. The base station is not a radar, and if the phone is “silent”, then it does not exist for it. So, in standby mode, a standard phone in a standard GSM network is completely invisible to the mobile network and cannot be "triangulated" by it. The phone itself is in a more advantageous position. The fact is that each base station broadcasts information about its “neighbors”, indicating the frequencies on which the nearest base stations of the same network operate. The phone in standby mode constantly measures the signal level (but not attenuation) from each of the “neighboring” base stations and, if necessary, selects the one from which the signal is “better heard” as the standby base station. If the phone has some information about where (at what coordinates) the base stations are located, then it can try to calculate the zone in which the hypothetical coverage areas of all "neighboring" base stations intersect. Somewhere within this area there will be a telephone. The more accurately the phone knows (or estimates) the boundaries of coverage areas, the more accurately this method will work. According to available information, that's how the Google Latitude app works. If there is no data on the location of the base stations, then the phone will not have any opportunity to “triangulate” its position.

**Phone in active mode**

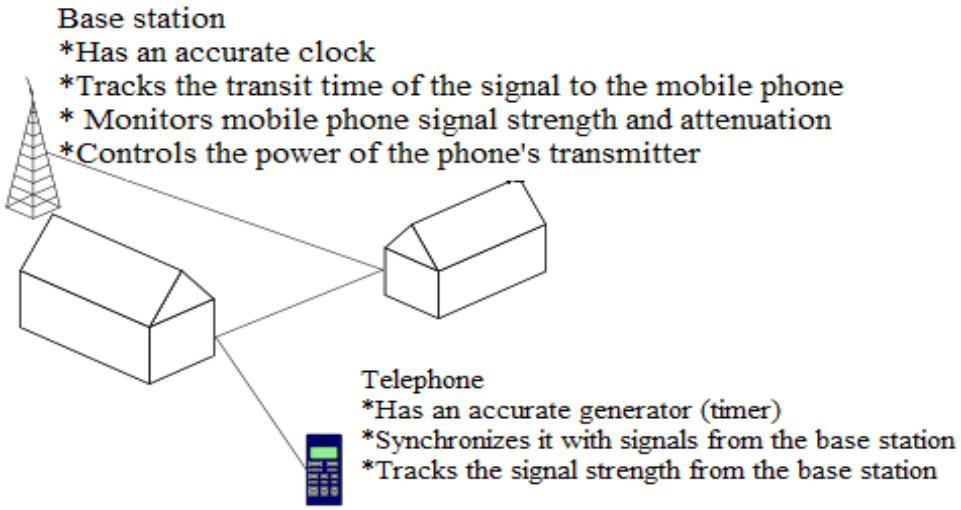



***Fig. 9***. Active Mode Schematic

In active mode, the phone sends signals to a single base station and receives response signals from it. Everyone is familiar with the fact that GSM networks can operate at frequencies of 900, 1800 and (rarely) 1900 mHz. In fact, we are talking about frequency ranges: 890-960, 1710-1880 and 1850-1990 mHz, respectively. Each base station

broadcasts only on one specific frequency from this range. Neighboring base stations, regardless of which operator they belong to, are always configured to create minimal interference with each other. In particular, neighboring base stations will never operate on the same frequency. The base station in the process of servicing the conversation performs control and regulatory functions. She is engaged in the calculation of the values of the so-called time shift (timing advance) and transmits them to the phone. The phone uses them to adjust its timer so that its and the base station's “clocks” are synchronized and the signals sent by the phone reach the base station within the “broadcast window” assigned to the phone. In order to correctly calculate the time shift, the base station measures the time it takes the signal to travel from itself to the phone, but it absolutely does not matter how many times the signal bounces off buildings and other obstacles along the way. The base station also evaluates the signal strength and attenuation of the phone and makes recommendations to the phone on the required transmission power. The phone during the conversation also has information about the time shift, the signal level from the base station and the power of its transmitter. It turns out that both the base station and the telephone can, in principle, somehow estimate the distance to each other along the path of the radio signal, but they cannot take into account all possible refractions and deviations. The accuracy of distance measurement by time shift is about 500 m.

**Conclusions** If we are not talking about a specific operator, but talking about GSM as a technology in general, then we can argue that:

1. The standard capabilities of the GSM network allow the construction of systems for determining the location of a subscriber based on measuring the parameters of the radio signal, but there is no GSM Phase 2+ standard for such systems / technologies.

2. If such a technology is not implemented in the operator's network, then by means of the network itself it is possible to determine only the last known location of the subscriber with an accuracy of the base station that serviced his call or registration in the network. You can send an SMS to your phone or call and update this information.

3. On the basis of a standard GSM telephone, it is possible to build a system for determining its location, but only if data on the coordinates of the installation of base stations are available.

4. The method of determining the location of the phone by means of the network using triangulation (in the form in which it is presented at the beginning of the article) is nothing more than a common fiction.

## References


- GSM Frequency Ranges
- Comparison of the accuracy of the work of different methods of localization of subscribers
- GSM Standard 03.22 "Functions related to Mobile Station (MS) in idle mode and group receive mode»
- GSM Standard 03.30 "Radio Network Planning Aspects»


## AUTHORS


Kyrianov Artemii – PhD student, Department of Computer Engineering, National Technical University of Ukraine “Igor Sikorsky Kyiv Polytechnic Institute”.

Heorhii Loutskii – Professor, Doctor of Technical Sciences. Head of the specialized academic council, National Technical University of Ukraine "Ihor Sikorskyi Kyiv Polytechnic Institute".

Oleksandr Chaikovskyi – PhD student, Department of Computer Engineering, National Technical University of Ukraine “Igor Sikorsky Kyiv Polytechnic Institute”.

**Oleksandr Honcharenko, Heorhii Loutskii**

# HETEROGENEOUS MULTISPACE DATAFLOW NETWORK

The article discusses a method for constructing a heterogeneous dataflow network based on the concept of a PF-network using the excess de Bruijn topology. The issues of the network structure, the main aspects of functioning, the principle of distribution of tasks and mechanisms for ensuring fault tolerance are considered. A superficial review of load balancing automation using tree decomposition and grain management was also done. A comparison with the original concept was made, the main gains, losses and prospects were identified.

**Key words:** fault tolerance, excess code, Latin square

*Fig.: 7. Tabl.: 1. Bibl.: 6.*

**Urgency of the research.** Currently, high-performance computing is increasingly at a dead end. Increasing the number of nodes increases the nominal performance of the system, as well as power consumption and space requirements, while the real performance barely increases. The reason for this is the problems of parallelism inherent in currently popular control flow systems. The solution to this problem is the transition to another paradigm - dataflow. However, this is not a panacea, because there are a number of dataflow architectures, each of which has its own advantages and disadvantages. To circumvent these problems, it is worth considering not one architecture, but a number of specialized architectural solutions combined into a single heterogeneous system. Of course, to ensure high productivity, such a system must be quite large - this raises the issue of its structure. At the moment, only distributed architecture, clusters and networks are capable of aggregating a large amount of computing resources. Thus, a heterogeneous distributed dataflow system or network is, in fact, the only acceptable solution to the given problem, which makes this issue relevant.

**Target setting.** The key characteristics for such a network are the following 4: ease of management - allocation of tasks and assignments, ease of searching for free resources, data transfer efficiency and fault tolerance. The issue of grain control is no less relevant. The main target of this research is to ensure these requirements within the framework of the proposed method or to create a basis for their implementation in the future.

**Actual scientific researches and issues analysis.** At the moment, there are a number of studies devoted to dataflow in the context of high-performance computing.

The application and advantages of dataflow computing in nonuniform networks are considered [1], the concept of an open dataflow network based on Petri net elements is proposed [2], the analysis of algorithms and implementations of systems of this type was performed [3]. At the same time, there is interest in the subject area from companies as well: for example, Maxeler Technologies produces dataflow accelerators based on FPGA, one of the areas of application of which is high-performance computing [4].

**Uninvestigated parts of general matters defining.** Although the subject area continues to develop, there are still unexplored or ignored points. One of them is the issue of granularity management, which is partially implemented in PF networks. Also, insufficient attention is paid to fault tolerance, which is critical for high-performance systems.

**The research objective.** The purpose of this study is to improve the efficiency of the PF network in the context of high-performance computing. The tasks of the research are the analysis of PF networks with the de Bruijn structure, an overview of their main properties and application for solving problems and meeting requirements, as well as - a conceptual analysis in comparison with the original solution.

**The statement of basic materials.** A typical PF network includes 3 management levels that provide dynamic parallelization [2, 5]. All of them are implemented in hardware in the form of corresponding elements and are connected by two highways along which data tokens circulate. This management structure is shown in Fig. 1.

At the upper, file level, the task is presented in the form of an executable file - the so-called p-script of subtasks, which describes complex algorithms that include data distribution (partial processing of arrays, operations on big data). It contains computational tasks of a lower order - p-scripts of formulas, between which there are dependencies on the data. The execution follows a model traditional for dataflow systems: a subtask can be started for execution when the ready condition is fulfilled for it. After execution, the result tokens are returned to the concentrator and activate dependent subtasks.

Similar manipulations occur at the operator level, where parallel operators act as objects. Each such operator is a sequential program - an s-script, processed on a functor - processor of classical architecture.


***Fig. 1.*** Management structure of PF network [5]

**Hardware structure of PF network.** The hardware components of the highest level in this kind of network are the so-called PF servers, interconnected by any network technology: for example, using Gigabit Ethernet. Each such server is a structurally and functionally complete computer, and consists of PF cells. There are 3 types of cells: A, B and C [2]. Fig. 2 shows the structure of these elements.

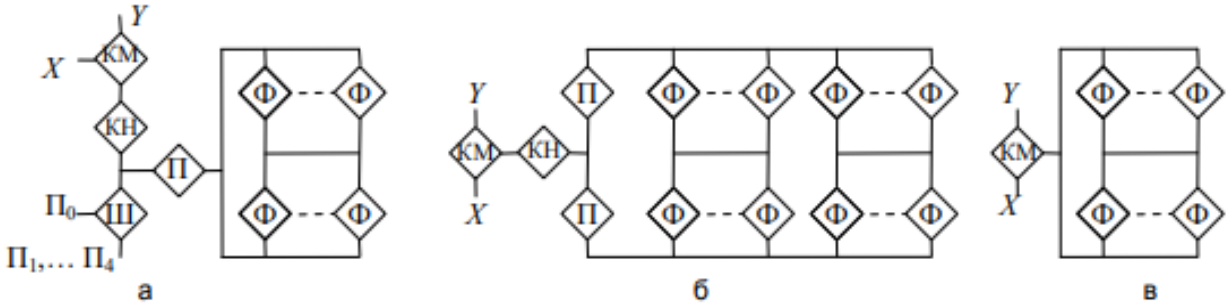



***Fig. 2.*** Structure of PF network elements [2]

The A-cell (foreground cell) is responsible for the distribution of tasks on the system (file level), as well as for the execution of those tasks that require constant uploading of data from external devices. The communicator of this element manages the distribution of tasks and, in case of failure of the background cells, redistributes the files associated with them to other devices.

B-cell (background cell) works mainly at the operator level and performs complex tasks that do not require constant data swapping - for example, processing

arrays. In the event that the B-cell lacks resources, it turns to the processing elements (C-cells), which contain only functors and all processing results are sent to the B-cell.

Combined in a certain way, these elements make up PF servers. There are various options for the structure of their computing field, including two-dimensional and three-dimensional options. In fig. 3 shows examples of the two-dimensional structure of the computing field. On the left is a linear version, where each B-cell corresponds to a line of C-cells. On the right is a matrix, where simultaneous grouping on two axes is performed, and from each B-cell it is possible to reach any processing cell in 2 steps.

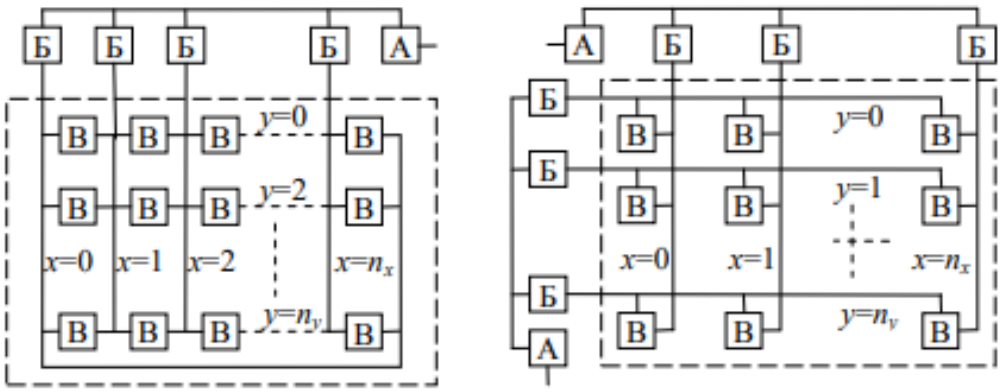



***Fig. 3***. Examples of the structure of the computing field of the PF server [2].

**Advantages of PF network conception.** The key advantage of the described solution is its scalability. The PF network is scalable at the level of servers, within each of them it allows additional elements to be connected to the buses, and within each of the elements it realizes the possibilities of separate scaling of the controlling (concentrators, schedulers) and executing (functors) parts. Similarly, this applies to scheduling: the addition of additional resources occurs automatically, and their failure is not critical for calculations and does not require human intervention.

Another interesting aspect is the distribution of management on several levels. Using this kind of dataflow-of-dataflow allows you to fine-tune the grain while sharing overhead between levels.

This makes the proposed solution extremely attractive for the construction of supercomputers and high-performance systems.

**Disadvantages of PF network.** The key disadvantage of this concept can be called homogeneity and attachment to classic processors as computers. The concept of functors involves the execution of sequential programs according to the MIMD

principle, while ignoring the possibility of combining classical processors with non-classical elements, including graphics processors, vector extensions and specialized FPGA-based chips.

In addition, the extensive use of buses in the structure of servers, on the one hand, provides good opportunities for communication and scaling, but it has its drawbacks. Thus, performing operations in a streaming system requires fairly frequent data transfers between elements, and the bus makes such transfers strictly sequential, unlike a network, where elements are able to exchange information independently of each other.

A final aspect worth noting is the presence of single points of failure in the form of foreground elements. If such an element fails, the entire line of B-cells associated with it, or even the entire PF server, will become unavailable.

**De Brujin network as basis.** From the point of view of management, the main advantage of this type of network is the simplicity of decomposition into trees. So, the classical de Bruyne topology can be decomposed into 2 binary trees, the excess one into 3 ternary trees [6]. At the same time, for different trees, the sets of non-finite nodes are different, as a result, the root nodes for one tree are finite for others, which allows the use of decomposition both for finding alternative routes and for parallel management of the system. Fig. 4 presents the de Bruyne network, built using the elements of the PF network, as well as the trees into which it is decomposed. At the same time, it is considered that the roots of trees are always A-cells, and other elements of the topology play the role of B-cells. As for C-cells, they are not represented on the topology and are considered abstracted "inside" foreground and background nodes.

In the context of building an effective dataflow network, this kind of property simultaneously solves several problems. On the one hand, independent trees make it possible to load the network with tasks from several directions at once, involving different nodes and in a different sequence, thus avoiding conflicts during transmission. Moreover, the tree-like structure of the hierarchy allows higher-order nodes to abstract from control aspects at lower levels, allowing for more precise control of the load on nodes. On the other hand, network connectivity makes it possible to balance the load on trees by redistributing tasks and subtasks. Another bonus is fault tolerance and static level 4, which makes it possible to implement such a system without unnecessary hardware costs.

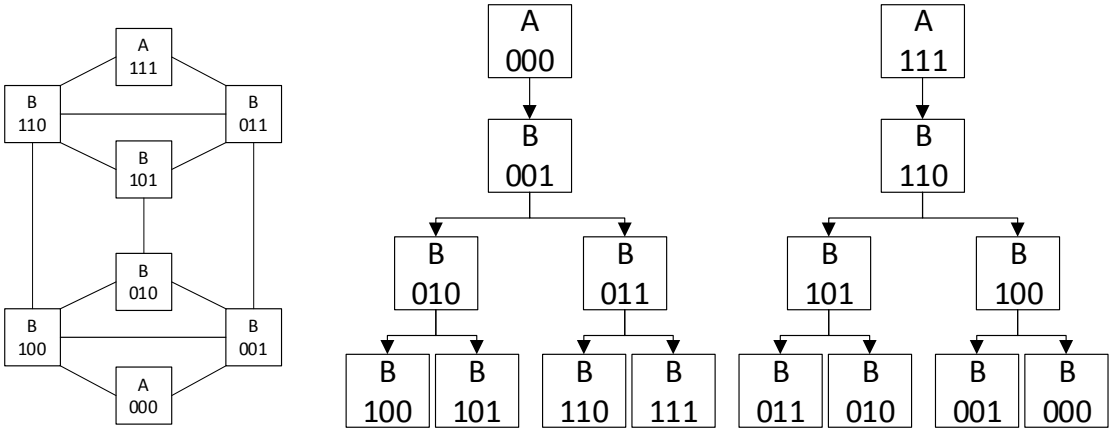



***Fig. 4.*** De Bruyne topology and its decomposition into trees, implemented in the form of a PF network [6]

However, the excess de Brujin network is much more interesting. Unlike the classic one, it uses redundant binary representation to encode nodes (RBR). In fig. 5 shows the model of the PF network based on the redundant topology of de Brujin together with the trees of its decomposition.

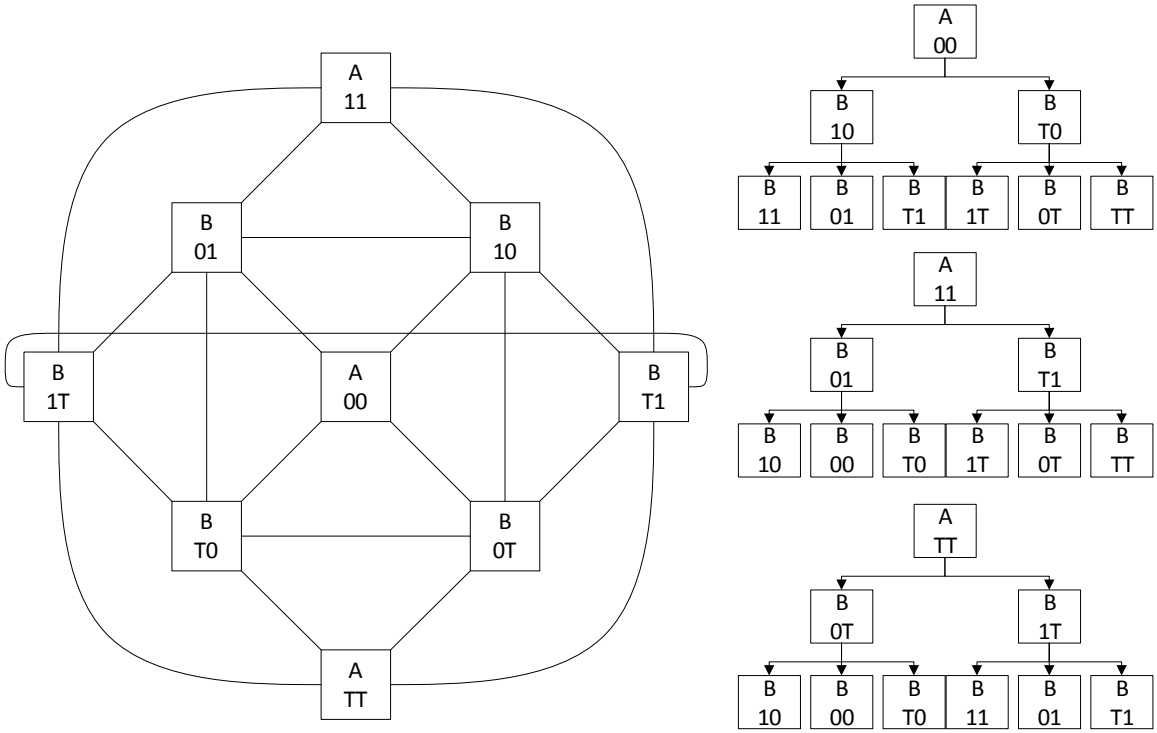



***Fig. 5.*** Excess de Brujin PF network and its decomposition

The specificity of this network compared to the previous one is the property of non-uniqueness of number representations, which can be used to abstract resources

inside a "logical" binary node. Its following properties contribute to this:

1. Each excess de Brujin network contains exactly 2 classical de Brujin subnetworks of the same rank. At the same time, these two subnets always have only one common node - node number 0. For the above network, these are the subnets 00-01-10-11 and 00-0T-T0-TT.

2. In addition to nodes included in 2 subgraphs, there is also a "hidden space" that contains nodes with the same numbers but different codes.

This allows you to logically divide the network into 3 parts - "spaces":

1. An open (direct, positive) space containing nodes that are part of a "positive" binary subgraph. These nodes contain only the numbers 0 and 1 in their code, and are therefore ideal candidates as a "facade".

2. A closed (inverse, negative) space containing the nodes of the "negative" subgraph. Is a complete copy of the "direct" space. This allows you to treat it as a virtual independent device or cluster with the same characteristics as the host system and use it for both balancing and redundancy. At the same time, in contrast to normal redundancy, in this case all working nodes of the main system remain available through redundancy, since they are hardware-only, which makes such redundancy much more effective.

3. Hidden space. Contains nodes with mixed codes. At the same time, due to the properties of RBR, for each hidden node there is one and exactly one node with the same number that is included in the direct or inverse space. This makes it possible to consider hidden nodes as an analogue of C-cells, which expand the computational capabilities of higher-order nodes, but at the same time, each such cell will have its own unique identifier, the node code, which allows it to be accessed not only directly from the owner node, but also and in a bypass, using a common network.

The only exception to this separation is node 0, which enters both open and closed space at the same time. On the one hand, this makes it a good candidate as a main node, but it also makes it the most vulnerable part of the system. This problem can be easily solved using classic redundancy. However, even it is not the only point of failure: thanks to the properties of decomposition, all serviceable nodes of the system are guaranteed to be available, provided that the number of failures in the system does not exceed 2.

**Fault tolerance and load balancing.** Applying such a separation allows you to apply the following work model. In normal mode, the two parts of the system work

independently. At the same time, node 0 is hardware duplicated: one element serves the "positive" subsystem, and the other serves the "negative". Thus, the two parts of the system will operate relatively independently.

What is the benefit of this? First of all, it allows you to load the system in parallel: on the one hand in 2 spaces at the same time, on the other - in each space from 2 directions. This makes it possible to talk about 4 independent "waves" or "streams" of computing tasks, spreading through the system and filling it with data, starting from the roots of the tree. In fig. 6 shows an example of task distribution over the network.

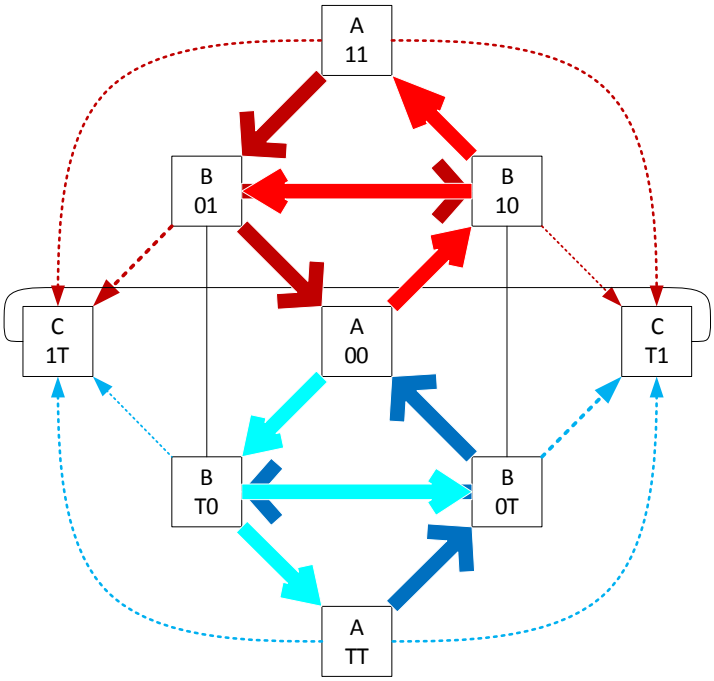



***Fig. 6.*** Distribution of tasks throughout the system.

At the same time, nodes 00, 11 and TT perform the role of A-cells. Other nodes - 01, 10, 0T and T0 - perform 2 roles at once: from the point of view of planning, they are responsible for grinding the grain of the task for transferring tasks to the next level, and from the point of view of calculations - performing those tasks that do not require a large amount of data swapping. As for "hidden" nodes 1T and T1, their role is to calculate specialized tasks and expand the capabilities of neighboring nodes. At the same time, nodes with the same number have priority in accessing their resource.

If a failure occurs, the following is done. If the failure is insignificant (for example, among hidden nodes or in the middle of the tree so that it does not destroy the tree completely) - it can be simply ignored. This will slightly reduce the speed of the system, but

not significantly: since the load streams intersect, the nodes lost to one will immediately become less loaded and more attractive to the other, which automatically balances the load. In the case when the failure is significant, there are several of them and this blocks the very possibility for the stream to continue its work in regular mode - the solution will be to switch to ternary trees. This will allow you to bypass the problem by using hidden nodes and access those parts of the system that are isolated. Fig. 7 shows these 2 cases of failure: on the left - insignificant failure, on the right - transition to ternary trees.

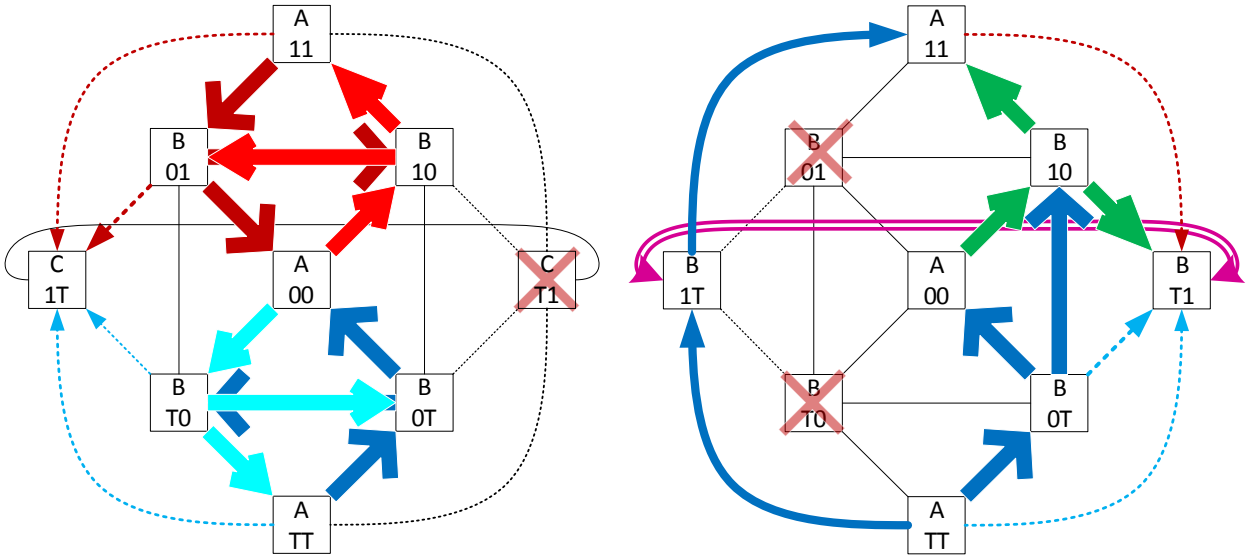



***Fig. 7.*** Solving the problems of fault tolerance.

**Results and discussions.** Although at this stage of research it is very difficult to evaluate the proposed method, nevertheless, based on the general properties, it is possible to analyze and qualitatively compare the original concept of PF networks and the proposed idea of a multi-space network. Table 1 shows the main characteristics of the system that follow from these two concepts.

Analyzing these properties, it can be seen that the only significant conceptual drawback is the limitation of scaling. This is normal given the topology binding. However, on the other hand, it makes it possible to significantly increase such important characteristics as fault tolerance and ease of resource search, as well as to make shipments much more independent thanks to the combination of bus and direct connections between elements. Another achievement is the distribution of management and loading: a total of 3 nodes manages the system, and as long as at least 1 is functioning, the system is able to fully perform its tasks.

***Table 1.*** **Properties of the system provided by the concept**

| Property | Basic SF network | Proposed network |
|---|---|---|
| Grain management | On 2 management levels | At 2 management levels and/or at each tree level |
| Count of loading "streams" | A maximum of 3 with a 3-dimensional field structure | 4 in standard mode, 3 – when switching to ternary trees |
| Roles of cells | Static, defined by the structure of the element. | Partially static: all nodes contain concentrators, schedulers, and functors, but their role is determined by their place in the tree. |
| Interchangeability of elements | Only within one line | Any element can act as a background. |
| What is a node in the model? | A specific element – the cell | Abstraction - both a specific cell and a certain group of specialized computers supplemented with control elements of the PF network. |
| Availability of resources in case of failure | Medium: When A-cells fail, access to associated background cells is lost. However, other cells can be easily replaced if they fail. | High: As long as at least one A-cell is working and connectivity is not broken, all viable nodes will be reachable. Substitution of nodes is also maximal. |
| Network topology | Multibus | Excess be Brujin with additions |
| Ease of scaling | Free: elements can be connected to any places of the network without changing its functioning. | Scaling is by topology, so you can't scale the system arbitrarily. |
| Ease of searching for a free resource | Via bus request | Through tree descent and bus request within the same number |
| Parallelism of transfers | Low: work on the bus always happens sequentially | High: the only limitation is the impossibility of parallel reception or transmission of 2 messages by one element. |

**Conclusions.** The article proposes a method of constructing a heterogeneous dataflow network, the nodes of which are divided into 3 "spaces" and perform different roles. This allows parallel loading of the system in 4 different directions, treating the 2 halves of the system as relatively independent devices, while allowing devices in one half to access devices in the other.

One of the key advantages of the proposed solution is the good potential for grain control. Since the network can be decomposed into trees (both binary and ternary), this makes it possible to divide the original task into parts - task packages, which will also be divided when descending the tree, which will allow automating load balancing and achieving maximum nodes utilization. Other positive aspects are high fault tolerance and load parallelism, thanks to which the system is filled with tokens not from one, but from 3 points at once, which minimizes latency.

The key disadvantages of the proposed solution are its closedness in scaling, as well as the use of RBR, which complicates the hardware structure.

This approach has great potential for development. A key issue is managing the granularity in task distribution across the tree. Solving this problem will make it possible to implement not only automatic parallelism, but also automatic load balancing and automatic grain management, which may ultimately combine threaded, coarse-grained and classic dataflow, realizing the advantages of each architecture.

## References

1. Klimov A. V., Levchenko N. N., Okunev A. S. Benefits of dataflow computation model within nonuniform networks //Informatsionnye Tekhnologii i Vychslitel'nye Sistemy. – 2012. – №. 2. – С. 36-45.

2. Ланцов Р. А. Открытые dataflow-системы с сетевой структурой //Научно-технический вестник информационных технологий, механики и оптики. – 2018. – Т. 18. – №. 6. – С. 1023-1033.

3. Milutinovic V. et al. DataFlow Supercomputing Essentials. – Cham : Springer, 2017.

4. Stroobandt D. et al. An open reconfigurable research platform as stepping stone to exascale high-performance computing //Design, Automation & Test in Europe Conference & Exhibition (DATE), 2017. – IEEE, 2017. – С. 416-421.

5. Ланцов Р. А. Основы параллельного управления в ПФ-сетях //Вестник Казанского государственного технического университета им. АН Туполева. – 2017. – Т. 73. – №. 1. – С. 96-105.

6. Olexandr G. et al. Routing method based on the excess code for fault tolerant clusters with InfiniBand //International Conference on Computer Science, Engineering and Education Applications. – Springer, Cham, 2019. – С. 335-345.

## AUTHORS

**Honcharenko Oleksandr Oleksiyovytch –** PhD student, Department of Computer Engineering**,** National Technical University of Ukraine “Igor Sikorsky Kyiv Polytechnic Institute” (Solomenskiy district, ave. Pobedy, 37, 03056, Kyiv, Ukraine).

**E-mail:** alexandr.ik97@ukr.net

**ORCID ID**: https://orcid.org/0000-0002-9086-6988

**Loutskii Heorhii Mychailovytch –** professor, PhD, Department of Computer Engineering**,** National Technical University of Ukraine “Igor Sikorsky Kyiv Polytechnic Institute” (Solomenskiy district, ave. Pobedy, 37, 03056, Kyiv, Ukraine).

**E-mail:** georgijluckij80@gmail.com

**ORCID ID**: https://orcid.org/0000-0002-3155-8301

**Oleksandr Pustovit, Rusinov Volodymyr,**
**Oleksii Cherevatenko, Leonid Pustovit, Artem Volokyta.**

# ISOEFFICIENT CALCULATION METHOD FOR DISCRETE FOURIER TRANSFORM

The paper considers the issue of isoefficiency of MPP systems and heterogeneous CPU-GPU systems on the problem discrete Fourier transform. The development of parallel applications as its goal can have not only reduction of execution time, but also provision of opportunities to solve problems of greater dimensions. Feature parallelization of the algorithm includes the effective use of hardware when increasing the dimensionality of the problem an important characteristic of parallel computing.

**Key words**: isoefficiency, heterogeneous calculations, Fourier transform

**Relevance of the research topic.** The creation of iso-efficient systems allows you to deploy a system for some task taking into account its efficiency. The efficiency of parallel computing depends on the algorithm for implementing the task mapping on the system. The purpose of this article is to consider the process of creating an isoefficient system for solving practical problems using the Fourier transform as an example, put the algorithm in the MPP system and use an empirical approach based on machine learning to approximate the isoefficiency function for systems with different architectural solutions.

The task that will be performed on a parallel and heterogeneous system is the discrete Fourier transform algorithm. The discrete Fourier transform is used to solve problems of spectral analysis - to study a signal through its division into a set of signals. The nonlinear complexity of this algorithm is interesting from the point of view of parallel processing.

The isoefficiency function will be described based on the results of modeling the problem on the considered systems. For a parallel MPP system, the result can be obtained analytically. For a heterogeneous CPU-GPU system, it is currently impossible to obtain the isoefficiency function analytically, instead, an approach using machine learning algorithms will be applied.

**Actual scientific researches and issues analysis.** There are a number of scientific publications that study the subject of isoefficient systems based on the MPP architecture [2, 3]. Methods of scaling parallel algorithms for solving certain applied problems for their display on the MPP of a system with a given topological

organization are considered. The approach of creating iso-efficient systems allows you to analyze the algorithm for its quality and parallelization capabilities on a given topology and, as a result, the capabilities to scale the system with the expected efficiency of the task.

Most modern supercomputers and large data centers use systems that have both central processing units (CPUs) and graphics processing units (GPUs) on the nodes. General Purpose GPU Computing (GPGPU) has opened the way for PC users to experiment and work on projects that involve GPUs to handle labor-intensive tasks. Heterogeneous systems based on CPU and GPU have been widely researched and are quite a promising direction for the development of parallel computing. Despite the proliferation of CPU-GPU-based systems, to date isoefficiency has not been investigated with respect to heterogeneous systems.

**The statement of basic materials.** Isoefficient systems are systems with a given efficiency of solving problems, which, being described in advance, is constantly maintained. Let's consider the theoretical possibility of creating such systems.

The formula for the efficiency of parallel processing on a parallel system is as follows, determining the possibility of achieving the required efficiency of parallel computing systems:

$$E = \frac{S}{N} \quad (1)$$

By varying the parameters n, that is, the dimension of the problem, and N, the number of processors, it is possible to achieve a linear increase in productivity with an increase in the number of processors. This means that when performing computational processes, it is possible to determine in advance the necessary efficiency of their implementation [4].

The task completion time is predicted using "precedential" information - how long has it taken for the task of the same type to be completed. A task scheduler is created, which forms the initial result, which is simulated from the analysis of already completed tasks, then the model is used to predict the time of completion of new tasks. The scheduler analyzes the size of the data with which tasks work and their execution time, and also takes into account the time of previous tasks of the same type.

Using a similar deterministic correlation model, the execution time of a task of the same type is predicted, where the input parameter is the size of its data.

