UDC 004

Information, Computing and Intelligent systems. – 2020. – № 1. – P. 86.

Recommended for publication by the Academic Council of the Faculty of Computer Science and Computer Engineering

The scientific journal “Information, Computing and Intelligent systems” is intended for the publication of the results of scientific research and scientific and practical developments in the field of technical sciences by students, masters, PhDstudents, scientists, and practicing specialists in the field of science "Information systems".

The thematic orientation of the journal “Information, computing and intelligent systems” is reflected in the following headings: computerized and computer systems and networks, information technologies, the Internet of Things, information transformation and processing, cloud computing, computer cryptography, data protection, intelligent systems, artificial intelligence, machine learning, automated design of software and technical tools, system control, diagnostics and control of parameters of complex systems, processes and environments; engineering knowledge, embedded systems, robotics, microelectronics.

ISSN 2708-4930

Certificate of state registration No. 23827-13667ПР from 20.02.2019
Magazine in English
Web- resource – https://itvisnyk.kpi.ua/
Format 60×84 1/8. Garnitura Times. Offset Folder № 1.

Ministry of Education and Science of Ukraine
National Technical University of Ukraine “Igor Sikorsky Kyiv Polytechnic Institute”

SCIENTIFIC EDITION

Information, Computing and Intelligent systems

The journal is the legal successor of the Collection of scientific works “Bulletin of NTUU “KPI”. Informatics, control and computer engineering”

Founded in 1964 years

Issue 1

Kyiv – 2020 (1)

SUMMARY

M. Novotarskyi, V. Kuzmich USAK method for the reinforcement learning………… 4

A. Doroshenko, O. Beketov, O. Yatsenko Loop parallelization automation for graphics processing units…………………………………………………………….. 15

Alireza Mirataei, H. Khalil, O. Markovskyi Protected discrete fourier transform implementation on remote computer systems………………………………………… 26

O. Markovskyi, I. Humeniuk, O. Shevchenko Hash search organization in e-dictionaries using block ciphers……………………………………………….. 34

O. Dolynnyi, S. Nikolskiy, Y. Kulakov The method of SDN clustering for controller load balancing…………………………………………………………………... 43

I. Stetsenko, A. Dyfuchyn Petri-object simulation: technique and software………….. 51

A. Kopiika, R. Piskun, V. Tkachenko, I. Klymenko Road monitoring system based on IoT technology for Smart City……………………………………………….. 60

I. Muraviov, V. Taraniuk, I. Klymenko Making an IoT development platform from a simple microcontroller demonstration board………………………….…………… 68

Abstracts……………………………………………………………………………… 77

Анотації……………………………………………………………………………… 82

UDC 004.056.57:032.26

NOVOTARSKYI M.,
KUZMYCH V.

USAK METHOD FOR THE REINFORCEMENT LEARNING

In the field of reinforcement learning, tabular methods have become widespread. There are many important scientific results, which significantly improve their performance in specific applications. However, the application of tabular methods is limited due to the large amount of resources required to store value functions in tabular form under high-dimensional state spaces. A natural solution to the memory problem is to use parameterized function approximations. However, conventional approaches to function approximations, in most cases, have ceased to give the desired result of memory reduction in solving real-world problems. This fact became the basis for the application of new approaches, one of which is the use of Sparse Distributed Memory (SDM) based on Kanerva coding. A further development of this direction was the method of Similarity-Aware Kanerva (SAK). In this paper, a modification of the SAK method is proposed, the Uniform Similarity-Aware Kanerva (USAK) method, which is based on the uniform distribution of prototypes in the state space. This approach has reduced the use of RAM required to store prototypes. In addition, reducing the receptive distance of each of the prototypes made it possible to increase the learning speed by reducing the number of calculations in the linear approximator.

Keywords: reinforcement learning, Kanerva coding, function approximation, prototype, value function

1. Introduction

Given the increased interest in the application of intelligent systems in various fields of science and technology, the relevance of the development and practical implementation of reinforcement learning algorithms has increased. In recent years, we can see such significant progress in this area that we can say that reinforced learning already has the main features of a particular field of science. A number of university courses have been created [1,2,3,4], there is a basic textbook on reinforced learning [5], which the authors constantly keep up to date. Tabular reinforcement learning methods, namely Multi-armed Bandits [6], Finite Markov Decision Processes [7], Monte-Carlo Methods [8], Temporal-Difference Learning [9] and their modifications are already considered as classical approaches. They are, in most cases, used as part of the development of modern approaches to reinforcement learning. The reason for this is the fact that the solution of modern intellectual tasks requires large or multidimensional state spaces, which leads to significant problems in creating optimal policies.

The natural direction in which reinforcement learning methods are developed is that learning agents can use approximate functions, which makes it possible to significantly improve the productivity of learning in cases of large-scale state spaces [10]. In this case, as a rule, we use parameterized functions that ensure the successful operation of intelligent systems that describe the tasks of the real world. However, constructing an approximate function is not an easy task. Such construction in most cases requires pre-configuration, which includes manual allocation of state spaces based on expert evaluation or uses complex heuristic algorithms. It is known that to build effective heuristic algorithms it is also necessary at the first stage to perform an analysis of the properties of the state space for their correct separation. Therefore, the mentioned approaches to the construction of approximate functions have a disadvantage, which is associated with the problems of dynamic scaling of the state space. This scaling is extremely important for real-world tasks, as in most cases it causes a significant increase in RAM, the available size of which is a major constraint on the way to improving reinforced learning. One effective solution to this problem has been proposed in [11] in the form of a new sparse distributed memory (SDM). A little later, this approach was called "Kanerva coding" after its creator Pentti Kanerva, who first proposed it in 1988. The main advantage of the proposed approach is that it requires a much smaller amount of RAM in the case of increasing state space than standard methods with approximate functions [12].

The essence of Kanerva coding method is that we choose a certain set of prototypes, each of which is a copy of one of the possible states of the environment. Each of these prototypes has a dimension that coincides with the dimension of the state space. Then the selected s state is called adjacent to the p_i prototype if the bitwise difference between the s state and the p_i prototype is less than some predetermined threshold number. Each p_i prototype has a corresponding value θ_i, which is a component of the parameter vector. Then the approximate value of each state-action pair is determined by the sum: $\sum_i \theta_i \mu_i(s)$. With this approach, Kanerva coding eliminates the exponential growth of memory with increasing state space dimension. However, modern reinforced learning tasks increasingly require the use of state space with a dimension of several thousand. In this case, it is necessary to look for new approaches, some of which have been considered in detail and partially proposed in [13]. Among the proposed approaches, the Similarity-Aware Function Approximation method is original and effective. The basics of applications of this method are presented in [14]. This method generalizes the concept of similarity by introducing a new continuous metric, which increases the resolution of the method and thus reduces the number of prototypes that need to be used.

However, the method does not solve the main problem of choosing the optimal number of prototypes. Therefore, the effectiveness of this method can be seen only after a detailed preliminary analysis, which aims to find a fine line between the prototype starvation and over-generalization. In this paper, we propose an approach that allows the use in the linear approximator only those prototypes that are adjacent to the current state of the environment. This approach makes it possible to optimize the use of RAM, provided that large-scale state spaces are used in reinforcement learning.

2. Basics of related methods

Reinforced learning systems are based on a single concept that includes agents, environments, and states, as well as actions and rewards. The general scheme showing the main interactions of these entities is shown in Fig. 1.

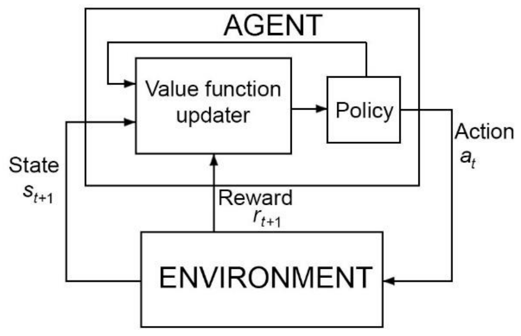


Fig. 1. Generalized structure of the reinforcement learning system

The main components of this system are the agent and the environment, which interact with each other. An agent is a certain entity that has the ability to produce actions that affect the environment. The environment is the world around the agent, which perceives the a actions and returns the r reward to the agent. The agent also has the ability to observe the current state, s, of the environment. Such a system is sometimes called a feedback system in which the role of feedback signals is played by signals of reward and state of the environment. The ways in which the agent chooses the next action are basic and are determined by the methods of his learning. At each point in time, the policy determines the behavior of the agent. It maps the set of all accumulated knowledge about the environment to the action to be performed. The sole purpose of the agent is to maximize the reward he can receive from the environment. The size of the reward signal determines the agent's policy choice. However, a policy that only takes into account the remuneration for the last action may conflict with the agent's overall goal if it leads to a significant reduction in remuneration in the future. The size of the reward signal determines the agent's policy choices. However, a policy that only considers reward for the last action may conflict with the agent's overall goal if it leads to significantly reduced reward in the future. Previous experience of interaction with the environment, which takes into account the value function, helps to solve this problem. Figure 1 shows that the input data for the policy is formed by the value function, which takes into account not only the current state and current reward, but also previous states and rewards with a certain discount.