The problem that will be simulated on the MPP system and the heterogeneous CPU-GPU system is the Discrete Fourier Transform. The implementation of the

discrete Fourier transform (DFT), which is the basis of spectral analysis, is an informal representation of signals, i.e. the investigated signals are represented by a sequence of counts x(k) .

$$F_x(p) = \sum_{k=0}^{N-1} x(k) e^{-jk\Delta t p \Delta\omega} \tag{2}$$

$$\omega \rightarrow \omega_p \rightarrow p\Delta\omega \rightarrow p;\ \Delta\omega = \frac{2\pi}{T} \tag{3}$$

It can be seen from the formulas that the signal presentation intervals are equal to 2, which is the period of low frequencies. To increase the accuracy, it is necessary to increase the interval T.

$$t \rightarrow t_k \rightarrow k\Delta t \rightarrow k;\ \Delta t = \frac{T}{N} = \frac{1}{k_{req}} f' \tag{4}$$

The DFT is a simple calculation procedure of the "matching" type, the estimate of its complexity is: N2 + N. To implement it, you need to calculate the turning coefficients of the DFT:

$$W_N^{pk} = e^{-jk\Delta t \Delta\omega p} \tag{5}$$

These rotation coefficients are recorded in the ROM, that is, they are constants.

$$W_N^{pk} = e^{-jk\frac{T}{N}p\frac{2\pi}{T}} = e^{-j\frac{2\pi}{N}pk} \tag{6}$$

In formula (4), $W_N^{pk}$ (rotation coefficients) do not depend on T, but only on the dimension of the transformation N, therefore they are not presented in exponential form, but in trigonometric form

$$W_N^{pk} = cos\left(\frac{2\pi}{N}pk\right) - jsin\left(\frac{2\pi}{N}pk\right) \tag{7}$$

The rotation coefficients are repeated, for this reason, the changes in values are described exactly to the specified values: p – up to (N-1), k – up to (N-1), with a period of N(2π). When taking out the sign of the coefficient, only half of the coefficients can be stored. The real and imaginary parts of the coefficients are stored separately in the ROM [6].

In its general form, the DFT can be represented as follows:

$$F_x(p) = \sum_{k=0}^{N-1} x(k) W_N^{pk} \tag{8}$$

From such a formulaic definition, it is appropriate to present the DPF in the form of a graph.

CUDA (Compute Unified Device Architecture) is a parallel computing platform and API developed by Nvidia. It allows developers to use Nvidia graphics cards for

general computing. The CUDA platform provides the developer with direct access to the system of video card commands and elements of parallel computing, for the execution of computing kernels (compute kernel).

The computing core is the main working unit, with the help of which the developer describes the algorithm. This term is not only used for GPUs, it is also used for FPGAs, TPUs, DSPs. For CUDA, the programming paradigm is very closely integrated with vector computing, based on the assumption that a kernel call is executed concurrently in a number of independent elements, allowing parallelism at the data level. However, there are also atomic operations that can be used to synchronize between elements. Each call receives indices for 1 or more dimensions, which are used for data addressing or buffering [7].

The architecture of the GPU belongs to the SIMD class, that is, data is sent to each core mentioned above, on which one operation is performed in one cycle. As an example, it is suggested to examine the TU102, which is the basis of the mainstream GPU RTX 2080Ti and in the professional GPU Quadro RTX 6000. It consists of 6 clusters of graphics processing, 36 clusters of texture processing and 72 Streaming Multiprocessors (SM). SM consists of 64 CUDA cores, 8 tensor cores, 256 kilobytes of register file, 4 Texture Units, 96 kilobytes of shared memory. Before that, each core has access to 6144 kilobytes of L2 cache.

**Results.** To study the efficiency of calculations, it is necessary to use metrics of the execution time of the algorithm sequentially (on one processor) and in parallel (on several processors). Based on the obtained time values, the effectiveness of the parallel algorithm can be investigated.

The first considered MPP system – a hypercube of degree 1. MPP (decoded as massive parallel processing) is a massively parallel architecture of computer systems. In this type of architecture, memory is physically separated. The system contains separate blocks (modules), inside which there are a processor, communication processors (routers), a local memory bank, network adapters, hard drives, input/output devices.

Only processors from the same module have access to the RAM of a separate node. Blocks are connected to each other by communication channels. It is possible for users to obtain the number of the processor and the processors to which it is connected, after which data exchange between them can be initiated.

The main advantages of systems with MPP architecture: good scalability, no need for clock synchronization of processors due to the fact that in each block only

"own" processors have access to the local RAM bank, high performance and effectiveness, practically proven on MPP -machines with a large number of processors (several thousand) [8].

The hypercube is one of the most widespread topologies, in particular for MPP systems, and has well-described characteristics and information on its application for various tasks. This topology is a special case of the grid structure, when there are only two processors for each dimension of the grid (that is, the hypercube contains 2N processors with dimension N) [9].

The hypercube topology is quite widespread in practice when combining parallel processors. A line connecting two nodes defines a one-dimensional hypercube. A square formed by four nodes is a two-dimensional hypercube, and a cube with eight nodes is a three-dimensional hypercube, etc. Since the system consists of several processor elements with local memory, the time spent on data transfer must be taken into account when performing the task. The following diagram shows the data transfer algorithm between nodes during DFT execution for 4 signals.

Based on the algorithm (Fig. 1), we will set the time functions of sequential processing and parallel processing

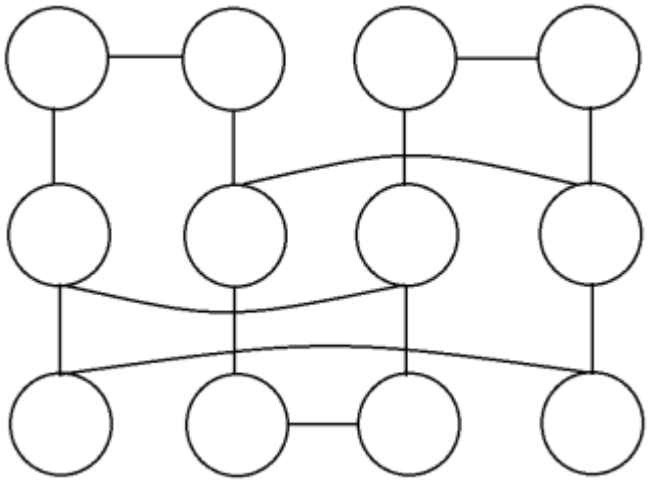


***Fig. 1.*** Interaction diagram of processors for the MPP system with parameters N and n=4

Let's apply the acceleration and efficiency formulas to obtain the analytical isoefficiency formula.

$$T_{1n} = n + k_n n \tag{9}$$

$$T_{pn} = n + k_n \frac{n}{N} \tag{10}$$

$$E_n = \frac{T_1}{NT_p} = \frac{1+k_n}{N+k_n} \quad (11)$$

where N is the number of processors, n is the dimension of the problem, kn is the dimension factor. The dimensionality factor can be calculated using the following formula:

$$k_n = 2k_{\frac{n}{2}} \quad (12)$$

$$k_2 = 1 \quad (13)$$

For the example shown in Figure 1, where N=4 and n=4, the value of parallel processing time will be:

$$T_{pn} = n + k\frac{n}{N} \quad (14)$$

Several different systems were used to obtain heterogeneous system results, listed in (table 1). The systems presented use different GPUs and CPUs, which adds complexity to the analytical approach to establishing isoefficiency.

***Table 1.***

| № | Процесор | Графічний процесор |
|---|---|---|
| 1 | AMD Ryzen 9 3900X | RTX 2060 |
| 2 | AMD Ryzen 5 2400G | GTX 1060-3GB |
| 3 | Intel Core i5-7200U | Geforce 940MX |
| 4 | Intel Core i5-9600KF | GTX 1080 |

First, it is necessary to set the execution time of tasks with the same dimension on the processor and graphics accelerator (Fig. 2). Based on the received data, a model will be developed that will allow you to distribute the load between processors in order to complete the task as quickly as possible.

The first thing that can be established from the graphs is the nonlinear complexity of the problem-solving algorithm. This necessitates the use of machine learning algorithms for the approximation of a nonlinear function. You can also see the difference between how the time increment changes with the change in the dimension of the problem. Before applying machine learning to distribute tasks between the processor and the graphics card, it is necessary to establish the data transfer time between the general memory of the system and the memory on the GPU.

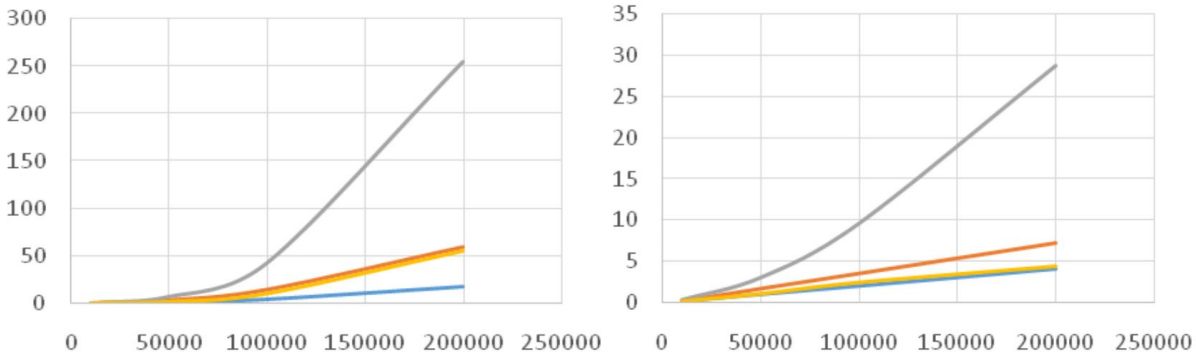



***Fig. 2.*** Task execution time on CPU (left) and GPU (right) [1]

Using an approach based on polynomial regression, let's establish the distribution of the problem's dimension on the CPU and GPU. Based on the time metrics of task execution on systems involving both processors, it is possible to empirically establish the efficiency of the system. Consider the time it takes to send data from the GPU memory to the shared memory (Fig. 3).

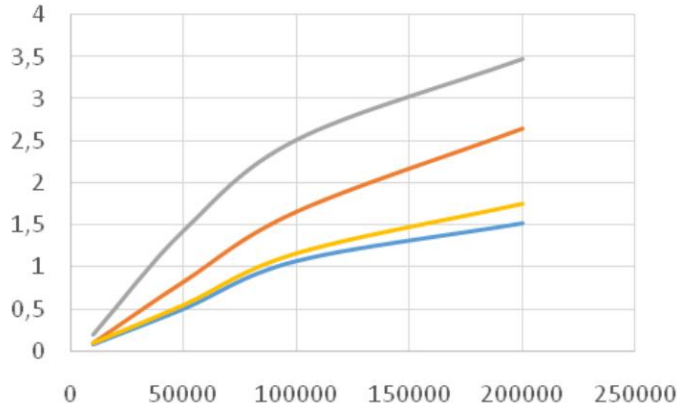



***Fig. 3.*** Time spent on sending data from GPU to shared memory [1]

Based on the diagram of the obtained time data (Fig. 4), it is possible to establish the efficiency of heterogeneous systems and establish the necessary dimension of the problem to obtain the same efficiency on another system.

The efficiency of the system can be established based on the time of execution of the task simultaneously on the processor and on the video accelerator (Table 2). The proposed approach below uses the distribution of the problem dimension across processors to establish the overall performance of a heterogeneous system in terms of speedup relative to the fastest processor of the two presented.

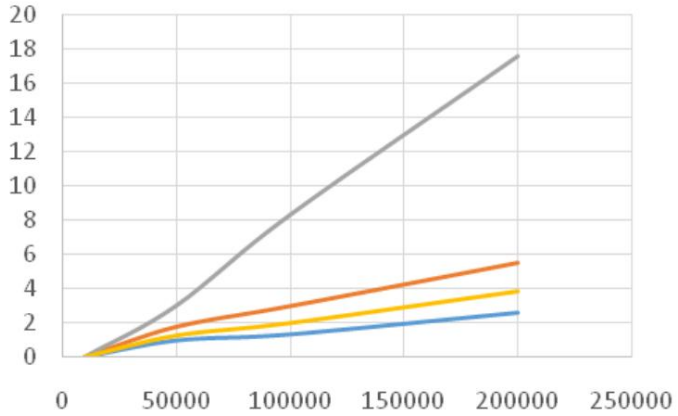



***Fig. 4.*** Task execution time on a heterogeneous system [1]

**Conclusions**. In the course of the research, the main result can be considered that heterogeneous CPU-GPU systems can be used for iso-efficient computing. The main advantage of this approach is predictability when building systems oriented to a specific task, within the framework of this work, such a task is the discrete Fourier transform. Based on the proposed approach, it is possible to scale the system due to accelerators based on a different architecture and develop isoefficient algorithms.

The simulation results show the nature of performing the calculation of the problem of nonlinear complexity. The use of machine learning methods, namely polynomial regression, allows you to create an algorithm for dividing the task between CPU and GPU while preserving the acceleration factor. From the results, it can also be concluded that the system with the most powerful processor shows the best results for n = 100000, while the results for n = 50000, 200000 are more balanced.

***Table 2.***

| № | Система | Графічний процесор | n = 50000 | n = 100000 | n = 200000 |
|---|---|---|---|---|---|
| 1 | AMD Ryzen 9 3900X | RTX 2060 | 1.004133 | 1.419191 | 1.446006 |
| 2 | AMD Ryzen 5 2400G | GTX 1060-3GB | 1.011028 | 1.360679 | 1.402923 |
| 3 | Intel Core i5-7200U | Geforce 940MX | 0.931325 | 1.228246 | 1.63615 |
| 4 | Intel Core i5-9600KF | GTX 1080 | 1.014851 | 1.245034 | 1.465462 |

## References

1. V. Rusinov, O. Cherevatenko, L. Pustovit, O. Pustovit, Method of Development of Isoefficient Heterogeneous System Using Machine Learning for the Problem of Discrete Transformation of Fourier, Herald of Khmelnytskyi National University, 297.3 (2021), 19–24

2. Hwang K. Scalability and programmability of massively parallel processor. Parallel Processing: CONPAR 94-VAPP VI. Springer, Berlin, Heidelberg, 1994. P. 1–4.

3. Grama A. Y., Gupta A., Kumar V. Isoefficiency: Measuring the scalability of parallel algorithms and architectures. IEEE Parallel & Distributed Technology: Systems & Applications. 1993. Vol. 1. Issue 3. P. 12–21.

4. Drozdowski M., Singh G., Marszalkowski J. M. Isoefficiency Maps for Divisible Computations in Hierarchical Memory Systems. PPAM (1). 2019. P. 224–234.

5. Ostertagová E. Modelling using polynomial regression. Procedia Engineering. 2012. Vol. 48. P. 500–506.

6. Bracewell R. N., Bracewell R. N. The Fourier transform and its applications. New York: McGraw-Hill, 1986. Vol. 31999. P. 267–272.

7. Harish P., Narayanan P. J. Accelerating large graph algorithms on the GPU using CUDA. International conference on high-performance computing. Springer, Berlin, Heidelberg, 2007. P. 197–208.

8. Hey T., Scott C., Surridge M. Simulation and modelling applications on mpp systems. Massively Parallel Processing Applications and Development. Elsevier, 1994. P. 15–21.

9. Yongchang J. et al. A scalability metric for algorithm-machine on NOW and MPP. Proceedings Fourth International Conference/Exhibition on High Performance Computing in the Asia-Pacific Region. IEEE, 2000. Vol. 1. P. 405–407.

## AUTHORS

**Volodymyr Rusinov –** PhD student, Department of Computer Engineering, National Technical University of Ukraine “Igor Sikorsky Kyiv Polytechnic Institute”.

**Oleksii Cherevatenko –** PhD student, Department of Computer Engineering, National Technical University of Ukraine “Igor Sikorsky Kyiv Polytechnic Institute”.

**Leonid Pustovit –** PhD student, Department of Computer Engineering, National Technical University of Ukraine “Igor Sikorsky Kyiv Polytechnic Institute”.

**Oleksandr Pustovit –** PhD student, Department of Computer Engineering, National Technical University of Ukraine “Igor Sikorsky Kyiv Polytechnic Institute”.

**Artem Volokyta** – associate professor, PHd, Department of Computer Engineering, National Technical University of Ukraine “Igor Sikorsky Kyiv Polytechnic Institute”.

**Mykyta Melenchukov,**
**Artem Volokyta, Olga Rusanova**

# METHOD FOR CALCULATING GAUSSIAN FUNCTIONS TO SOLVE THE PROBLEM OF IMAGE BLUR ON A HETEROGENEOUS SYSTEM

The article examines the Gaussian image blurring method using heterogeneous system.

**Keywords:** Gaussian function, heterogeneous system, CPU, GPU.

**Relevance of the research topic.** Heterogeneus computations is underdeveloped currently and there are many areas where they can be efficiently used [1]. At the same time, there are many problems that are solved by algorithms, certain parts of which are better performed on a CPU or GPU[2, 3]. In addition, on not very powerful systems, simple parallelization of calculations on the CPU and GPU can significantly reduce the calculation time[4]. Performing calculations on heterogeneous systems allows you to solve these problems

**Target setting.** Simultaneous execution of calculations on GPU and CPU helps to overcome the disadvantages of calculation on CPU and GPU for different algorithms by optimizing the execution process and taking advantage of the strengths of both.

**Actual scientific researches and issues analysis.** There are many scientific studies that describe approaches to the effective use of heterogeneous systems for solving problems in various directions. They differ in approaches to the distribution of the share of calculations between the CPU and GPU, optimization methods of calculations of the algorithm parts that are not adapted to calculations on the CPU or GPU.

**Uninvestigated parts of general matters defining.** Heterogeneous computing is a modern topic that is rapidly developing now and has many directions for research. This article examines the effectiveness of a heterogeneous system depending on the volume of input data, its distribution between the CPU and GPU, and the computational complexity of the algorithm.

**The research objective.** The purpose of this work is to investigate the optimization of a heterogeneous system for the problem of Gaussian image blurring, its dependence on the volume of input data, and the complexity of blurring.

**The statement of basic materials.** This article considers the solution of the Gaussian image blurring problem [5] using a heterogeneous system. In this algorithm,

image blurring is achieved by calculating the color of each pixel of the resulting image from the color values of its surrounding pixels (Figure 1).

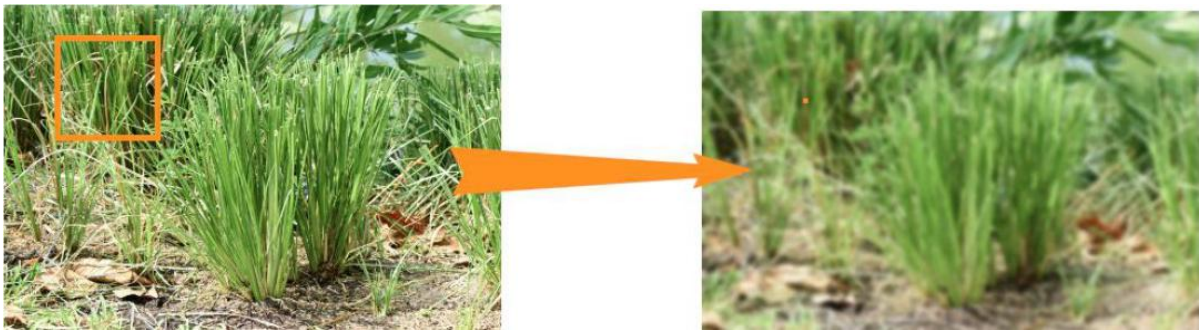


***Fig. 1.*** The principle of operation of the Gaussian image blurring algorithm

The algorithm starts by creating matrix operator - a filter that will be applied to each image pixel. The values of the cells of this matrix are calculated by the formula (Equation 1).

$$val(i,j) = e^{-\frac{i^2+j^2}{2\sigma^2}} \quad (1)$$

where σ - blur uniformity coefficient. In the next step, the matrix is normalized. For example, for a 3x3 matrix and σ = 2.411, there will be such a result (Table 1).

***Table 1.*** **Example of matrix operator**

| | | |
|---|---|---|
| 0.104745 | 0.114153 | 0.104745 |
| 0.114153 | 0.124407 | 0.114153 |
| 0.104745 | 0.114153 | 0.104745 |

The color of each pixel is encoded by RGB values - 3 numbers that represent red, green and blue color levels. For each pixel of image should be counted new value of RGB color ($R_{new}, G_{new}, B_{new}$) to perform blur operation (Equation 2).

$$\begin{aligned} R_{new} &= \sum \quad k_i * R_i + 0.124407 * R_{old} \\ G_{new} &= \sum \quad k_i * G_i + 0.124407 * G_{old} \\ B_{new} &= \sum \quad k_i * B_i + 0.124407 * B_{old} \end{aligned} \quad (2)$$

where $k_i$ is a coefficient from matrix operator. $R_{old}, G_{old}, B_{old}$ – previous color of pixel, $R_i, G_i, B_i$ – colors of neighboring pixels.

An image of the size width x height is submitted to the input of the algorithm. It is processed in the form of a matrix. The total number of pixels in the image was used to compare the results on the volume of input data.

Three approaches to the solution that were chosen:

- Multi-threaded on CPU using OpenMP
- Multi-threaded on GPU using CUDA
- Combination of multi-threaded solutions on CPU and GPU.

To combine the CPU and GPU, image should be divided into columns - 75% of the columns are computed by the GPU, 25% by the CPU. The calculation of the transformation matrix was performed on the CPU in all cases.

Solutions were tested on images with resolutions of 320x320, 412x275, 600x450, 1000x563, 1200x900, 2048x1306, 4250x2833 with operator matrix sizes 3 and 5 (Table 2-3).

***Tab. 2*. Execution time for 3x3 matrix operator**

| Image resolution | Pixel count | CPU execution time, ms | GPU execution time, ms | CPU + GPU execution time, ms |
|---|---|---|---|---|
| 320x320 | 102400 | 14 | 103 | 17 |
| 412x275 | 113300 | 8 | 68 | 41 |
| 600x450 | 270000 | 27 | 69 | 27 |
| 1000x563 | 563000 | 41 | 68 | 26 |
| 1200x900 | 1080000 | 68 | 140 | 40 |
| 2048x1306 | 2674688 | 155 | 91 | 46 |
| 4250x2833 | 12040250 | 665 | 179 | 184 |
| 5120x2880 | 14745600 | 825 | 125 | 218 |

As the result, calculations on the CPU were the slowest, the calculations on the GPU + CPU were faster than the GPU until some critical point in the calculations (Figure 2-3), which is related to the size of the operator matrix - the larger it is, the sooner the GPU will overtake the CPU + GPU solution. With an increase in the operator matrix, the complexity of calculations grows quadratically and the CPU solution no longer has time to process its share of calculations. If it is reduced from 25% to 10%, then the separation from the GPU will come later and will not grow so fast (Figure 4).

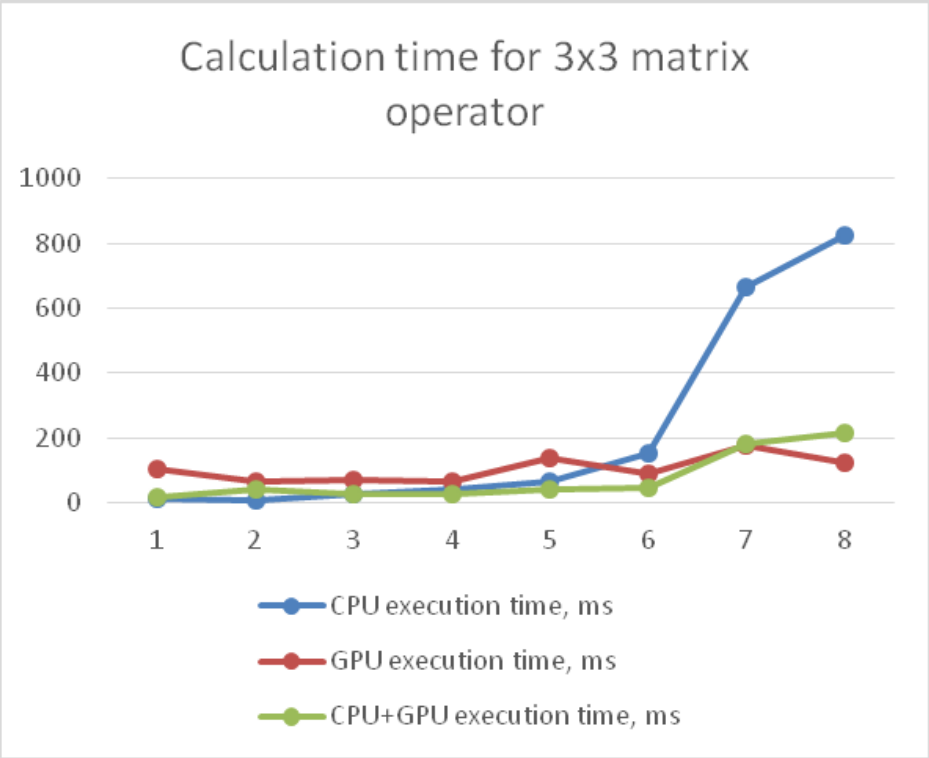



***Fig. 2.*** Calculation time for 3x3 matrix operator

***Tab. 3.*** **Execution time for 5x5 matrix operator**

| **Image resolution** | **Pixel count** | **CPU execution time, ms** | **GPU execution time, ms** | **CPU + GPU execution time, ms** |
|---|---|---|---|---|
| 320x320 | 102400 | 39 | 89 | 62 |
| 412x275 | 113300 | 22 | 67 | 25 |
| 600x450 | 270000 | 35 | 65 | 36 |
| 1000x563 | 563000 | 75 | 68 | 64 |
| 1200x900 | 1080000 | 210 | 71 | 57 |
| 2048x1306 | 2674688 | 390 | 90 | 110 |
| 4250x2833 | 12040250 | 1620 | 121 | 399 |
| 5120x2880 | 14745600 | 1905 | 146 | 581 |

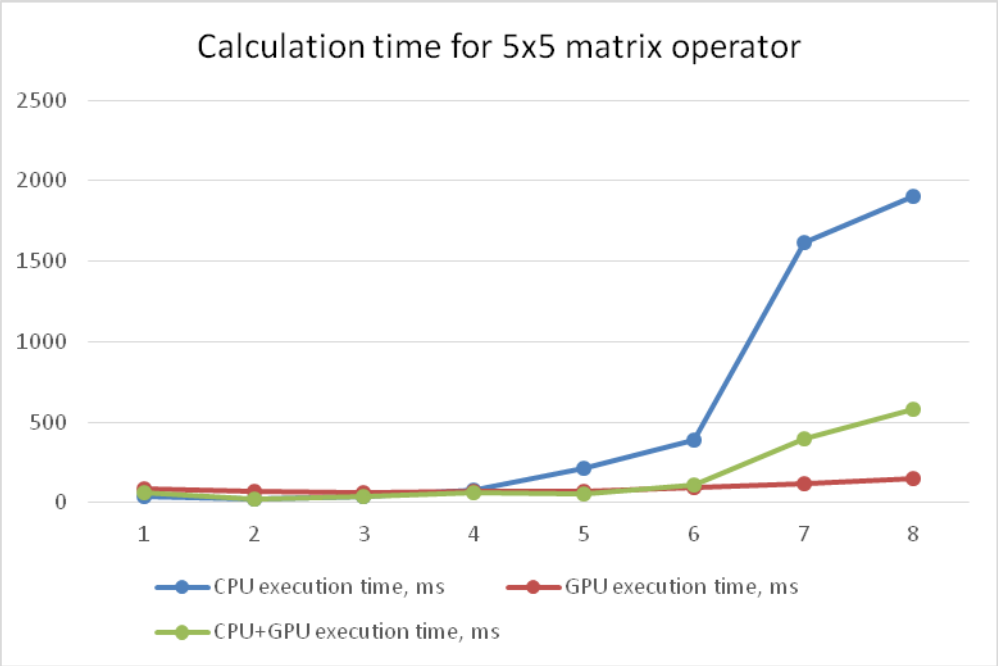



***Fig. 3.*** Calculation time for 5x5 matrix operator

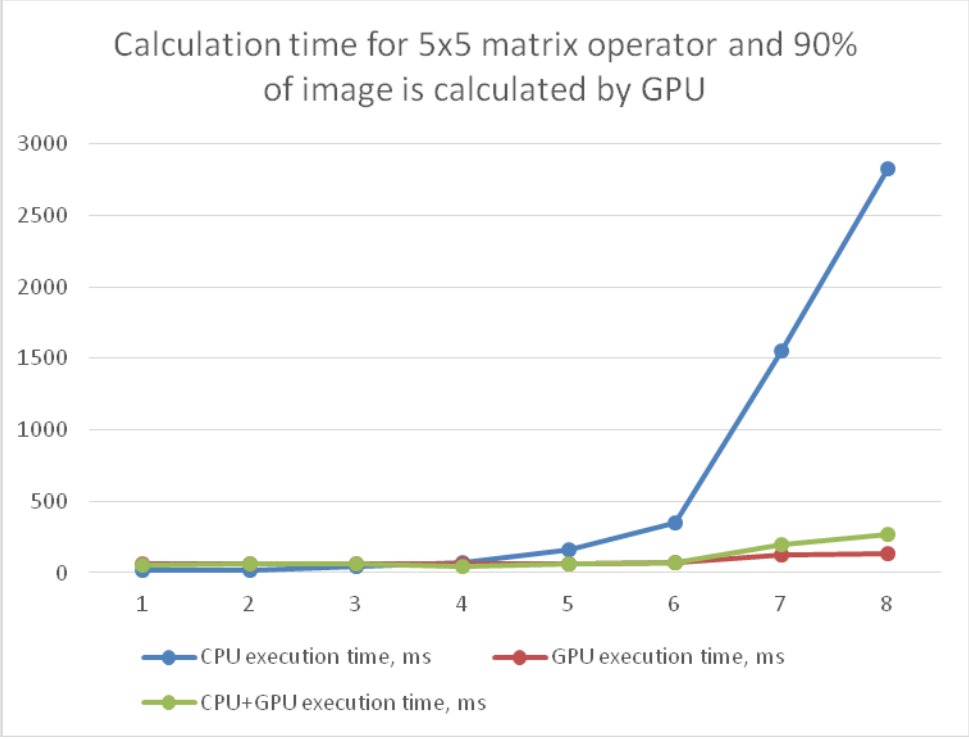



***Fig. 4.*** Calculation time for 5x5 matrix operator and 90% of image is calculated by GPU

**Conclusion.** This article examines the relationship between the efficiency of using a heterogeneous system and the size of the input data, the complexity of the

algorithm, the different sizes of the image portions assigned to the calculation of the GPU and the CPU.

The GPU does not highly dependent on the image size - with an increase of input data by 4.5 times (with image dimensions from 2048x1306 to 4250x2833), the computation time increased by 1.3 times.

The CPU is very dependent on the size of the input data - on the same interval, the computation time increased by 4.15 times.

The CPU+GPU solution occupies an intermediate stage and achieves better results with the correct combination of the proportions of calculations parts and their difficultness (calculated through the size of the matrix operator).

## References

1. Lusher, David J., Satya P. Jammy, and Neil D. Sandham. "OpenSBLI: Automated code-generation for heterogeneous computing architectures applied to compressible fluid dynamics on structured grids." Computer Physics Communications 267 (2021).

2. Tang, Xiaoyong, and Zhuojun Fu. "CPU–GPU utilization aware energy-efficient scheduling algorithm on heterogeneous computing systems." IEEE Access 8 (2020).

3. Benatia, Akrem, et al. "Sparse matrix partitioning for optimizing SpMV on CPU-GPU heterogeneous platforms." The International Journal of High Performance Computing Applications 34.1 (2020).

4. Bateni, Soroush, et al. "Co-optimizing performance and memory footprint via integrated cpu/gpu memory management, an implementation on autonomous driving platform." 2020 IEEE Real-Time and Embedded Technology and Applications Symposium (RTAS). IEEE, 2020.

5. Bozkurt, Ferhat, Mete Yaganoglu, and Faruk Baturalp Günay. "Effective Gaussian blurring process on graphics processing unit with CUDA." International Journal of Machine Learning and Computing 5.1 (2015).

## AUTHORUS

**Melenchukov Mykyta** – student, Department of Computer Engineering, National Technical University of Ukraine “Igor Sikorsky Kyiv Polytechnic Institute”.

E-mail: melenchukov.nikita@gmail.com

**Artem Volokyta** – associate professor, PHd, Department of Computer Engineering, National Technical University of Ukraine “Igor Sikorsky Kyiv Polytechnic Institute”.

**Rusanova Olga Veniaminivna** – associate professor, PHd, Department of Computer Engineering, National Technical University of Ukraine “Igor Sikorsky Kyiv Polytechnic Institute”.

## **Parallel Section GN.** Global Networks, Grid and Cloud.

**Yurii Kulakov, Olga Rusanova,**
**Iryna Hrabovenko, Yuliia Hrabovenko**

# THE EFFICIENCY EXPLORATION OF PARALLEL WAVE ROUTING ALGORITHM WITH GPU COMPUTING COMPARED TO CPU

The present paper concerns the issues of speeding up the execution time of the modified reverse wave routing algorithm in a software-defined network of large size. The parallel version of the algorithm is executed on the predefined network sizes with the same edge probability on a multi-core CPU and GPU separately, partly on a multi-core GPU and partly on a multi-core CPU. The exploration results of the parallel algorithm help to define the most suitable way of algorithm computing in networks of different sizes.

**Keywords:** modified inverse wave algorithm, CPU, GPU, software-defined network.

Fig.: 4. Tabl.: 3. Bibl.: 3.

**Relevance of the research topic.** The modified inverse wave algorithm is an effective traffic engineering method for software-configured networks, as it reduces the time complexity of forming multiple paths and reduces reconfiguration time. However, the execution time of the algorithm increases significantly in large networks. This research considers the application of graphic processor technology to improve the performance of modified reverse wave routing algorithm in a large mobile network.

**Target setting.** The research target is to speed up the execution time of the routing algorithm in software-configured networks of large size by using GPU computing.

**Actual scientific researches and issues analysis.** Many scientific papers in the field of mobile networks are devoted to solving the problem of choosing an optimal algorithm for execution in large networks [1], [2], [3]. As powerful GPUs become more available and suitable for massively parallel computing, performing parallel processing of the algorithm on the GPU can solve the problem of speeding up the routing execution in a scalable network. Today there are many scientific works devoted to choosing CPU, GPU or CPU+GPU implementation that provides minimum execution time for different applications [4].

**Uninvestigated parts of general matters defining.** This article is devoted to the parallelization of the reverse wave routing algorithm and exploration of its execution efficiency in three cases including separate execution with GPU, CPU, and partly on GPU and partly on CPU to improve the algorithm performance characteristics in large mobile networks.

**The research objective.** The main task is to use the technology of graphic processors to make the reverse wave algorithm find paths in the large networks faster.

**The statement of basic materials.** First of all, the possibility of performing 'for' cycle iterations in parallel can be used to improve execution time of the algorithm. This is possible because the routers of the current wave can be computed separately and the results of their calculations can be combined to form the next set of routers and so on. Besides, the factors increasing the execution time of the parallel algorithm version will include the number of iterations and the maximum number of operations of minimum delay metrics change of adjacent nodes on each iteration.

1. Set initial number of routers $W_1 = \{R_n\}$;
2. $D_i$=0;
3. $J$=0; *can be executed in parallel*
4. for $j$=$j$+1 step 1 form the routers set $W_{j+1}$=$\{R_i|\ i=1,\ldots k\}$ adjacent to the routers set from $W_{j1}$=$\{R_i|\ i=1,\ldots k\}$, where $k$ - the sum of the degrees of the routers set $W_{j1}$=$\{R_i|\ i=1,\ldots k\}$;
5. if $W_{j+1}$=∅ then go to 10 do
6. for $i$=1 step 1 до $k$ calculate $Z_i\{V_n, V_l, M_{i,n}, d_i\}$
7. if $d_j > D_i$ then $D_i = d_j$
8. end;
9. go to 4
10. end.

***Fig. 1.*** Pseudo code of the parallel algorithm

Then, the parallel wave algorithm execution on a multi-core CPU and a multi-core GPU depends on its implementation with the use of special libraries and data structures that fit specific architecture needs.

**Experiments.** Firstly, the proposed parallel algorithm was executed on a multi-core CPU only. The CPU characteristics included 4 processors Intel(R) Core(TM) i5-7200U CPU @ 2.50GHz 2.70 GHz.

To implement the algorithm on the CPU, parallel processing tools in the Python programming language were used, namely, the multiprocessing module, which supports

the process generation using the API. Due to the possibility of bypassing the Global Interpreter Lock (GIL), this module allows full use of several processors on the user's computer. With the use of a multiprocessing library in Python, processes are generated by creating a Process object and then calling the start() method. This package also includes special data types for exchange between processes. Table 1 shows the results of running the algorithm with CPU depending on the number of nodes from 100 to 1000 in a random connected graph with a step of 100 with an edge probability of 0.01.

***Table 1.*** **Table of algorithm execution results with CPU**

| **t, sec** | 1,49 | 2,07 | 15,89 | 20,06 | 21,9 | 36,15 | 60 | 74,87 | 101,86 | 112,43 |
|---|---|---|---|---|---|---|---|---|---|---|
| **N** | 100 | 200 | 300 | 400 | 500 | 600 | 700 | 800 | 900 | 1000 |

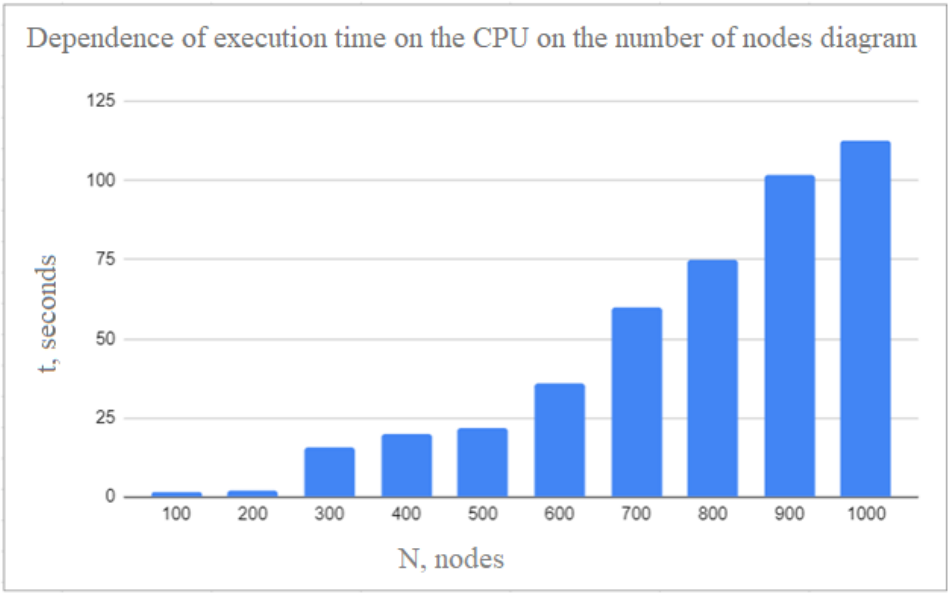



***Fig. 2.*** Dependence of execution time on the CPU with nodes number

As can be seen from the results of execution at a given computing power obtained a small execution time with the number of nodes of the graph from 100 to 200, but then with an increasing nodes number the execution time of the algorithm does not give optimal results. It can be explained by the impossibility to process all the nodes in parallel because of an insufficient number of processors, and because the number of levels of the graph also increases, which also increases the execution time of the algorithm.

Secondly, the parallel algorithm was executed on a multi-core GPU. It was decided to use the CUDA simulator from the cudatoolkit package in Python that

implements the functions of one GPU device with a capacity of 5.2 which is sufficient for writing kernel functions with GPU support. This cudatoolkit package also includes GPU-accelerated libraries and the CUDA runtime for the Conda ecosystem and the Numba library tools that support CUDA GPU programming by directly compiling a limited subset of Python code into CUDA kernels and device functions according to the CUDA execution model. Kernels written in Numba seem to have direct access to NumPy arrays. NumPy arrays are transferred between CPU and GPU automatically. The algorithm was executed on the same range of graph nodes number from 100 to 1000 and using the same coefficient of edge probability in a graph that equaled 0.01.

Table 3 shows the results of running the algorithm with GPU depending on the number of nodes from 100 to 1000 in a random connected graph with a step of 100 with an edge probability of 0.01.

***Table 2.*** **Table of algorithm execution results with GPU**

| **t, sec** | 0,07 | 0,16 | 0,73 | 0,87 | 0,94 | 1,78 | 2,21 | 3,31 | 3,85 | 4,13 |
|---|---|---|---|---|---|---|---|---|---|---|
| **N** | 100 | 200 | 300 | 400 | 500 | 600 | 700 | 800 | 900 | 1000 |

As can be seen from the results of execution at these graph sizes, there is a gradual increase in the execution time of the algorithm, which generally gives good execution results even at a graph size of 1000 nodes. Execution time intervals with the number of nodes from 100 to 200, from 300 to 500, from 600 to 700, and from 800 to 1000 give approximately similar execution times.

Then, the algorithm was executed partly on a multi-core GPU and partly on a multi-core CPU in ratio proportion of fifty-fifty using the same characteristics of CPU and GPU. To implement partial parallelization of the algorithm on CPU and GPU in a 50/50 percentage ratio, a pre-implemented functionality was used to parallelize the algorithm on CPU and GPU, so that half of the graph is processed on the CPU and the other half with all its associated data transferred for processing on the GPU. Table 3 shows the results of running the algorithm partly on CPU and GPU depending on the number of nodes from 100 to 1000 in a random connected graph with a step of 100 with an edge probability of 0.01.

This combination gives good results of parallelization at the number of nodes from 100 to 300. After that, there is a significant increase in execution time at each subsequent interval, which may be due to an insufficient number of cores on the CPU and time spent on data transfer from CPU to GPU and vice versa.

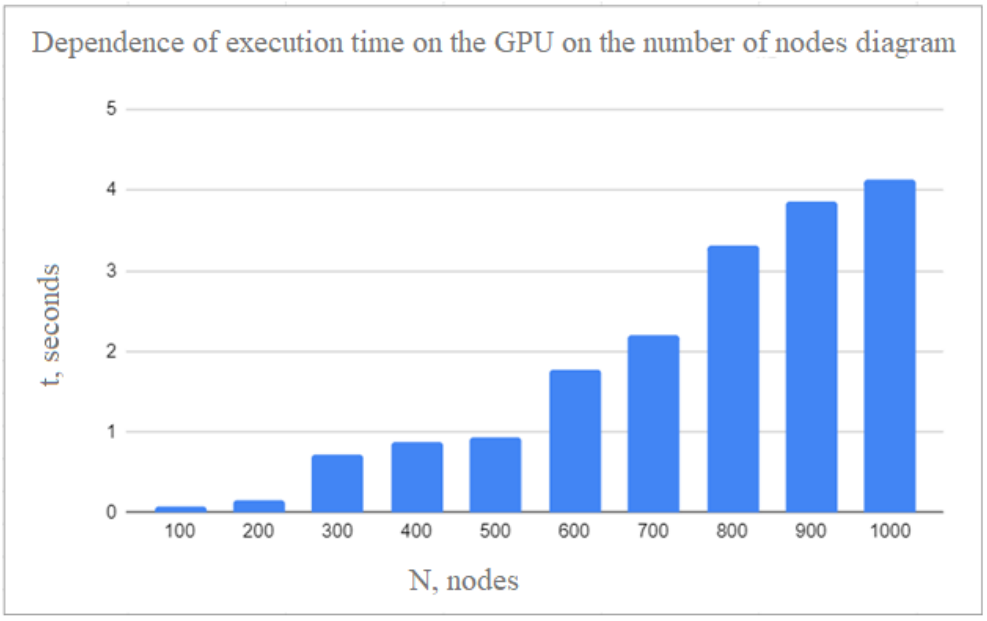



***Fig. 3.*** Dependence of execution time on the GPU with nodes number

***Table 3.*** **Table of algorithm execution results partly on CPU and GPU**

| **t, sec** | 1,71 | 3,9 | 5,03 | 11,51 | 16,26 | 18,32 | 22,41 | 26,9 | 41,53 | 63,85 |
|---|---|---|---|---|---|---|---|---|---|---|
| **N** | 100 | 200 | 300 | 400 | 500 | 600 | 700 | 800 | 900 | 1000 |

To sum it up, the results from all three experiments are given in the form of a bar chart on Fig 5.

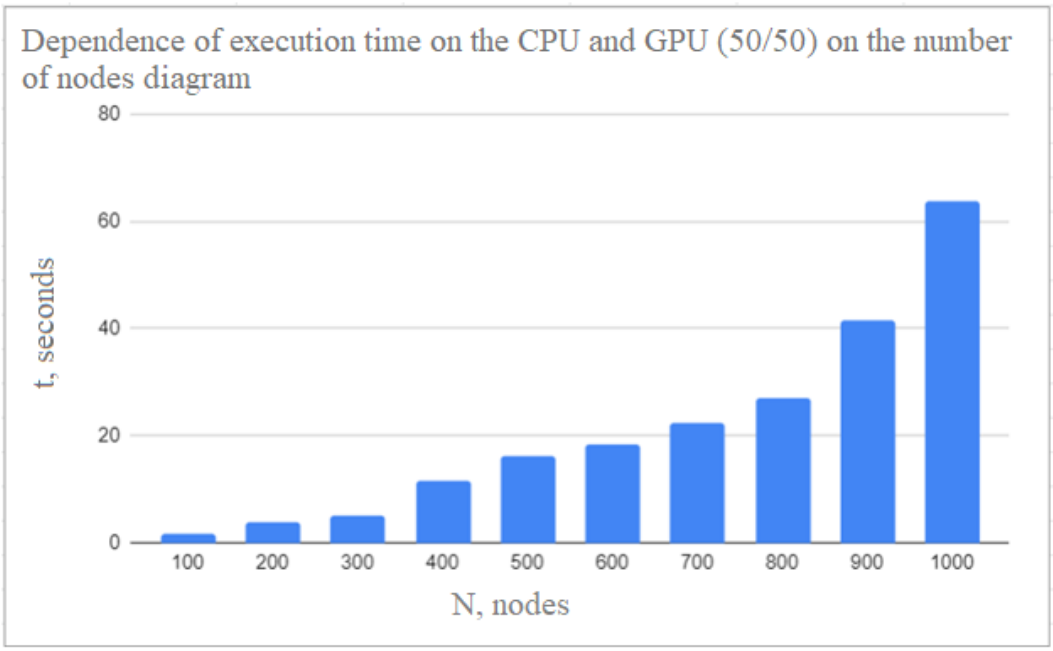



***Fig. 4.*** Dependence of execution time on the CPU and GPU with nodes number

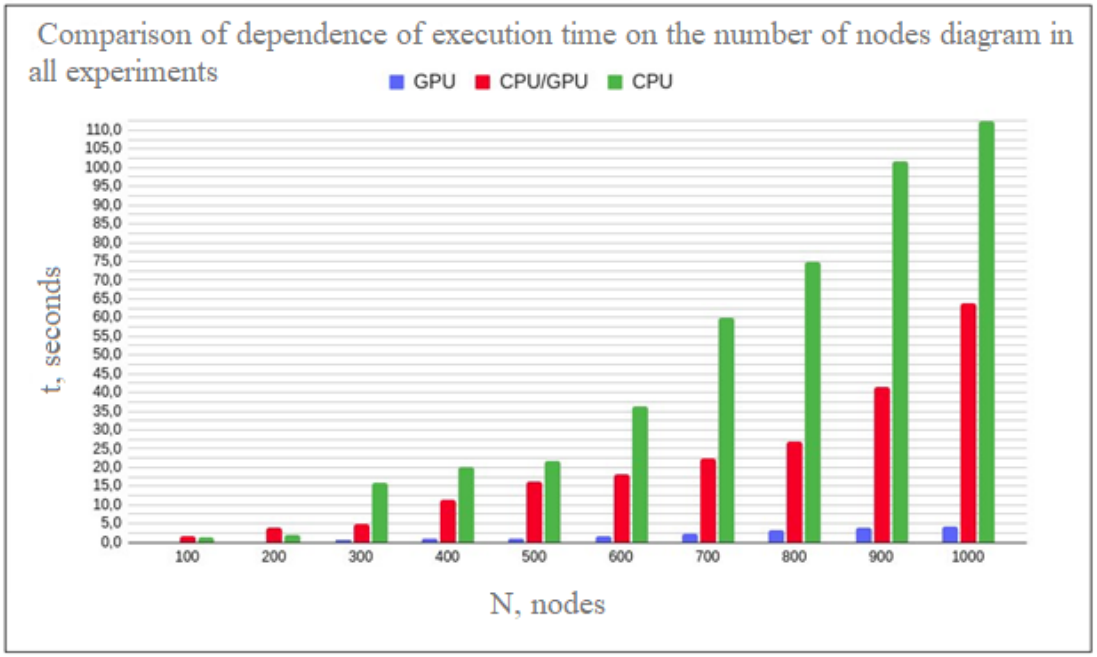



***Fig. 5.*** Comparison of execution time results from all experiments

**Conclusions.** It has been proven that with an increasing number of graph nodes the algorithm execution time increases, and the best execution time is provided on the GPU, due to the architecture, because GPUs have enough cores to process large amounts of data in parallel, which in this case is determined by the number of vertices. Due to the insufficient number of cores, the execution time on the CPU is much longer, as all vertices in the queue are not processed in parallel by each core, but are distributed to available cores and wait for their execution sequentially. When running the algorithm partly on the CPU and partly on the GPU, we have better results than those obtained when running only on the CPU, although for graphs with less than 200 nodes it is more profitable to apply the algorithm only on the CPU than partly on the CPU and GPU associated with additional time delays for data transfer from CPU to GPU.

## References

1. Kulakov Y. et al. Traffic engineering in a software-defined network based on the decision-making method //Восточно-Европейский журнал передовых технологий. – 2019. – №. 2 (9). – С. 23-28.

2. Kulakov, Y., Kogan, A., “The method of plurality generation of disjoint paths using horizontal exclusive scheduling“, The advanced science journal. Issue 10. Volume 2014. ISSN 2219-746X. DOI: 10.15550/ASJ.2014.10. Pp. 16-18.

3. Kulakov Y. et al. Load Balancing in Software Defined Networks Using Multipath Routing //International Conference on Computer Science, Engineering and Education Applications. – Springer, Cham, 2020. – С. 384-395.

4. Rusanova O.V.,Korochkin A.V.,Shevelo O.P. Classification of scheduling problems for modern parallel computer systems// Збірник тез доповідей XII Міжнародної науково-технічної конференції «Комп'ютерні системи та мережні технології,м.Київ,28-30 березня 2019р.-К.:НАУ,2019.-С.102-103.

## AUTHORS

**Yurii Kulakov** (supervisor) – professor, Department of Computer Engineering, National Technical University of Ukraine "Igor Sykorsky Kyiv Polytechnic Institute".

**Olga Rusanova** (supervisor) – associate professor, Department of Computer Engineering, National Technical University of Ukraine "Igor Sykorsky Kyiv Polytechnic Institute".

**Iryna Hrabovenko** - student, Department of Computer Engineering, National Technical University of Ukraine "Igor Sykorsky Kyiv Polytechnic Institute".

**Yuliia Hrabovenko** - student, Department of Computer Engineering, National Technical University of Ukraine "Igor Sykorsky Kyiv Polytechnic Institute".

Email: yuliia.hrabovenko@gmail.com

UDC 004.272.2

**Oleksii Krutko, Oleksandr Korochkin**

# ANALYSIS OF THREADS CONTROL TOOLS IN MODERN LANGUAGES AND LIBRARIES OF PARALLEL PROGRAMMING

The paper deals with the analysis tools for thread control used in modern language and libraries of parallel programming. Languages Java, Ada, C#, Python, libraries WinAPI, OpenMP, MPI are considered. Means optimal for solving the problems of mutual exclusion and synchronization for scalable parallel programs are defined.

**Key words:** threads, organization of treads communication.

Tabl.: 1. Bibl.: 6.

**Target setting.** The problem of developing software for parallel computer systems is becoming more urgent in connection with the growing market of multi-core processors. This work is devoted to the analysis of various means of programming and threads control in modern parallel programming libraries and languages.

**Actual scientific researches and issues analysis.** The organization of the interaction of threads is an important part of a parallel program, the execution of which is critical depending on both the choice and the application of means of interaction. The increase in the number of processors in modern computer systems and, accordingly, the number of interacting threads poses the task of choosing and using reliable thread synchronization tools. This is especially true for scalable systems.

**Uninvestigated parts of general matters defining.** This article is devoted to the selection and application of thread interaction tools for scalable computer systems where the number of processors and, accordingly, the number of threads, can change dynamically. The development of applications for scalable systems has its own characteristics and requires the use of optimal tools that will ensure the correct execution of the program and the absence of deadlocks. Therefore, this work focuses on the analysis thread control tools for scalable systems.

**The research objective.** The task is to analyze the existing means of interaction of threads, which will provide the possibility of choosing the means optimal for scalable systems when solving the task of reliable interaction of a large number of threads, the number of which may change.

The purpose of this work is to increase the efficiency of development and execution of parallel programs for scalable computer systems.

**The statement of basic materials.** Software development for parallel computer systems is based on the use of modern parallel programming languages and libraries. The emergence of new and improvement of existing means of interaction of threads requires their constant tracking and analysis for the purpose of optimal use when building parallel programs, including for scalable computer systems.

Well-known parallel programming languages and libraries were selected for the analysis of the means of creating and organizing the interaction of threads of different levels: Java, Ada, C#, Python, WinAPI, OpenMP, MPI [1-6].

Creation (declaration) of a thread is related to the description of the thread (group of threads), the formation of the ID of the thread, setting the priority, choosing the processor for execution, the size of the stack, actions of the thread, starting and terminating.

This can be done in different ways:

– through the use of special modules (classes) (Java: class Thread, Ada: module task);

– through thread functions that define the actions and parameters of threads (C#, WinAPI) - with the help of so-called thread functions. (C#, WinAPI) Here, the thread's actions are specified through a pre-designed function that defines the thread's behavior;

– through the definition in the sequential program of the sections that will be run in parallel (OpenMP)

– through creating copies of the entire program and parallel execution of these copies (MPI, PVM).

Additional possibilities are provided by combining threads into groups (pools) and using queues of various types, which allow optimizing the execution of threads (Java, Python, Ada, C#, MPI). This is a development of the queuing mechanisms (previously proposed in the Ada language) and communicators (MPI).

Each approach has its advantages, the use of which allows you to simplify the development of a parallel program, its debugging, modification, and scaling.

Important for scalable parallel systems are solution:

– problem of dynamically creating thread

– problem of access to shared resources (mutual exclusion problem)

– thread synchronization problem.

Solutions to the first problem are provided by modern languages (libraries) of parallel programming, where the possibility of dynamic creation of threads, characteristic of scalable systems, is realized. Dynamic generation of threads requires

correspondingly dynamic identification of threads. In the OpenMP and MPI libraries, identification is carried out automatically; each new thread receives an integer identifier in a corresponding parallel block or communicator. Another situation occurs if the thread name is formed directly by the developer. In Ada, dynamic thread creation provides a task type that allows you to create arrays of threads

task type RS is . . . end RS;

T is an array of RS (1.. N);

and together with the use of a discriminate, form an internal identifier:

task type RS(Id: integer ) is . . . end RS;

T1: RS(1); . . . T10: RS(10);

The second problem. As a rule, when the number of threads changes, the number of shared resources does not change, so the problem of mutual resolution can be solved by low-level means (semaphores, mutexes) and lock type means. But if it is necessary to take into account quantitative characteristics for complex accesses to shared resources, then more powerful tools are needed - monitors, for example, protected inputs in the protected unit of the Ada language, which provides additional logic for accessing shared data.

Recently, non-blocking means of thread interaction (atomic variables, non-blocking queues) have been developed, which allow minimizing the time of using shared resources.

In scalable systems, the solution to the problem of thread synchronization is the most complicated. If the use of binary semaphores and events is enough to synchronize two threads, then multiple synchronization requires more powerful means.

To implement multiple synchronization, you can use the barrier mechanism, which was developed in the form of a counting barrier and is used in almost all languages and libraries.

The most powerful tool remains the monitor mechanism. In Ada language, special constructions of the protected module are proposed for solving the synchronization task - protected entries, which have barriers that additionally define various conditions for blocking and unlocking flows. The use of these entries allows to program complex forms of thread interaction, which is important for scalable programs:

entry Wait_Task when Cond

Here, in Wait_Task entry barrier ( when Cond construct), a logical variable is formed that defines the condition of blocking and unblocking the thread.

Table 1 provides data on the presence in the considered languages and libraries of parallel programming means of organizing the interaction of threads.

***Table 1***

| Tools | Java | C# | Ada | Python | WinAPI | OpenMP | MPI |
|---|---|---|---|---|---|---|---|
| *Semaphores* | + | + | + | + | + | | |
| *Mutexes* | | + | | + | + | | |
| *Events* | | + | | + | + | | |
| *Critical Section* | + | + | | + | + | + | |
| *Barriers* | + | + | + | + | + | + | + |
| *Atomic/Volatile* | + | + | + | + | + | + | |
| *Monitors* | | | + | | | | |
| *Queue* | + | + | + | + | | | |
| *Pool* | + | + | | + | + | + | |
| *Messages* | | | + | | | | + |

**Conclusions.** Means of modern languages and parallel programming libraries for creating and control threads are analyzed. The given results will make it possible to choose the optimal tools when creating software for scalable parallel computer systems.

## References

1. Oaks S. Java Performance: In-depth Advice for Tuning and Programming Java 8, 11, and Beyond. O'Reilly Media, Inc.; 2end Edition, ( Februare 4, 2020), p.452.

2. Barnes J. Programming in Ada 2012. Cambridge University Press; 2nd edition (May 19, 2022), p. 992.

3. Nagel Chrisian, Professional C# and .NET. Wrox; 2021st Edition, (September, 2021), pp. 1008.

4. Gorelick M., Ozsvald I. High Performance Python, O'Reilly Media, Inc.; 2end Edition, ( April 30, 2020), p. 470.

5. Klemm M., Cownie J. High Performance Parallel Runtimes Design and Implementation. Berlin, Boston: De Gruyter OldenDurg 2021, p.328.

6. Muhammad Nufail Farooqi, Miquel Pericàs Vectorized Barrier and Reduction in LLVM OpenMP Runtime In Proceeding of 17th International Workshop on OpenMP, IWOMP 2021, Bristol, UK, September 14–16, 2021, pp. 18-32.

## AUTHORS

**Krutko Oleksii** – student of National Technical University of Ukraine “Igor Sikorsky Kyiv Polytechnic Institute”.

E-mail: aleksey.krutko @gmail.com

**Korochkin Oleksandr** – associate professor, Department of Computer Engineering, National Technical University of Ukraine “Igor Sikorsky Kyiv Polytechnic Institute”.

E-mail: avcora@gmail.com

UDC 004.8

**Ivan Holubov, Iryna Klymenko**

# PERFORMANCE COMPARISON OF POPULAR RDBMS

The article discusses the comparison of the productivity of popular RDBMS. Popular RDBMS are used for comparison.

**Key words:** RDBMS, database, read operation

Fig.: 1. Bibl.: 1.

**Target setting.** Due to the growing demand for RDBMS. The analysis of popular RDBMS has become a hot topic in recent years.

**Actual scientific researches and issues analysis.** In connection with the invention of new methods and approaches in the field of databases, the topic of choosing an appropriate RDBMS has become more studied.

**Uninvestigated parts of general matters defining.** Despite a significant number of works devoted to comparing the productivity of popular RDBMS, the problem remains poorly understood. Therefore, this work is focused on comparing popular RDBMS in many ways.

**The research objective.** The purpose of this article is to compare the performance of popular RDBMSs. As a solution, the article will focus on the parameters for reading records.

**The statement of basic materials.** You can build ratings in different ways, in some cases the ratings are based on user surveys, in some cases you can estimate the number of copies sold. In some cases, you can estimate the cash equivalent of sales.

But no matter what rating you take, in any case you will see at the top of the list those DBMS. These are Oracle, Microsoft SQL Server, PostgreSQL and MySQL. [1]

## General model structure.

Let's start with a system called Oracle database from the company of the same name. Oracle was founded in 1977. And it was she who launched the first commercially available DBMS, because the existing DBMSs were only research in nature. I must say that the DBMS market is now about $ 30 billion. This company provides about 46% of the market for modern databases.

The next product to mention is SQL Server from Microsoft, which is the market leader in all software in the world. It is based on a software product developed together with Sybase. The first product was produced in 1988. Unlike other multi-platform databases, Microsoft's SQL Server runs on the Windows operating system, and SQL

Server owns 53% of the market, but the overall market share of this software is less than 20%. Microsoft has announced the release of the first version, which runs on the Linux operating system since 2017.

PostgreSQL is a free, open-source software product. The first version was developed at the University of California, Berkeley. At the moment, this database is supported by a team of enthusiasts.

The MySQL database has the same basis, it is a database management system that is freely distributed, it is supported by Oracle. This database can be a good basis for developing small and medium-sized applications.

## Experiments.

Below is a graph that describes the relationship between the time of the read operation and the number of records. (Fig. 1) It can be concluded that in the read operation, Oracle and PostgreSQL are more productive.

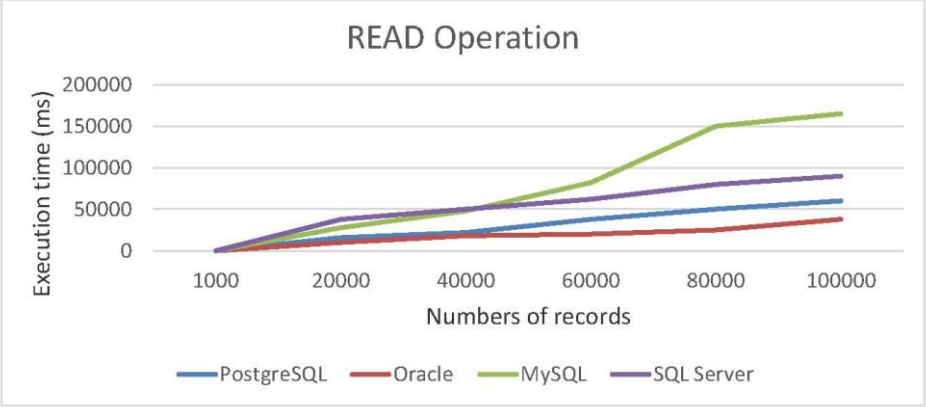



***Fig.1.*** The relationship between the time of the read operation and the number of records

**Conclusions.** The paper demonstrates a comparison of the productivity of popular RDBMS. It's clear that. A comparative characteristic has been developed that makes it possible to choose an RDBMS depending on the required parameters.

There are several areas for future work. One is to compare DBMSs that have the properties of consistency and partition tolerance, as well as availability and partition tolerance. Another direction is to increase the number of RDBMS comparison characteristics to give a better analysis.

## References

1. C.J.Date (2005). An Introduction to Database Systems. (p. 75).

## AUTHORS

**Iryna Klymenko** - Doctor of Technical Sciences, Department of Computer Engineering, National Technical University of Ukraine “Igor Sikorsky Kyiv Polytechnic Institute”.

**Volodymyr Rusinov, Oleksii Cherevatenko**

# METHOD OF NEURAL NETWORK TRAINING FOR EDGE ARCHITECTURE

The article analyzes the issue of heterogeneous CPU-GPU system use in accelerating applied tasks, related to the neural network learning process.

**Keywords:** edge computing, neural networks, machine learning, CPU, GPU.

Fig.: 3. Tab.: 0. Bibl.: 7.

**Relevance of the research topic.** The current state of embedded computing and IoT presence creates an opening for the development of intelligent systems based on the idea of edge computing. Thanks to the recent advances in wireless technology, multiple devices that can work independently can form a single computing unit, not so different from supercomputers, albeit on a smaller scale. [1] Being small, power-efficient devices, some problems should be addressed, for example, such devices are not very powerful, latency can be disruptive, a specific setup is required and all of this directly influences the user experience.

**Target setting.** After the advent of neural networks, AI-enabled applications are becoming more widespread every day. At the same time, IoT devices are becoming more powerful and power-efficient. New research shows that when coupled IoT devices can reach high performance, allowing them to train machine learning models and run them more seamlessly than by using dedicated servers.

**Actual scientific researches and issues analysis.** Numerous studies have been published exploring how to run machine learning-enabled tasks on embedded devices and edge nodes. Approaches range from using one designated device to train a model and distribute the finished model over the network to creating SDN-powered nodes to distribute the learning process over the devices. This subject is quite novel, therefore no ubiquitous solution has yet been proposed.

**Uninvestigated parts of general matters defining.** Edge computing and machine learning are both relatively new technologies that are constantly evolving. Many of the issues are still unresolved or require additional investigation. This article presents one of the methods to resolve the issue of consumption of time and power during the training process, thereby allowing to deploy AI models faster by distributing work across multiple devices.

**The research objective.** The goal of this article is to develop a method that speeds up the training process significantly by distributing the workload across edge

devices. Also, the “cut” model will be tested to determine whether it is feasible to use only part of the model for faster deployment of AI models.

**The statement of basic materials.** Deep Neural Networks are the most commonly used way to adopt machine learning. Since their first introduction, as a perceptron, DNNs have developed into a versatile tool for absorbing patterns in massive amounts of data. [2] There is a range of different ways to compose a DNN, some of them have names like AlexNet or GoogleNet. Unfortunately, DNNs demand high computational power, and embedded devices are generally tuned for relatively simple tasks. The most prominent example of DNNs is convolutional neural networks (CNN) known for their high level of performance when working with visual input. [3]

Recent developments in IoT and ARM architecture saw IoT devices such as Raspberry Pi 4 run desktop OSes and applications on par with desktop PCs. On top of that, there are IoT devices that sport a GPU or a TPU onboard for tasks like machine learning or fast processing of video input. Recently, ARM-based CPUs have been able to perform on par and better than existing x86-based CPUs in various tests, among which is video processing.

However, to be power efficient, chips must have a TDP of around 5 to 15 W at a high load, therefore limiting their capabilities. To overcome problems presented by low-power SoCs on IoT devices, connecting several devices can be useful in increasing overall productivity. This approach is called edge computing, and it aims to exploit the growing amount of IoT devices. Coupled with recent developments in SDN technology it is possible to achieve low latency and fast data delivery at a low cost.

Edge computing has advantages over cloud computing or using specialized equipment, making it at least an alternative to these approaches. Edge nodes are usually close to their target datalakes. This leads to two important points of consideration: secure access to the data and decreased latency. Also, edge provides services at an affordable cost. As mentioned earlier, IoT devices can run desktop OSes, which in turn provide access to many third-party libraries as their code need not be translated for a specific device, therefore creating a level of abstraction increasing both flexibility and sustainability.

There are multiple ways to construct an AI-enabled architecture. [4] With many articles devoted to the creation of NN models in resource-constraint environments, effort goes into NN design for embedded devices. Some of the ideas like pruning and quantization proved to be efficient when used in training on low-powered devices. [5] However NN models have to be small thereby limiting their flexibility.

The suggested solution is to distribute the model over the edge devices. This will enable fast delivery of AI on the edge device, by letting the other parts of the edge train on the output of the model present. Suppose we have a random fully-connected, feedforward DNN, if we take the output of any layer, we can use it later on the next layer with the only rule being that the next layer accepts the output from the former layer.

To distribute the learning process over the devices, we use SDN-powered nodes. Using of SDN technology allows us to increase network capabilities such as traffic, capacity, number of nodes. This would come handy in computer processing when we need to transfer big amounts of data between devices. [6]

To test the mentioned solution we first need to choose a tested model. For that purpose, we take MobileNet, which performs very well on IoT devices. Architecturally it is also very streamlined, having multiple blocks consisting of convolution layer, depthwise convolution, and pointwise convolution layers, together forming potent depthwise separable convolution blocks. By adding a dense layer at the end of such block it can be used to train separately from the rest of the model. [7]

Dataset is made up of over 40 million pictures of doodles provided by Google, of which only 100,000 were used during training. An app gave users one of the 340 prompts to which they have a minute to draw the best representation of that prompt drew these 64x64 doodles. During training, these doodles were also split into mini-batches of 256 to speed up the training process.

Testing is performed on Raspberry Pi 3 with following hardware:

Cortex-A53 ARM 64-bit SoC at 1.4 GHz, 1 GB LPDDR2 RAM, Bluetooth 4.2 BLE, Wi-Fi 802.11.b/g/n/ac with microSD card acting as a permanent storage.

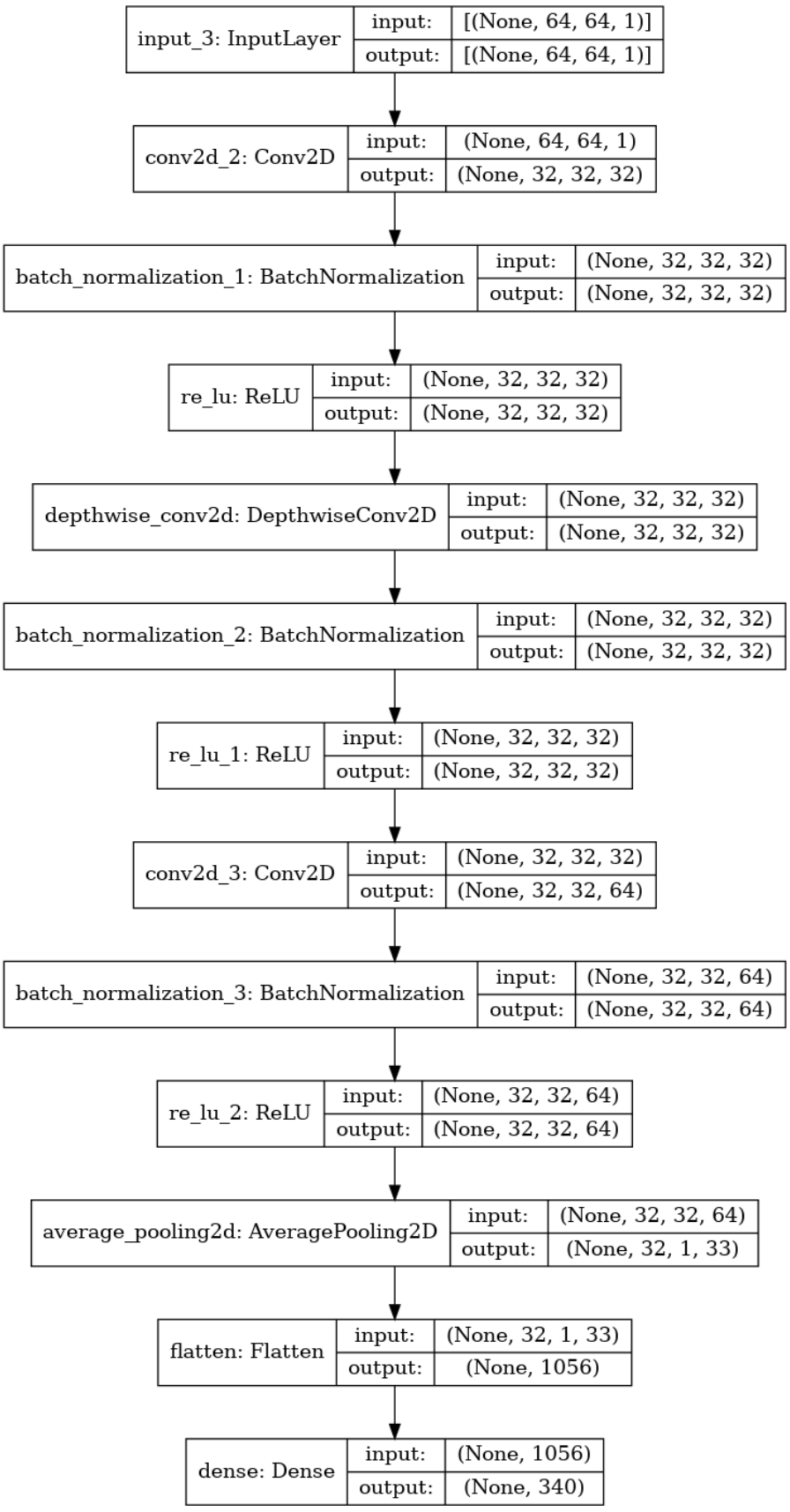



***Fig 1.*** The architecture of the MobileNet-based model

***Fig 2.*** Doodles from the dataset

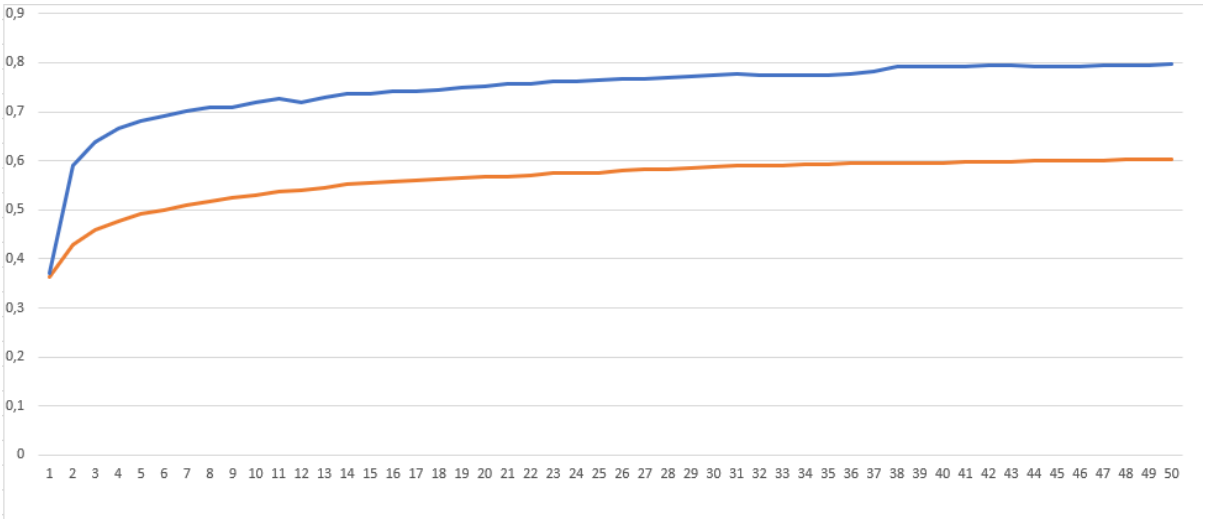



***Fig. 3.*** Test accuracy of the full MobileNet (blue) and first block of the MobileNet (orange)

**Conclusion.** In this article, we explored the idea of distributing learning on Edge architecture, creating on-site AI presence. Results show that accuracy of the MobileNet for this dataset reaches 79.74% and our MobileNet-based model gets up to 60.03% on 50th epoch. As for speed, the MobileNet model is much slower on the Raspberry Pi, however MobileNet-based test model is approximately as fast on Raspberry Pi as the whole model is on dedicated machine, sporting an NVidia Tesla GPU.