2.1. Reinforced learning with function approximation

We will consider a reinforcement learning system, which includes an agent represented by an algorithm that has the property of training. At a t time, the agent performs an a_t action in accordance with a $\pi\left(a_t \middle| s_t\right)$ policy that specifies the probability of performing the action under the s_t state of the environment. At the next $t+1$ time, the agent receives a r_{t+1} reward and the result of observation the s_{t+1} state of the environment. At each time step, the agent tries to modify his π policy so as to maximize the total amount of rewards in each episode using the expression:

$$g_t = r_{t+1} + \gamma r_{t+2} + \gamma^2 r_{t+3} + \ldots = \sum_{k=0}^{T-t-1} \gamma^k r_{t+k+1} \tag{1}$$

where g_t is the total return of the agent, starting from t time; γ is the rate of the discount, which varies in $0 \le \gamma \le 1$ range; T is the maximum length of the history, which in the general case can be equal to infinity. Consider the principle of the agent, which combines Q-learning with the mechanisms of approximation for the successful operation of reinforcement learning systems using high-dimensional state spaces. Q-learning is based on the use of the Q-function in order to implement π policy. This function is also called the state-action value on-policy function. It is denoted as $Q^{\pi}\left(s,a\right)$. Using the Bellman optimality equation, we can find the $Q^*\left(s,a\right)$ optimal value as the maximum return; starting from the s state under condition we provide the actions and operations in accordance with the π policy:

$$Q^*\left(s,a\right) = \mathrm{E}\left[r + \gamma \max_{a'} Q^*\left(s',a'\right)\right]. \tag{2}$$

The reinforcement learning algorithm consists in realization of iterative process of the value function estimation:

$$Q_{i+1}\left(s,a\right) = \mathrm{E}\left[r + \gamma \max_{a'} Q^*\left(s',a'\right) \middle| s,a\right]. \tag{3}$$

This sequence converges to the optimal value: $\lim\limits_{i\to\infty} Q_i = Q^*$. However, the achievement of this result encounters problems of practical implementation due to the fact that the representation of the Q-function in tabular form leads to an exponential increase in the amount of RAM when using high-dimensional state spaces. To solve this problem, we present the Q -function as a parameterized function: $Q^*\left(s,a\right) \approx Q\left(s,a;\boldsymbol{\theta}\right)$, where $\boldsymbol{\theta}$ is parameter vector. Then the problem of representing the state-action function can be reduced to choosing values of a much smaller number of components of the $\boldsymbol{\theta}$ vector. In most cases, this uses a linear approximator, which is represented by the expression:

$$Q\left(s,a;\boldsymbol{\theta}\right) = \boldsymbol{\theta}^T \mathbf{x}(s,a) = \sum_{i=0}^{n-1} \theta_i x_i\left(s,a\right), \tag{4}$$

where $\mathbf{x}\left(s,a\right) = \left(x_0\left(s,a\right), x_1\left(s,a\right), \ldots, x_{N-1}\left(s,a\right)\right)$ is the feature vector, which represents the state-action function; $\boldsymbol{\theta}^T$ is the transposed parameter vector with the number of components equal to the number of features; n is the number of dimensions of the state space. Features in this case are $x_i\left(s,a\right)$ basic functions, because they form a linear basis to create the $Q\left(s,a;\boldsymbol{\theta}\right)$function approximation. There are different approaches to the selection of the feature vector components. One such approach is, for example, the stochastic gradient-descent method, for which $\mathbf{x}(s,a)=\nabla Q\left(s,a;\boldsymbol{\theta}\right)$. Then the iterative process of finding the vector of parameters is determined by the expression:

$$\boldsymbol{\theta}_{t+1}\leftarrow\boldsymbol{\theta}_t+\alpha\left[Q^{\pi}\left(s_t,a_t\right)-Q\left(s_t,a_t;\boldsymbol{\theta}\right)\right]\mathbf{x}\left(s_t,a_t\right). \qquad (5)$$

As is known, for a stochastic gradient-descent method, this iterative process converges to a local minimum, which does not always coincide with the global minimum. Since in most cases it is necessary to find out a global minimum, it is important to find other function approximation techniques.

2.2. Algorithms for high-dimensional state spaces

Algorithms for high-dimensional state spaces are extremely relevant because in most cases they are the only possible approach to solve practical tasks of reinforcement learning. A well-known algorithm for large state spaces is the tile coding algorithm [5]. This algorithm has become widespread due to its simplicity and efficiency. An important disadvantage of this algorithm is that it requires pre-allocation of state space manually. Such preliminary selection significantly affects the further efficiency of the algorithm.

There are usually a large number of modifications of this algorithm that use different heuristic solutions to implement an efficient distribution of state space
[16, 17]. However, such heuristic algorithms require the use of a large number of computing resources to process current information on each of the state space dimension and pre-configure a large number of parameters that characterize the environment. Therefore, the effective use of these algorithms for reinforcement learning with the high-dimensional state space is difficult and available only after gaining some experience.

Radial basis functions (RBF) is the second known algorithm for the implementation reinforcement learning in large state spaces [5]. For this algorithm, the features have a continuous value in the range [0,1]. Typically, RBF-features are subject to normal distribution. The advantage of such features over binary ones is that the corresponding functional dependencies are differentiated, which expands the possibilities of management and analysis. The disadvantages of the RBF method repeat the same disadvantages as tile coding. These disadvantages include the increased computational complexity and the need for manual adjustment, which requires an expert level of use of this method.

The most promising algorithm has every reason to consider the Kanerva coding algorithm. Today, this is one of the algorithms that is characterized by the smallest increase in RAM with increasing dimensionality of the state space.

In this method, as in the previous ones, it is necessary to choose the vector of basic functions. They are represented by prototypes of states in this method. New $p=\left(\varphi_0,\varphi_1,\ldots,\varphi_{n-1}\right)$ prototypes are created based on the components of the state space. The method uses similarity for each of the dimensions of the state space, which is equal to the code distance between the binary representation of the vectors of the prototype and the state. The $\mu\left(s,p_i\right)$ function is equal to one, provided that the s state is close to the p_i prototype and zero otherwise. This function is called the membership grade function. It can be represented by the expression:

$$\mu_i\left(s,p_i\right)=\begin{cases}1 \;\; if \; \sum\limits_{j=0}^{n-1}\left(\phi_j \; XOR \; \left(\varphi_j\right)_i\right)\le c,\\ 0 \;\; otherwise.\end{cases} \qquad (6)$$

where j is the bit number, n is the state space dimensionality, c is the threshold value.

For Kanerva coding, there is also the problem of choosing the optimal set of prototypes, which in practice is solved by randomly selecting from the available set of states. Such a choice does not solve the problem of optimality. In this case, there may be significant disturbances in the distribution of receptive fields and, as a consequence, there is a decrease in the efficiency of representation of certain areas of the state space and the corresponding value functions. An adaptive adjacency method [15] has been proposed as one of the known ways to improve the coverage of state space by prototype receptive zones [15], but such an approach also cannot always solve the problem, but only reduces the percentage of failed coatings.

Also known methods that partially solve the problem of optimal coverage based on the use of fuzzy logic. But such approaches are a separate area, which we do not consider in this paper.

An original method called Similarity-aware Kanerva or SAK for short has been proposed recently.

2.3. SAK method

The SAK method significantly changes the principle of determining the similarity of states and prototypes and thus solves the problem of determining the fields of perception for the selected set of prototypes. The main difference of this approach is that the features are no longer binary, but are represented as continuous quantities, which are represented by real numbers [14].

For a feature with the j index, we determine the d_j distance between the p prototype and the s state by the expression:

$$d_j = \frac{\left|\phi_j - \varphi_j\right|}{range_j / \rho}, \tag{7}$$

where $range_j$ is the range of possible values of the j feature, ρ is a fixed factor, the value of which exceeds 1 and provides sensitivity to changes in the $\left|\phi_j - \varphi_j\right|$ difference.

Define the similarity grade for the j feature using the expression:

$$m_j\left(s,p\right) = e^{-d_j(s,p)}. \tag{8}$$

Fig.2 shows the graph of this function:

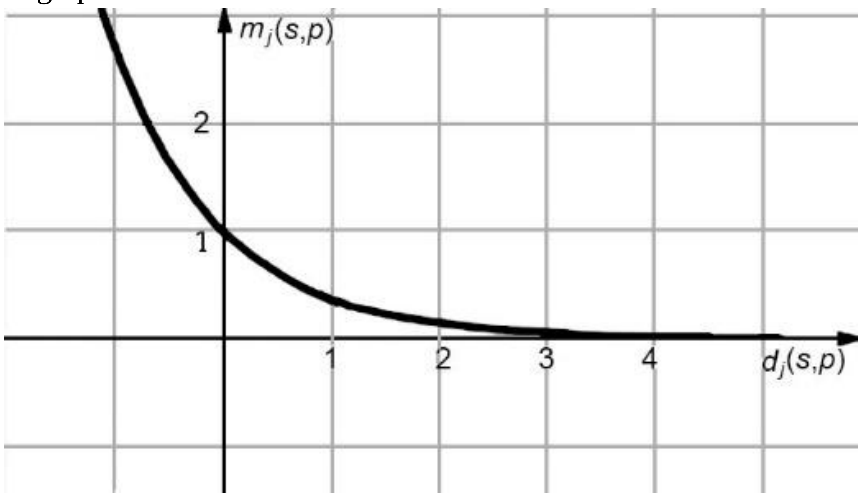


Fig. 2. Graph of the $m_j\left(s,p\right)$ function

The figure shows that the value of the $m_j\left(s,p\right)$ function is equal to 1, if the value of the difference between the dimension of the ϕ_j state and the φ_j prototype approaches zero with increasing distance between dimensions.

The $\mu\left(s,p\right)$ membership grade of states and prototypes is equal to the minimum similarity between their dimensions:

$$\mu(s,p) = \min_{j=0,1,\ldots n-1} m_j(s,p). \tag{9}$$

Thus, the membership grade for states is a continuous value that varies in the range [0,1].

When using Kanerva coding, we can specify a fixed number of prototypes. In the case of using a linear approximator for Kanerva coding, the value of the state-action function approximation is determined from the expression:

$$Q(s,a;\theta) = \sum_{i=0}^{|P(s)|} \theta(p_i,a) \cdot \mu(s,p_i), \tag{10}$$

where $|P(s)|$ is the power of the indexed list of prototypes that are similar to the s state.

The $\theta(p_i,a)$parameter value will be stored and updated for each p_i prototype when selected the a action by the agent. When using the Sarsa algorithm, the following update is determined from the expression [15]:

$$\theta_i(p_i,a) \leftarrow \theta_i(p_i,a) + \delta(s,p_i)\alpha(s,a)\left[r + \gamma\left(Q(s',a';\theta_{i-1}) - Q(s,a;\theta_i)\right)\right], \tag{11}$$

where $\delta(s,p_i) = \frac{\mu(s,p_i)}{\sum_{k=0}^{|P(s)|} \mu(s,p_k)}$ is a current fraction of the membership grade, $\alpha(s,a)$ is the step size parameter.

Although the SAK algorithm significantly improves the basic parameters of reinforcement learning compared to the Kanerva coding algorithm, it does not solve the main problems of this type of algorithms. In particular, the algorithm uses random generation of prototypes. While it is known that using prototype tuning, actually increase the average solution rate from 67.9% to 97.1% [18]. In addition, the difficult question of choosing the optimal number of prototypes remains unclear. The choice of the optimal number of prototypes makes it possible to avoid the prototypes starvation due to the insufficient number of active prototypes for the value function approximation, as well as to avoid over-generalization of prototypes.