Further areas of exploration for this approach may include using specialized accelerators such as Intel Compute Stick, or NVidia Jetson and using real-time video feed for testing unsupervised classification.

## References

1. Zhao, Z., Barijough, K. M., & Gerstlauer, A. (2018). Deepthings: Distributed adaptive deep learning inference on resource-constrained iot edge clusters.

IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems, 37(11), 2348-2359.

2. Yamashita, R., Nishio, M., Do, R. K. G., & Togashi, K. (2018). Convolutional neural networks: an overview and application in radiology. Insights into imaging, 9(4), 611-629.

3. Li, E., Zeng, L., Zhou, Z., & Chen, X. (2019). Edge AI: On-demand accelerating deep neural network inference via edge computing. IEEE Transactions on Wireless Communications, 19(1), 447-457.

4. Xu, X., Ding, Y., Hu, S. X., Niemier, M., Cong, J., Hu, Y., & Shi, Y. (2018). Scaling for edge inference of deep neural networks. Nature Electronics, 1(4), 216-222.

5. Merenda, M., Porcaro, C., & Iero, D. (2020). Edge machine learning for ai-enabled iot devices: A review. Sensors, 20(9), 2533.

6. Lemeshko A. V., Evseeva O. Yu., Garkusha S. V. (2014). Research on Tensor Model of Multipath Routing in Telecommunication Network with Support of Service Quality by Greate Number of Indices. Telecommunications and Radio Engineering, 73(15), 1339—1360.

7. Abich, G., Reis, R., & Ost, L. (2021, February). The impact of precision bitwidth on the soft error reliability of the MobileNet network. In 2021 IEEE 12th Latin America Symposium on Circuits and System (LASCAS) (pp. 1-4). IEEE.

## AUTHORS

**Volodymyr Rusinov** – PhD student, Department of Computer Engineering, National Technical University of Ukraine “Igor Sikorsky Kyiv Polytechnic Institute”.

E-mail: VRusinovIO51@office365.fiot.kpi.ua

**Oleksii Cherevatenko** – PhD student, Department of Computer Engineering, National Technical University of Ukraine “Igor Sikorsky Kyiv Polytechnic Institute”.

E-mail: chereva@ukr.net

**Vladyslav Kuchin, Alireza Mirataei,**
**Olexander Markovskyi**

# METHOD OF SECURE MODULAR EXPONENTIATION ON REMOTE COMPUTING PLATFORMS

The paper deals with a method of secure calculation of the modular exponent, which speeds up the operation by using remote capacities. In the proposed method, protection against disclosure of the base and exponent is implemented. The computation speedup provided by the developed method is estimated.

**Key words:** modular exponentiation, secure computation, remote resources, homomorphic encryption.

**Target setting.** Due to the widespread distribution of portable low-power devices with support for cryptographic information protection protocols, which use the modular exponentiation operation in their implementations, the question of attracting powerful cloud resources to accelerate the execution of this operation is relevant.

**Actual scientific researches and issues analysis.** In connection with the rapid spread of cloud technologies in recent years, the topic of remote secure information processing is increasingly common in scientific research.

**Uninvestigated parts of general matters defining.** Despite the existence of secure remote modular exponentiation methods, the computational speedup they provide remains low. This paper is aimed at increasing the level of computational acceleration that can be achieved through the use of remote capacities.

**The research objective.** The purpose of this paper is to increase the efficiency of the use of remote computing resources in the implementation of cryptographic algorithms for information protection, which are based on the operation of modular exponentiation.

**The statement of basic materials.** The task of the secure computation of the modular exponent is to create such an organization of computations that would at the same time ensure the protection of the base and the exponent and allow to speed up the calculations, thanks to the involvement of remote resources.

**Опис запропонованого методу.** The main idea of the method is to decompose the exponent in the form

$$d = a \cdot x + b, \tag{1}$$

where $a$ and $b$ are secret decomposition coefficients that ensure the hiding of the exponent, $x$ is exponent that is passed to the remote platform for computation.

The described decomposition occurs once at the start of the program. The digit capacity of the coefficients $a$ and $b$ is chosen according to the required level of security of the exponent $d$. Another important component of the developed method is the base $m$ hiding mechanism. The basis of this mechanism is the RSA asymmetric encryption algorithm. The user needs to select the public and private keys ($E$, $n$) and ($D$, $n$) that match the condition $E \cdot D \bmod \varphi(n) = 1$, where $\varphi$ is the Euler function, $n$ is modulus. The user also calculates the product $y = D \cdot x$ where $x$ is component of decomposition of exponent $d$. This product is sent to the remote platform as an exponent for calculations. Since the values of $D$ and $x$ are independent of the base $m$, $y$ can be calculated once at the beginning of the program.

Before sending data, the user raises $m$ to the power of $E$ modulo $n$, obtaining $c = m^E \bmod n$. Taking into account that this operation is performed by user resources, $E$ should have a small bit size to reduce the computation time.

The value of $c$ is sent to the remote platform as the base, and the number $y$ as the exponent. The remote platform must calculate and return to the user the result $r = c^y \bmod n$, which, taking into account the transformations described above, is $r \equiv c^y \equiv (m^E)^{D \cdot x} \equiv (m^{E \cdot D})^x \equiv m^x \pmod n$.

After the platform returns the calculated result $r$, the user is left to decipher the answer by performing such calculations: $z = (r^a \bmod n \cdot m^b \bmod n) \bmod n$, where $r$ is result returned from the remote platform, $m$ is base; $a$, $b$ are coefficients of exponent decomposition.

**Assessment of the level of security of secret data.** The reliability of $m$-base encryption depends on the reliability of the RSA algorithm. As mentioned above, in order to reveal the encrypted value $c = m^E \bmod n$, a potential attacker needs to solve the problem of factorization of the modulus $n$ in an acceptable time, which today is considered an unsolvable problem.

The problem of revealing the encrypted exponent $d$ is reduced to the problem of selecting such coefficients $a$ and $b$ that satisfy equality (1). Taking into account the capacity of the coefficients $a$ and $b$, the total number of pairs ($a$, $b$) will be determined by the formula

$$N = 2^{l_a} \cdot 2^{l_b} = 2^{l_a + l_b}, \tag{2}$$

where $l_a$, $l_b$ are the bit lengths of the coefficients $a$ and $b$, respectively. The above formula allows us to estimate the number of pairs of coefficients among which the attacker will brute force, trying to restore the secret exponent.

**Assessment of efficiency.** One of the ways to evaluate the efficiency of the method of secure calculation of the modular exponent is to compare the number of elementary operations performed in this method on the user side with the number of such operations that need to be performed when calculating the modular exponent without involving remote resources. The ratio of the size of calculations performed by the user and the remote platform is called the acceleration factor *k*. The modular exponentiation operation requires $l_d$ to $2l_d$ modular multiplication operations, where $l_d$ is the bit capacity of exponent *d*. Based on this, we can determine the average estimate of the number of multiplications performed as $N_{\text{заг.}} = 1{,}5l_d$.

The number of multiplications performed by the user will be determined by the capacity of the numbers *a*, *b* and *E*. Let the bit capacities of numbers *a*, *b* та *E* be equal to $l_a$, $l_b$ and $l_E$, respectively. The $m^E$ mod *n* operation performed to hide the message *m* requires, on average, $1{,}5l_E$ multiplications. The remaining calculations to be performed after the remote platform returns the result require $1 + 1{,}5l_a + 1{,}5l_b$ multiplications. Thus, the total number of multiplication operations to be performed by the user in the method proposed by the author is

$$N_{\text{кор.}} = 1 + 1{,}5(l_a + l_b + l_E), \tag{3}$$

and the formula for the acceleration factor takes the form

$$k = \frac{1{,}5l_d}{1 + 1{,}5(l_a + l_b + l_E)}. \tag{4}$$

Based on the computational capabilities of modern computer systems, we can assume that the value of 50 for the bit capacity of the coefficients *a* and *b* will provide a high level of security for the secret exponent in most cases of practical application of the proposed method. Based on this value, and also taking into account that the most common value of the bit capacity of numbers currently used in the RSA cryptographic information protection algorithm is 2048, it is possible to calculate the acceleration factor that the developed method will provide in conditions of its practical use. Calculations performed using formula (4) allow us to conclude that when using a fairly common value of 16 for the bit capacity of the public key *E*, the calculation acceleration factor will be approximately equal to 18.

**Conclusions.** Theoretically substantiated, developed and experimentally investigated a method for the secure calculation of the modular exponent on remote computing power, based on the decomposition of the exponent and encryption of the base of the power using the RSA algorithm. It is shown that the proposed method makes it possible to speed up the calculation of the modular exponent by about 18 times while maintaining the security of the secret part of the data.

## References

1. Boroujerdi N. Cloud Computing: Changing Cogitation about Computing/ Boroujerdi N., Nazem S. // IJCSI International Journal of Computer Science Issues, - Vol. 9, - Issue 4. -2012.- No 3.- PP. 169-180.

2. Xiaofeng Chen. New Algorithms for Secure Outsourcing of Modular Exponentiations / Xiaofeng Chen, Jin Li, Jianfeng Ma, Qiang Tang, Wenjing Lou // ESORICS 2012, LNCS 7459, - 2012.- PP. 541–556.

3. Can Xiang. Verifiable and Secure Outsourcing Schemes of Modular Exponentiations Using One Untrusted Cloud Server and Their Application // IACR Cryptology ePrint Archive 2014: PP.500 .- https://eprint.iacr.org/2014/500.pdf

4. Markovskyi O.P. Secure Modular Exponentiation in Cloud Systems./ Oleksandr P. Markovskyi, Nikolaos Bardis, Nikolaos Doukas, Sergej Kirilenko // Proceedings of The Congress on Information Technology, Computational and Experimental Physics (CITCEP 2015), 18-20 December 2015, Krakow, Poland, C. 266-269.

## AUTHORS

**Vladyslav Kuchin** – bachelor, Department of Computer Engineering, National Technical University of Ukraine “Igor Sikorsky Kyiv Polytechnic Institute”.

E-mail: vladislavkuchin2001@gmail.com.

**Alireza Miratаei** – PhD student, Department of Computer Engineering, National Technical University of Ukraine “Igor Sikorsky Kyiv Polytechnic Institute”.

E-mail: alirezaataei@gmail.com.

**Markovskyi Oleksandr** – associate professor, Department of Computer Engineering, National Technical University of Ukraine “Igor Sikorsky Kyiv Polytechnic Institute”.

E-mail: markovskyy@i.ua

UDC 004.272.2

**Mykola Shadler, Artem Volokyta.**

# THE METHOD OF SELECTING COMPONENTS OF A COMPLEX SYSTEM BASED ON EVOLUTIONARY CALCULATIONS

*Abstract:* There is EA examines, namely genetic algorithms and methods of speeding up their work on the example of solving the problem of building a complex system - a personal computer.

As a practical part, a genetic algorithm was implemented, which solves the task using the proposed method of selecting components of a complex system, which, due to the use of evolutionary calculations, namely a genetic algorithm with a modified fitness function, allows to increase the efficiency of the process of composing elements of complex systems on the example of selecting components for personal computers or servers.

*Keywords:* PC, personal computer, genetic algorithms, evolutionary calculations, fitness function, complex system.

## Introduction

Most of the technological concepts of mankind are based on a wide variety of natural phenomena. Therefore, according to Darwin's words, the human brain is considered the most powerful tool for solving various problems, which invented the evolutionary mechanism that formed the basis of genetic algorithms.

The increasing complexity of tasks solved by machines requires the use of such mimicry, resource-intensive and, on the other hand, powerful tools as evolutionary algorithms and neural networks. EAs are able to perform the necessary research many times, taking into account and involving the best results in the next iterations of the research. Currently, the practical application of such algorithms requires experienced specialists who will monitor the input sample, the process of recombination and mutation of research objects, which limits the use of EA to solving academic problems, namely, modeling and research of various processes.

We believe that the further development of these algorithms depends on the ease of practical application of these algorithms in applied fields, so the purpose of this work is the study of EA, namely genetic algorithms and methods of speeding up their work on the example of solving the problem of building a complex system - a personal computer.

## Personal computer components

A complex system is solved - a personal computer or server, consisting of the following components:

- processor;
- motherboard;
- RAM;
- video card;
- power supply;
- permanent memory;
- body

Each of which is distinguished by a certain set of indicators that characterize the corresponding component and the level of its physical and logical compatibility with others. A successful combination of components by characteristics allows to avoid such a phenomenon as "bottle neck", which leads to excess power of some components over others and disrupts the balance in a complex system, leading to an increase in costs in accordance with the received final power.

As for first version of algorithms only several components were selected, such as CPU, Motherboard, GPU and RAM.

CPU (Central Processing Unit) is the main and main part of a personal computer, which is responsible for directly performing calculations. The CPU characteristics that will form the assessment of the power of this component are as follows:

- clock frequency;
- norms of the lithographic process (technical process, technical process);
- architecture;
- number of cores;
- volume of cache memory.

This component has only one characteristic, which will physically limit its choice in accordance with other components, first of all, the video card - the socket.

Due to logistical constraints, we will have a level power distribution with other components relative to the declared characteristics.

RAM is a memory that is volatile and is used in computer systems as a temporary storage of information that must be quickly accessed.

The most important parameters for selection are:

- frequency;
- latency scheme (timings);

These parameters directly affect the speed of memory, which is a powerful criterion for optimization.

RAM also has two parameters that exclude its ability to combine with incompatible motherboards, that is

- generation (DDR2, DDR3, DDR4);
- form factor (DIMM, SODIMM).

The motherboard is the main element of a personal computer, which provides communication between other components, namely the processor, RAM and main memory, video card, power supply and peripherals. It has the following characteristics, which are important to consider when assembling a computer:

- form factor;
- chipset (chipset);
- socket;
- support of a certain generation of RAM;
- availability of commonly used interfaces for connecting peripheral devices.

First, the socket of the motherboard must match the socket of the processor, because otherwise its installation in the processor connector is physically impossible due to the difference in interfaces (legs and contact pads). Secondly, the supported generation of RAM - differences in the physical connector and the ability of the processor to work with it, the amount of desired memory and the possible number of threads. The chipset is responsible for the maximum power of the personal computer, and the form factor affects the choice of the case.

Responsible for displaying graphics on the monitor, it can be discrete (a separate board connected to the motherboard) or integrated into the processor. The choice depends on the tasks faced by the personal computer. If this is a gaming solution, then it needs a powerful discrete video adapter, if not, then you can be satisfied with an inexpensive discrete solution, or even pay attention to those integrated into the processor.

The most important influencing parameters are length and power supply to choose the housing of the appropriate sizes and the power supply unit.

Accordingly, test models of components were created, containing the most important parameters that must be taken into account by the algorithm in the process of modeling possible final combinations of components and creating stable connections, taking into account the restriction introduced into the system regarding

the physical compatibility of individuals, which will block the occurrence of such impossible combinations. For the simplicity of the study, it was decided to reduce the number of components required for selection to four, namely:

− CPU, for example AMD Ryzen 9 5950X;

− Motherboard on the example of ASRock Z370 Pro4;

− GPU on the example of AMD Radeon RX 6950 XT;

− RAM on the example of Kingston DDR4-2933 8192.

The table with all the characteristics of these components is in appendix B under number 1, and under number 2 you can see the table with simplified components due to the reduction of the least weighty characteristics and their separation by powerful or limiting categories.

Using the website www.pcbenchmarks.net, we will determine the following benchmark scores for specific components:

1. AMD Ryzen 9 5950X – 45942 points at a price of $548;

2. AMD Radeon RX 6950 XT – 27283 points at a price of $1,100;

3. ASRock Z370 Pro4 – 40,000 – points (subjective assessment of the author according to the used chipset) at a price of $195;

4. Kingston DDR4-2933 8192 – 15342 points at a price of $50.

**Developed genetic algorithm**

On the basis of the types of genetic algorithms, selections and practical applications of genetic algorithms that I have considered, it was decided to simulate the process of selecting computer components in the form of solving a Diophantine equation of the form:

$$a + b + c + d = price$$

where price is the planned amount of money planned to be spent on the computer.

Taking into account the benchmarks of the components described in the previous paragraph, we will calculate the coefficients for the Diophantine equation based on the mathematical expectation of the final price depending on the price of each of the components:

$$548 + 1100 + 195 + 50 = 1893 \quad => \quad 1893/4 = 473{,}25$$

And therefore, the Diophantine equation will have the following form:

$$1{,}157a + 2{,}32b + 0{,}4c + 0{,}1d = 1893$$

Let's reduce the coefficients to whole numbers by multiplying by 10:

$$11a + 23b + 4c + d = 1893$$

That is, the most optimal solution that satisfies the condition of the equation is selected components of the same power class. It is worth noting that depending on the field of use of a personal computer, the balance may shift in one direction or another, such as an overload of about 30% in the direction of the video card when designing a computer for gaming, 20% in the direction of the processor for office computers or double the amount of RAM for server machines.

It is customary to calculate the advantage of one solution over another using dispersion - the closer the components are to each other in terms of power, the more optimal the system will be.

The fitness function of the applied algorithm looks as follows:

$$f = x\Delta + yD$$

where x and y are weighting factors

The algorithm creates a new population as follows:

1. Sorts the existing population according to the fitness indicator;

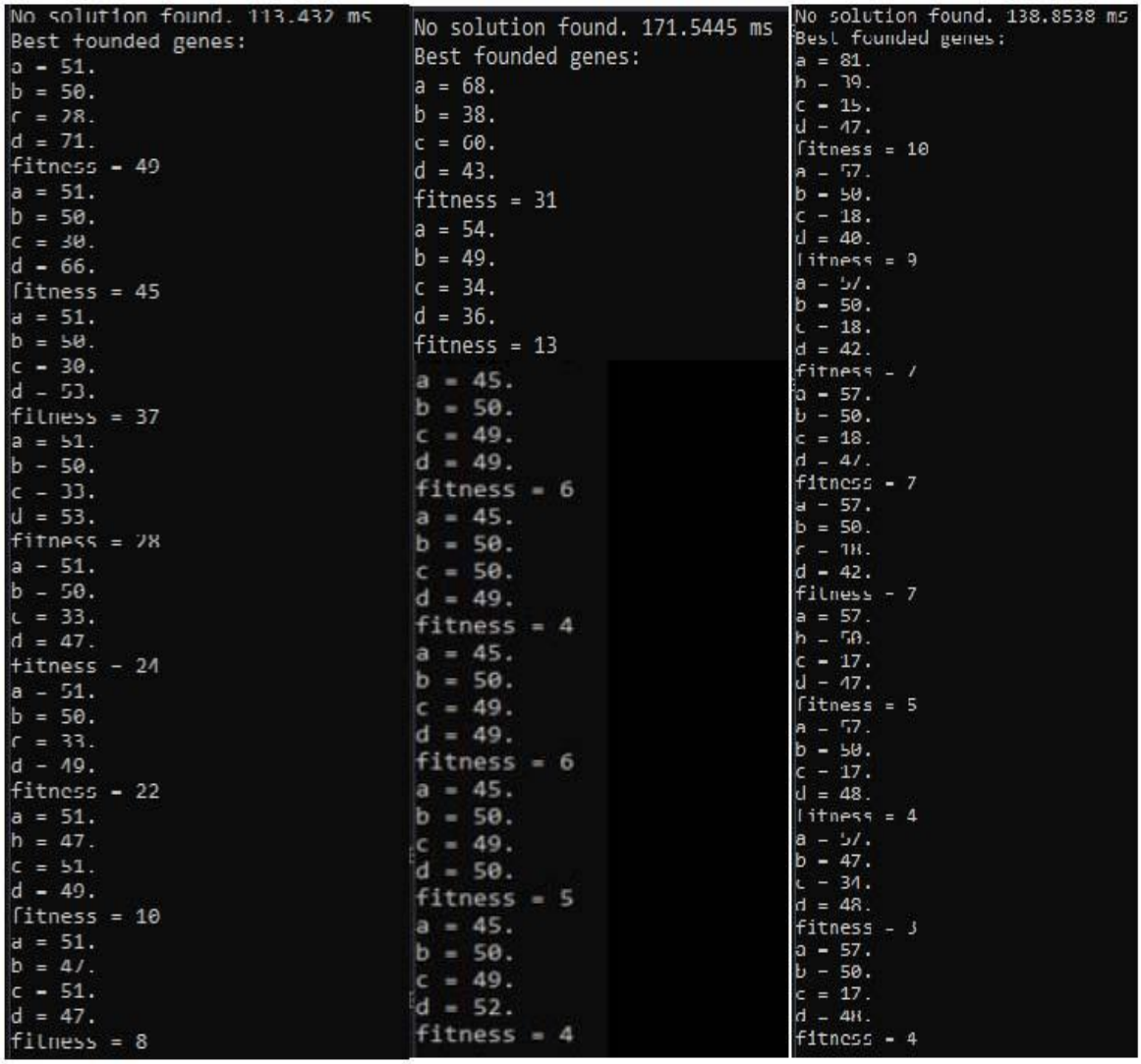
2. Makes a record of the best gene with a fitness deviation of less than 5% in the list of optimal solutions;

3. Selects parents with a survival rate of 20%, i.e. 20% of the best representatives of the population are crossed with each other according to the principle of 2 parents - 2 children;

4. Other representatives of the population are replaced by new randomly generated individuals.

The crossing of individuals takes place according to the crossover principle, in addition, both the number of genes to be exchanged and their position in the individual are chosen randomly. With a probability of 5%, a mutation occurs in both children.

If fitness by price and fitness by variance are taken with the same coefficients, then an ideal solution in the form of a gene is expected { 49, 49, 49, 49 }. With such a fitness function, the possibility of exact GA convergence is close to zero, so let's analyze the obtained solutions that fall into the top 5%. We will conduct an experiment and run the algorithm with a population size of 125 individuals and 1000 iterations for the coefficients (1; 1) (a), (1; 0.4) (b) and (1; 0.1) (c).




a) b) c)

## Conclusions

Information from scientific articles on the use of genetic algorithms in solving various modern problems was processed, and a conclusion was made about the relevance of this method of finding approximate solutions. Several main directions can be named as future trends in the development of EA. The first of the modern trends is the hybridization of two or more algorithms to obtain better results. Currently, in the literature you can find an increasing number of works that present hybrid algorithms. Also, many researchers are working on modifications of EAs to improve their computational performance.

A genetic algorithm was developed that solves the problem of Diophantine equations, to which the process of selecting components was reduced. Considering that

for the solution of this type of applied problems there is no need to achieve one hundred percent accuracy in solving Diophantine equations, we have a large number of optimal solutions that arise in the process of searching for the ideal, which indicates the effectiveness of the application of genetic algorithms to solve the problem of component selection.

A fitness function has been implemented, which evaluates the individual's fitness according to two criteria - the individual's proximity to the solution of the Diophantine equation and dispersion, because a balanced system is considered the most optimal option. The developed algorithm has a disadvantage in the form of a tendency to find local extrema, which can be solved later by increasing mutation, introducing cataclysms, etc.

## References

1. Holland J. H. Adaptation in Natural and Artificial Systems, Univ. of Michigan Press, 1975.

2. Tobias Blickle and Lothar Thiele "A Comparison of Selection Schemes used in Genetic Algorithm", 1995, 2 Edition.

3. Dorigo M, Optimization, learning and natural algorithms. PhD thesis, Dipartimento di Elettronica, Politecnico di Milano, Italy, 1992 [in Italian]

4. Glover F.(Ed.) Tabu search methods for optimization. Feature Issue of Europen J. Oper. Res. v106 (1998), N2–3.

## AUTHORS

**Mykola Shadler** – student, Department of Computer Engineering, National Technical University of Ukraine “Igor Sikorsky Kyiv Polytechnic Institute”.

**Artem Volokyta** – associate professor, PHd, Department of Computer Engineering, National Technical University of Ukraine “Igor Sikorsky Kyiv Polytechnic Institute”.

UDC 004.272.2

**Mykola Serpuchenko, Oleksandr Rokovyi**

# MULTIFACTOR AUTHENTICATION IN CORPORATE VPN NETWORKS

The paper deals with the modern methods of multifactor authentication when connecting to VPN networks. On the example of Microsoft Direct Access and Forticlient SSL VPN technologies, the main approaches to solving this problem today are shown. The shortcomings of these existing approaches are shown and a new approach that corrects them is proposed.

**Key words:** MFA, VPN, Direct Access, SSL VPN.

Fig.: 3. Bibl.: 6.

**Target setting.** According to Forbes with reference to data scientists at Ladders 25% of all professional jobs in North America will be remote by the end of 2022, and remote opportunities will continue to increase through 2023 [1]. As Virtual Private Network provides the external access to the internal resources it is one of the most vulnerable parts of the corporate network.

**Actual scientific researches and issues analysis.** Nowadays various corporations offer their own ways to securely connect to a VPN. MFA or Multi-Factor Authentication helps secure company resources by additionally verifying the identity of the remote user and device. It serves to protect critical resources from some common identity attacks. There are several corporate solutions of implementing MFA in VPN. Most of them are vendor-specific (like Microsoft DirectAccess, FortiClient VPN, Cisco AnyConnect, DUO security check) and have some drawbacks in implementation.

**Uninvestigated parts of general matters defining.** Despite a considerable number of existing variants of using MFA in corporate VPN, proposed solutions have some drawbacks. For example, Microsoft DirectAccess is invisible and transparent from user perspective, as it automatically creates IPsec VPN tunnel using user and computer certificates, however, due to its architecture it is difficult routable and based on IPv6 which creates problems to some applications. SSLVPN, for example FortiClient VPN, has no problem with IPv4 and routing, but it requires user to manually connect their devices to the VPN. That mean that VPN connection cannot be established before user logon and the device cannot be manageable and controlled until the connection.

**The research objective.** The task is to analyze the existing technologies of securing VPN access with multifactor authentication, to find their weaknesses and to propose the solution of eliminating those drawbacks.

The purpose of this work is to develop MFA-based VPN structure that is transparent to a user, stable, cost-effective and scalable.

**The statement of basic materials.** Recently Microsoft introduced the replacement of Direct Access, Microsoft Always On VPN, which mitigates some problems of its ancestor. It supports IPv4 routing and is third-party vendors compatible. However, Microsoft Always On VPN, still has some weaknesses, and the main one that VPN tunnels, as it is shown on Fig. 1, are terminated on a Windows Server which is not designed to handle a large amount of various VPN connections.

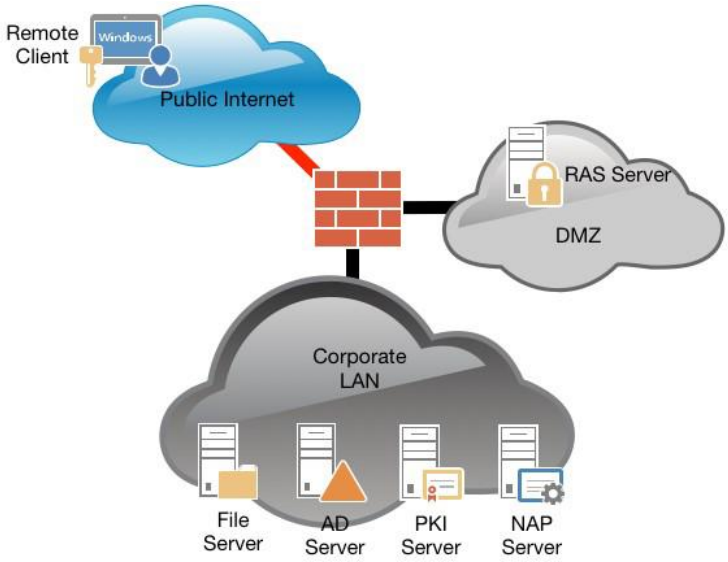



***Fig. 1.*** Microsoft Always On VPN general structure

In order to mitigate the disadvantages of "pure" Microsoft Always On VPN solution it is possible to combine Always On VPN and firewall-based VPN.

**General model structure.** The chosen structure is similar to the one that is used in Microsoft Always On VPN, as it is a common way to organize the internal infrastructure, that usually has some Active Directory, Certification Authority and Radius servers. In the proposed solution the role of RAS (Remote Access Server) is taken by a corporate firewall. Fortigate firewall was chosen as an example in this research. The process of establishing VPN connection consists of 5 steps and is divided on 2 stages. The first stage (Fig. 2) is the establishing the device tunnel. After a PC was powered on it automatically initiate the creation of IPsec tunnel between itself and the corporate firewall.

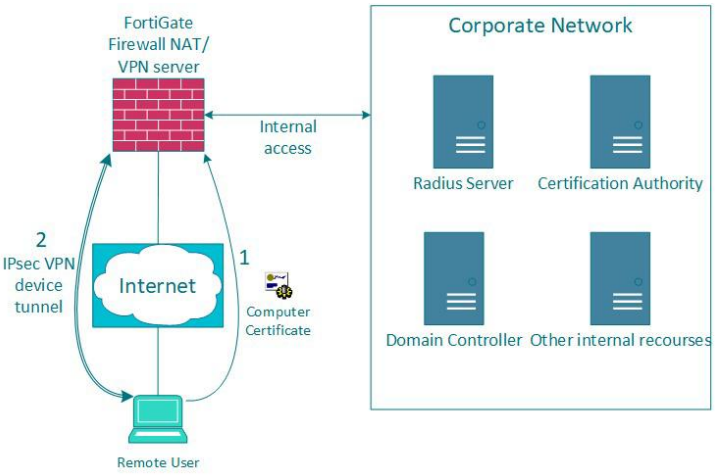



***Fig. 2.*** The first stage in Firewall based Always On VPN

If the computer certificate is valid and trusted, the Firewall establish the VPN tunnel with the remote device. At this step the first stage is completed, VPN device tunnel is established. Device tunnel is an extremely limited connection which aim is to make the remote computer be manageable by the corporate infrastructure. The second stage (Fig. 3) aims to create the user tunnel – fully operational VPN tunnel with a granular access to the internal resources.

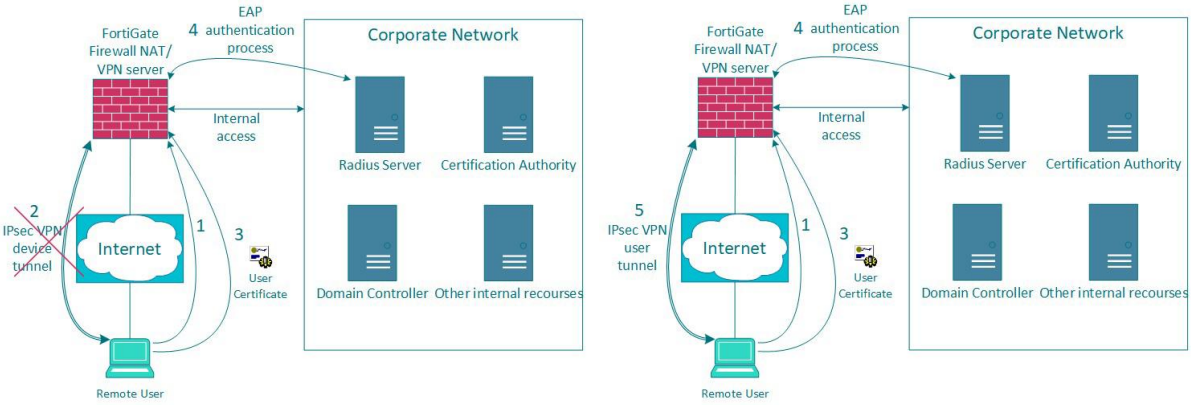



***Fig. 3***. The second stage in Firewall based Always On VPN

After the user logs in, the computer initiates the new IPsec tunnel based on the user certificate. At this step the firewall delegates the authentication process to a radius server. Authentication process uses secure EAP algorithms to prove the identity of the user and to complete the authorization. After the EAP authentication, Firewall establishes the user tunnel VPN connection and terminates the device tunnel.

**Conclusions.** The paper has demonstrated the new approach of creating MFA-based VPN structure that takes best parts of modern well-known techniques,

liquidating most of their limitations. The results of this research shows that the proposed solution exceeds other similar technologies in cost-effectiveness, compatibility, scalability, transparency and stability (unlike Windows server, firewalls like FortiGate or Cisco ASA are designed for handling a large amount of various VPN connections which makes the structure more stable).

## References

1. This Is the Future Of Remote Work In 2021 // Forbes Dec 27, 2020. URL: https://www.forbes.com/sites/carolinecastrillon/2021/12/27/this-is-the-future-of-remote-work-in-2021/?sh=18fea8a1e1de (дата звернення: 10.06.2022).

2. How remote work is quietly remaking our lives // VOX Oct 9, 2019. URL: https://www.vox.com/recode/2019/10/9/20885699/remote-work-from-anywhere-change-coworking-office-real-estate (дата звернення: 10.06.2022)

3. ISO/IEC 27001:2013(en) Information technology – Security techniques – Information security management systems – Requirements (Інформаційні технології. Методи безпеки. Системи управління інформаційною безпекою. Вимоги).

4. DirectAccess // Microsoft Documentation article Jul 29, 2021. URL: https://docs.microsoft.com/en-us/windows-server/remote/remote-access/directaccess/directaccess (дата звернення: 14.06.2022).

5. Configuring the SSL VPN tunnel // FortiOS - Cookbook версія 6.0.0. Дата оновлення 24.06.2020 URL: https://fortinetweb.s3.amazonaws.com/docs.fortinet. com/v2/attachments/a4a06ec3-12a7-11e9-b86b-00505692583a/FortiOS-6.0.0-Cookbook.pdf (дата звернення: 26.05.2022).

6. Always On VPN technology overview // Microsoft Documentation article May 19, 2022. URL: https://docs.microsoft.com/en-us/windows-server/remote/remote-access/vpn/always-on-vpn/always-on-vpn-technology-overview (дата звернення: 18.06.2022).

## AUTHORS

**Serpuchenko Mykola** – PHD student of National Technical University of Ukraine “Igor Sikorsky Kyiv Polytechnic Institute”.

E-mail: nikolay.serpuchenko@gmail.com

**Rokovyi Oleksandr** – associate professor, Department of Computer Engineering, National Technical University of Ukraine “Igor Sikorsky Kyiv Polytechnic Institute”.

E-mail: rokovoy@comsys.kpi.ua

UDC 004.056.5

**Polina Buhaichenko,**
**Al-Mrayat Ghassan Abdel Jalil Halil**

# ONE APPROACH TO ORGANIZATION OF MODULAR EXPONENTIATION ON MULTI-CORE PROCESSORS

The article proposes an approach to the organization of the modular exponentation on multicore processors, which is based on the parallelization of the computational process due to the difference in the time of modular multiplication and modular elevation to the square. It is shown that the optimal number of cores that could be involved for the organization of parallel processing is two. The speed of modular exponentation increases by a 30%.

Keywords: modular exposure, Montgomery reduction, parallel calculations.

Fig.: 1. Bibl .: 8

**Actual scientific researches and issues analysis.** Virtually all public-key cryptographic algorithms are based on modular exposure operations on numbers whose bit rate (2048 or 4096) significantly exceeds the bit size of the processor. The classical method of breaking algorithms of this class consists in factorization of the module, ie its expansion into two prime numbers. The computational complexity of this problem is determined by the bit size of the numbers.

The emergence of cloud technologies has the effect of reducing the level of protection of algorithms. To restore the balance you need to increase the level of security, and the only way to achieve this for algorithms based on number theory is to increase the bit rate. However, for modular exposition, the computational complexity is cubic in nature, ie when the bit size doubles, the computational volume increases eightfold.

Therefore, the urgent task is to find ways to accelerate the calculation of the modular exponent.

**Uninvestigated parts of general matters defining.** The basic computational operation of a wide class of public key cryptographic algorithms is modular exponentation, which is performed on numbers whose bit size significantly exceeds the processor capacity. The classical algorithm of its execution has strictly consistent character and it cannot be parallelized. The only reserve for accelerating its execution is to reduce the execution time of its component - modular multiplication. It consists of multiplication and reduction, ie finding the remainder of the division of the product by the module.

One of the possible reserves for accelerating the computational implementation of modular multiplication is the combination in time of execution of several components of multiplication by reduction.

**Analysis of recent research and publications**. Both classical algorithms of modular exposition: from the senior and from the least significant digit of the code exponents have strictly consistent character and theoretically cannot be parallelized [3]. These algorithms consist of two basic operations - modular multiplication and modular squaring, and as is clear from the algorithms, the squaring accounts for 2/3 of the computational volume [4]. Barrett's technology [5], Montgomery technology [9] or precalculation technology [5] can be used for reduction both when multiplying and when squaring.

In paper [5] it was shown that in the absence of restrictions on memory resources on the critical path of the algorithm are only operations of modular elevation to the square. Therefore, a promising way to accelerate modular exposure through parallelization is to find approaches to reduce the implementation time of modular elevation to the square and the organization of the computing process on multiple processor cores with minimal synchronization and data exchange.

**The research objective**. The purpose of the work is to increase the speed of computational implementation of the basic operation of a wide range of cryptographic algorithms for information protection - modular exponentation due to the organization of parallel process processing on multi-core processors for cryptographic information protection systems.

**The statement of basic materials**. When implementing multiplication on an *r*-bit processor, *n*-bit numbers are divided into *k* fragments, where $k=n/r$. The number of processor multiplications of fragments is $k^2$. When calculating the modular square, the number of $K_к$ rocessor multiplications is determined by the formula:

$$K_к = \frac{k^2 + k}{2}. \quad (1)$$

Thus, due to the organization of modular squaring using two factors: savings on operations of multiplication of symmetric sections and the use of Montgomery group reduction using modulus-dependent preselections, the organization of modular squaring is proposed, which allowed to accelerate this operation four times. compared with modular multiplication.

This opens up the possibility for the computational process of modular exposure to be parallelized due to the fact that the basic procedures of this operation - modular

multiplication and modular squaring - have a significant difference in the time of computer implementation.

To implement this idea, a method of modular exposure on multicore processors has been developed.

The proposed method of implementing modular exposure on $m$ processor cores is based on the idea that the $n$-bit code of the exponent $E = \{e_1,e_2,\ldots,e_n\}$, $\forall l \in \{1,2,\ldots,n\}$: $e_l \in \{0,1\}$ is forming $m$ partial exponents $\varepsilon_1$, $\varepsilon_2,\ldots,\varepsilon_m$. Each i-th,, $i \in \{1,2,\ldots,m\}$, partial exponent $\varepsilon_i$ contains a group of $n_i$ adjacent bits of the exponent $E$ at the same bit positions as the codes of the exponent $E$. All other bits of each of the partial exponents $\varepsilon_1$, $\varepsilon_2,\ldots,\varepsilon_m$ are equal to zero. It is obvious that $n_1+n_2+\ldots+n_m = n$. This means that subsets of groups of digits of exponent $E$ in partial exponents do not intersect. In other words, the following condition is met:

$$\varepsilon_1 \cup \varepsilon_2 \cup \Lambda \cup \varepsilon_m = E \,. \tag{2}$$

For each partial exponents $\varepsilon_1$, $\varepsilon_2,\ldots,\varepsilon_m$ the initial numbers $\eta_1$, $\eta_2,\ldots,\eta_m$ are determined, starting from which the partial exponents contain a group of digits of the complete exponent E. In other words, the initial numbers $\eta_1$, $\eta_2,\ldots,\eta_m$ determine the number of high zeros in the codes of partial exponents. The numerical values of these numbers are determined by the formula:

$$\eta_1 = 0,\ \forall l \in \{2,3,\ldots,m\} : \eta_l = \sum_{h=1}^{l-1} n_h \,. \tag{3}$$

Technologically, as described, the partial exponents $\varepsilon_1$, $\varepsilon_2,\ldots,\varepsilon_m$ are formed when calculating the modular exponent as follows. Based on the condition of equal load of each of the processor cores, the values of $n_1$, $n_2,\ldots,n_m$ are determined - the number of significant binary digits in each of the partial exponents. It is obvious that the values of $n_1,n_2,\ldots,n_m$ do not depend on the number A, which is the operation of modular exposition, or on the code of the exponent E, nor on the module M, but only on the bit size n of the module and the number m of processor cores the modular exposure operation is performed. Therefore, the determination of the values of $n_1,n_2,\ldots,n_m$ can be done in advance. The results obtained can be used for hundreds of thousands of calculations of the modular exponent. According to the results of preliminary calculations of the values $n_1,n_2,\ldots,n_m$, the mask codes $\mu_1$, $\mu_2,\ldots,\mu_m$ are formed. Each i-th, $i \in \{1,2,\ldots,m\}$, mask $\mu_i$ contains a group of ni adjacent units, starting with the $\eta$i-th digit. All other bits of each mask $\mu_1$, $\mu_2,\ldots,\mu_m$ are equal to zero. Thus,

we can say that each *i*-th mask with its single bits localizes a group of significant bits of the corresponding partial exponent.

Before the start of the calculations, the pre-formed masks and values of $n_1,n_2,\ldots,n_m$, as well as the values of $\eta_1, \eta_2,\ldots,\eta_m$ are transmitted to the processor cores. Direct calculation of the modular exponent $A^E$ mod M begins with the fact that all processor cores are transmitted the same codes of the exponent E, the number A over which the exposition is performed, as well as the mask M. Subsequently, on each i-volume, $i \in \{1,2,\ldots,m\}$, the processor core performs operations in the following sequence:

1. The code of the partial exponent $\varepsilon_i$ is calculated by the bit conjunction of the exponent code and the corresponding mask in the form: $\varepsilon_i = E \,\&\, \mu_i$.

2. The index j of the current bit of the exponent is set in $\eta_i$ : $j = \eta_i$.

3. The counter with the number of operations is set to zero: c = 0.

4. The value of the current result R is set to one: R = 1.

5. If the value of the current j-th bit of the partial exponent $\varepsilon_i$ is equal to one, then perform a modular multiplication: $R = R \cdot A \bmod M$.

6. Perform the operation of modular elevation to the square of the code of the current result: $R = R2 \bmod M$.

7. Perform the increment of the index $j$: $j = j + 1$ and the counter $c = c + 1$.

8. If the counter c is less than $n_i$ : $c < n_i$, then proceed to re-execution of item 5

9. Perform the operation of modular elevation to the square of the code of the current result: $R=R^2$ mod M.

10. Perform the increment of the index $j$: $j = j + 1$ and the counter $c = c + 1$.

11. If the counter c is less than $n$: $c < n$, then go to the re-execution of paragraph 9, otherwise - the end.

An important element of the proposed method is the choice of values of $n_1$, $n_2,\ldots,n_m$ – the number of significant binary digits in each of the partial exponents based on the condition of equal load on each of the processor cores.

The maximum effect of acceleration of modular exposure is achieved with such a ratio between the number of bits of the exponent processed on the second and first processors:

$$\frac{n_2}{n_1} = 1 + 2 \cdot \gamma \,. \qquad (4)$$

The coefficient β of the acceleration of the calculation of the modular exponent is determined as follows:

$$\beta = \frac{t_0}{t_1} = \frac{n \cdot (\gamma + 0.5) \cdot t_{mm}}{(n \cdot \gamma + n_1 \cdot 0.5) \cdot t_{mm}} = \frac{\gamma + 0.5}{\gamma + 0.5 \cdot \frac{n_1}{n}} = \frac{\gamma + 0.5}{\gamma + \frac{1}{4 \cdot (1 + \gamma)}}. \quad (5)$$

In particular, if the same procedure is used for modular squaring and modular multiplication, then $t_{ms} = t_{mm}$, the value of γ=1 and, according to formula (5) the numerical value of acceleration β=1.33. This means that the use of two processors by the proposed method, accelerates the calculation of the modular exponent by 30%.

**Conclusions**. The method of parallelization of the modular exposition operation is theoretically substantiated, developed and investigated, characterized in that the exponent code is divided into a number of partial exponent codes containing fragments of the main exponent and zeros, due to which the calculation of modular partial exponents can be performed independently the final result is formed as a modular product of partial results, which allows to accelerate the computer implementation of modular exposure on multicore processors by parallelization.

## References

1. Menezes Alfred, Handbook of Applied Cryptography. / Alfred Menezes, Paul C. van Oorschot, and Scott A. Vanstone // CRC Press. – 2001. – 780 p.

2. Bardis N. Secure Implementation of Modular Exponentiation on Cloud Computing Resources / Bardis N., Markovskyi O. // Proceeding of International Conference Applied Mathematics, Computational Science and Systems Engineering. Athens, Greece, October 6-8, 2017. P.90-96.

3. Montgomery P. Modular multiplication without trial division. // Mathematics of Computation. – 44(170). – 1985. – P. 519–521.

4. Markovskyi Oleksandr The Employment of Montgomery reduction for acceleration of exponent on Galoise fields calculation / O. Markovskyi, V. Masimyk, O.Kot // Proceeding of International Conference on Security, Fault Tolerance, Intelligence" (ICSFTI2020), 13-14 may 2020, Kyiv.- P.44-49.

5. Brickell E.F. Fast exponentiation with precomputation / E.F. Brickell, D.M. Gordon, K.S. McCurlay, D.B. Wilson // Advances in cryptography –Proceeding of EUROCRYPT'12, LNCS-2059, Springer-Verlag. – 2012. – P. 200-207.

6. Boroujerdi N. Cloud Computing: Changing Cogitation about Computing / N. Boroujerdi, S. Nazem // IJCSI International Journal of Computer Science Issues. – Vol. 9. – Issue 4. – 2012. – №3. – PP. 169-180.

7. Kawamura S. A fast modular exponentiation algorithm / S. Kawamura, K. Takabayashi, A. Shimbo // IEEE Transaction on Information Theory. – Vol. 94. – № 6. – 2015. – P.2136-2142.

8. Xiaofeng Chen New Algorithms for Secure Outsourcing of Modular Exponentiations / Xiaofeng Chen, Jin Li, Jianfeng Ma3, Qiang Tang, Wenjing Lou // ESORICS 2012, LNCS 7459. – 2012. – PP. 541–556.

## AURHORS

Buhaichenko Polina – student of Department of Computer Engineering, National Technical University of Ukraine “Igor Sikorsky Kyiv Polytechnic Institute”. E-mail: pbuhaichenko@gmail.com

**Al-Mrayat Ghassan Abdel Jalil Halil** – PhD student, Department of Computer Engineering, National Technical University of Ukraine “Igor Sikorsky Kyiv Polytechnic Institute”. E-mail: grayatjo@gmail.com

## **Parallel Section AI.** Machine learning, Big Data.

**Anastasiia Holovash, Olga Rusanova**

# IMPROVING THE QUALITY OF INDIVIDUAL SPORT ACTIVITIES USING COMPUTER VISION TECHNOLOGY

The paper deals with the issues of comparing human movements using computer vision technology to ensure training quality during sports. The system is designed for devices with the iOS operating system, uses a smartphone camera. The ARKit library is used to recognize the position of a person. The development is aimed at use for strength training, gymnastics, etc.

**Keywords:** Computer Vision, ARKit, Swift, movement comparison, sports.

**Relevance of the research topic.** As well as Artificial Intelligence Computer Vision technology is developing very fast which is increasingly used in sports both to improve the spectator experience and to increase the effectiveness of training.

However, most of the existing sports systems are designed to train professional teams where user interfaces are complex, require the work of a coach and are too expensive for ordinary users.[1, 2, 3]

**Target setting.** For now there is no existing convenient systems designed for sports for ordinary users. That could provide feedback for the performed exercise without the participation of the trainer and the ability to add your own exercises.

**Actual scientific researches and issues analysis.** One of the most active technological fields today is an artificial intelligence, which is basic for Computer Vision technology, which is increasingly spread in our everyday life and sport industry.

This especially applies to sport professional part. We can recognize patterns between human body movements for classification and accuracy evaluation. With the help of this technology, it is possible to track the postures of several players and assess the situation on the field. Computer Vision programs are capable of detecting and classifying hits to analyze plays and further improve athletic skills. Computer vision-based sports video analytics helps reduce feedback time for time-constrained tasks. In addition, tracking is used to help newscasters and analysts interpret and analyze sports play and tactics more quickly. [1, 2, 3]

**Uninvestigated parts of general matters defining.** Most of the existing systems for sports, such as the training system from Stats Perform, which uses 3D sensing technology, are too expensive for the average user to use. In addition, most

modern systems are designed for professional training, where user interfaces are complex and often require the work of a trainer.

Therefore, this work focuses on development the product for people who are not professionals in sports, who want to exercise correctly without harming their health. The program works with the use of the camera, without other means of recognizing the position of the body.

**The research objective.** Monitor the position of the human body during sports and compare with the exemplary exercise using the Computer Vision technology to improve the quality of individual sport activities. Create a user interface for visual feedback during sports training that will be accessible to everyone.

**The statement of basic materials.** The main tool for recognizing a person's position is the ARKit library, which includes motion capture technology. In this way, the movement of a person is tracked in real time. [4]

**Movement tracking.** In this paper we are tracking the position of 18 points [6]. These points are represent human joints. They are shown in the figure 1. These positions of the points are characterized by a Transformation matrix relative to the root joint (there is a hip joint) indicated by a black dotted arrow on the figure 2. [7, 8]

Therefore, the position of each point on the human body (x', y' z') is characterized by a matrix of transformations relative to the root joint, which is the origin of coordinates (x, y, z) in this case [5]:

$$[x'\,y'\,z'\,w'] = [a_1\ a_2\ a_2\ b_1\ a_4\ a_5\ a_6\ b_2\ a_7\ a_8\ a_9\ b_3\ 0\ 0\ 0\ 1\ ][x\ y\ z\ 1\ ]$$

Transformation matrix - numerical values that we can compare. So we can find the difference between two positions using this values.

Comparing with the largest possible difference we can find similarity of positions as a percentage for convenience.

***Fig. 1***. Image of tracked joints.

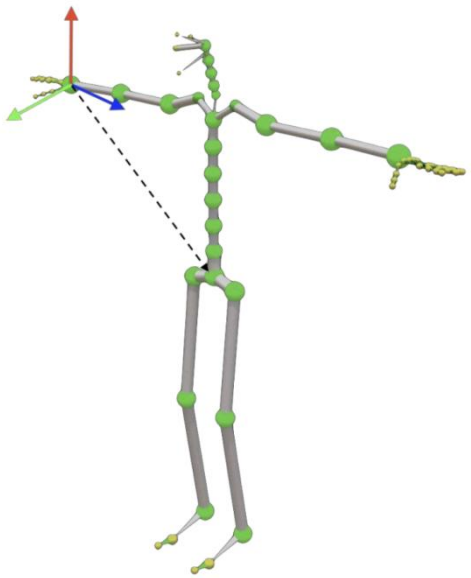


***Fig. 2.*** Location for the joint of the right hand relative to the root node of the skeleton.

**Data comparison.** So the movement can be detected by getting and comparing data from each frame of the video.

**Movement detection.** To analyze the performance of the exercise execution we need to be able to track Start and End of iteration.

To detect start of iteration for each updated frame, the system compares just received frame with the previous one. If the position has changed, the difference between the frames is large enough to be considered a movement, the movement is considered started, if the position has not changed, the system continues to compare frames.

It is possible to determine the end of the iteration if the half of the duration of the sample repetition is passed. The next step is to check the current position for similarity with the final one, and if the similarity is high enough, then we need to check whether the user has stopped in the final position. If all the described checks are successful, then the iteration is considered complete.

**General system structure.** The system consists of a mobile application containing the user interface with which the user interacts, an exercise recording module and a training module, here with the help of a camera and artificial intelligence programs built into ARKit, the video stream is processed to obtain data about the position of the human body during the exercise, the data of the user's movements are processed and analyzed with the help of developed algorithms. Processing results are displayed in the user interface or, using a service for interacting with the database, sent to the database.

The next part is a database with storage that is necessary for storing and downloading data and media materials from various devices. Figure 3 illustrates system main components.

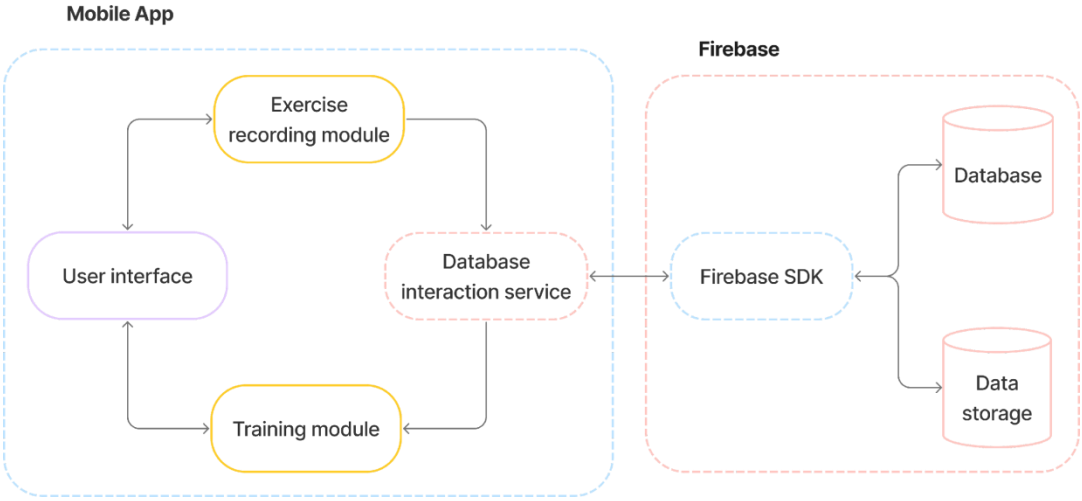



***Fig. 3.*** System main components.

**Conclusions.** As a result, an application for sports which allows the user to add a variety of exercises and gives feedback of the correctness of performance in a user-friendly form has been implemented. That improves a quality of individual sport activities and at the same time does not require the presence of a coach. It can be concluded that the goal was achieved.

Areas for further improvements:

- Display tracked points during recording (so user can understand that system detects his body correct).
- Use an AR model to explore the exemplary exercise from different sides and angles.
- Provide more accurate information about the results (for example, that the position of the hands was not correct).
- Create a setting for adding an exercise (for example, trainer can specify whether you need to follow the same speed or not).

## References

1. Virtual Trainer [Electronic resource] – Mode of access to the resource:https://apps.apple.com/ua/app/virtual-trainer/id1519100123

2. Historical & Real-Time Player Tracking [Electronic resource] – Mode of access to the resource: http://sportspower.ai/tech-rd/

3. REAL-TIME OPTICAL TRACKING [Electronic resource] – Mode of access to the resource: https://www.statsperform.com/team-performance/football-performance/optical-tracking/

4. Capturing Body Motion in 3D [Electronic resource] – Mode of access to the resource: https://developer.apple.com/documentation/arkit/content_anchors/_capturing_body_motion_in_3d

5. Елементарні перетворення матриці. [Electronic resource] – Mode of access to the resource: https://studfile.net/preview/5025632/page:4/

6. ARSkeleton.JointName [Electronic resource] – Mode of access to the resource: https://developer.apple.com/documentation/arkit/arskeleton/jointname

7. Bringing People into AR [Electronic resource] – Mode of access to the resource: https://developer.apple.com/videos/play/wwdc2019/607

8. Advances in AR Quick Look [Electronic resource] – Mode of access to the resource: https://developer.apple.com/videos/play/wwdc2019/612

## AUTHORS

**Holovash Anastasiia** – bachelor, Department of Computer Engineering, National Technical University of Ukraine “Igor Sikorsky Kyiv Polytechnic Institute”.

E-mail: nastyuha.holovasch@gmail.com

**Rusanova Olga** – associate professor, Department of Computer Engineering, National Technical University of Ukraine “Igor Sikorsky Kyiv Polytechnic Institute”.

E-mail: olga.rusanova.v@gmail.com

**Yevheniia Kolomiiets, Polina Shakhova, Artem Volokyta**

# AUDIO FEATURES EXTRACTION FOR NEURAL NETWORKS USAGE

The article deals with the issue of feature extraction of an audio signal for using the result data by a neural network that identifies the elements of harmony of a musical piece. The developed system is an audio data preprocessing service and uses such algorithms as the tonal profile algorithm (KSH algorithm) and Python, Numpy, and Librosa language tools to determine key audio.

**Key words:** audio data, audio analysis, music analysis, Python, audio processing.

Fig.: 4. Bibl.: 5.

**Relevance of the research topic.** Audio analysing services are a popular area of software development because of the demand in tools like audio identification, classifying audio and extraction of audio characteristics [1] in audio streaming and creation services for improving their algorithms (such as recommendations, storing and editing).

Tools for identifying audio characteristics are also widely used by practicing musicians.

**Target setting.** Due to the relevance of the subject of audio analysis, there are multiple services that deal with different aspects of the feature extraction, the difference being that this research focuses on preprocessing algorithms for normalizing audio signal values.

**Actual scientific researches and issues analysis.** Audio features extraction presents a complex problem and as a result has an entire field dedicated to evaluating and improving the findings of different audio processing systems.

One of the persistent research practices are present as The Music Information Retrieval Evaluation eXchange (MIREX). [2] This is a state-of-the-art, research-based approach to music analysis coordinated and managed by the International Music Information Retrieval Evaluation Laboratory. There are many different methods used by researchers being used and evaluated in the field of music data retrieval (MIR).

Some of the researched methods and functions, such as Sound Onset Detection, are small-scale MIR detection (e.g., identifying the locations of music starting points in audio files that match the index). Others, such as Symbolic Melodic Similarity, are

the MIR studies that operate at a high level (e.g. the creation of music based on patterns of similarity).

**Uninvestigated parts of general matters defining.** As a result of reviewing the existing audio analysis systems and comparing the components related to low-level data about musical pieces, the inefficiency of these algorithms according to the preprocessing aspect was found. Some of the systems have a complete set of methods for obtaining data about works, but it are limited critically – users can only get the pre-calculated information about audio that is present in the service's database.

Some systems on the other hand support analyzing and loading of user data but do not return data in the required format - only the result of calculations and are often critically inaccurate when it comes to individual values.

**The research objective.** This article has objective of researching music and signal parameters that are useful for extracting harmonic audio components, [1] developing modules that calculate these features and creating a service for audio preprocessing that allows to split into segments and calculate tonal characteristics of an audio and to save this data for usage by chord recognizing neural network.

**The statement of basic materials.** The system operates on such concepts and algorithms as reading the audio in WAV format, performing a preliminary analysis and determining the core characteristics of the piece – key and BPM, which will be saved for other stages, using the algorithm of tonal profiles and Numpy tools for the key identification and onset detection functions of Librosa [4] for the latter purpose.

The main component of work is defining the beat of the music and forming frames so that one frame represents the duration of the audio from one beat to the next and calculating the tonal features (in this case MFCC or Mel spectrogram) for each of the frames separately.

**General system structure.** The developed system should contain all the components needed to extract audio features and to perform normalization and storage of the extracted data in a file in order to save it on disk for further operations.

The chosen way perform this task is illustrated on the figure 1. The diagram illustrates the concepts of working on the segments of the audio separately and treating the subsystem of chord identification as a “black box”, where only preservation of tone-rhythm values relation is important.

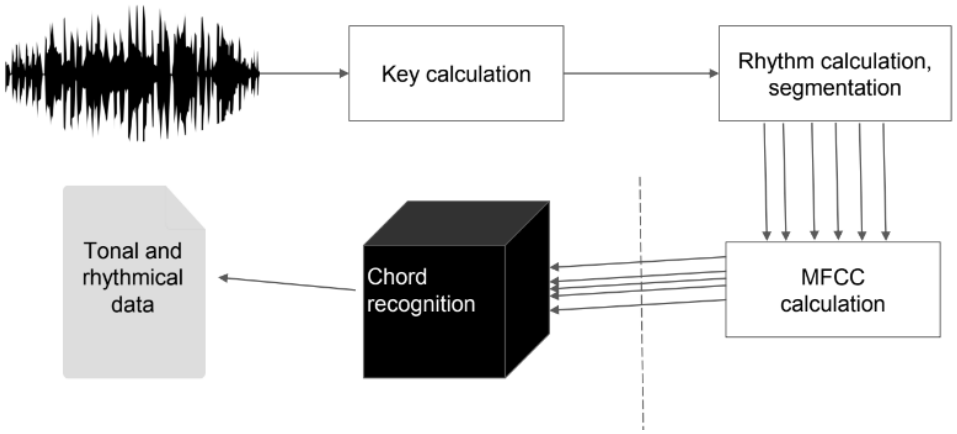



***Fig. 1.*** General system components

**Key calculation.** For key calculation the tonal profiles algorithm is used, another name being the Krumhansl-Schmuckler algorithm. [5] For effective array operations NumPy is used.

The main steps of the algorithm can be distinguished as:

1. Receiving the input data.
2. Calculating the tonal representation – chroma features from input data.
3. Calculating the sums of the values of each of the 12 notes over the duration of the entire audio – thus obtaining an audio profile.
4. Finding the correlation coefficients between the obtained audio profile and each of the 12 tonality profiles, more specifically between the theoretical tonality profile and the profile of the studied audio.
5. From the received coefficients, choosing the largest value - it corresponds to the most likely tonality result. It is also possible to obtain an alternative tonality corresponding to the second highest coefficient.
6. Returning the received values.

The flowchart representation is illustrated at figure 2.

**Audio features.** Different spectral features represent different relations. [4] Wave plot demonstrates the time-amplitude relation, chroma feature illustrates the tone-time values and STFT shows the relation between absolute frequency and time (figure 3).

The chosen formats of audio spectral features are MFCC та Mel spectrogram, which represent frequency-time relation with the help of mel cepstral coefficients. They are different from STFT because they are preserving timbre, don’t store absolute frequency, but store value according to the human ear, and are the most popular features for training neural networks – more details in related work.

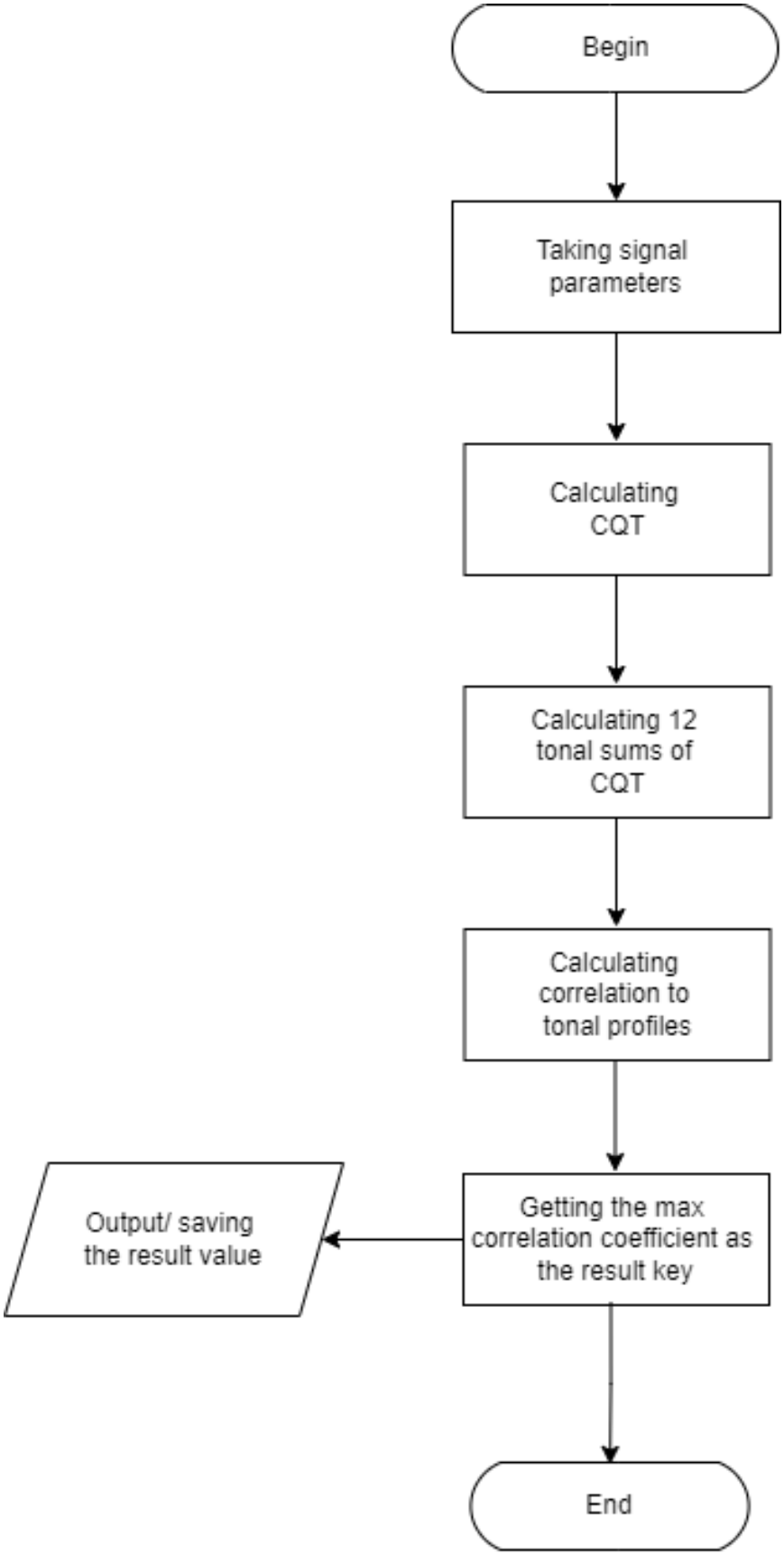



***Fig.*** 2. Flowchart of key detection component

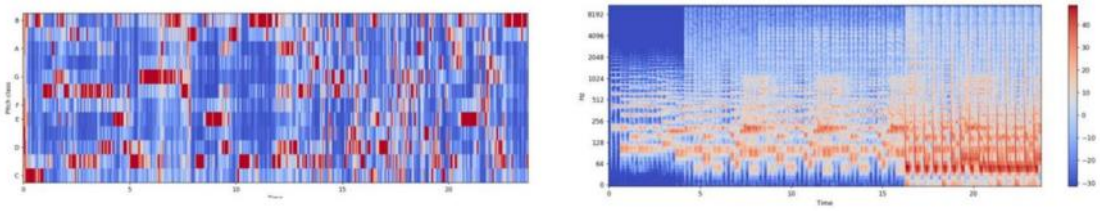


***Fig. 3.*** Spectral features of audio signal

**Evaluation of the results.** Algorithm of rhythm calculation was tested for the results correctness: accuracy of results is preserved up to 0.03 deviation, where Spotify and visualisation was used for theoretical values.

Algorithm of key calculation was also tested for the results correctness: inaccurate results could be observed in songs with weak harmonical components with average accuracy being 0.8. Spotify [3] and musical theory were used for obtaining theoretical values.

Files with the results data are presented as JSON files with metadata header, where one of the fields has segment data, presented as headers and MFCC features (figure 4).


```
{
    "name": "Alejandro Gaga Lady",
    "BPM": 99.38401442307692,
    "key": "G",
    "alt_key": "D",
    "frame_size": 512,
    "sample_rate": 22050,
    "mfccs": [
        {
            "header": {
                "id": 0,
                "frame_start": 0,
                "frame_end": 586
            },
            "mfcc": [
                [
                    -715.8488159179688,
                    0.500795304775238,
                    0.5006826519966125,
                    0.5004936456680298,
                    0.5002302527427673,
                    0.4998905062675476,
                    0.49947649240493774,
                    0.49898630380630493,
                    0.49841994047164917,
```


***Fig. 4.*** Results data file

A tool for data extraction was also developed, which allows to get WAV audio file with a YouTube URL. Tool is working correctly, but some predictable loss of quality is observed compared to pure WAV file.

**Conclusions.** As a result of research, the following conclusions were made:

1. The audio parameters were researched and it has been found that key, rhythm and tonal features should be extracted for proper audio evaluation and neural network teaching.

2. The tools for audio visualization and feature calculation were developed, and it has been affirmed that the developed system works correctly and has an acceptable accuracy of results.

3. The format of the result data was found to be readable and convenient.

As the results of research and evaluation it is safe to say that the results could be useful for appliance in the field of Musical Information Retrieval. The next step of music data usage is the neural network for chord recognition.

Some possible improvements could be suggested for future work: adding the ability to identify the work not by name and regular expression, but by audio fingerprint and implementing of working with more audio formats.

## References

1. Phillip Magnuson. A Structural Examination of Tonality, Vocabulary, Texture, Sonorities, and Time Organization in Western Art Music [Electronic resource] / Phillip Magnuson – Mode of access to the resource: https://academic.udayton.edu/phillipmagnuson/soundpatterns/compbasics/5otherparts.html.

2. The Music Information Retrieval Evaluation eXchange (MIREX) [Electronic resource] – Mode of access to the resource: https://www.dlib.org/dlib/december06/downie/12downie.html

3. Spotify. Web API Guides [Electronic resource] / Spotify – Mode of access to the resource: https://developer.spotify.com/documentation/web-api/guides/.

4. Mcfree B. librosa: Audio and Music Signal Analysis in Python / Mcfree Brian, 2015. – p. 24.

5. Madsen S. KEY-FINDING WITH INTERVAL PROFILES / Madsen Søren – Linz. – p. 4.

## AUTHORS

**Kolomiiets Yevheniia** – bachelor, Department of Computer Engineering, National Technical University of Ukraine “Igor Sikorsky Kyiv Polytechnic Institute”.

E-mail: jennichka10155@gmail.com

**Shakhova Polina** – bachelor, Department of Computer Engineering, National Technical University of Ukraine “Igor Sikorsky Kyiv Polytechnic Institute”.

E-mail: schachowa.paulina@gmail.com

**Volokyta Artem** – associate professor, Department of Computer Engineering, National Technical University of Ukraine “Igor Sikorsky Kyiv Polytechnic Institute”.

E-mail: artem.volokita@kpi.ua

**Polina Shakhova, Yevheniia Kolomiiets, ArtemVolokyta**

# METHOD BASED ON CONVOLUTIONAL NEURAL NETWORK FOR MUSICAL CHORD RECOGNITION

The article deals with the issue of Automatic Chord Recognition (ACR) using Convolutional Neural Networks (CNN). Recognition is based on the Mel-Frequency Cepstral Coefficients (MFCC) of the input audio signal. The proposed system allows to recognize 25 basic chords. Python language and TensorFlow library were used for the development.

**Keywords:** Automatic Chord Recognition, Convolutional Neural Networks, MFCC, TensorFlow.

**Relevance of the topic.** The problem of the automatic chord recognition has been known since the last century. Automatic chord recognition systems are widely used in many areas, for example for music generation, for classification of the music into various categories (genre, mood), for song identification, etc.

**Target setting.** Nowadays there are many conferences, competitions and other events in the field of Music Information Retrieval (MIR), including the annual ISMIR conference to exchange ideas and innovations related to this field, MIREX competition of algorithms for musical chords recognition, etc. However, there is no unambiguous and completely optimal solution of this problem at the moment and the existing solutions need improvements.

**Actual scientific researches and issues analysis.** A lot of research in this area has focused on deep neural networks: convolutional (CNN) [4, 5], recurrent (RNN) [2], long short-term memory (LSTM) [7].

**Uninvestigated parts of general matters defining.** The following issues were noticed in reviewed works:

- training datasets usually consist of songs of one artist or one genre, that can affect the model's quality of recognition on the real data [1, 2];
- model input is usually audio features, that require to store a large amount of data [2, 4, 7];
- some of the models is able recognize only a small number of chords [3].

**The research objective.** The purpose of this paper is to investigate the application of the convolutional neural networks to recognize musical chords from music audio recordings. As a solution, the article focuses on creating a model that

takes into account the previous observations. Proposed model is able to classify 25 types of chords: 12 minor and 12 major chords and 1 “non-chord” type. A set of Mel-Frequency Cepstral Coefficients or MFCC extracted from each audio signal is used as model input features. The model is implemented using Python and TensorFlow library.

**The statement of basic materials.** In general, the task of recognizing song chords can be defined as: for a given sound signal $x(t)$, where $t \in [t_{start}, t_{end}]$ and a set of possible chord classes $Y$, for each moment of time $t$ it is necessary to determine the chord that sounds at that moment.

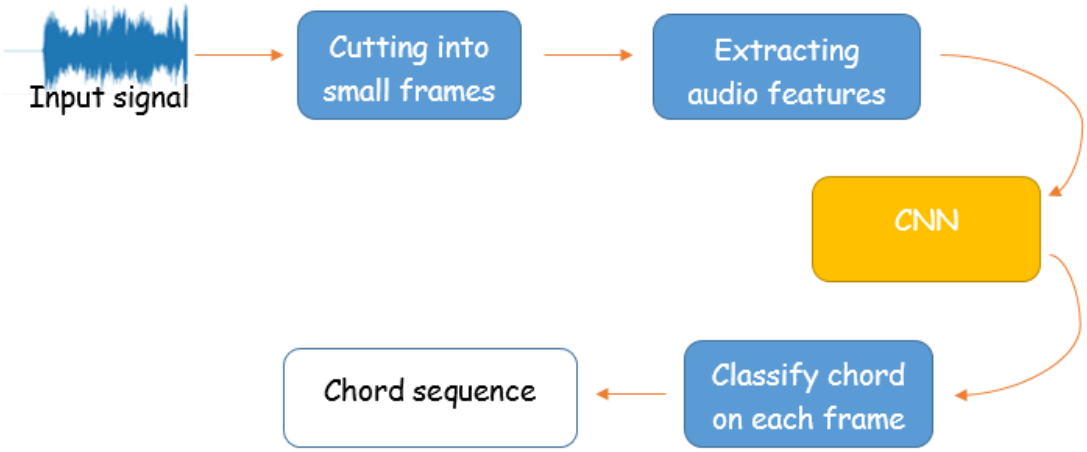



***Fig. 1.*** General system structure

**Model input features.** MFCC were decided to use as model input features, because unlike the other sound characteristics they represent a small amount of data, providing a lot of information at the same time.