3. USAK algorithm

This paper proposes a Uniform Similarity-Aware Kanerva algorithm (USAK) that differs from SAK by adding a heuristic that improves the use of prototypes and proposes some modifications to include prototypes in the ActiveN list [14].

For an arbitrary j dimension, we determine the d_j distance between the p prototype and the s state from the expression:

$$d_j = \left|\phi_j - \varphi_j\right|. \tag{12}$$

The similarity grade for a j feature is determined from the condition of uniform placement of prototypes in the range of this feature.

$$m_j(s,p) = 1 - \frac{2b_j d_j}{\rho \cdot range_j}, \tag{13}$$

where ρ is the coefficient of overlap of the receptive zones of the prototype: $1 < \rho \le 1.5$, $range_j$ is the range of the j feature, b_j is the number of prototypes in the range of the j feature.

The graph of the function is shown in Fig.3.

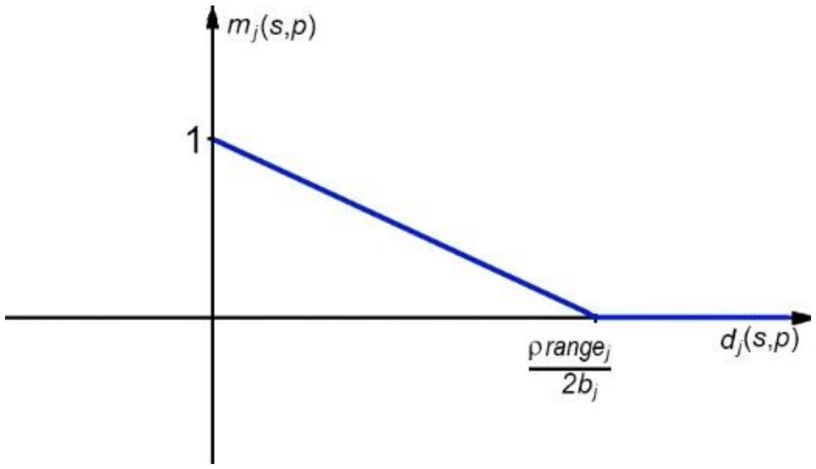


Fig. 3. Graph of the $m_j(s,p)$ function for the USAK algorithm

The graph in fig. 3 specifies a linear normalized value of the $m_j(s,p)$ similarity grade for the j feature. The value of $m_j(s,p)$ is equal to 1 if the value of the difference between the ϕ_j state dimension and the ϕ_j corresponding dimension of the prototype is zero. This $m_j(s,p)$ function linearly decreases to zero with increasing distance between dimensions and is equal to 0 at the perceptive limit of the prototype respectively to the j feature.

The $\mu(s,p)$ membership grade of the s state and the p prototypes is equal to the minimum similarity grade for their dimensions (9). Thus, the membership grade for states is a continuous quantity that varies linearly in the range [0,1]. Since the prototype has clear and limited perceptive zones for each of its features, in this case it is important to fully cover the state space with prototypes for each of the state dimensions. In this paper, an algorithm for uniform coverage of the state field is proposed.

In the general case, consider the n -dimensional state space with uniform coverage of this space by prototypes. Then there is no need to save all prototypes, because they can be dynamically generated by setting the values of their dimensions with a predetermined step. Consider the pseudocode of the USAK algorithm, which includes the dynamic generation of features of evenly distributed prototypes.

Algorithm 1 : Uniform Similarity-Aware Kanerva Coding

Input:

$\mathbf{b} = (b_0, b_1, ..., b_j, ..., b_{n-1})$: the number of prototypes for each dimension.

$\mathbf{range} = (range_0, range_1, ..., range_j, ..., range_{n-1})$: range of features for each dimension.

N : number of episodes.

M : the maximum number of iterations in the episode.

Output: $\boldsymbol{\theta}$ vector as a learning outcome.

```
def P_generator($s'$):
  for $i$ in range($B$)
     for $j$ in range(n):
            $\phi_j = s'(j)$
            $\varphi_j = i \times \frac{range_j}{b_j + 1}$
            $d_j = |\phi_j - \varphi_j|$
            $m_j = 1 - \frac{2b_j d_j}{\rho \cdot range_j}$
            if $m_j > 0$ then $\mu.append(m_j)$ else $\mu.append(0)$
        $\mu_i = \min(\mu)$
        if $\mu_i > 0$ then $\mathbf{p}_{active}.append(i)$; $\boldsymbol{\mu}_{active}.append(\mu_i)$
    return $\mathbf{p}_{active}$, $\boldsymbol{\mu}_{active}$

def Teta_iter( $s_t, a_t, r_t, s_{t-1}, a_{t-1}$):
    for $i$ in range$(|\mathbf{p}_{active}|)$:
        $\delta(i) = \frac{\mu_i}{\sum_{k=0}^{|\mathbf{p}_{active}|} \mu_k}$
        $\theta_i = \theta_i + \alpha_t \left[ r_t + \gamma Q(s_t, a_t; \boldsymbol{\theta}_t) - Q(s_{t-1}, a_{t-1}, \boldsymbol{\theta}_{t-1}) \right] \delta(i)$
def Main():
    Determining the number of prototypes: $B = b_0 \times b_1 \times ... \times b_{n-1}$
    Initialization of the parameter vector: $\boldsymbol{\theta} = 0$, where $|\boldsymbol{\theta}| = B$.
    for episode in range(1,N):
        Randomly choose $s_0$ and $a_0$.
        for t in range(1,M):
            Perform $a_{t-1}$ in $s_{t-1}$.
            Get $r_t$ and $s_t$.
            Determine the $\mathbf{p}_{active}$ vector of active prototypes
            and the $\boldsymbol{\mu}_{active}$ distance vector:
            $\mathbf{p}_{active}, \boldsymbol{\mu}_{active} = \text{P\_generator}(s_t)$
            Determine the value of the $Q(s_t, a_t; \boldsymbol{\theta})$ function approximation:
            $Q(s_t, a_t; \boldsymbol{\theta}_t) = \sum_{i=0}^{|\mathbf{p}_{active}|} \theta_i \cdot \mu_{active}(i)$
            Update $\boldsymbol{\theta} : \boldsymbol{\theta}_{t+1} = \text{Teta\_iter}( s_t, a_t, r_t, s_{t-1}, a_{t-1})$
            Determine $(s_{t+1}, a_{t+1})$ using the $\varepsilon$-greedy method:
            $(s_{t+1}, a_{t+1}) = \arg\max_s Q(s_t, a_t, \boldsymbol{\theta}_t)$
                $s_{t-1} = s_t$, $a_{t-1} = a_t$
```

4. Practical implementation

Let us consider the work of the proposed method on the example created on the basis of the popular problem "WaterWorld", which was first proposed by Andrej Karpathy [19]. Its essence is that the agent must try to survive by avoiding collisions with objects that move freely in the environment. The author of this problem compares it with the problem of spacecraft navigation in the field of asteroids.

The agent receives +1 for survival for one time step and -100 points in case of collision with the simultaneous end of the current episode. To prevent collision, the agent has a set of sensors arranged evenly in a circle. The number of sensors can be changed during the experiments. Obviously, the dimensionality of the state space depends on their number. Each sensor allows you to determine the coordinates of the nearest object in the area of its observation, speed and direction of its movement. The agent also has additional sensors that determine its own coordinates, speed and direction of its movement.

The total number of features varied from 44 to 104, creating a state space with the appropriate dimensionality. The agent can perform 4 actions: left, right, up and down. Each of these actions is represented by a real number that indicates the modulus of velocity of the object in a certain direction. The experiments use agents that have from 4 to 20 sensors. Figure 4 shows a screenshot of the task, which shows the position of the agent with 18 sensors and the environment in which objects float freely.

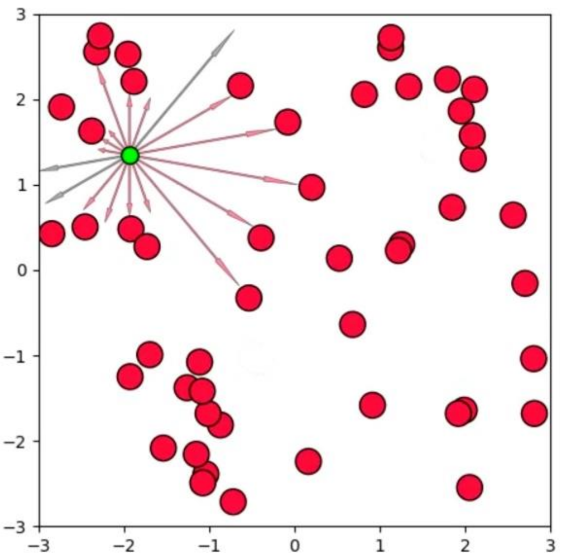


Fig.4. An agent in an environment with "hostile" objects

As shown in Figure 4, the agent is also able to distinguish the wall of the area from floating objects. The modulus of the inverse vector of the velocity of repulsion from the wall is proportional to the modulus of the vector of pushing on the wall. The purpose of the agent is to prolong its existence as much as possible, avoiding collisions with these "enemy objects".

The studies were performed using the USAK algorithm for the described problem. Such studies aimed to experimentally determine the indicators of its effectiveness in a given range of parameters.

Figure 5 shows the rewards for USAK algorithms with a different dimension of the state field. From this graph we can conclude that the speed of learning has no critical dependence on the dimensionality of the state speed. However, the advantage of the USAK algorithm is a 42% reduction in the number of calculations per episode due to the exclusion of remote prototypes from the calculations.