**Creating a model.** A convolutional neural network architecture was chosen to perform this task. Convolutional Neural Networks (CNNs) are widely used in image classification, but they also have shown very good results in audio processing (speech recognition, music identification).

The output of each convolutional layer is activated with the ReLU function. Batch normalisation is performed after two first max pooling layers. Dropout with probability 0.3 is applied after the second batch normalization layer, last max pooling layer and fully-connected layer.

***Table 1.*** **Configuration of the proposed CNN**

| | |
|---|---|
| Convolution × 3 layers | N × 40 × 32 |
| Max pooling | 3 × 3 |
| Convolution × 2 layers | 9 × 20 ×64 |
| Max pooling | 3 × 3 |
| Convolution × 2 layers | 5 × 10 × 128 |
| Max pooling | 2 × 2 |
| Flatten | 1920 |
| Fully-connected | 128 |
| Softmax | 25 |

**Experiments.** In addition to the Isophonics collection of 180 transcribed *The Beatles'* songs, which is traditionally used for chord recognition tasks, the training set in this work is extended with also popular Isophonics *Queen* dataset and a collection of popular rock and pop songs presented at one of the stages of MIREX competition. The total dataset consists of 379 songs with their transcriptions stored in the format: *Start_time – End_time – Chord*

| | Start | End | Chord |
|---|---|---|---|
| 0 | 0.000000 | 1.053119 | N |
| 1 | 1.053119 | 3.593854 | B:min |
| 2 | 3.593854 | 6.090000 | G |
| 3 | 6.090000 | 8.655804 | E |
| 4 | 8.655804 | 11.140340 | A |
| 5 | 11.140340 | 13.659705 | A |
| 6 | 13.659705 | 16.109410 | C#:min |
| 7 | 16.109410 | 18.686825 | F#:min |
| 8 | 18.686825 | 19.371814 | D |

***Fig. 2.*** Example of chord transcription file

Chords in each song transcription file were simplified to 25 basic chord classes. Both audio and chord transcription files were aligned by time and cut into small (about 0,2 seconds) frames. After extracting MFCCs from each audio frame, results were saved into JSON-file.

A large range of input data can affect neurons and lead to incorrect adjustment of coefficients, so the training data was additionally aligned before feeding into the neural network. This means reducing the data to the interval [-1; 1].

**Evaluation the results.** The main metrics for evaluating the results of training NN are accuracy function and loss function. Loss function shows the average cost of training for the epoch, and accuracy function - the amount of correctly classified data.

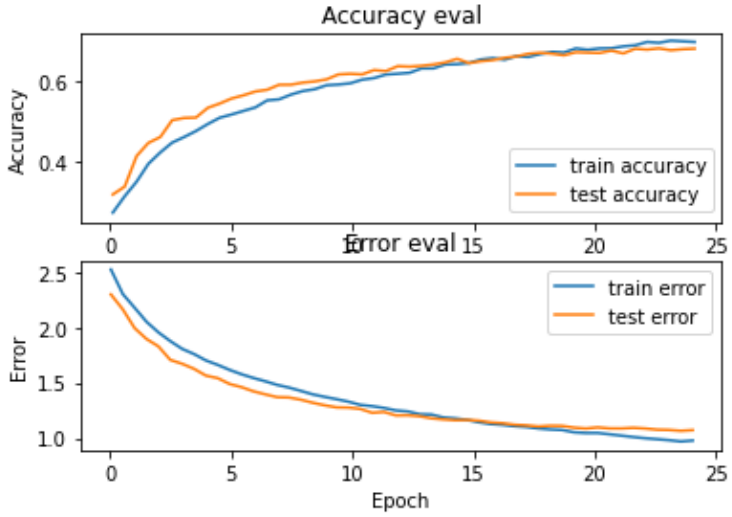



***Fig. 3.*** Loss and accuracy function results

Throughout the learning process, the amount of loss is constantly decreasing, while the accuracy is increasing. Both graphs for training and test data are similar in shape and change synchronously.

**Conclusions.** This paper demonstrates the method based on Convolutional Neural Networks for Automatic Chord Recognition tasks. Proposed model is able to recognize all basic minor and major triads and “non-chords” from a set of Mel-Frequency Cepstral Coefficients extracted from each input audio signal. It can be seen that the use of such combination produces good results. The model was trained on extended dataset of the songs, so it is expected to perform qualitative recognition on the real data.

The possible direction for future work is to increase the number of chords that the model is able to recognize. This task is primarily complicated by the uneven use of different chords and could require a more complex neural network architecture.

## References

1. H.-T. Cheng, Y.-H. Yang, Y.-C. Lin, I.-B. Liao, and H. H. Chen. (2008). *Automatic chord recognition for music classification and retrieval.* In 2008 IEEE International Conference on Multimedia and Expo. (pp. 1505–1508).

2. N. Boulanger-Lewandowski, Y. Bengio, and P. Vincent. (2013). *Audio chord recognition with recurrent neural networks.* In ISMIR. (pp. 335–340).

3. J. Osmalskyj, J-J. Embrechts, S. Piérard, M. van Droogenbroeck. (2012). *Neural networks for musical chords recognition.* In Journées d'Informatique Musicale, hal ID: hal-03041758.

4. F. Korzeniowski, G. Widmer. (2016, Sept. 13–16). *A fully convolutional deep auditory model for musical chord recognition*. In IEEE International Workshop on Machine Learning for Signal Processing.

5. E. J. Humphrey, J. P. Bello. (2012). *Rethinking automatic chord recognition with convolutional neural networks*. In 11th International Conference on Machine Learning and Applications IEEE.

6. M. McVicar, R. Santos-Rodriguez, Y. Ni, T. D. Bie. (2014). *Automatic Chord Estimation from Audio: A Review of the State of the Art*. In IEEE ACM Transactions on Audio, Speech, and Language Processing. (pp. 556 – 575).

7. S. Nakayama and S. Arai. (2018). *DNN-LSTM-CRF model for automatic audio chord recognition*. In Proceedings of the International Conference on Pattern Recognition and Artificial Intelligence (pp. 82–88).

## AUTHORS

**Shakhova Polina** – bachelor, Department of Computer Engineering, National Technical University of Ukraine "Igor Sikorsky Kyiv Polytechnic Institute".

E-mail: schachowa.paulina@gmail.com

**Kolomiiets Yevheniia** – bachelor, Department of Computer Engineering, National Technical University of Ukraine "Igor Sikorsky Kyiv Polytechnic Institute".

E-mail: jennichka10155@gmail.com

**Volokyta Artem** – associate professor, Department of Computer Engineering, National Technical University of Ukraine "Igor Sikorsky Kyiv Polytechnic Institute".

E-mail: artem.volokita@kpi.ua

**Fedir Prokhnytskyi, Oleksandr Rokovyi.**

# MAIL MESSAGE FILTERING BASED ON ARTIFICIAL INTELLIGENCE

**Abstract.** The purpose of the work is to analyze the effectiveness of the email filtering module. The research uses a dataset from the Kaggle platform that has been processed and supplemented with additional messages, as well as classifiers based on two different models: a naive Bayesian classifier; support vector machines method. The effectiveness of each of the approaches based on the same sample was analyzed using model metrics: precision and recall. Each model was further tested on the test data set.

**Keywords:** artificial intelligence, machine learning, classifier, neural network.

## Introduction

The increase in the number of unsolicited emails, called spam, has created a need for spam filters to reduce the time and effort involved in managing inboxes as well as managing storage. Spam does not allow the user to fully and effectively use time, memory and network bandwidth. The sheer volume of spam flowing through computer networks wreaks havoc on mail servers' memory space, communication bandwidth, processor power, and user time. Effective spam filters can prevent cyber fraud to users and data can also be protected from spammers. Recently, machine learning techniques have been extremely successful in detecting and filtering spam. These models mainly "learn" on data sets that are previously formed and are able to find "commonality" in new data that comes from the outside.

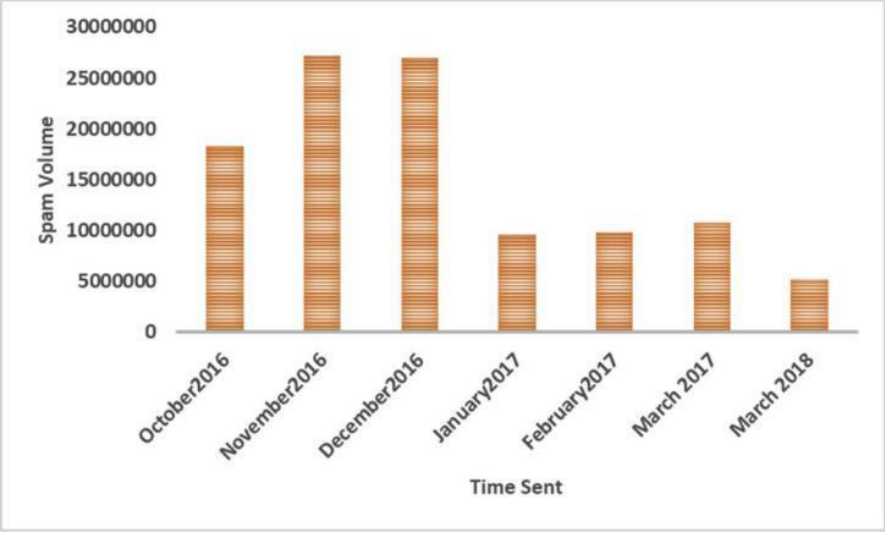



***Fig. 1***. Volume of spam from 2016 to 2018

According to Kaspersky Lab's report, in 2015 the volume of spam was down to a 12-year low. Spam volume fell below 50% for the first time since 2003. According to anti-virus software developer Symantec, spam fell to 49.7% in June 2015 and 46.4% in July 2015. The latest statistics from 2016 show that spam accounted for 56.87% of email traffic worldwide, with the most prominent types of spam being medical and dating spam.

## Analysis of existing solutions

The Naive Bayes classifier is a supervised learning method based on probability and statics. This method of filtering letters uses an adaptive set of rules, and the corresponding set of probabilities is set according to the classification decisions and received letters. Each mail is described by a set of attributes, and each attribute is assigned a probability according to the number of times it occurred in the training set. A naive Bayes classifier for spam filtering uses a simple probability formula that can be interpreted as (where c=spam): "The probability that a message will be spam, given its characteristics, is equal to the probability that any message will be detected as spam, multiplied by the probability of features occurring in spam, divided by the probability of detecting those features in any message."[1]

The advantage of this approach is that the sample size requirements are reduced from exponential to linear. The disadvantage is that the model is accurate only if the independence assumption holds.

The support vector machines method is a supervised learning algorithm that has shown much better performance than other classifiers due to its multidimensional bounds and simplicity. It maximizes the distance to the nearest reference point, and points equidistant from a given reference point are called support vectors. A linear combination of these support vectors forms a classifier or partition hyperplane.

Algorithm: the training set S is introduced, and the kernel function is determined in the form {c1, c2,…..,cn} and {d1, d2,…..,dn}. The number of nearest neighbors, say k, is assigned. Then a two-stage for loop is designed, c=ci from 1 to n is set for the outer loop. The inner loop is performed for j from 1 to q, where the SVM classifier function f(x) is designed with the fusion parameters (c, d). Using the if-else condition, the classifier function f(x) is compared with the best classifier given by the k-fold cross-validator. Therefore, a return command is given to classify the message as spam or non-spam, shown in Figure 2.

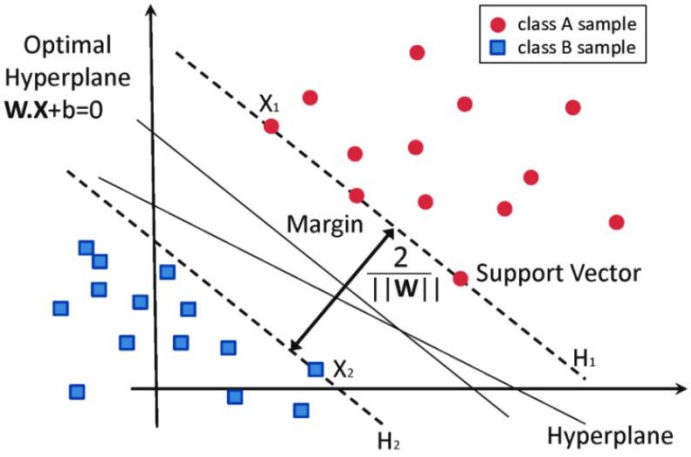



***Fig. 2***. Support vectors separate elements from different classes (spam/non-spam)

**Description of the features of the work results**

The assessment of the relevance of the models is based on the comparison of the metrics of these models. Two metrics will be used in this study: precision and recall. Precision (also known as positive predictive value) is the proportion of relevant instances among retrieved instances, while recall (also known as sensitivity) is the proportion of relevant instances that were retrieved. Therefore, both precision and recall are based on relevance.

Consider a computer program for recognizing dogs (the corresponding element) in a digital photograph. After processing an image containing ten cats and twelve dogs, the program identifies eight dogs. Of the eight items identified as dogs, only five are actually dogs (true positives), while the other three are cats (false positives). Seven dogs were missed (false negatives) and seven cats were correctly excluded (true negatives). The program's precision is then 5/8 (true positives / selected items) and its recall is 5/12 (true positives / matched items). When a search engine returns 30 pages, only 20 of which are relevant, instead of returning 40 additional relevant pages, its precision is 20/30 = 2/3, which tells us how valid the results are, while its recall is 20/ 60 = 1/3, which tells us how complete the results are. Adopting a statistical hypothesis testing approach in which the null hypothesis in this case is that the subject is irrelevant, i.e. not a dog, no Type I and Type II errors (i.e., perfect specificity and sensitivity of 100% each) corresponds to perfect accuracy, respectively ( no false positives) and perfect recall (no false negatives). In general, recall is simply the addition of the Type II error rate, that is, one minus the Type II error rate. Accuracy is

related to the Type I error rate, but in a slightly more complicated way because it also depends on the prior distribution of viewing the relevant and irrelevant item.

The cat and dog example above contained 8 − 5 = 3 Type I errors, for a Type I error rate of 3/8, and 12 − 5 = 7 Type II errors, for a Type II error rate of 7/12. Precision can be thought of as a measure of quality and recall as a measure of quantity. Higher precision means that the algorithm returns more relevant results than irrelevant ones, and high recall means that the algorithm returns most of the relevant results (whether or not irrelevant results are also returned). In fig. 3.1 shows the distribution of false positive/negative values and formulas for calculating accuracy and recall.[2]

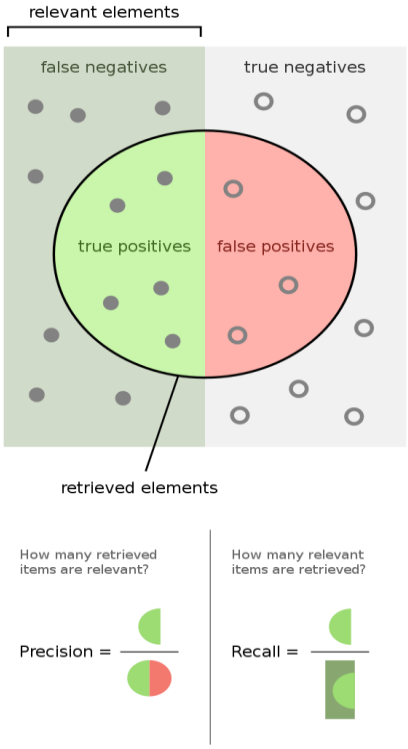



***Fig. 3.*** The set of results of machine learning models

Also, a data set is specially selected for model training. It is not necessary that this set should contain the same amount of data of both classes (spam, useful mail). In this study, model training takes place taking into account that the model should not contain false positive values, accordingly, it is much worse to misclassify useful mail: add it to the spam box, than to misclassify spam - add it to the inbox. This study used a data set with the following ratio of spam to useful mail.

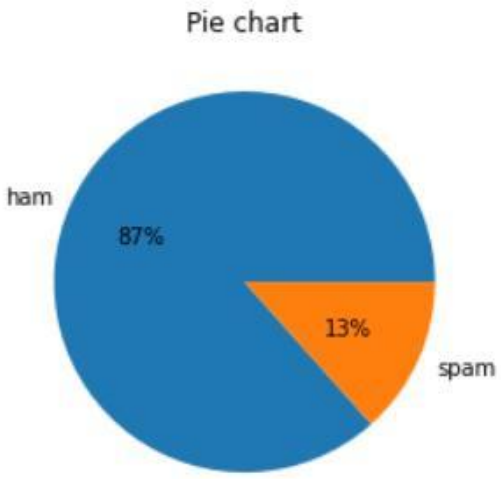



***Fig. 4.*** The ratio of spam to useful mail in the data set

**Comparison results of trained and tested models**

The result of the study of models based on the Bayes classifier and based on the support vector machines method. Training and testing will be carried out, as well as metrics will be taken: accuracy, recall level, which will help to choose the best model. The best model will be the one whose metrics will be the highest, and also according to the results of testing, which will have the least false positive results, that is, such results when useful mail was determined as spam, while spam mail determined as useful is allowed, but nevertheless the model that will have fewer such results will still be better, that is, the goal of the study will be a certain compromise between the models with the least number of false positive results and the largest number of true negative results.

**A model based on a naive Bayesian classifier**

Many experiments with different hyperparameter alpha will be performed for this model. In machine learning, hyperparameter optimization or tuning is the problem of choosing the optimal set of hyperparameters for a training algorithm. A hyperparameter is a parameter whose value is used to control the learning process. In contrast, the values of other parameters (as a rule, node weights) are studied.[3]

In this study, the alpha parameter will be randomly selected from 0.00001 to 20 in steps of 0.11. Also, in the course of training, lists with precision and recall will be formed for each iteration, which will allow choosing the best iteration.

```
[12]: alpha              12.430010
      Train Accuracy      0.973205
      Test Accuracy       0.970636
      Test Recall         0.785124
      Test Precision      0.989583
      Name: 113, dtype: float64
```

***Fig.5.*** The best model selected on the basis of metrics is built on the basis of the Bayesian classifier

The selected model has an accuracy index of 0.989 and a recall value of 0.785. Based on this model, you can build a table of spam/non-spam detection results.

| | True result | False result |
|---|---|---|
| Ham | 1592 | 5 |
| Spam | 43 | 199 |

***Fig. 6.*** The result of model testing

As a result of testing, you can see that the model has five false-positive results and 43 true-negative results (spam emails that got to the spam box)

**A model based on support vector machines methods**

For this model, many experiments will be conducted with different hyperparameter C. In this study, the parameter C will be chosen randomly from 500 to 1000 with a step of 100. Also, during training, precision and recall lists will be formed for each iteration, which will allow to choose the best iteration. The selected model has an accuracy index of 0.994 and a recall value of 0.8099. Based on this model, you can build a table of spam/non-spam detection results.

```
C                  500.000000
Train Accuracy       1.000000
Test Accuracy        0.974443
Test Recall          0.809917
Test Precision       0.994924
Name: 0, dtype: float64
```

| | True result | False result |
|---|---|---|
| Ham | 1596 | 1 |
| Spam | 46 | 196 |

***Fig. 7***. The result of training and testing the best model, which was selected on the basis of metrics and built by the SVM method

As a result of testing, you can see that the model has one false-positive result and 46 true-negatives (spam emails that got to the spam box).

## Conclusion

As a result of studying and testing two models based on different types of machine learning algorithms and based on the selected dataset, it was determined that the most usable model is the model based on the support vector machines method, because it has the highest accuracy, the highest recall and high speed, which makes it suitable for use on large data sets. This model is used as the core of the mail filtering system.

## REFERENCES

1. Bayesian Optimization in a Billion Dimensions via Random Embeddings https://jair.org/index.php/jair/article/view/10983

2. Fundamentals of Deep Learning http://perso.ens- lyon.fr/jacques.jayez/ Cours/Implicite/Fundamentals_of_Deep_L earning.pdf

3. Bergstra, James; Bengio, Yoshua (2012)."Random Search for Hyper-Parameter Optimization". Journal of Machine Learning Research. 13: 281–305. - https://jmlr.csail.mit.edu/papers/volume13/bergstra12a/bergstra1 2a.pdf

## AUTHORS

**Fedir Prokhnytskyi** – PhD student, Department of Computer Engineering, National Technical University of Ukraine “Igor Sikorsky Kyiv Polytechnic Institute”.

**Rokovyi Oleksandr** – associate professor, Department of Computer Engineering, National Technical University of Ukraine “Igor Sikorsky Kyiv Polytechnic Institute”.

UDC 004.8

**Andrii Kobyliuk, Artem Volokyta**

# METHOD OF SCHEDULING BASED ON ARTIFICIAL INTELLIGENCE

The article examines the method of planning tasks in a computer system using artificial intelligence approaches. A problem-modified genetic algorithm was used for its implementation.

**Keywords:** scheduling, artificial intelligence, genetic algorithm.

Fig.: 7. Tabl.: 1.

**Relevance of the research topic.** Because heuristic planning algorithms are specialized for a certain choice of inputs, the task of choosing the universal and most efficient one is a non-trivial one, always sacrificing in some cases to gain in others. If there are many algorithms, the choice can be made using prediction. New methods based on the field of artificial intelligence can work effectively without prior knowledge of the problem space and can be used to solve the scheduling problem.

**Target setting.** A genetic algorithm can be used to optimize a scheduling problem in hopes of further improving the overall performance and efficiency of a computer system.

**Actual scientific researches and issues analysis.** One of the first attempts was made [1] by Lawrence Davis using genetic algorithms (GA) to solve the scheduling problem. Davis noted the effectiveness of using the method stochastic search where a genetic algorithm operated on a given list which was subsequently used to form the actual planning schedule. Later, the same idea was developed by Philip Husband, who isolated [2] all the most modern, at that time, genetic planning algorithms. He noted similarity between the planning problem and other problems from the class of such problems problems such as the traveling salesman's problem, the problem of layout and packing in a backpack etc.

**Uninvestigated parts of general matters defining.** Despite the large number of works devoted to the application of genetic algorithms for planning, the problem of using new algorithms for this purpose remains understudied. The task of optimizing the planning problem primarily concerns the NP-class of algorithms, so a deterministic algorithm cannot be created to solve ill-posed problems.

**The research objective.** The purpose of this work is to research methods and models of planning, which would be based on methods of artificial intelligence, development of a software product that would implement the data model in practice.

**The statement of basic materials.** Steps below explain the basic flow of the algorithm, and shown in Figure 1:

1. Randomly generate initial population of individuals (first generation).

2. Evaluate the fitness of each individual according to the objective function or goals; distance and time taken

3. The following steps are repeated until termination criterion is met:

i. Reproduce the best individuals.

ii. Best individuals are put through crossover and mutation phase

iii. Birth of new offspring/individuals. iv. Evaluate the new individuals.

v. Replace least-fit individuals.

4. The termination occurs when either:

i. The goals are achieved, or

ii. The growth/improvement of the generation stops (iteration limit).

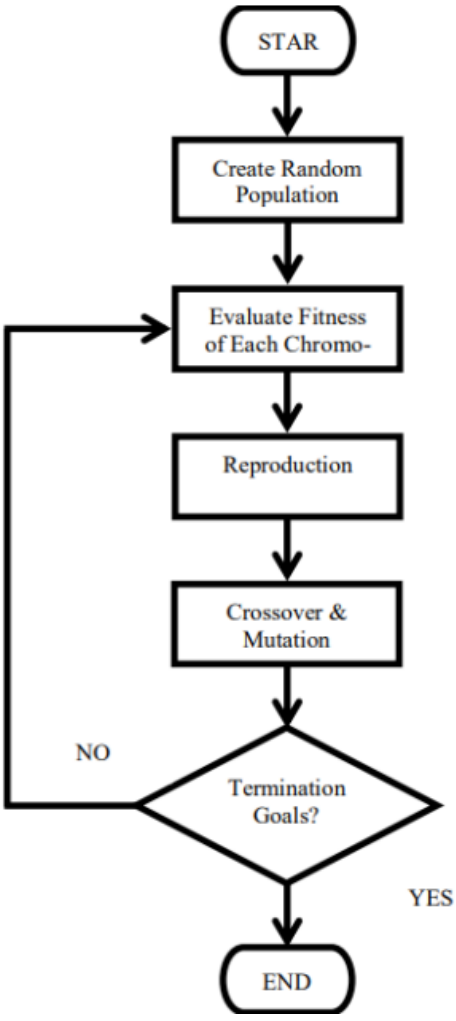



***Fig. 1.*** GA steps

A total of 100 chromosomes were generated randomly before further improved using genetic operators in GA. Table 1 shows the example of genes for each chromosome. The C(n) is the chromosome number for the population which was set to 100 (n=100). While, the T(i) is the task number. In this study, there are k processors utilized to complete the overall tasks.

***Table 1.*** **Chromosome structure**

| Chromosome | T1 | T2 | ... | Tm |
|---|---|---|---|---|
| C1 | P1 | P2 | P1 | P1 |
| C2 | P2 | P4 | Pk-1 | P1 |
| ... | | | | |
| Cn | P1 | Pk | P2 | P3 |

The fitness value of every single chromosome is calculated to evaluate its quality towards the objective function. The fitness value is measured as in unit of time. The evaluation process was conducted by using the following equation:

$$F(i) = maxFT - FT(i) + \frac{1}{(maxFT - minFT + 1)}$$

where minFT and maxFT are scalar quantities denoted respectively the maximum and minimum completion time among all chromosomes in the data population. 55(5) is the termination time for the ith chromosome.

The entire population of chromosomes is then sorted and the best 30 chromosomes are sorted out from the population. The selection is inspired from the natural selection in evolution. In which the best chromosome with the best fitness value has the higher probability of surviving compared to those whom have less. These selections were meant towards for a better solution.

Each chromosome is given the proportional size of the sector, which corresponds to its value of the fitness function in the given population. The larger the value of the fitness function, the larger the areas of the sectors will occupy chromosomes and, accordingly, they are the ones with a greater fate the probabilities will be carried over to the next generation and will be selected.

The crossover is something much similar to the act of sexual reproduction of the living thing. In attempt of pushing for a better solution, the selected chromosomes are then combined and creating a new chromosome or a child. Meanwhile, the new chromosome would have both the characteristics of the parents. The idea of

a crossover is both to collect and merge both qualities of the parents and increase the chances of producing a better offspring.

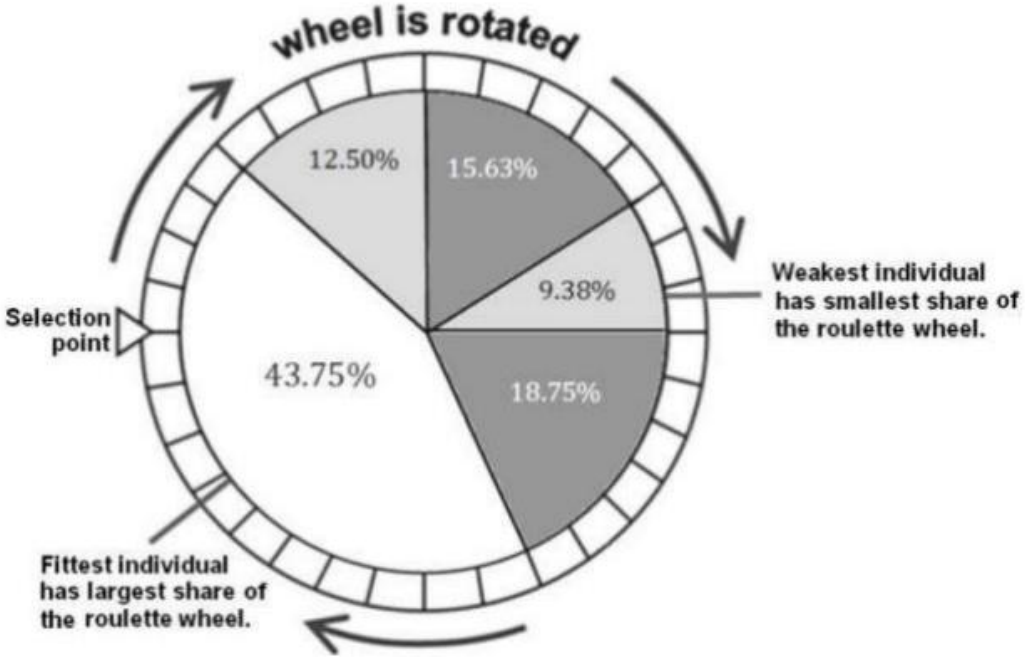



***Fig. 2.*** Process of selection

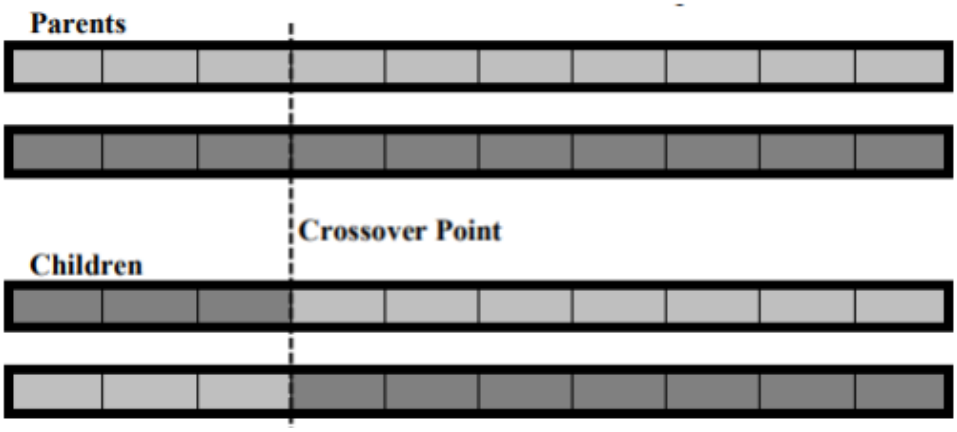



***Fig. 3***. Process of crossover

Mutation is one of the genetic operations, which consists in a change one or more values of a gene in a chromosome in relation to it previous state (Fig. 4).

After preliminary tests conducted in the beginning of this study, the best probability rate was found to be at 0.1 (10%).

After the process of crossover and mutation, the new chromosomes (offspring) will be determined for its fitness value. This population of offspring will then be sorted based on the fitness value. A new population is then created; which involves 30 child

chromosomes, 30 parent chromosomes and 40 randomly generated chromosomes as shown in Figure 5. This new population will undergo the same processes as mentioned earlier until the predetermined stopping criterion is exhausted.

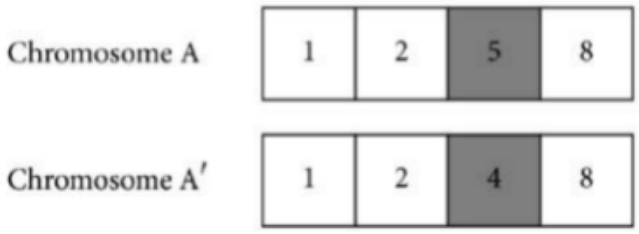



***Fig. 4.*** Process of mutation

**New population**

| | |
|---|---|
| C1 | Child chromosomes |
| … | |
| C30 | |
| C31 | Parent chromosomes |
| … | |
| C60 | |
| C61 | Randomly Generated chromosomes |
| … | |
| C100 | |

***Fig. 5.*** Process of mutation

The figure 6 shows the user interface of the developed scheduler program. Through the interface, you can edit and set graphs of tasks, computer systems, and also generate data for testing.

As can be seen from fig. 7 significant acceleration is developed the scheduling method (orange color) does not give with a small amount vertices in the problem graph. When increasing them to 12, efficiency can be noted and the advantage of the method for the average connectivity of the problem graph. The ordinate axis is the value of the acceleration coefficient, and the abscissa axis is the value of the connectivity coefficient. If you look at the graphs below, you can see that with a small number of vertices, all methods of scheduling give practical results the same result within the margin of error in terms of acceleration factor.

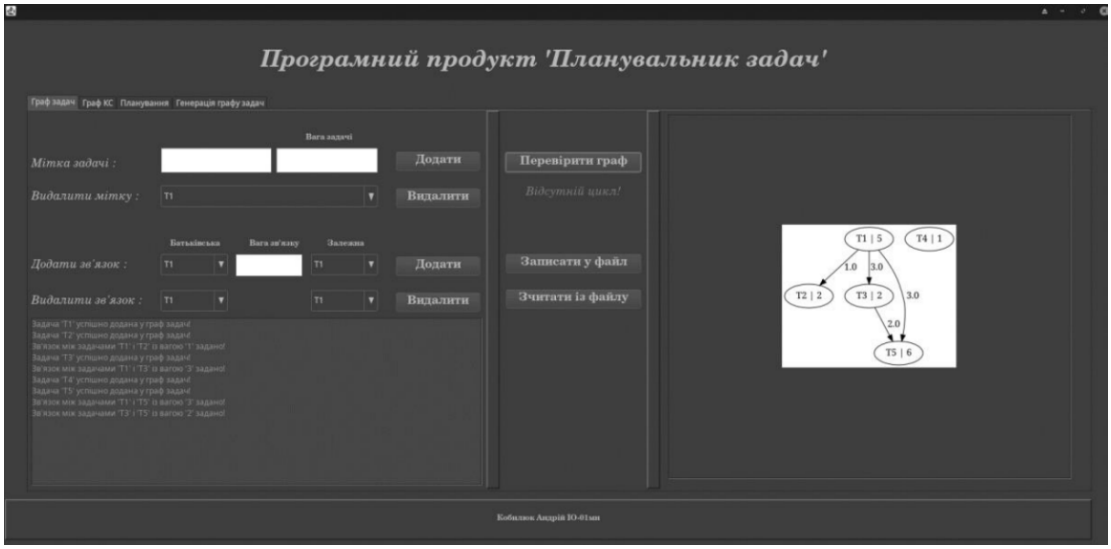



***Fig. 6.*** Developed GUI

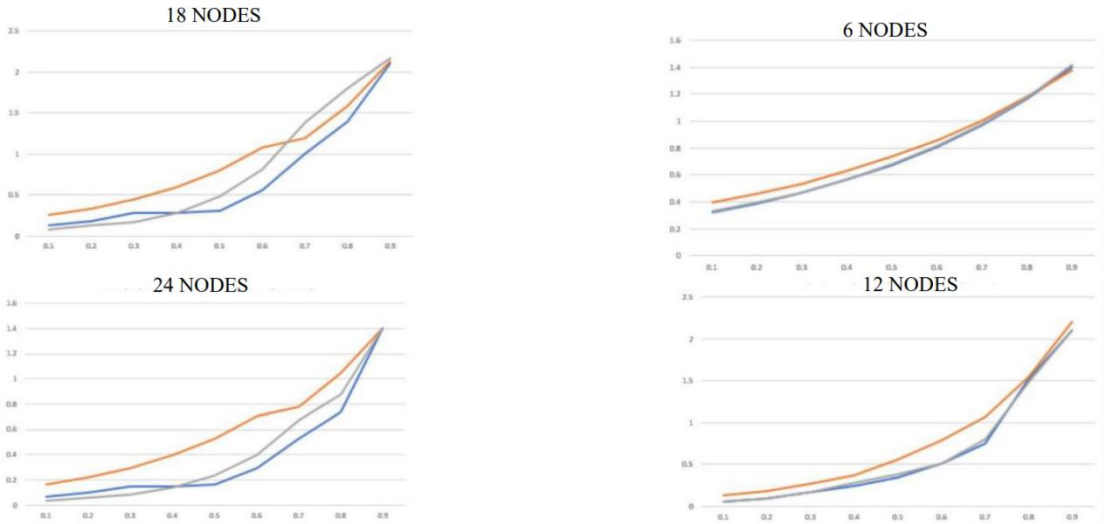



***Fig. 7.*** Testing of method

When the number of vertices increases, the genetic one clearly stands out planning algorithm. Taking into account the connectivity of problem graphs, it is possible note that the developed method works better with medium connectivity and degrades in efficiency to conventional methods, respectively, at low and high connectivity.

**Conclusion.** Theoretically, the genetic algorithm has the ability to significantly improve the overall performance of processors by fully utilizing their power. In addition, GA offers a solution for parallel execution of tasks by processors. Scheduling on a multi-core processor has several problems such as task allocation, CPU idle time, and task sequencing, but all these problems can be solved by GA. The results obtained

as a result of this study clearly support the theories regarding the ability of the used method to solve the planning problem. The developed program was able to reproduce the characteristics and qualities of GA. Furthermore, the produced GA is capable of simulating and completing the specified PPJ graph. As a conclusion, the general experiment was successful. The obtained results are reliable and significant.