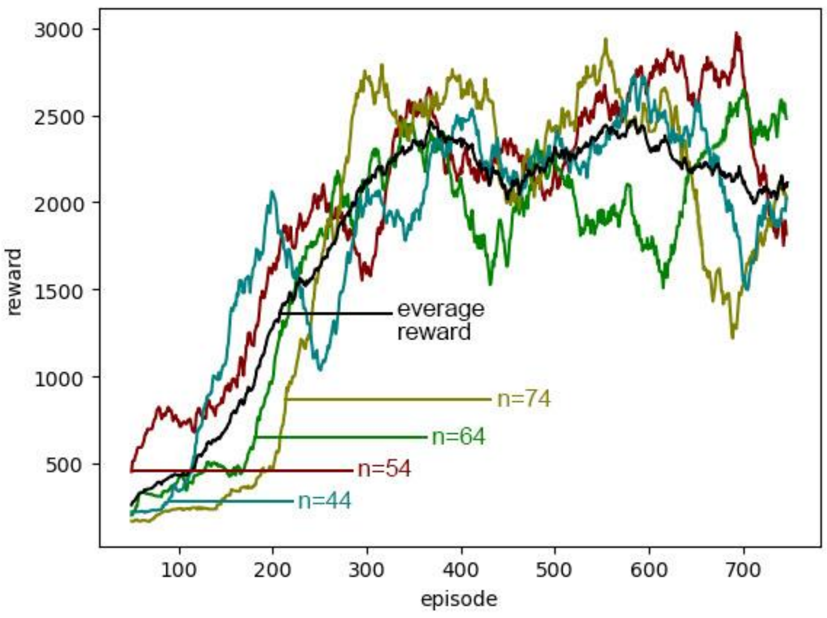


Fig. 5. Dynamics of reward change for different dimensionalities of the state space

Figure 6 shows the increase in occupied RAM over time. This growth does not depend significantly on the size of the state space, which indicates the effectiveness of the proposed approach to the uniform arrangement of prototypes that cover the state space. The advantage of this approach is that remote prototypes do not participate in the calculation, which leads to a linear increase for data to be stored. In this study, as in the previous case, we considered the state spaces with 44, 54, 64, and 74 dimensionalities. These dimensionalities of the state spaces are formed by agents with 8,10,12 and 14 sensors, respectively.

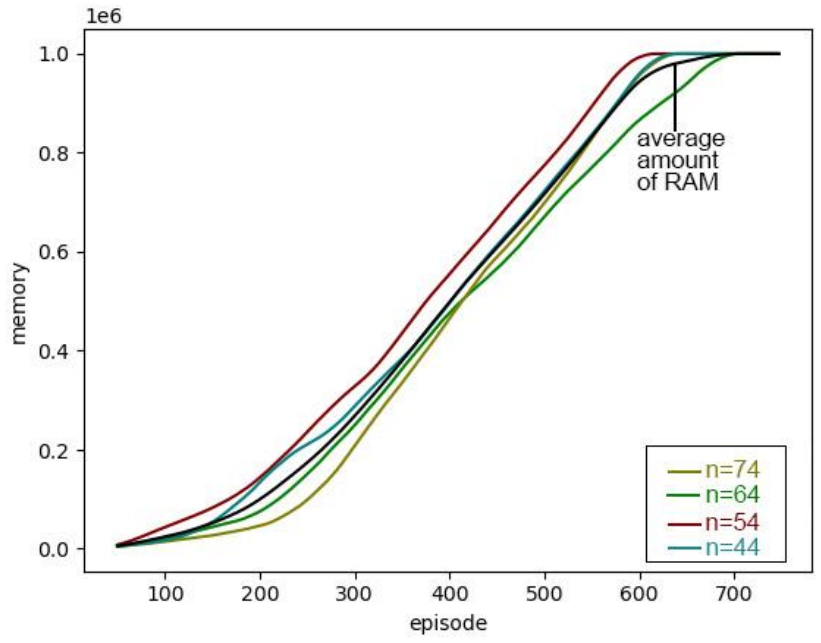


Fig. 6. Linear growth of occupied RAM as a function of time for state spaces of dimensionalities 44, 54, 64 and 74.

5. Conclusion

This paper discusses modern approaches to reducing the use of computer RAM in solving reinforced learning problems with high-dimensional state spaces. The USAK method is proposed, which improves the characteristics of the SAK method due to the uniform distribution of prototypes that cover the state space. Even distribution of prototypes has reduced the amount of RAM for their storage because such placement of prototypes allows you to generate prototypes automatically with the appropriate step of their dimensions. The second advantage of uniform placement of prototypes is that it is possible to use a predetermined fixed perceptivity of prototypes. This fact has a positive effect on the reduction of the volume of calculations in the linear approximation of the value function, because in this case only those prototypes that have a membership value greater than zero are taken into consideration. A number of experiments were conducted on the basis of the well-known problem "WaterWorld", which allowed to determine the advantages of this method, as well as to form their vision of ways to further improve it.

References

1. The Stanford University (2020), “CS234: Reinforcement Learning”, available at: https://web.stanford.edu/class/cs234/
2. The University of Edinburg (2020), “Reinforcement Learning”, available at: http://www.inf.ed.ac.uk/teaching/courses/rl/
3. The University of Alberta (2020), “Fundamentals of Reinforcement Learning”, available at: https://www.classcentral.com/course/fundamentals-of-reinforcement-learning-14497

4. Silver D. (2020), “UCL Course on RL”, University College London, available at: https://www.davidsilver.uk/teaching/
5. Sutton R. S. and Barto A.G. (2018), “Reinforcement Learning: An Introduction”, Cambridge: The MIT Press, , available at: http://www.academia.edu/download/38529120/9780262257053_index.pdf
6. Slivkins A. (2019), “Introduction to Multi-Armed Bandits”, available at: https://arxiv.org/abs/1904.07272v5
7. Levin D.A. and Peres Y. (2017), “Markov chains and mixing times”, available at: https://www.statslab.cam.ac.uk/~beresty/teach/Mixing/markovmixing.pdf.
8. Wiering M. fand van Otello M. (2012), “Reinforcement Learning”, Berlin: Springer-Verlag.
9. Hester T. (2013), “TEXPLORE: Temporal Difference Reinforcement Learning for Robots and Time-Constrained Domains”, Berlin: Springer-Verlag.
10. Whiteson Sh. and Stone P. (2006), “Evolutionary Function Approximation for Reinforcement Learning“, Journal of Machine Learning Research, Vol. 7, pp. 877-917
11. Kanerva P. (2003), “Sparse Distributed Memory”, Cambridge: MIT Press.
12. Cheng Wu and Yiming Wang (2017), “Learning From Big Data: A Survey and Evaluation of Approximation Technologies for Large-Scale Reinforcement Learning”, IEEE, Computer and Information Technology (CIT), International Conference,-DOI: 10.1109/CIT.2017.11
13. Wei Li (2019), “Function Approximation-based Reinforcement Learning for Large-Scale Problem Domains”, PhD dissertation, Northeastern University, Boston, Massachusetts..
14. Wei Li and Meleis W. (2018) “Similarity-Aware Kanerva Coding for On-Line Reinforcement Learning”, Proceedings of the 2-nd International Conference on Vision, Image and Signal Processing.
15. Wei Li and Meleis W. (2018), “Adaptive Adjacency Kanerva Coding for Memory-Constrained Reinforcement Learning“, International Conference on Machine Learning and Data Mining in Pattern Recognition, pp.187-201.
16. Sherstov A. A. and Stone P. (2005),” Function Approximation via Tile Coding: Automating Parameter Choice”, International Symposium on Abstraction, Reformulation, and Approximation, pp.194-205.
17. Waskow S. J. and Bazzan A.L.C. (2010), “Improving Space Representation in Multiagent Learning via Tile Coding”, Brazilian Symposium on Artificial Intelligence, pp.153-162.
18. Cheng Wu (2010), “Novel Function Approximation Techniques for Large-scale Reinforcement Learning”, PhD dissertation, Northeastern University, Boston, Massachusetts.
19. Karpathy A. (2020), “REINFORCEjs. WaterWorld: DQN”, available at: https://cs.stanford.edu/people/karpathy/reinforcejs/waterworld.html

UDC 004.4’24

DOROSHENKO A.,
BEKETOV O.,
YATSENKO O.

LOOP PARALLELIZATION AUTOMATION FOR GRAPHICS PROCESSING UNITS

A technology that allows extending GPU capabilities to deal with data volumes that outfit internal GPU’s memory capacity is proposed. It involves loop tiling and data serialization and can be applied to utilize clusters consisting of several GPUs. Applicability criterion is specified and a semi-automatic proof-of-concept software tool is implemented. The experiment to demonstrate the feasibility of the proposed technology is described.

Keywords: CUDA, general-purpose computing on graphics processing units, loop optimization, parallelization methods.

1. Introduction

The complication of computing problems and improvements of hardware, on the other hand, resulted in the appearance of multiprocessor systems, that are naturally suited to employ parallel algorithms. Nowadays, many new parallel platforms are emerging, one of the most popular is the direction using graphics processing units (GPUs) as general-purpose computing devices. As for now, GPUs have the highest level of parallelism compared to other computational devices. These devices can provide big performance boosts. However, efficient programming of GPUs is not an easy task. Developers should know many technical details about GPU architecture.

The interest in graphics accelerators is ever-growing due to their superior performance compared to conventional processors, availability, and low energy consumption. However, the development of appropriate tools is still a problem. In particular, we consider the problem of modeling parallel systems with heterogeneous components that contain both CPU and GPU. Along with spreading of GPGPU technology [1] that allows the employment of graphics accelerators for solving computational tasks, new challenges arise. As far as GPU is not a standalone device and is managed by a host operating unit, it should be considered within the context of heterogeneous computational platforms. Composing programs for such platforms demands knowledge in architecture and specific programming tools. Generally, concurrent software development passes through the stage of sequential implementation that becomes a starting point for further platform-dependent and hardware environment specific implementations.

Existing automatic code parallelizing tools [2–5] don’t account for the limited amount of GPU’s onboard memory space while real-life problems demand huge amounts of data to be processed. To embrace those cases of massive computational tasks that involve large amounts of data, we propose a technique that provides the ability to rip the loop and to split the data and calculation operations.

This paper considers the problem of automated parallelizing transformation of embedded loops for the target platform of a computing system of heterogeneous architecture that includes graphic processors. A technology for a semi-automated parallelization method of nested loops for graphics processors is proposed. A technique that allows to extend GPU capabilities to deal with data volumes that outfit internal GPU’s memory capacity is revealed and proved. The technology involves loop tiling and data serialization and can be applied to utilize clusters consisting of several GPUs [6, 7].

A formal transformation of the computation loop nest that allows the transition from sequential algorithm to parallel is illustrated on solving linear systems with Cholesky decomposition method, matrix multiplication, and *N*-body problems.

2. A formal model of the loop and parallelization technique

Loop parallelization is a long-standing problem of computational programming. Loops give a fair parallelization opportunity for numerous scientific modeling problems that involve numerical methods. This section introduces the idea of loop transformation.