## References

1. Davis L. Job Shop Scheduling with Genetic Algorithms / Lawrence Davis. // Proceedings of the 1st International Conference on Genetic Algorithms. – 1985. – С. 136–140

2. Hasband P. The Natural Way to Evolve Hardware / P. Hasband, I. Harvey, T. Adrian. – 1996.

3. Debanjan K., Siddhartha B. A Multi-Objective Quantum-Inspired Genetic Algorithm (Mo-QIGA) for Real-Time Tasks Scheduling in Multiprocessor Environment. Procedia Computer Science. 2018. Vol. 131. P. 591–599

4. Artificial Intelligence-based Scheduling. // The International Conference on Security, Fault Tolerance, Intelligence. – 2020. – С. 5.

5. Студентська робота [Електронний ресурс]. – 2022. – Режим доступу до ресурсу: https://ela.kpi.ua/handle/123456789/34324

**Authorus**

**Kobyliuk Andrii** – student, Department of Computer Engineering, National Technical University of Ukraine “Igor Sikorsky Kyiv Polytechnic Institute”.

E-mail: andrii.kobyliuk@gmail.com

**Кобилюк Андрій Григорович** – студент, кафедра обчислювальної техніки, Національний технічний університет України «Київський політехнічний інститут імені Ігоря Сікорського».

**Artem Volokyta** – associate professor, PHd, Department of Computer Engineering, National Technical University of Ukraine “Igor Sikorsky Kyiv Polytechnic Institute”.

**Волокита Артем Миколайович** – доцент, кандидат технічних наук, кафедра обчислювальної техніки, Національний технічний університет України «Київський політехнічний інститут імені Ігоря Сікорського».

УДК 681.5.004.4

**Bohdan Smishchenko, Artem Volokyta**

# CREATING METHOD FOR ROAD IMAGE SEGMENTATION

This article examines the method used to generate street markings from existing photographs. The use of neural networks for image generation is demonstrated. The method was implemented using tensorflow based on data from the cityscape dataset.

**Keyword**: cGAN, convolutional neural networks, image generation, data generation.

Fig.: 9, Bibl.: 5

**Actuality:** This article demonstrates approach of using conditional GAN for segmented image generation. Shows usage of neural network for image generation and teaching neural network for making markup out of image. To implement neural network will use tensorflow as frameworks and cityscape dataset as data source.

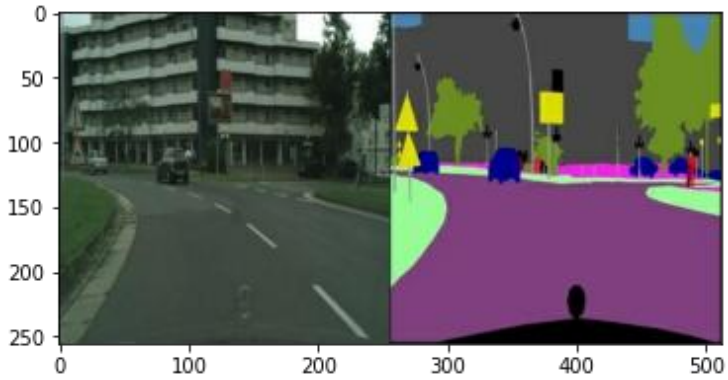



***Fig 1.*** Data example from dataset.

As you can see on Fig 1 we got two images: plain image and segmented one. Idea behind this app is to make ai create layout and learn it to find out patterns inside usual images of streets, nevertheless results are not industry breakthrough and similar task was solved tens of times other ways this approach can be interesting as on of possible applications of this architecture.

**Actual scientific research and issues analysis.** Methods applied to solve problem of image segmentation usually include usage of convolutional neural networks and deep learning. Multiclass segmentation is one of possible solutions of this task.

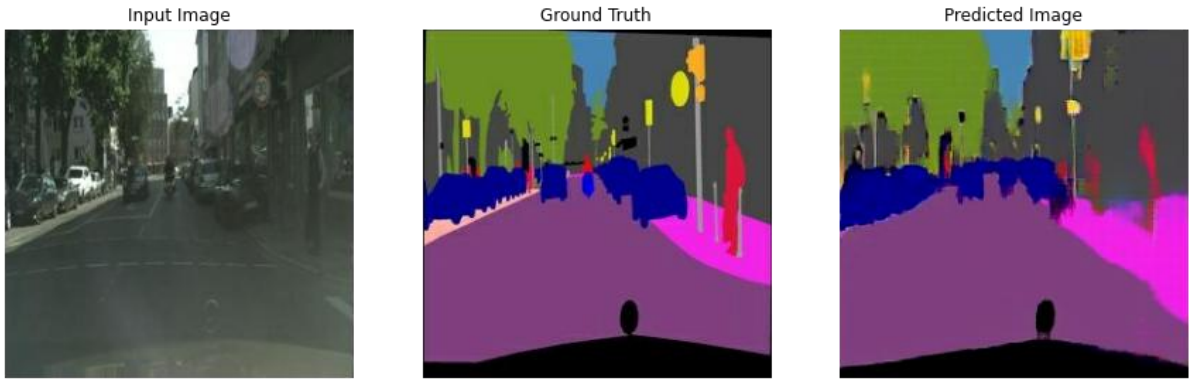



***Fig 2.*** Image segmentation process training result.

Each neural network architecture differently approaches to solution of this problem, GAN trains two networks one for generating image, second for discriminating generated image and real one.

**Uninvestigated parts of general matters defining.** Despite having multiple solutions that make segmentation images out of street views usage of GANs might offer nice and effective implementations with high accuracy.

**The research objective.** Purpose of this article is to create method for image segmentation and make research about ways to improve quality of generated images.

**The statement of base material.** As input image we got two images 256*256*3 size and result of neural networks are, for generator 256*256*3, for discriminator 30*30*1, where 30 is number of classes inside cityscape dataset.

| | | | |
|---|---|---|---|
| conv2d_transpose_8 | input: | (None, 128, 128, 128) | (None, 256, 256, 3) |
| Conv2DTranspose | output: | | |

***Fig 3.*** Generator output.

Let`s see output of discriminator.

| | | | |
|---|---|---|---|
| conv2d_13 | input: | (None, 33, 33, 512) | (None, 30, 30, 1) |
| Conv2D | output: | | |

***Fig 4.*** Discriminator output.

After discriminator compares generated image and real one, weights of both are updated and next step of training begins.

After 40000 training steps we got decent results on first taken image, but still requires some clean up, making edges sharper.

Some graphs that show speed of learning neural network, we have two values: discriminator loss and generator loss.


***Fig 5.*** Results of 40000 steps of training on fist image.

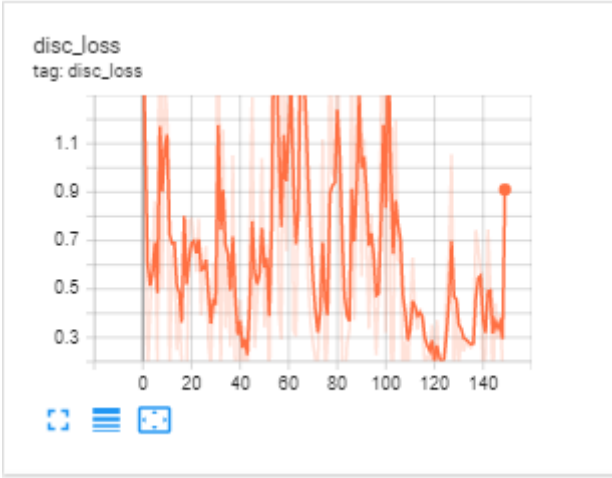



***Fig 6.*** Graph of discriminator loss value changed

Similar graph for generator.

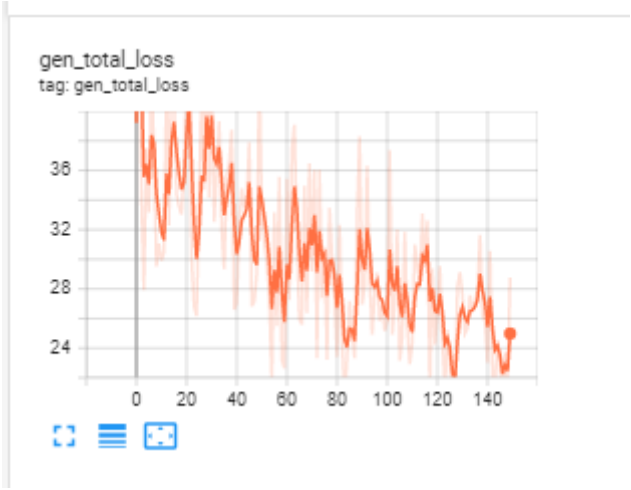



***Fig 7.*** Graph of generator loss value

For generator it`s clearly visible that loss reduces that means neural network has less mistakes over training.

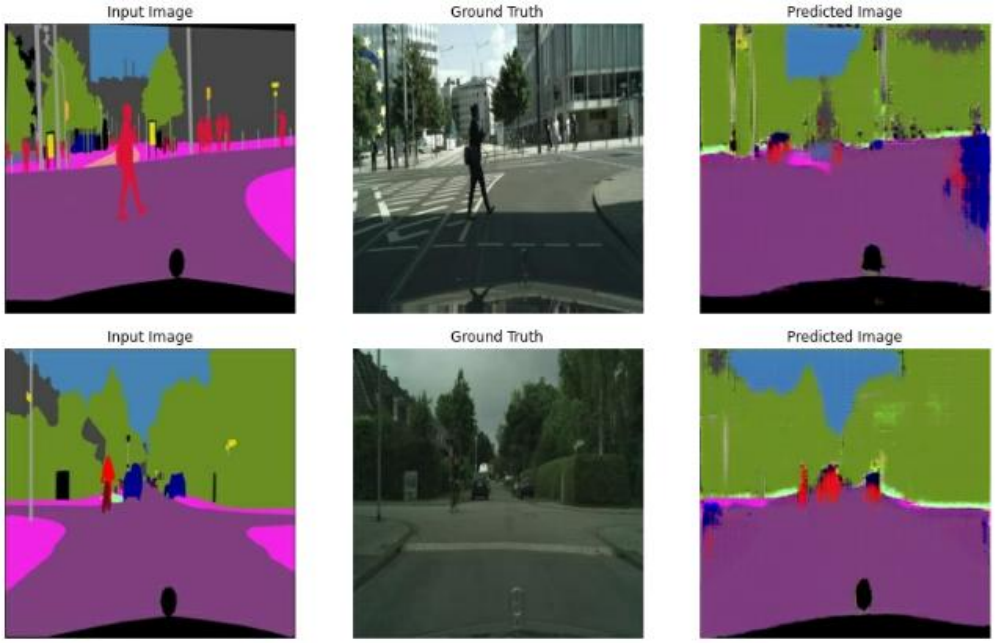



***Fig 8.*** Generated examples on random input from datasource.

**Possible ways to improve model.** One of the ways to improve quality of generated images is to change architecture of generator or making more steps for training, change amount of classes taking into account, for example reducing number of small details.

**Experiments.** One possible experiment is to change amount of training steps up to 60 thousand. Result of this is sharper edges of items therefore better quality of generated images.

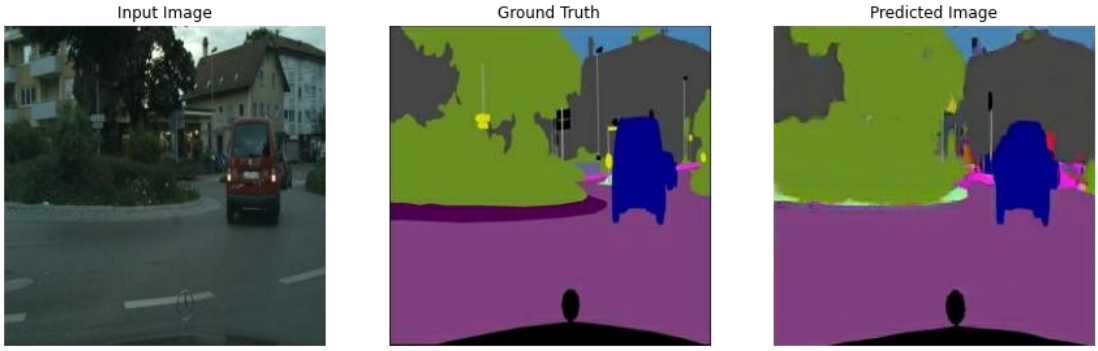



Fig 9. Result of 60000 steps.

Despite huge quality improve this way is limited up to a point where next training steps won`t give any significant quality improvement but required time for training will skyrocket.

**Conclusion.** In this article was described idea of using this architecture to solve usual problems. In theory described possible ways to improve generated image quality. Provided example of one of such approaches and described it`s drawbacks.

In future works some advance in architecture are main possible approach to make results better and maybe reduce computation time.

## References

1. pix2pix: Image-to-image translation with a conditional GAN: https://www.tensorflow.org/tutorials/generative/pix2pix#build_the_discriminator

2. How to Develop a Conditional GAN (cGAN) From Scratch: https://www.tensorflow.org/tutorials/generative/pix2pix#build_the_discriminator

3. How to Develop VGG, Inception and ResNet Modules from Scratch in Keras: https://machinelearningmastery.com/how-to-implement-major-architecture-innovations-for-convolutional-neural-networks/

4. Aurelien Geron Hands-on Machine Learning with Scikit-Learn, Keras, and TensorFlow: Concepts, Tools, and Techniques to Build Intelligent Systems, 2019 600c.

5. ADVANCING PRODUCT DESIGN WORKFLOWS IN MANUFACTURING: https://www.nvidia.com/content/dam/en-zz/es_em/Solutions/design-visualization/industries/manufacturing/quadro-manufacturing-industry-brochure-us-nvidia-681594-FNL-web.pdf

## AUTHORS

**Volokyta Artem** – associate professor, Department of Computer Engineering, National Technical University of Ukraine “Igor Sikorsky Kyiv Polytechnic Institute”.

E-mail: artem.volokita@kpi.ua

**Bohdan Smishchenko** – student, Department of Computer Engineering, National Technical University of Ukraine “Igor Sikorsky Kyiv Polytechnic Institute”.

E-mail: bohdan.smishenko@gmail.com

## **Parallel Section RT.** IoT, Real Time Systems.

**УДК: 004.896**

**Andrii Shapran, Oleksandr Dolholenko**

# DIVISION USING THE BASE RADIX16 NUMBER SYSTEM TO FORM FRACTION DIGITS

The operation of dividing numbers in floating-point form is the most complex operation performed in a microprocessor core. To speed it up, the Intel company, starting with the Sandy Bridge architecture, uses a division algorithm using to represent the fraction of the number system with a base of r=16 (Radix 16).

The article analyzes the requirements of the IEEE Std 754™-2008 standard for floating-point arithmetic. A basic structural scheme for the implementation of floating-point division operations has been developed, that has similar features in many specific implementations of microprocessor cores. To reduce the calculation time of the floating-point division operation, the implementation of the mantissa divider block using the Radix base 16 calculation system to form the quotient digits has been considered. Separate blocks of the divider are designed to the level of the functional scheme.

**Key words:** mantissa, order, division algorithm, normalization, number system with base Radix 16.

**Introduction.** The floating-point representation of numbers is similar to the commonly used form of number representation in scientific computing and consists of two parts: the significant part of the number (or mantissa) $f$ and the exponent (exponent, or order) $e$. A floating-point number $x$ is represented as $\pm,e_x$, $f_x$ and has a value*:* $x = \pm f_x r^{ex}$, where $r$ – base of the number system (base of the exponent, or base of the degree).

Some reductions and abbreviations (acronyms) adopted in the standard *IEEE 754* [1]:

*LSB* – least significant bit;

*MSB* – the most significant bit;

*NaN* **–** not a number.

The absolute value of the mantissa of a normalized number is in the range [1, 2]. During normalization, it shifts so that in $|f|$ $MSB = 1$. An operation is performed on each left shift $e = e - 1$, with each right shift $e = e + 1$. Thus, the two-digit dot in the representation of f is always located after the bit *MSB.*

The MSB bit of the mantissa of a normalized number, which is always equal to 1, is removed from the representation of the number and only a small part of the mantissa is stored in memory. In an arithmetic device, this hidden bit is restored and thereby contributes to increasing the accuracy of the representation of operands without taking up memory space.

A floating-point number has two signs: the sign of the number (sign) is displayed by a separate bit; the sign of the order is displayed by the order bias (bias).

The standard requires using several data types (formats) for floating-point calculations [1].

A *denormalized number* is a floating-point number, exponent of wich is zero (0 is $e_{min}$) and non-zero mantissa: $\boldsymbol{e + bias} = e_{min}$, $f \neq 0$, $sign = 0\ V\ 1$. When performing an arithmetic operation with a denormalized operand, the hidden bit is not restored in it. The use of denormalized operands makes the effect of loss of significance less drastic. Without using of such operands, some small values that cannot be represented as normalized numbers would have to be rounded to 0. Such solution would degrade the accuracy of floating-point calculations in some cases. For example, a number $(0.1)_2 \times 2^{-126}$ does not have a normalized representation in the IEEE single format. If we use 0 instead of this number in further calculations, then the "gradual loss of accuracy of the result" will occure. At this time, the implementation of the ability to process denormalized numbers in a computing device can cause unwanted hardware and time costs. In this regard, the standard does not require mandatory implementation of denormalized number processing.

**The purpose of the article.** The purpose of this article is to develop a device for performing the operation of dividing floating-point numbers using the Radix base 16 numbering system for the intermediate representation of the quotient digits.

**Presenting main material.** As a result of the research, a structural scheme for the implementation of division operations has been developed, which has similar features for many specific implementations of microprocessor cores. Based on this, the developed scheme can be considered as basic, commonly used, in its general features, when developing specific options for implementation of floating-point division operations. This divider scheme is shown in the fig. 1.

The divider usually consists of: an operand unpacking block, a mantissa divider, a number of additional circuits used for processing: orders, exceptions ($\pm 0$, $\pm\infty$, *NaN*), normalization and rounding of the result, as well as its packaging.

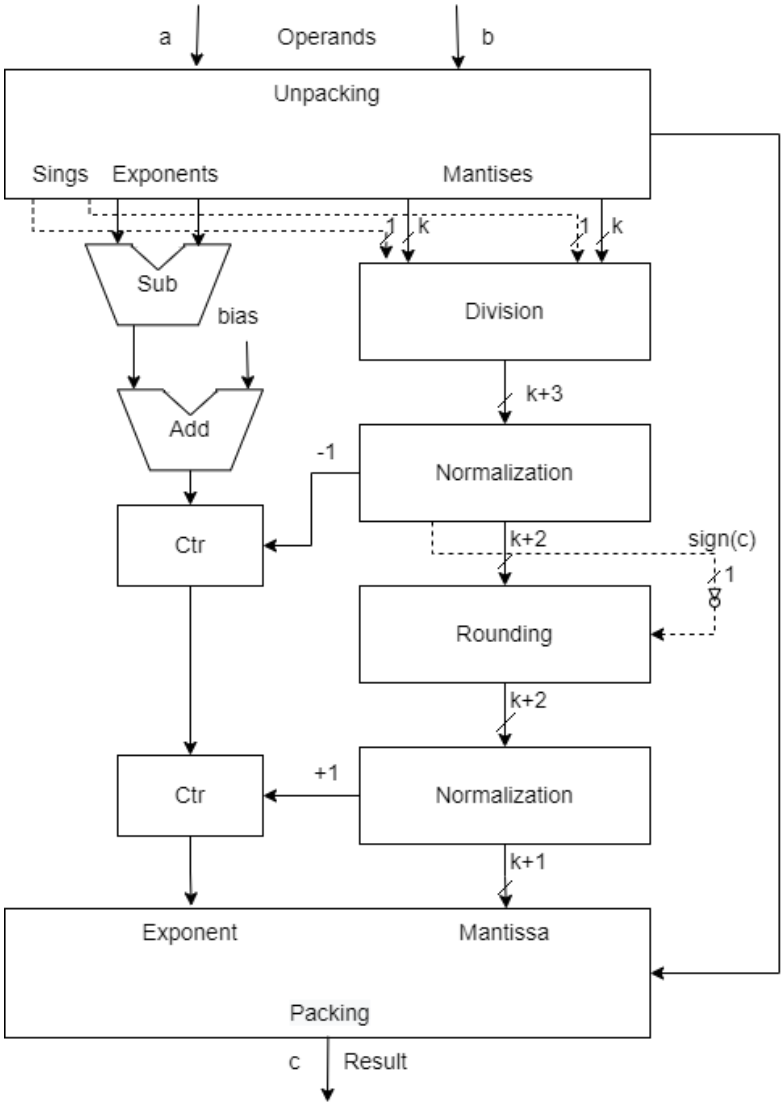



***Fig. 1***. Floating point divider

Performing a division operation on floating-point operands $c = a/b$ is reduced to the following operations:

$$(\pm f_a \text{ x } 2^{(ea+bias)})/(\pm f_b \text{ x } 2^{(eb+bias)}) = \pm f_a / f_b \text{ x } 2^{(ea - eb+bias)} = f_c \text{ x } 2^{(ec+bias)}.$$

When constructing a floating-point divisor, care must be taken to ensure correctness and avoid unjustified loss of accuracy. In addition, the possibility of handling any exceptions should be implemented.

Unpacking includes the selection of: the sign of the mantissa, as well as the recovery of the hidden MSB bit for each of the operands. During unpacking, the format of the operands is also converted to the internal format of the arithmetic device (for example, to the format of quadruple precision). The unpacked operands are tested for the presence of exceptions among them: 0, *NaN*, ±∞. If there is an exception, the

result of the operation is formed in accordance with the relations given in [1] and sent to the package, bypassing the division of the mantissa.

The preliminary order of the quotient is calculated by subtracting the two shifted orders of the operands and adding shift sum to the resulting:

$$(e_a+bias) - (e_b+bias)+bias= (e_a - e_b)+bias= e_c+bias.$$

Division of mantissas with signs represented by an additional code is carried out with the help of a matrix divider, or with the help of a sequential divider with a high base, for example, *Radix*-16 [2]. After dividing two normalized mantissas, each of which is in the range [1, 2], the mantissa of the quotient can be in the range (1/2, 2). In this regard, for its normalization it may be necessary to shift by one digit to the left with a decrease of 1 in the preliminary value of the order of the quotient. After the first normalization, the $k$+3 digit mantissa of the quotient is truncated to $k$+2 digits.

When rounding the mantissa of the quotient $f_c$ loss of its normalization may occur again. At the same time $| f_c |$ may be equal to 2, and for its normalization, a shift of one digit to the right with an increase of 1 value may again be required $e_c+bias$.

To increase the speed, it is possible to pre-calculate the value of ec+bias increased by 1 at each normalization step and choose the correct value after it becomes clear whether a shift is required after rounding. Since mantissa division is the most complicated part of floating-point division, there is enough time for such calculations. In addition, rounding should not be a separate step at the end of the operation. It can be combined with mantissa division equipment.

The speed of the divisor based on the algorithm of recurrent calculation of numbers depends mainly on the delay of the function that generates the digit of the quotient.

The basis of the development of the mantissa division block is the SRT algorithm of mantissa division in the number system with the base $r$ = 16, which is described in [4]. The article presents an analytical approach that extends the well-known theory [5-8] for performing standard SRT division and allows to implement the function of predicting the number of the part more easily.

In fig. 2 the block diagram of the mantis division unit performing the operation $q = x/d$ is shown and can process all floating-point number formats provided by the standard [1]. There are five such formats: half precision (SF) - 16 bits, single precision (F) - 32 bits, double precision (DF) - 64 bits, double extended precision (DEF) - 80 bits and quadruple precision (QF) (128 discharges). To achieve this goal, the mantissa

division block must be able to handle, according to the format: 12, 25, 54, 66 and 114 bit binary mantissas d and x (together with the sign and hidden bits) in positive code.

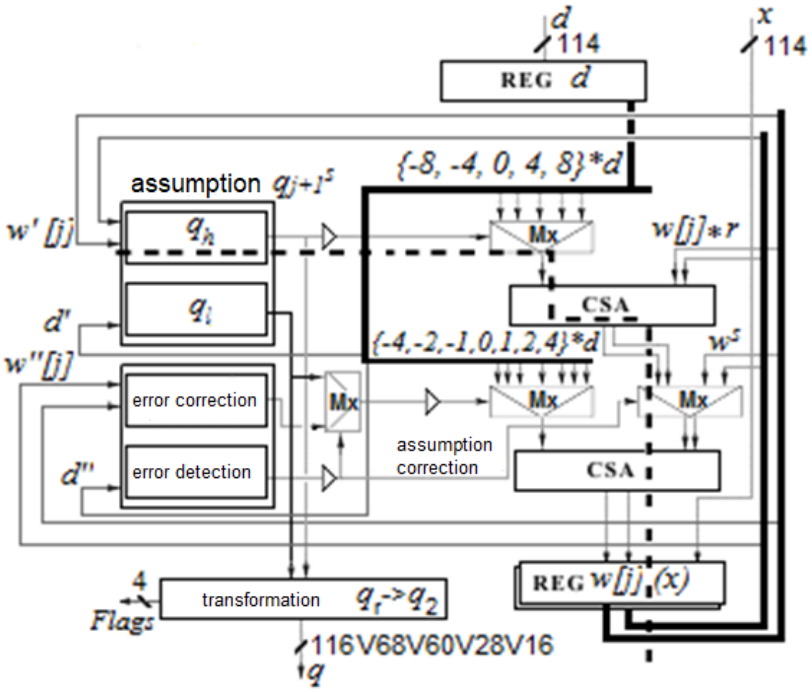



***Fig. 2.*** Structural diagram of the mantissa division block in the counting system with r=16 and memorization of transfers during additions.

At each cycle of this scheme, the prediction of the next digit of the share is carried out $q_{j+1}$. The mantissa of the fraction q is calculated in the redundant number system with the base $r$ = 16 and is immediately converted into a binary positive code, as described in [8]. Before conversion, each of the provided digits of the share $q_{j+1}$ is a signed number $|q_{j+1}| \le a$, for which the redundancy factor is fulfilled $p = a/(r-1) = 12/15$, or in other words: each digit of the fraction in the device is assumed from the range $q_{j+1} \in \{-12,-10,-9,-8,-7,-6,-5,-4,-3,-2,-1,0,1,2,3,4,5,6,7,8,9,10,12\}$ and consists of two components, according to [4]. Each digit of the fraction from the redundant sixteen-year numbering system, in the process of calculation, is converted into four digits of the binary system. As a result, the necessary number of calculation cycles for predicting the digits of the fraction is reduced by four times, compared to the calculation of the digits of the fraction directly in the binary number system. The usage of a redundant digital set for fractional prediction significantly increases the speed of each individual calculation cycle by using carry-preserving adders.

The duration of one calculation cycle in the operation of the mantissa division block can be calculated by the signal propagation delay in the critical (longest) chain of the circuit from Fig. 2, where it is shown by a dashed line.

In order to reduce the number of CSAs (carry-saved single-bit adder lines), forming *d* multiples of $q_{js}$ *€{-11, 11},* are not used in the device under development, as it would require three CSAs. Instead of it, the correction function is used. As a result, the required number of calculation cycles for predicting the quotient digits can be reduced by less than four times. In order to increase the speed of the scheme, the prediction of the fraction number at the jth step of the calculation is carried out simultaneously with the detection of an error at the (j -1)th step.

Each digit $q_{j+1s}$ of parts is calculated as the result of of two component parts sum calculated on two different combination schemes:

$$q_{j+1s} = q_h + q_l, \text{ where: } q_h \text{ €}\{\pm 8, \pm 4, 0\}, q_l \text{ €}\{\pm 4, \pm 2, \pm 1, 0\}.$$

In fig. 3 the logic of the assumption from fig. 4 in detail is shown. In this figure: CPA - Carry Propagate Adder (adder with propagated transfers); CS - combination formation schemes $q_h$ and $q_l$, in accordance.

Older grades of the partial remainder *w'[j]*, calculated in the form with stored hyphens, are pre-processed using CPA. The outputs of the CPA are connected to the inputs of two different combinational circuits as shown in Fig. 3. The second inputs of the combinational circuits are connected to the two higher digits of the divider *d*. The first combinational circuit calculates $q_h$ and requires only the five most significant bits of the value. The second calculates $q_l$ and needs the entire evaluation of the partial balance.

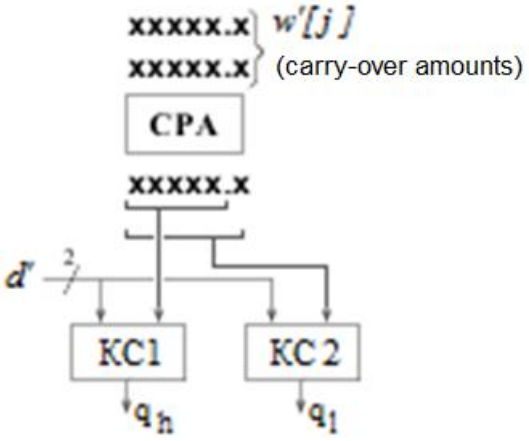



***Fig. 3.*** Details of the implementation of the assumption logic.

The scheme for converting the mantissa of the fraction *q* from the redundant number system into a binary positive code, according to [8], is built as an accumulating adder/subtractor.

**Conclusions.** In this work, a reconfigurable floating-point divider has been developed, that can dynamically reconfigure to divide operands of all five operand formats required by the standard for floating-point arithmetic *IEEE Std 754™-2008.*

The calculation of the mantissa of the quotient is carried out using the redundant number system with a base to predict the digits of the quotient *r =16* and numbers *{-12,-10,-9,-8,-7,-6,-5,-4,-3,-2,-1,0,1,2,3,4,5,6,7,8,9,10,12}.* In the process of calculating the numbers, the fraction is converted into a normal binary positive code.

Using such a redundant numbering system for temporarily representation of the quotient's mantissa digits reduces the required number of iterative calculation steps by 4 times. In fact, it is achieved by four binary digits of the mantissa calculation at each iteration, which can be adjusted at the next iteration.

The performed development will be useful for designing fixed and floating point division operation device, for a microprocessor core with a superscalar architecture compatible with the ***x86-64*** family.

***References:***

1. IEEE 754: Standard for Binary Floating-Point Arithmetic [Электронный ресурс] / 3 апрель 2014. – URL: http://grouper.ieee.org/groups/754/.

2. Behrooz Parhami. Computer Arithmetic. Algorithms and Hardware Designs, New York, Oxford University Press, 2000 – 491 p.

3. Pippenger, N., "The Complexity of Computations by Networks," IBM J. Research and Development, Vol. 31, No. 2, pp. 235-243, March 1987.

4. C VLSI, 1999. Proceedings. Ninth Great Lakes Symposium on Year: 1999, Pages: 74 - 77, DOI: 10.1109/GLSV.1999.757380

5. L. Benini, E. Macii, and M. Poncino. Telescopic Units: Increasing the Average Throughput of Pipelined Designs by Adaptative Latency Control. In 34th Design Automation Conference, 1997.

6. J. Cortadella and T. Lang. Division with Speculation of Quotient Digits. In 11th Symposium on Computer Arithmetic, pages 87–94, 1993.

7. J. Cortadella and T. Lang. High-Radix Division and Square Root with Speculation. IEEE Transaction on Computers,C-43(8):919–931, August 1994.

8. M.D. Ercegovac and T. Lang. Division and Square Root. Digit-Recurrence Algorithms and Implementations. Kluwer Academic Publishers, Norwell, MA, 1994.

## AUTHORS

**Oleksandr Dolholenko** - associate professor, candidate of technical sciences, Senior Research Fellow, Department of Computer Engineering, National Technical University of Ukraine "Ihor Sikorskyi Kyiv Polytechnic Institute".

**Andrii Shapran** – student, Department of Computing, National Technical University of Ukraine "Ihor Sikorskyi Kyiv Polytechnic Institute".

E-mail: andriyito.ti99@gmail.com

**Anatolii Haidai, Iryna Klymenko**

# THE METHOD OF THE FUNCTIONAL PARAMETERS ESTIMATION OF THE SLEEP MONITORING SYSTEM BASED ON NEURAL NETWORK

The article considers the choice of environment and user parameters for the neural network that will monitor the status, provide advice or directly influence the microclimate, with prototypes of the data acquisition device and the neural network for processing the data.

**Key words:** neural networks, environment data collection, sleep monitoring system.

Fig.: 3. Bibl.: 8.

**Target setting.** The choice of data for the sleep monitoring system is very important, as it directly affects its effectiveness during implementation, and for a system that aims to control the microclimate, the selected parameters must be controlled.

**Actual scientific researches and issues analysis.** The main area of research and publications that have been developed is the diagnosis of sleep apnea, narcolepsy [1], [2], [3], [4]. To collect data in these publications, use special devices that record the activity of the brain, lungs. These can be as sensors attached to the chest, special bracelets and bandages that are attached to certain parts of the body.

**Uninvestigated parts of general matters defining.** The reviewed scientific works are aimed at the study of a particular disease and represent a method of its diagnosis, which is based on the use of narrowly focused equipment, which in turn limits its use because the devices are expensive, which will not provide them to many people. This can be achieved by changing the data collected, using new devices, the cost of which will provide a large part of the population and using a neural network that will process this data.

**The research objective.** Identify parameters that have a direct impact on sleep quality, select components that will receive this data, their transmission to the neural network and its further processing.

**The statement of basic materials.** The main parameters that a person can control to improve sleep are ambient temperature, humidity, air quality, light level. Each of these parameters affects the quality of our sleep.

Temperature [5] is one of the key parameters of the environment that is important and affects the quality of sleep. The DS18B20 sensor was selected to obtain temperature readings.

This sensor is digital, has an error of 0.5 degrees Celsius, in the environment that will be used in this system.

To control the temperature, you can use air conditioners, which will need to be connected to the neural network, so that it can change the parameter according to the results of training according to the optimal for a particular user.

Humidity [6] is also one of the important parameters of the environment, as it affects the nasopharynx, eyes, skin. Low humidity can lead to damage that will lead to eye and breathing problems, which in turn can affect sleep quality. You can measure the humidity level with a sensor.

Although this sensor has an error of 3% at a temperature of 25 degrees, but due to its compactness and the ability to measure values in the range from 0 to 99.9% humidity, can be used in the system. The control of this indicator can be carried out by means of an air humidifier, at its connection to system and adjustment of water supply to the tank.

Air [7] quality affects the condition of the heart and lungs, and research has a direct effect on sleep quality, people with more air pollution sleep worse and more anxiously. The MQ-135 sensor can be used to analyze indoor air.

The control of this indicator clearly depends on the place of residence, and this in turn determines how the level of air pollution can be influenced. So when placing near the green zone of air in which it is much cleaner possible to control the ventilation capacity of the room, at the same time when living in an industrial region, you can use a home air purifier.

The level of illumination [8] of the room affects the productivity of the body and how quickly the human body goes to sleep. Finding the optimal values of sleep that will not interfere, and further control with a gradual decrease in brightness can accelerate the transition of the body to sleep, to assess this parameter, you can use the module of the light sensor analog-digital.

This indicator can be controlled by reducing the brightness of monitors, table lamps and other lighting devices, at a constant speed. For example, a change in brightness of 25 percent per hour will be quite invisible, but will have an impact on processes in the human body.

To obtain these parameters, a prototype device was developed (Fig. 1) according to the scheme (Fig. 2).