Let us consider a set of finite sets I_k, $0 \le k \le N$, each set consists of an ordered set of elements, i.e.

$$I_k = \{ i_{k,0} \prec_k i_{k,1} \prec_k \ldots \prec_k i_{k,\#I_k - 1} \},$$

where $\prec_k$ is a partial order over the set I_k, $\# I_k$ indicates the number of elements of the set I_k.

The for loop operator

$$for \;\; i_k \in I_k : S(i_k)$$

is a form of notation of the following sequence

$$\{ S(i_{k,0}), S(i_{k,1}), \ldots, S(i_{k,\#I_k - 1}) \},$$

where $S(i_k)$ is an expression containing the dependence on the loop iterator i_k.

Let D be a set of data with a subjective mapping $T : D \mapsto D$ over it. Let us also introduce subjective mappings $p : I \mapsto D$ and $q : I \mapsto D$.

Let us consider the pairs of elements of the set D of the form $(a, v) \in D^2$, for those elements $a \in D$, for which there exist preimages of the set I under mappings p and q, respectively. Sets of all such pairs are denoted as

$$P = \{ (a, v) \,|\, \exists \overline{i} \in I : p(\overline{i}) = a \in D \,|\, v = T(a) \} \subset D^2$$

and

$$Q = \{ (b, w) \,|\, \exists \overline{i} \in I : q(\overline{i}) = b \in D \,|\, w = T(b) \} \subset D^2 .$$

Hereinafter, the pair $(a, v) \in D^2$ will be referred to as a data cell, $a \in D$ a cell address, and $v \in D$ a cell value. Mapping T provides a cell value by cell address. The set of all cells involved in the calculations is called memory.

Let $F : I \times D \mapsto D$ is the transformation mapping of the data set. Let us consider the nested loop of the following form:

$$\begin{aligned} & for \;\; i_N \in I_N : \\ & \quad for \;\; i_{N-1} \in I_{N-1} : \\ & \quad\quad \ldots \\ & \quad\quad\quad for \;\; i_0 \in I_0 : \\ & \quad\quad\quad\quad T \cdot q(\overline{i}) := F(\overline{i}, T \cdot p(\overline{i})), \end{aligned} \tag{1}$$

where symbol $\overline{i} = \{ i_0, i_1, \ldots, i_N \} \in I_0 \times \ldots \times I_N = I$ denotes a vector of iterators. The number of operators involved in the loop is called nesting depth.

Nesting depth for the loop (1) is equal to $N + 1$. Sets P and Q are called a set of initial data and a set of final data of the loop (1), respectively.

The loop iteration is a calculation that executes a loop for a certain fixed value of the vector of iterators $\overline{i}$, which acquires value from the set $I_0 \times \ldots \times I_N$. We assume that iterations are independent by data:

$$\forall \overline{i}, \overline{j} \in I, \overline{i} \neq \overline{j} : p(\overline{i}) \neq q(\overline{j}), q(\overline{i}) \neq q(\overline{j}), \tag{2}$$

i.e. no iteration of the loop changes the initial data of other iterations, and different iterations do not change the values of the same cells.

Having numbered the elements of the sets I_k according to the lexicographic order, let us proceed to the loop with a linear counter by performing the following substitution:

$$\mathit{for}\;\; i_N \in I_N \mapsto \mathit{for}\;\; 0 \leq i_n < \# I_n,$$

where counter i_n changes with a unit step. Let us perform decomposition of the loop nest. To achieve this, at first we perform the following substitution for each for operator:

$$\begin{array}{l} \mathit{for}\;\; 0 \leq i_n < \# I_n \mapsto \\ \mathit{for}\;\; 0 \leq s_n < S_n : \\ \quad \mathit{for}\;\; s_n \cdot L(\# I_n, S_n) \leq i_n < \min((s_n + 1) \cdot L(\# I_n, S_n), S_n), \end{array}$$

where

$$L(a,\ b) = \left\lfloor \frac{a}{b} \right\rfloor + 1 - \delta_{0,\, a \bmod b},$$

$\lfloor \cdot / \cdot \rfloor$ denotes the integer part of a quotient, δ is Kronecker symbol, S_n is the desired number of steps to subdivide the loop, $1 \leq S_n < \# I_n$. This transformation is commonly known as loop tiling [8]. After subdivision of subloops and regrouping, the loop takes the form, in which an internal loop is similar to the initial loop but of a reduced scale. We keep internal $N+1$ loops intact and group outer loops:

$$\begin{array}{l} \mathit{for} \quad 0 \leq e < \prod_{i=0}^{N} S_i : \\ \quad \vec{i} = g(e); \end{array} \tag{3}$$

Here $g(e)$ is a mapping that restores the vector of counters $\vec{i}$ and is constructed in the following way:

$$g_0(e) = e \bmod S_0,$$

$$g_k(e) = \left\lfloor \left(e - \sum_{j=k+1}^{N} g_j(e) \prod_{l=0}^{j-1} S_l \right) \Big/ \prod_{j=0}^{k-1} S_j \right\rfloor, \; 0 < k < N,$$

$$g_N(e) = \left\lfloor e \Big/ \prod_{j=0}^{N-1} S_j \right\rfloor,$$

$$0 \leq e \leq \prod_{k=0}^{N} S_k .$$

The obtained loop (3) maintains the sequence of the vector of counters equal to the sequence produced by the initial loop (1).

Let's denote the inner $N+1$ loops of the cycle (2) along with $g(e)$ as a $kernel(e)$. We intend to delegate the *kernel* execution to GPU and to run it concurrently thus diminishing the depth of the inner loop nest. As the GPU's memory space is isolated from the host's device one, we introduce *serialize* operation that is to prepare the input data required to perform calculations for the step *e* and *deserialize* operation to store the output data processed by GPU. The further implementation of these procedures is out of our scope and depends on the particular problem. Finally, we get:

$$\begin{aligned}
&for \quad 0 \le e < \prod_{i=0}^{N} S_i : \\
&\quad serialize(e, inputData, dataPull); \\
&\quad transfer2device(inputData); \\
&\quad kernel(e, inputData, outputData); \\
&\quad transfer2host(outputData); \\
&\quad deserialize(e, outputData, dataPull);
\end{aligned} \tag{4}$$

Iterations of the loop (4) can be distributed over concurrently running threads through involving several additional data exchange buffers. This approach could be applied to any distributed memory computational system, e.g. GPU cluster or heterogeneous cluster of any other computation empowered devices. To preserve equivalence in a sense of output results equality for the same given input data, Bernstein's [9] conditions must be met. This roughly means that iterations should not overwrite the other's iterations input data and should store their output data apart. The set of S_k ($0 \le k \le N$) is the transformation's tunable parameters that are chosen in a way to satisfy Bernstein's conditions and to optimize processing time that is to find a trade-off on time spent on data preparation, transfer, and kernel execution. These timings depend on the input and output data load size which is restricted by the total available amount of GPU's memory and hardware configuration parameters, such as input and output memory transfer rate and GPU compute capabilities.

The loop of the initial form is a form of notation of the data transformation sequence, given by the mapping F. However, a loop in this form is not suitable for execution by a parallel computing device, because for operator defines a sequential computational procedure for individual iterations. To perform the parallelization of the loop, it is necessary to properly distribute the iterations and corresponding data between the threads. The loop in the parallel form will be equivalent to the original loop if the above condition (2) holds. Let us prove this statement.

Let there is a procedure P performing data operations. Let us classify the memory cells used by this set of commands according to the mode of their use:

1) $W,\, W \subset D$ denotes a set of cells which are only read, i.e. $\forall d \in W$:

$$\forall \bar{i} \in I : q(\bar{i}) \ne d; \tag{5}$$

$$\exists \bar{i} \in I : p(\bar{i}) = d; \tag{6}$$

2) $X,\; X \subset D$ is a set of cells which are only written, i.e. $\forall d \in X$:

$$\exists \bar{i} \in I : q(\bar{i}) = d; \tag{7}$$

$$\forall \bar{i} \in I : p(\bar{i}) \ne d; \tag{8}$$

3) $Y,\, Y \subset D$ is a set of cells which are read and then written:

$$\forall d \in Y : \exists \bar{i}, \bar{j} \in I, \bar{i} \prec \bar{j} :\;\; p(\bar{i}) = d,\; q(\bar{j}) = d; \tag{9}$$

4) $Z,\ Z \subset D$ is a set of cells which are written and then read:

$$\forall d \in Z : \exists \overline{i}, \overline{j} \in I, \overline{i} \prec \overline{j} :\ q(\overline{i}) = d,\ p(\overline{j}) = d\ .$$

Let there are three procedures P_1, P_2 and P_3, the corresponding indices denote their sets of cells. Consider two algorithms that perform these procedures: the first calls the procedures sequentially one after the other, and the second performs the first two procedures simultaneously, and then the third. According to Bernstein [9], the following conditions must be met in order for the results of sequential and parallel algorithm calculations to coincide:

$$\left(W_1 \cup Y_1 \cup Z_1\right) \cap \left(X_2 \cup Y_2 \cup Z_2\right) = \varnothing,$$

that is, those memory cells that are read by the first procedure do not intersect with those cells that are written by the second procedure; condition symmetric to the previous one

$$\left(X_1 \cup Y_1 \cup Z_1\right) \cap \left(W_2 \cup Y_2 \cup Z_2\right) = \varnothing,$$

and

$$X_1 \cap X_2 \cap \left(W_3 \cup Y_3\right) = \varnothing,$$

that is, those cells that are used by both the first and the second recording procedures simultaneously will not be subsequently read without preliminary rewriting. Summarizing Bernstein's conditions for a set of procedures P_i, $0 \le i \le C$, we obtain:

$$\begin{gathered} \forall 0 \le i, j \le C: \\ (W_i \cup Y_i \cup Z_i) \cap (X_j \cup Y_j \cup Z_j) = \varnothing, \\ (X_i \cup Y_i \cup Z_i) \cap (W_j \cup Y_j \cup Z_j) = \varnothing, \\ X_j \cap X_j \cap (W_C \cup Y_C) = \varnothing. \end{gathered}$$

On the one hand,

$$d \in (W \cup Y \cup Z) \Leftrightarrow \overline{i} \in I : p(\overline{i}) = d\ .$$

On the other hand,

$$d \in (W \cup Y \cup Z) \Leftrightarrow \overline{i} \in I : q(\overline{i}) = d\ .$$

Therefore, the first two Bernstein conditions are equivalent to the following:

$$\forall \overline{i}, \overline{j} \in I : p(\overline{i}) \ne q(\overline{j})\ .$$

Since condition (8) requires

$$\forall \overline{i}, \overline{j} \in I : p(\overline{i}) \ne p(\overline{j}),$$

then

$$X_i \cap X_j = \varnothing,$$

and therefore the condition (9) also holds. Thus, conditions (7), (8) are equivalent to the combined conditions of Bernstein and ensure that the memory status of the device does not depend on the order of operations. Therefore, the statement is true.