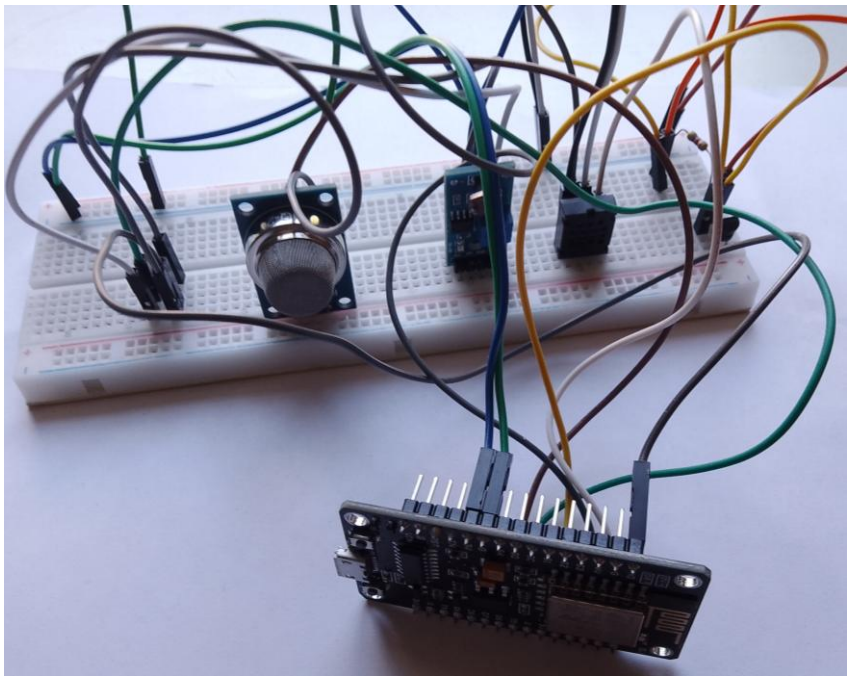


***Fig. 1***. Photo of the prototype

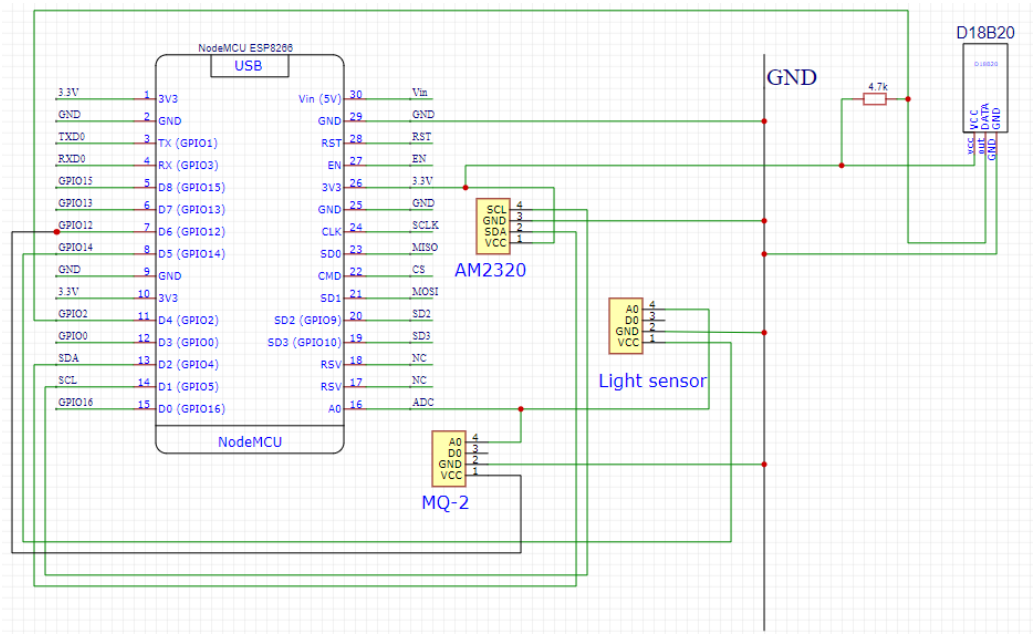



***Fig. 2.*** Scheme of the data collection device

Data on a person's heart rate and motor activity can be obtained from smart bracelets or watches, and based on the obtained values to draw conclusions about whether a person is sleeping or not and to determine the phase of sleep.

The neural network based on the data obtained from these sensors should determine the optimal sleep parameters for the user. This is done by identifying the relationship between the parameters of the environment and how much a person changes motor activity and heart rate. In the phase of deep sleep, the pulse will be reduced, motor activity is minimal, and in the case of fast sleep, these values will be higher.

It is proposed to use a network with such an architecture for data processing (Fig. 3).

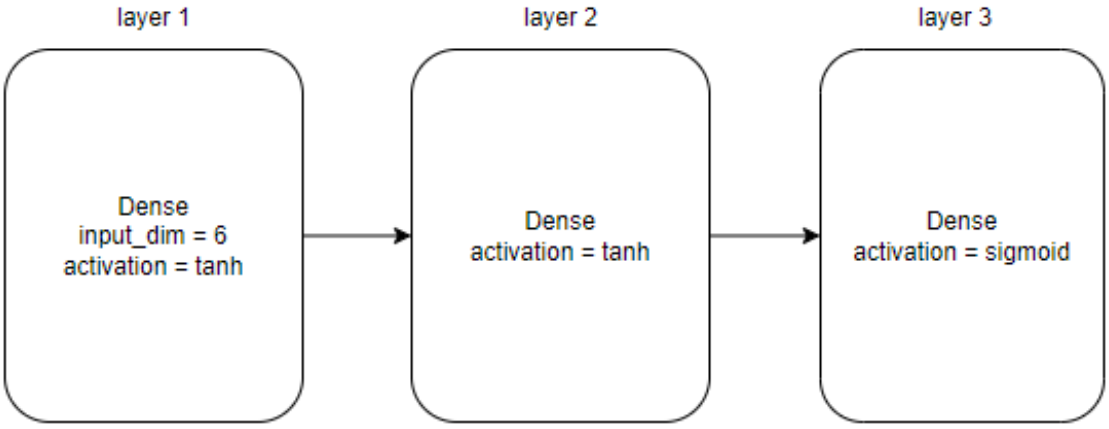



***Fig. 3.*** Neural network model architecture.

You also need to add the loss and optimization function to the network, binary cross entropy and Adam, respectively.

**Conclusions.** The selected set of parameters of the environment and the user is important for determining the quality of sleep and its phase, and adding control parameters through these devices allows you to create the best conditions, increase sleep efficiency, which in turn will affect user performance. The above network architecture can process the data received from the sensors, but requires further study.

## References

1. Zhenghua Chen, Min Wu, Wei Cui, Chengyu Liu (2020) *An Attention Based CNN-LSTM Approach for Sleep-Wake Detection with Heterogeneous Sensors*. IEEE Journal of Biomedical and Health Informatics.

2. Thijs E Nassi, Wolfgang Ganglberger, Haoqi Sun, Abigail A Bucklin, Siddharth Biswal, Michel JAM van Putten, Robert J Thomas, M Brandon Westover (2021) *Automated Scoring of Respiratory Events in Sleep with a Single Effort Belt and Deep Neural Networks*. IEEE Transactions on Biomedical Engineering

3. Jens B. Stephansen, Alexander N. Olesen, Emmanuel Mignot (2018) *Neural network analysis of sleep stages enables efficient diagnosis of narcolepsy*. Nature Communications volume 9, Article number: 5229

*4.* Gary Garcia-Molina, Keith Baehr, Brenda Steele, Tsvetomira Tsoneva, Stefan Pfundtner, Brady Riedner, David P. White, Giulio Tononi (2017) *Automatic characterization of sleep need dissipation using a single hidden layer neural network.* 25th European Signal Processing Conference (EUSIPCO)

5. Harding, E. C., Franks, N. P., & Wisden, W. (2019) *The Temperature Dependence of Sleep*. Frontiers in neuroscience, (v. 13, p. 336.)

6. Md. Dilshad Manzara, Mani Sethia, M.Ejaz Hussain*a. (2011) *Humidity and sleep: A review on thermal aspect. Biological* Rhythm Research

7. American Thoracic Society. (2017, May 22). *Air pollution may disrupt sleep*. ScienceDaily.

8. Czempik, P. F., Jarosińska, A., Machlowska, K., & Pluta, M. (2020). *Impact of Light Intensity on Sleep of Patients in the Intensive Care Unit: A Prospective Observational Study*. Indian journal of critical care medicine: peer-reviewed, official publication of Indian Society of Critical Care Medicine, (v. 24, p. 33–37.)

## AUTHORS

**Haidai Anatolii** – graduate, Department of Computer Engineering, National Technical University of Ukraine “Igor Sikorsky Kyiv Polytechnic Institute”.

**Klymenko Iryna Anatoliivna** – associate professor, Department of Computer Engineering, National Technical University of Ukraine “Igor Sikorsky Kyiv Polytechnic Institute”.

**Illia Verbovskyi, Valerii Zhabin**

# IMPROVING THE EFFICIENCY OF FUNCTIONS COMPUTATION IN ON-LINE MODE ON FPGA

The paper considers the possibility of reducing resources and increasing the speed of computing functions in on-line mode on the basis of recursive-digital low-pass filter.

**Key words:** redundant system, FPGA, dependent operations overlapping, on-line mode, recursive-digital low-pass filter.

Fig.: 6. Bibl.: 5.

**Urgency of the research.** Most of the algorithms designed to accelerate computations, do not involve optimizing the input and output process. Which requires more resources and time, particularly when it is implemented on FPGA. Therefore, there is a need to analyze and increase the number of operations that use this method. It can improve method characteristics.

**Target setting.** In the operation of computer systems in real time, when the duration of data processing is limited by external factors, we can parallelize calculations using a certain set of individual computing modules. But when it comes to the chain of operations, this method is impossible, because the operations are dependent. In this case, the result of one operation is used as an operand for the next. However, partial overlapping dependent operations is possible by on-line mode. This approach is now being used and is giving excellent results [5]. In addition, there is lack of a large number of algorithms for calculating functions, that implement on-line mode, for usage in FPGA.

**Actual scientific researches and issues analysis.** Over time, the number of algorithms that use on-line mode is increasing, but many of them still do not use the possibility of bitwise input, which does not fully solve the issue of reducing resources. In particular, in [1] the high radix algorithms that shortens the critical path of the multiplier is studied, and in [3] the application of algorithm for logarithm, exponential, and powering computation is investigated. Each of them does not implement full bitwise processing of operands, which requires the consumption of a large number of pins when working with multi-bit operands.

**Uninvestigated parts of general matters defining.** This article covers the analysis and usage of bitwise input when calculating functions in on-line mode. The study focuses on increasing speed and reducing equipment spending.

**The research objective.** Analyze the possibilities of improving the efficiency of calculations based on a recursive-digital low-pass filter in on-line mode.

**The statement of basic materials.** One of the approaches to solving the problem of reducing the number of connections between system components is the use of quasiparallel computing blocks (CB) that exchange data with each other using a serial code. Although the numbers are represented by a serial code at the inputs and outputs of such CB, their internal organization is closer to parallel devices. In this connection, they were called quasiparallel [4]. Based on the method and formulas considered in [4], we can obtain formulas for generalizing addition, subtraction and multiplication:

$$N_i = 2R_{i-1} + F_i, \tag{1}$$

$$R_i = N_i - z_i, \tag{2}$$

$$z_i = \begin{cases} -1, \ if \ \ N_i < -2^{-1}; \\ 0, \ \ if \ \ -2^{-1} \le N_i < 2^{-1}; \\ 1, \ \ if \ \ 2^{-1} \le N_i. \end{cases} \tag{3}$$

where $N_i, R_i$ – internal variables, $z_i$ – result digit, $F_i$ – function increment, calculated from partial operands $X_i$ and $Y_i$, according to the following formulas (addition/subtraction and multiplication, respectively):

$$F_i = 2^{-p}(x_i \pm y_i), \tag{4}$$

$$F_i = 2^{-p}(x_i Y_i + y_i X_{i-1}), \tag{5}$$

where $p$ – on-line delay.

**General model structure.** The structure of the device for calculation is built by special modules using formulas (1-5). For example, to calculate a recursive-digital low-pass filter according to the following formula:

$$y_i = b \cdot (x_i - y_{i-2}) + c \cdot (x_{i-1} - y_{i-1}) + x_{i-2} + x_{i-1}, \tag{6}$$

it is necessary to construct the Synchronous Dataflow Graph (SDFG) [2] then to turn all formulas from SDFG into a tree of operations (Fig. 1).

Then, in the tree, the operation nodes, must be replaced with special computing blocks. These blocks perform one of the considered operations in formulas (4-5). The structure of the modular system that performs the calculations is shown in Fig. 2.

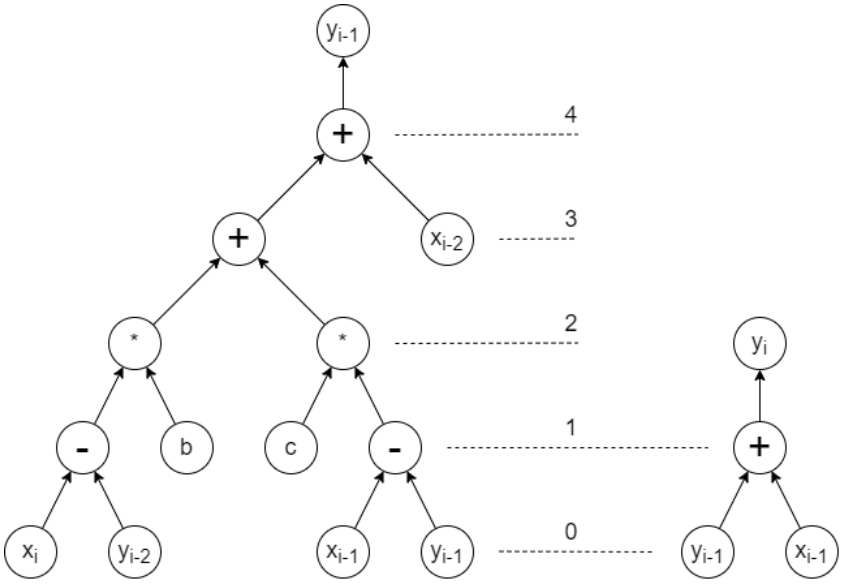



***Fig. 1.*** Tree of operations for SDFG

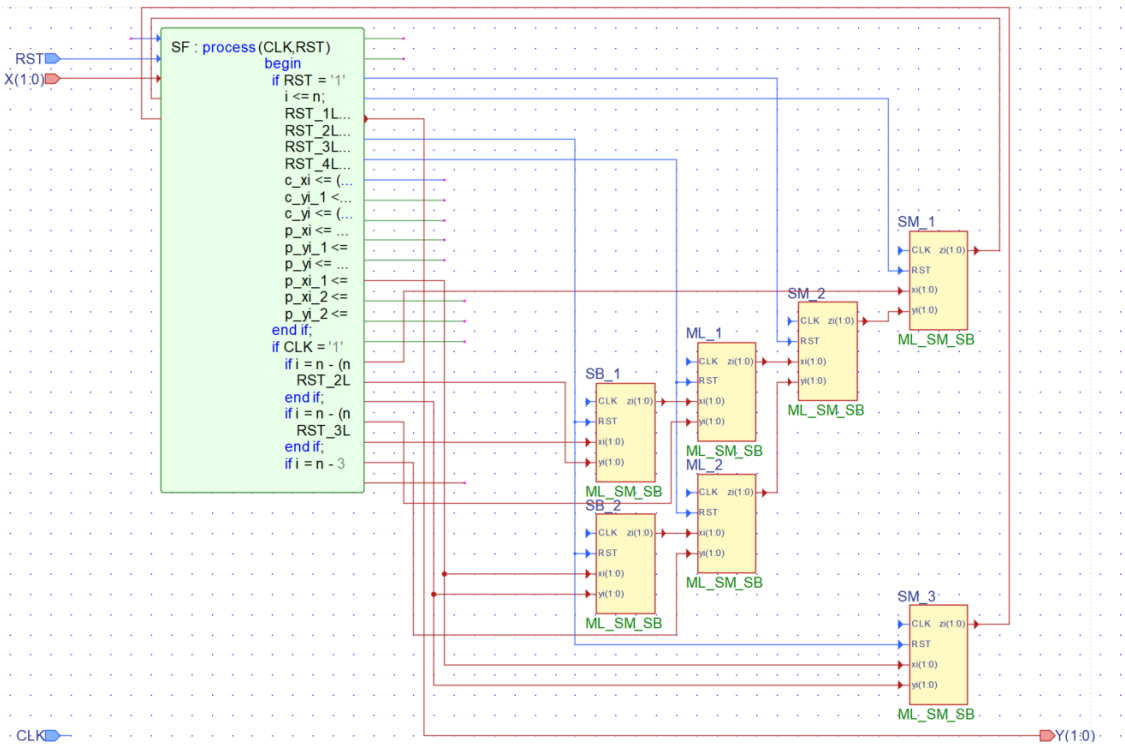



***Fig. 2.*** Structure of the modular system

**Experiments and analysis.** As a result of the filter, the amplitude-frequency characteristic (AFC) is similar to the correct low-pass filter: there are decays at the appropriate frequencies, then phase change and repetition in the opposite direction, while the input and output values of the filter also remain systematic and similar to the sinusoidal and cosine signals, respectively (Fig.3).

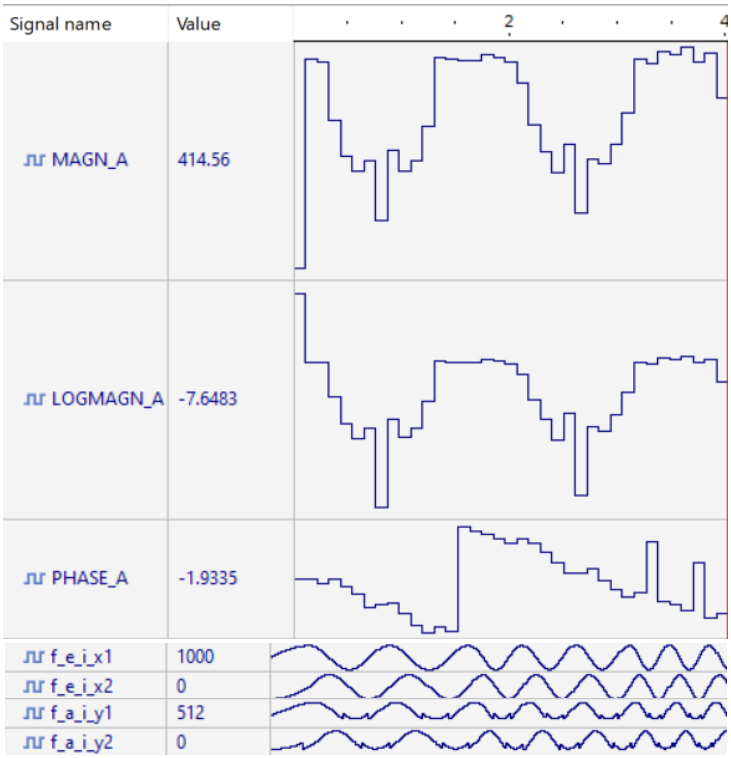
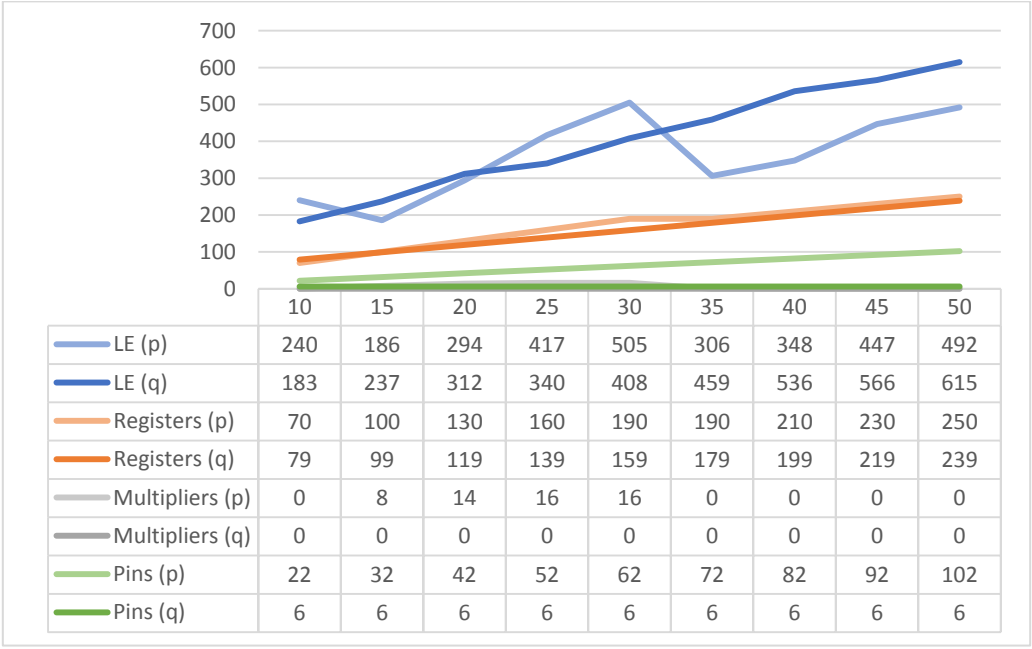


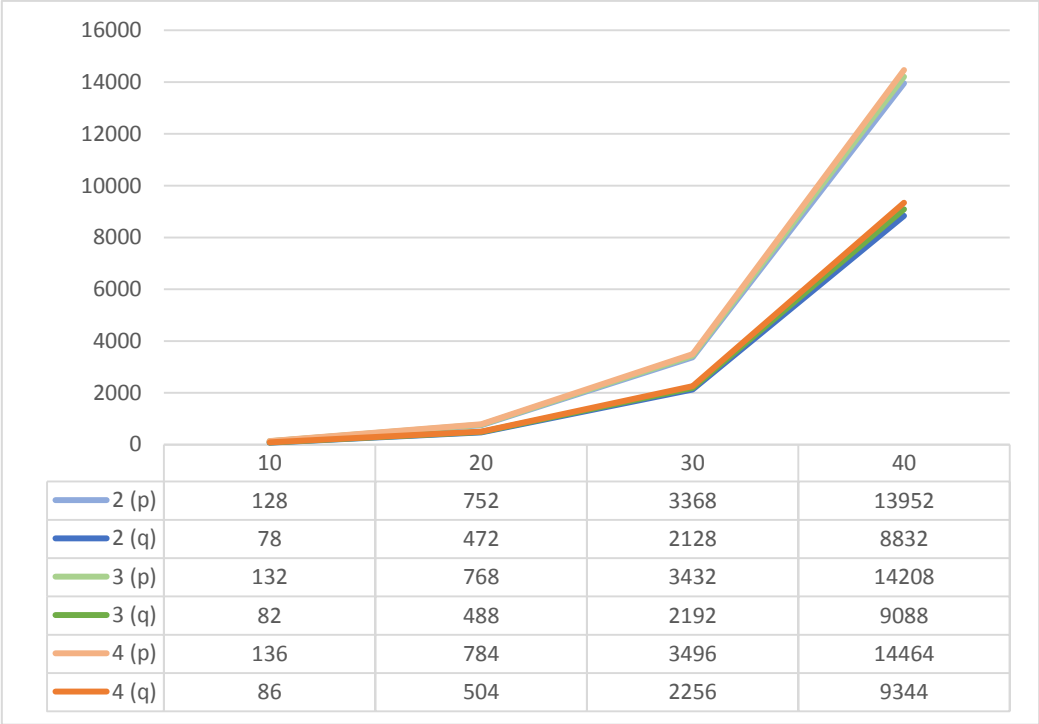

***Fig. 3.*** AFC with I/O values of the developed filter

After testing and synthesizing the system on FPGA, were obtained the results of comparing the parallel and quasiparallel system in terms of resources used (Fig.4) and reducing the number of iterations (Fig.5).




| | 10 | 15 | 20 | 25 | 30 | 35 | 40 | 45 | 50 |
|---|---|---|---|---|---|---|---|---|---|
| LE (p) | 240 | 186 | 294 | 417 | 505 | 306 | 348 | 447 | 492 |
| LE (q) | 183 | 237 | 312 | 340 | 408 | 459 | 536 | 566 | 615 |
| Registers (p) | 70 | 100 | 130 | 160 | 190 | 190 | 210 | 230 | 250 |
| Registers (q) | 79 | 99 | 119 | 139 | 159 | 179 | 199 | 219 | 239 |
| Multipliers (p) | 0 | 8 | 14 | 16 | 16 | 0 | 0 | 0 | 0 |
| Multipliers (q) | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Pins (p) | 22 | 32 | 42 | 52 | 62 | 72 | 82 | 92 | 102 |
| Pins (q) | 6 | 6 | 6 | 6 | 6 | 6 | 6 | 6 | 6 |

***Fig. 4.*** Comparing the parallel (p) and quasiparallel (q) system in terms of resources used




| | 10 | 20 | 30 | 40 |
|---|---|---|---|---|
| 2 (p) | 128 | 752 | 3368 | 13952 |
| 2 (q) | 78 | 472 | 2128 | 8832 |
| 3 (p) | 132 | 768 | 3432 | 14208 |
| 3 (q) | 82 | 488 | 2192 | 9088 |
| 4 (p) | 136 | 784 | 3496 | 14464 |
| 4 (q) | 86 | 504 | 2256 | 9344 |

***Fig. 5.*** Comparing the parallel (p) and quasiparallel (q) system in terms of number of iterations with on-line delay (p = 2, 3, 4)

Thanks to the on-line mode, dependent operations can overlap in time, which allows to reduce the time of calculations. A comparison of the calculation of function (6) in conventional and on-line mode of operation is shown in Fig. 6.

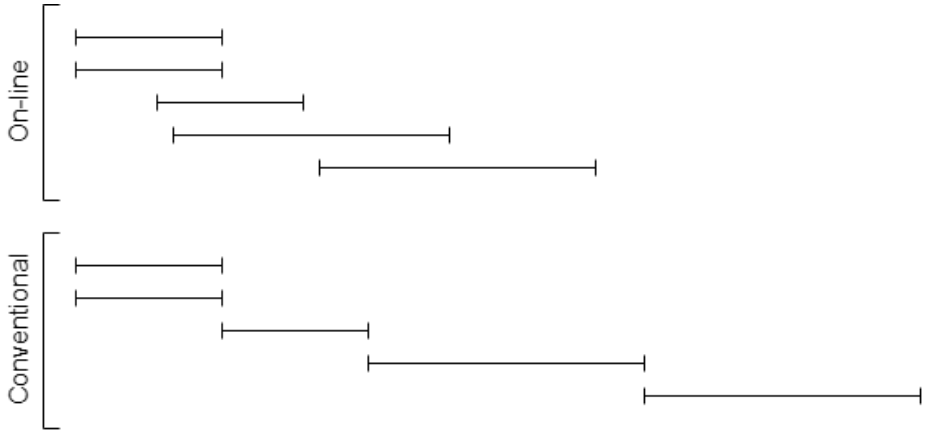



***Fig. 6.*** Reduction of calculations time

**Conclusions.** The article shows the possibility of reducing resources and accelerating the calculation of a function such as a recursive-digital filter. A device has been developed which, for this example, allows to achieve an acceleration of up to 40% , as shown in Fig. 5, and reduce the number of pins to a constant value.

At the same time, this device has room for improvement. If after certain operations you do not increase the size of the number to increase accuracy, and leave it as in the input operands, the percentage of acceleration will increase. In addition, support for floating-point operations should be added, which will expand the range of numbers. All these changes will definitely improve the result.

## References

1. Amin A. A. M., Shinwari M. High-Radix Multiplier-Dividers: Theory, Design, and Hardware. IEEE Transactions on Computers. 2010. Vol. 59, no. 8. P. 1009–1022.

2. Khan S. A. Digital Design of Signal Processing Systems. Chichester : John Wiley & Sons Ltd, 2011. 586 p.

3. Piñeiro J., Ercegovac M. D., Bruguera J. D. Algorithm and Architecture for Logarithm, Exponential, and Powering Computation. IEEE Transactions on Computers. 2004. Vol. 53, no. 9. P. 1085–1096.

4. Zhabin V. I., Korneĭchuk V. I., Tarasenko V. P. Computation of rational functions for many arguments. Automation and Remote Control. 1978. Vol. 38, no. 12. P. 1864–1871.

5. Zhabin V., Zhabina V., Verba O. Asynchronous On-Line Float-Point Computations in Systems with Direct Connections between Computation Units. IEEE 2nd International Conference on System Analysis & Intelligent Computing : Proceedings, Kyiv, 5–9 October 2020. Kyiv, 2020. P. 1–5.

## AUTHORS

**Verbovskyi Illia** – student, Department of Computer Engineering, National Technical University of Ukraine “Igor Sikorsky Kyiv Polytechnic Institute”.

E-mail: illyaverb@gmail.com

**Zhabin Valerii** – Doctor of Technical Sciences, Professor, Professor of Department of Computer Engineering, National Technical University of Ukraine “Igor Sikorsky Kyiv Polytechnic Institute”.

E-mail: v.zhabin@kpi.ua

УДК 004.75

**Anton Kopiika, Valentyna Tkachenko**

# DATA PROCESSING SYSTEM FOR SMART CITY BASED ON NEURAL NETWORK

This work is devoted to the problem of automatic road quality control, which can be used by road repair services. This paper provides a survey of some known techniques and algorithms of detecting potholes on the road and describes our own method, using trained neural network based on data gained from accelerometer. It will be shown our concept of system for detecting potholes on the road, which can be used, as a part of the IoT system. It uses data from an accelerometer for finding road bumps using neural network.

**Keywords**: Internet of Things, SmartCity, Potholes detection, Accelerometer, CNN.

Fig.: 2. Tabl.: 0. Bibl.: 8.

**Relevance of the research topic.** The ability to control the quality of the road surface as an element of Smart City system would be an extremely useful to prevent further damage for road surface and safe a lot of money for repairing.

**Target setting.** The main aim is to provide our own method, using trained neural network based on data gained from accelerometer to classify road surface quality.

**Actual scientific researches and issues analysis.** In general, there are several approaches to solving this problem, which are presented in open sources. We can mention both algorithmic and theoretical purposes, but they all have some critical disadvantages. Some of them are needed for qualified personnel for data collection and analysis, and mostly all this approaches do not cover the driver's driving style, because the most important quality indicators can also be considered, abrupt braking or acceleration, the treatment of which can’t be tracked using an accelerometer.

**Uninvestigated parts of general matters defining.** In this paper, the possibility of creating a full-fledged system for finding pits on the roads using accelerometer data based on convolutional neural networks, in contrast to the generally accepted approach in the use of recurrent neural networks for time series analysis. It was also possible to integrate it into a large-scale Internet of Things system.

**The research objective.** In this paper the main purpose was to offer our way for solving the problem of finding road defects with the help of modern linear acceleration sensors for monitoring the quality of road sections. To solve this problem,

we considered and analyzed the algorithms for assessing the quality of the road surface and marking road sections by quality and create own algorithm based on neural network solutions. We offer a hardware that allows real-time detection of defects on the road surface and in a special way to mark such road sections.

**Materials and methods of research.** First of all, hole system is divided into two parts like on figure 1. The first part is a device for collecting linear acceleration data. It was proposed to use the lis3dsh accelerometer and the stm32f407vg microcontroller as hardware for this part. Their choice is due to low energy consumption and high popularity in the Internet of Things, as well as the availability of convenient software for their programming and debugging. It should also be noted that you can easily use devices of other lines stm32 with minimal reconfiguration of configuration files.

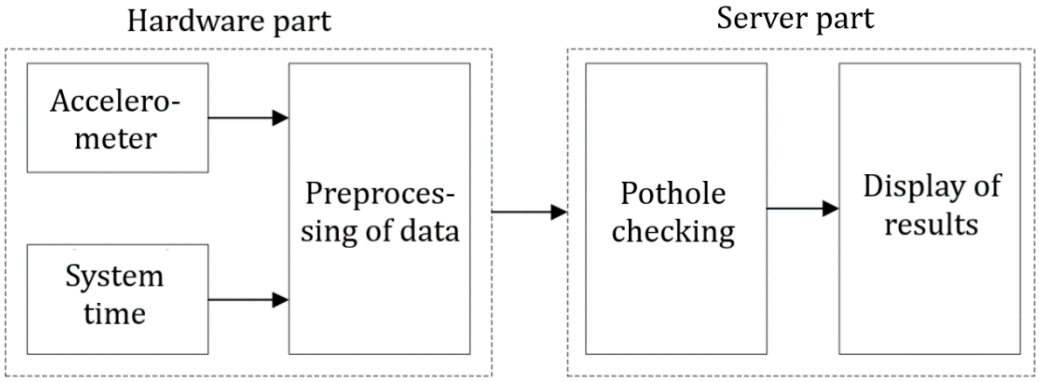



***Fig. 1.*** System architecture

The second part of the overall system is the server, which collects data from all devices connected to it by the communication protocol, and is responsible for the secondary processing and subsequent classification of the quality of the road surface. It is worth noting that this approach allows you to scale the size of the system, add other peripherals. In this work the own dataset was collected. To do this, a test run was performed on sections of road of different quality. The csv file with three parameters, each of which represents the rate of linear acceleration in one of the axes, is considered as a dataset. The total size of the training dataset is 3600 frames. Each frame is a matrix of 3 * 64 in accordance with the number of records for 2 seconds. For this work was made our own model of CNN network. In total it can classify a road quality level on 4 types:

1. Smooth road
2. Curb, lying policeman, railway crossing

3. Small pits

4. Large pits (deeper than 2 cm, more than 20 cm in diameter)

**Research results.** Testing of our device was held on the road section which has some different roughness such as (large potholes, small potholes, pothole clusters, gaps, pavements and rail crossings). It helps to find profits and disadvantages of methods we used. As a result, a neural network was obtained, which classifies the quality of the road surface with an accuracy of 85.2%. The graph of accuracy indicators obtained depending on the number of epochs is shown in Fig. 2.

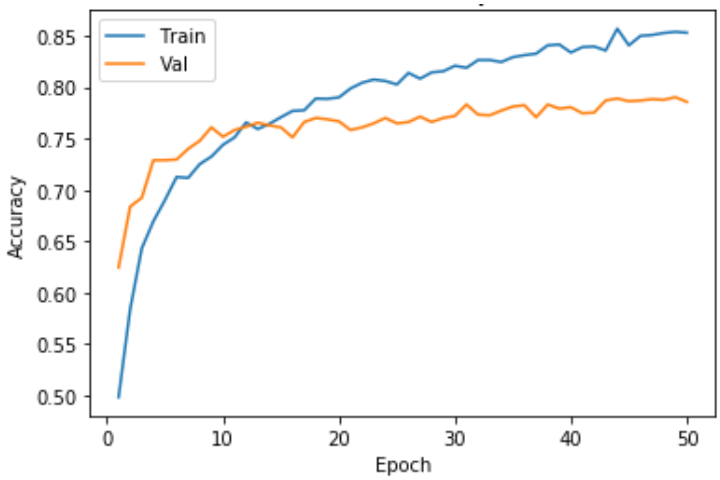



***Fig. 2.*** Neural network accuracy depending on the number of training epochs

**Conclusions.** This paper describes a system for identifying potholes on roads using developed device as part of the IoT system. This system was developed based on an overview of such examples, their advantages, and their disadvantages. Using a neural network solutions help to find more features that can't be found using human eye. Such system concept in the future may become part of a larger project, and increase its functionality. In the future, it is possible to improve the process of detecting holes by increasing the dataset to make network familiar to different manners of driving.

Based on the obtained result, we can clearly say that the system has certain advantages over analogues that exist in the world. First of all, the created model has a low level of energy consumption, competitive for similar microcontroller systems and much more prevalent in comparison with the systems as used by mobile devices. Disadvantages include the attachment to the correct position of the sensor, but this problem is solved by rigid fixation, or by adding a method to recalculate the acceleration relative to the static coordinate system.

## References

1. Komninos, N. Intelligent Cities: Innovation, Knowledge Systems, and Digital Spaces; Taylor & Francis: Abingdon, UK, 2002.

2. R. Bishop, "A survey of intelligent vehicle applications worldwide", IEEE Intelligent Vehicles Symposium (IV 2000), Dearborn, MI, USA, May, 2000, pp. 25-30.

3. Devapriya, W.; Babu, C.N.K.; Srihari, T. Real time speed bump detection using Gaussian filtering and connected component approach. In Proceedings of the World Conference on Futuristic Trends in Research and Innovation for Social Welfare (Startup Conclave), Coimbatore, India, 29 February–1 March 2016; pp. 1–5.

4. Mohamed, A.; Fouad, M.M.M.; Elhariri, E.; El-Bendary, N.; Zawbaa, H.M.; Tahoun, M.; Hassanien, A.E. RoadMonitor: An intelligent road surface condition monitoring system. In Intelligent Systems' 2014; Springer: Berlin, Germany, 2015; pp. 377–387.

5. Y.-c. Tai, C.-w. Chan, and J. Y.-j. Hsu, "Automatic road anomaly detection using smart mobile device," in Proceedings of the2010 Conference on Technologies and Applications of Artificial Intelligence (TAAI2010), November 2010

6. De Zoysa, Kasun, Chamath Keppitiyagama, Gihan P. Seneviratne, and W. W. A. T. Shihan. "A public transport system based sensor network for road surface condition monitoring." In Proceedings of the 2007 workshop on Networked systems for developing regions, pp. 9. ACM, 2007.

7. Jakob Eriksson, Lewis Girod, Bret Hull, Ryan Newton, Samuel Madden, and Hari Balakrishnan. 2008. The Pothole Patrol: Using a Mobile Sensor Network for Road Surface Monitoring. In Proceedings of the 6th International Conference on Mobile Systems, Applications, and Services (MobiSys '08). ACM, New York, NY, USA, 29–39.

8. Marius Hoffmann, Michael Mock, and Michael May. 2013. Road-quality classification and bump detection with bicycle-mounted smartphones. In Proceedings of the 3rd International Conference on Ubiquitous Data Mining-Volume 1088. CEUR-WS. org, 39–43.

## AUTHORS

**Kopiika Anton** – Graduate Magister of Department of Computer Engineering at National Technical University of Ukraine "Igor Sikorsky Kyiv Politechnic Institute". Legal address of Working place: 37, Prosp. Peremohy, Kyiv. E-mail: toxxa099@gmail.com. Phone: +380680202074. Orcid id:0000-0003-4090-7507

**Valentyna Tkachenko** – PhD, Associate Professor of Department of Computer Engineering. Working place: National Technical University of Ukraine "Igor Sikorsky Kyiv Politechnic Institute". Legal address of Working place: 37, Prosp. Peremohy, Kyiv. E-mail: tkavalivas@gmail.com. Phone: 0442049092. Orcid id: 0000-0002-1080-5932