Example. As an illustration, consider parallelizing Cholesky decomposition. Cholesky decomposition method [10] of solving linear equation systems replaces the initial system with a square matrix by two systems with lower and upper triangular matrices. Cholesky method is used for solving systems with symmetric positive-definite matrices. Such systems may arise as a result of the least square method for example. LU-factorization method is more common, and it results in solving triangular systems as well. The following loop solves the system $Ax = B$, where A is a lower triangular matrix of size $N \times N$ with Gaussian elimination method [11]:

```
$for\ \ i,\ 0 \le i < N:$
  $s = 0;$
  $for\ \ j,\ 0 \le j < i:$
    $s := s + A_{i,j} \cdot X_j;$
  $X_i := (B_i - s)\ /A_{i,i}.$
```

This form of the algorithm does not satisfy the property of iterations independency, as *i*-th iteration of the outer loop depends on X_{i-1} value obtained on the previous loop iteration. The properties don’t comply for the inner loop as well as the iterations are linked through s variable. Moreover, this algorithm is not scalable, as splitting the matrix makes it rectangular and requires another solution algorithm. These drawbacks of the transformed algorithm would not preserve the correctness of the results of the calculation. So let’s consider the algorithm of the following more convenient form:

```
$for\ \ i,\ 0 \le i < N:$
  $X_i := B_i\ /A_{i,i};$
  $for\ \ j,\ i \le j < N:$
    $B_j := B_j - A_{j,i} \cdot X_i.$
```

By introducing the parameter B_j, the inner loop of the previous algorithm was deprived of the dependency on the parameter s and thus created an opportunity to parallelize the inner loop by $N - i$ independent threads. Although this form of the algorithm maintains a dependency on the parameter X_i, and the conditions (7), (8) are not satisfied, as the matrix of the system remains square, the algorithm doesn’t change and therefore acquires scalability. The transformation of the loop is valid if synchronization by the iterator i is held. The parallelization of an upper triangular matrix can be carried similarly.

3. Program execution flow

In this section, we observe the execution flow of a program parallelized with the help of the proposed method. Consider the computing node that consists of one multicore CPU and one GPU. Modern GPUs support direct memory access technology thus allowing to carry out data transfer and kernel execution asynchronously. To optimize the data exchange process, dual buffering is involved. Four buffers at both host and device sides are involved — two for the input and two for the output data exchange. On the host side, computations are made by two threads executing $kernel$, $serialize$ and $deserialize$ procedures simultaneously. One of them serializes the input data and fills the input data buffer, then transmits the buffer to the GPU and launches the kernel, and the other receives output data buffer from GPU and deserializes it. Besides the calculations, GPU carries out bidirectional data transfers through the asynchronous data transfer mechanism. Computations are performed in three stages — initial, cyclic and finalizing.

At the starting point, data buffers are empty and GPU waits for the data transfer. It doesn't matter what thread will carry out the initial step, as all the operations are executed sequentially and asynchronous transfer mode is not involved. At the initial step, CPU serializes input data buffers of the first two iterations and transfers the buffer containing the first iteration data to the accelerator. After the initial step, the cyclic stage starts. The execution flow is shown in Fig. 1. The iteration number is given after the buffer's name. One step of the loop is divided into odd and even parts. Both odd and even parts of the first step skip deserialization as the host output buffers are empty yet. At the odd part of the first step, the accelerator-to-host transfer is omitted too. Meanwhile, the accelerator performs computations over a current buffer, host threads fetch data buffer from a previous step, deserialize penultimate step buffer, send input data buffer, and serialize buffer for the next step. In one step, two kernel launches are executed. After each part (odd or even) is finished, the processes are synchronized. Two final steps depend on the actual number of kernel launches. If the number of kernel launches is odd, the final step of the cyclic part excludes an even part and does not involve serialization and host-to-accelerator transfer, and the even part of the penultimate step skips serialization. Otherwise, if the number of launches is even, the last loop step is full, but the even part of the final step omits serialization. The final step deserializes output data buffer transferred at the last loop step and then fetches and deserializes the final output data buffer consequently finishing the computations.

4. Application of the proposed technique for constructing CUDA programs

In this section, we illustrate the application of the proposed technique to matrix multiplication and N-body problems. The time measurements were collected on the hardware system composed of Intel Core i5-3570 CPU (4 cores, 3.8 GHz) with 16 GB of host memory and NVIDIA Tesla M2050 GPU (3 GB of global memory, 384 bits memory bandwidth, connected through PCIe2.0 x8) running Ubuntu 16.04 host operating system.

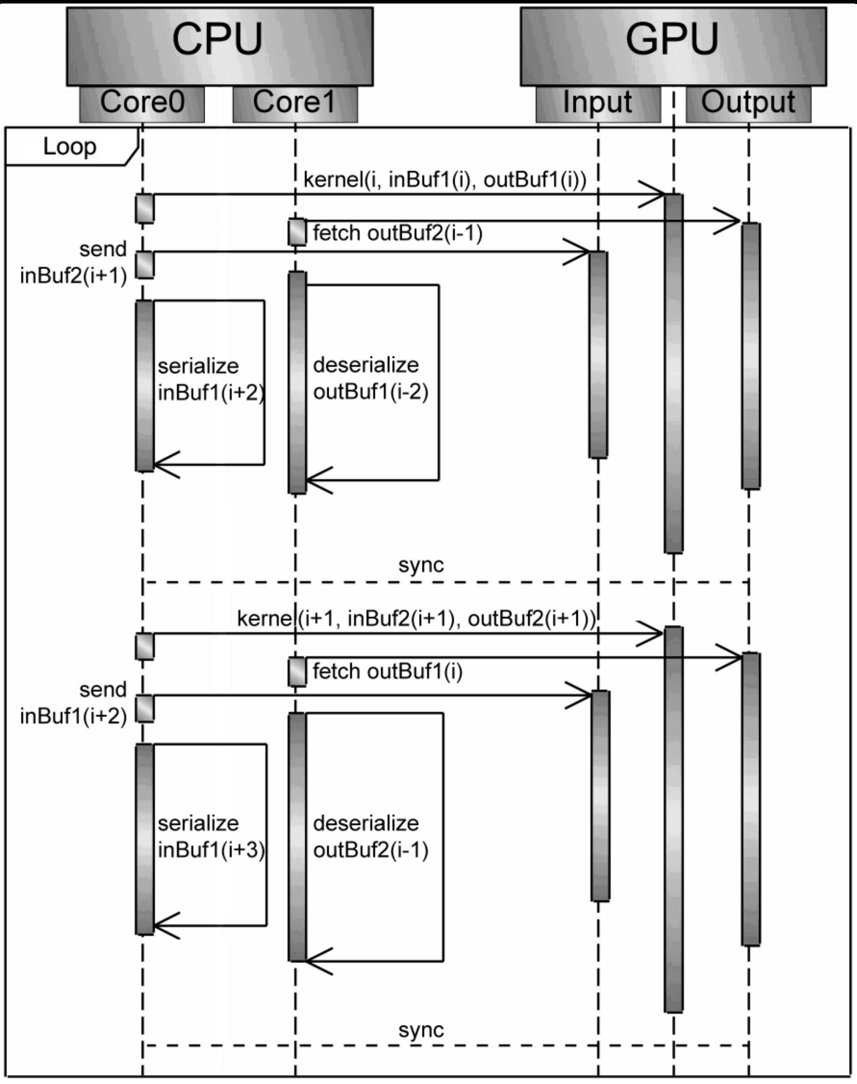


Fig. 1. Execution flow diagram of the cyclic stage of the concurrent program for the system of one accelerator and two control flow threads with four data exchange buffers.

A semi-automatic source-to-source code transformation tool based on the TermWare rewriting system [12] aiming to assist in constructing a new concurrent program was implemented. It takes the initial loop marked with pragmas, applies the transformations (3), and provides a template of the code of a new loop to be substituted. The remaining actions include serialization and deserialization routines implementation; the kernel could be implemented as well as generated by another tool and adapted in place.

The algorithm of the initial sequential matrix multiplication program involved a three-dimensional loop nest. It was transformed using the proposed technique and C-to-CUDA compilers PPCG and Par4All. Both of the programs generated by PPCG and P4A showed comparable results. After applying the slicing technique, the initial matrices were split into submatrices. The internal loop subdivision parameter S_0 was set to 1, the roles of parameters S_2 and S_1 is adjusting submatrices width. The schema with double data exchange buffering and two CPU threads were used. Even not involving GPU, adjusting the slicing number allowed to reach about 12 times acceleration over the initial loop due to CPU caching. For the GPU implementation, the parameterized PPCG generated kernel was used; the source codes of the constructed matrix multiplication program are available at GitHub [13]. The chart in Fig. 2 shows the constructed program’s timings and the timings of the program obtained with PPCG relatively to the matrix dataset size.

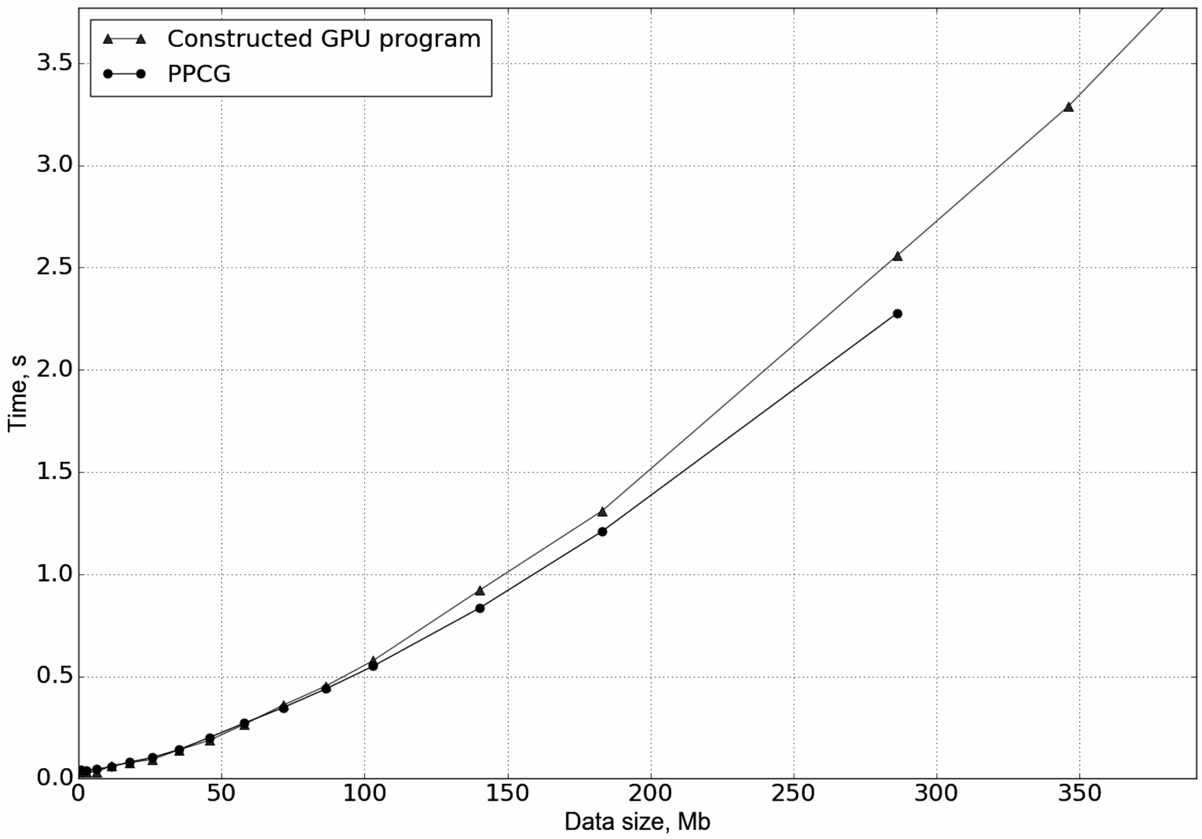


Fig. 2. The dependency of the execution time on the size of the input data for the concurrent PPCG-generated and constructed matrix multiplication programs

The relative acceleration of about 430 times in comparison to the sequential program executed on the CPU for the datasets of square matrices of the size of 5000×5000 single-precision floating point numbers was reached. It could be seen from the chart, that the PPCG generated program has achieved maximum data set size less than 300 Mb that is 10% of GPU's global memory available. This is because PPCG is limited to static memory usage, thus blocking to link programs with too large static arrays.

It is worth mentioning that involving two CPU threads is excessive as the part of serialization and deserialization is negligible compared to GPU kernel computation time, which can be seen from the GPU execution profile given in Fig. 3. Thus, involving just one thread instead won't decrease performance substantially, however, two concurrent threads are required to avoid gaps in kernel launches and to gain maximum performance from the GPU.

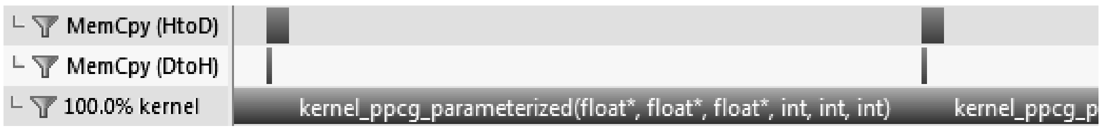

MemCpy (HtoD)
MemCpy (DtoH)
100.0% kernel
kernel_ppcg_parameterized(float*, float*, float*, int, int, int)
kernel_ppcg_p

Fig. 3. The fragment of the profile of matrix multiplication execution

Another application examined was a predictor routine from the N-body problem with the predictor-corrector time-iterative [14] algorithm. The model of the system consists of a set of particles that interact pairwise PPCG applying caused a slowdown effect and led in about 500 times decrease in performance in comparison to the sequential CPU implementation. As for the constructed program with a self-implemented kernel not involving shared memory usage, the relative CPU to GPU acceleration at the selected data size range reached 13 times. The plot in Fig. 4 shows the dependency of sequential CPU and transformed GPU programs execution time on the data size that is scaled by the alteration of the number of particles N. The timings are measured for the one time-step of the prediction routine. The memory limit was not reached as it would take approximately 30 years to process one time-step of the algorithm for a fully loaded GPU that was used in the experiment, however, the applicability of the technique is confirmed.

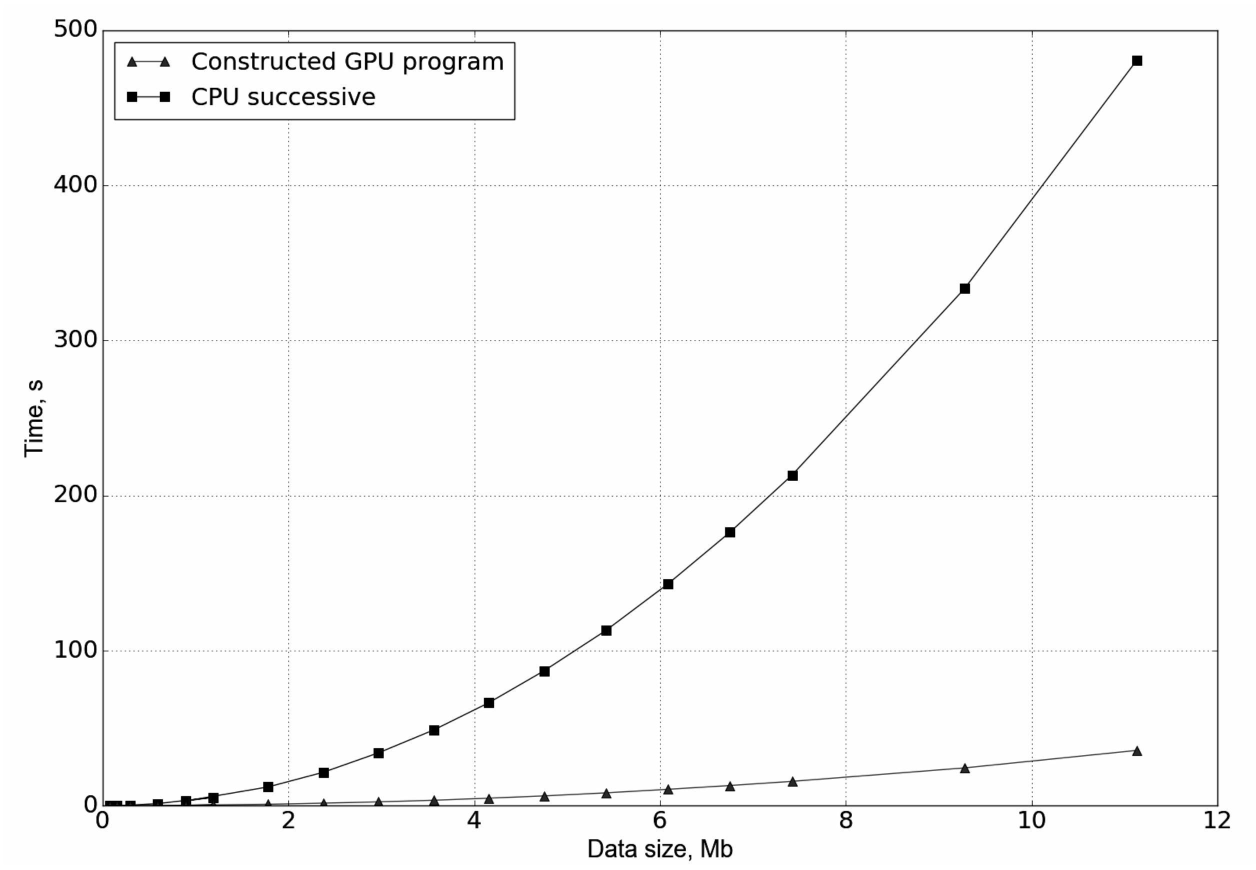


Fig. 4. The dependency of the execution time on the size of the input data for the sequential and constructed concurrent N-body programs

Thus, the difference in the constructed multiplication and N-body programs consists in the kernel, serializer, and deserializer implementation while the control flow structure remains identical.

5. Conclusions

A technology for semi-automated parallelization of nested loops for GPUs was proposed. The advantage of the proposed technology is the ability to process data volumes that exceed the GPU memory size and simultaneously use multiple GPUs. The usage of the technology is illustrated by the development of CUDA programs for solving matrix multiplication and N-body problems. An assistant semi-automatic code transformation tool was implemented. Further work is associated with developing unified methods and tools in loop parallelization.

References

1. Harris M. J. Real-time cloud simulation and rendering : a dissertation for a Ph.D. degree in the department of computer science. Chapel Hill, NC : University of North Carolina, 2003. 151 p.
2. PIPS: Automatic Parallelizer and Code Transformation Framework. PIPS4U : website. URL: http://pips4u.org (accessed: 17.08.2020).
3. Polyhedral parallel code generation for CUDA / Verdoolaege S. et al. ACM Trans. Architec. Code Optim. 2013. Vol. 9, № 4, Art. 54. P. 1–23.
4. Split tiling for GPUs: automatic parallelization using trapezoidal tiles / T. Grosser et al. GPGPU-6 : Proc. 6th Workshop on General Purpose Processor Using Graphics Processing Units, March 16, 2013. New York : Association for Computing Machinery, 2013. P. 24–31.

5. Automatic parallelization of tiled loop nests with enhanced fine-grained parallelism on GPUs / P. Di et al. Proc. 41st International Conference on Parallel Processing, September 10–13, 2012. Washington, D.C. : IEEE Computer Society, 2012. P. 1–12.
6. Automated design of parallel programs for heterogeneous platforms using algebra-algorithmic tools / Doroshenko A., Beketov O., Bondarenko M., Yatsenko O. ICTERI 2019 : Post Proc. 15th Int. Conf. “ICT in Education, Research and Industrial Applications. Integration, Harmonization and Knowledge Transfer”, June 12–15, 2019. CCIS. Vol. 1175. Cham : Springer, 2020. P. 3–23.
7. Doroshenko A., Beketov O. Large-scale loops parallelization for GPU accelerators. ICTERI 2019 : Proc. 15th Int. Conf. “ICT in Education, Research and Industrial Applications. Integration, Harmonization and Knowledge Transfer”, June 12–15, 2019. Cham : Springer, 2019. P. 82–89.
8. Wolfe M. More iteration space tiling. Supercomputing’89 : Proc. ACM/IEEE Conference on Supercomputing, November 1989. New York : Association for Computing Machinery, 1989. P. 655–664.
9. Bernstein A. J. Analysis of programs for parallel processing. IEEE transactions on electronic computers. 1966. Vol. EC-15, № 5. P. 757–763.
10. Dereniowski D., Kubale M. Cholesky factorization of matrices in parallel and ranking of graphs. Proc. 5th International Conference on Parallel Processing and Applied Mathematics, September 7–10, 2003. Berlin : Springer, 2004. P. 985–992.
11. Gentle J. E. Gaussian elimination. Numerical Linear Algebra for Applications in Statistics. Berlin : Springer, 1998. P. 87–91.
12. Doroshenko A., Shevchenko R. A rewriting framework for rule-based programming dynamic applications. Fundamenta Informaticae. 2006. Vol. 72, № 1–3. P. 95–108.
13. GitHub repository. github : website. URL: https://github.com/o-beketov/matmul (accessed: 17.08.2020).
14. Aarseth S. J. Gravitational N-body simulations. Cambridge : Cambridge University Press, 2003. 430 p.

UDC 004.383.3, 004.056

ALIREZA MIRATAEI,
HANA KHALIL,
MARKOVSKYI O.

PROTECTED DISCRETE FOURIER TRANSFORM IMPLEMENTATION ON REMOTE COMPUTER SYSTEMS

Annotation: Effective method of the discrete Fourier transform acceleration with the use of cloud computing is theoretically substantiated and developed. The reigning feature of the suggested method is homomorphic encryption of the input signals, which provides efficient protection during the remote calculation. It has been shown theoretically and experimentally that the proposed method allows one to 1-2 orders of magnitude to speed up the processing of signals while maintaining their confidentiality. The proposed method can be applied for effective signal stream processing in clouds.

Keywords: discrete Fourier transform, cloud computer systems, remote signal processing, homomorphic encryption.

1. Introduction

The cloud technologies development radically changed the organization of computer information processing. In response to cloud systems, a wide range of users obtain access to powerful computing systems. The emergence of such opportunity allows solving qualitatively new classes of tasks, multiply speeds up solution of a wide range of tasks with high computational complexity. Wide variety of applied tasks related to the interface implementation between computer systems and the outside world is used digital signal processing methods, including the discrete Fourier transform. Most of these tasks are performed in real time, that sets strict conditions for the fast discrete Fourier transform implementation. One of the most innovative lines of attack on the problem is using the cloud technologies capabilities [1]. Remote multiprocessor computer systems open wide possibilities for parallelization of the discrete Fourier transform, respectively accelerating their computing implementation.

For the vast majority of real-life signal processing systems, it is important to maintain confidentiality of the signals and their processing results. This factor significantly restricts the cloud technologies using for remote signal processing. These determinates the necessity for development of the homomorphic signals encryption methods in their remote processing in clouds [2].

Thus, the scientific task of developing the method for realization of the discrete protected Fourier transform on remote computer systems is relevant for the modern stage in the evolution of information technology.

2. Problem statement and review of methods for its solution

Digital signal processing is one of the most important tasks of modern computer technologies. Actually, the digital signal processing implements the interface between the real world and the computer systems [3]. For most practical uses, the problem of signal processing needs to be carried out in real time. This dictates hard conditions for the time-consuming implementation of computing that implement digital signal processing and, in particular, the discrete Fourier transform. For this purpose, specialized chips are developed and serially produced, which in favor of parallelization of computing at the hardware level solve the problem of Fourier transform accelerating [4]. Wide range of related hardware tools are developed by well-known firms like Texas Instruments INC, Motorola, Intel and Analog Devices. In particular, Texas Instruments INC produces series of fast Fourier transform processors that characteristically differ in speed performance and functional peripherals.

Fourier transform is one of the most used transforms for function decomposition. This transform allows to get the signals of different origin (images, voice signals, radio signals etc.) organized as a set of real numbers. It allows to compare, analyze, and read the signals.

Any kind of signals, that are specified by the set of measurements from identical periods of time, could be written as a sum of sinusoids. Sinusoidal functions have such characteristics: amplitudes (A1, A2, …), frequencies (ω, 2·ω, 3·ω, 4·ω …) and phases (φ1, φ2, φ3…).

As for input array (x0, x1, … xn-1) for Discrete Fourier Transform (DFT) we use calculated signal data through fixed time periods. Array from n complex elements (y0, y1, … yn-1) is output data of DFT, its components have amplitudes and phases of sinusoids, their sum recreates initially set input signal. If we take gi as real yi component, qi as imaginary yi component, Ai as spectral amplitude and φi as phase transport of yi then will be used next formulas [3]:

$$\forall i \in \{0,1,...,n-1\}: d_i = \frac{1}{n} \cdot \sum_{j=0}^{n-1} x_j \cdot \cos\frac{2 \cdot \pi \cdot i \cdot j}{n}, \quad q_i = \frac{1}{n} \cdot \sum x_j \cdot \sin\frac{2 \cdot \pi \cdot i \cdot j}{n} \quad . \tag{1}$$

Using the obtained values (from formulas (1)) of the real and imaginary components of the sinusoidal signal with the frequency i·ω we can calculate the amplitude of this signal, as well as its phase shift using the formulas [3]:

$$\forall i \in \{0,1,...,n-1\}: A_i = \sqrt{q_i^2 + g_i^2}, \quad \varphi_i = arctg(\frac{g_i}{q_i})^2 \quad . \tag{2}$$

For most practical applications of signal processing, the calculation according to the formula (2) is not performed [5], therefore the Fourier transform time is determined by the calculations described by the formula (10. It is obvious that the number of floating-point multiplication and floating-point additions for the implementation of the formula (1) is equal n2. This means that as the number of n increases, the number of floating-point multiplication and addition operations increases rapidly. Consequently, it is necessary to organize a remote calculation of formulas (1) on multiprocessing computing systems to reduce the execution time of the DFT. From the structure of the formulas (1) it is obvious that they can be calculated simultaneously on n processors.

Cosines and sinuses do not depend on the values x0, x1, ... xn-1 and they can be considered as constant numbers, the values of which depend only on indexes. Therefore, the components of formulas (1) can be calculated and organized in advance as a set of coefficients, which, in consequent researches, can be considered as permanent. The calculation of coefficients can be carried out in accordance with the following formulas [4]:

$$\forall i, j \in \{0,1,...,n-1\}: \quad a_{ij} = \frac{1}{n} \cdot \cos\frac{2 \cdot \pi \cdot i \cdot j}{n}, \quad c_{i,j} = \frac{1}{n} \cdot \sin\frac{2 \cdot \pi \cdot i \cdot j}{n} \quad . \tag{3}$$

If you use constant coefficients that are calculated by formulas (3), the basic formulas (1) can be simplified to the following form [4]:

$$\forall i \in \{0,1,..,n-1\}: g_i = \sum_{j=0}^{n-1} a_{ij} \cdot x_j, \; q_i = \sum_{j=0}^{n-1} c_{ij} \cdot x_{ij} \quad . \tag{4}$$

It needs to organize a secure Fourier transform: encrypt the values x0, x1, ... xn-1, which are transmitted to the cloud. Consider that in the cloud values v0, v1, ... vn-1 and w0, w1, ... wn-1 are being calculated using formulas (4) and go back to the user who has to restore the real values g0, g1, ... gn-1 and q0, q1, ... qn-1.

As noted above, the implementation of DFT on remote computer multiprocessor systems offers great possibilities for parallelization of computational process. Theoretically, when using an unlimited number of processors, multiply operations can be performed simultaneously by 2·n2 processors.

In addition, to form 2·n sums according to formulas (4) it needs to perform log2n sum operations on 2·n processors. Doing so, theoretically minimal execution time T0 of DFT on remote computer multiprocessor systems are determined by the formula:

$$T_0 = t_a \cdot \log_2 n + t_m \,, \tag{5}$$

where tm is the time for performing the floating-point multiplication and ta is the time for performing the floating-point adding.

When performing a DFT on single processor, then the Fourier transform is usually characterized by the number of multiples and additions that needs to perform during the process of the transformation. Number of multiplies for implementation of DFT is n2, and the number of additions is n·(n-1) [6]. The numerical value of T1 time execution of DFT on a single processor is determined by the following formula:

$$T_1 = t_m \cdot n^2 + t_a \cdot n \cdot (n-1) \quad (6)$$

Considering the fact that for modern processors, the time to perform an floating-point multiplication operation is many times greater than the time when the floating-point addition operation is performed, the attraction of remote multiprocessor computing systems with use of the cloud technologies capabilities allows speeding up the calculations in ξ times when DFT is performed. Numerical value ξ can be determined by substitution (5) and (6) to the following formula:

$$\xi = \frac{T_1}{T_0} \approx \frac{t_m \cdot n^2}{t_m + t_a \cdot \log_2 n} \quad (7)$$

If we take into account that the execution time of the multiplication is about 30 times longer than the execution time of the addition, then, according to the formula (7) when n = 16 the value of ξ is 226, and when n = 256 ξ is equal to 51739.

In practice, digital signal processing DFT is often implemented in the form of fast Fourier transform, which allows to reduce the number of multiplication and addition operations by using the properties of symmetry of the Fourier transform coefficients. When using a fast Fourier transform (FFT), amount of floating-point multiplication operations same as amount of floating-point addition operations are both equal to n2/2 [7].

There are two important conclusions to draw from:

The usage of remote multiprocessor computing systems to speed up the calculating of DFT gives a significant effect in terms of accelerating the computations, that is important for practical use which function in real-time mode.

The efficiency of using remote multiprocessor computing systems in frames of cloud technologies to accelerate the computational implementation of DFT is increasing as the size of the transformation is increased.

To realize these benefits provided by the modern cloud technologies, we need to develop a method that makes it impossible to get unauthorized access to the numerical values of the signal simples while running DFT over the signals on remote and out of user control computer systems.

One of the powerful reserves that can be used for the faster calculation of the discrete Fourier transform is cloud technologies. They provide the user with the global network access to virtually unlimited computing resources. These opportunities are actively used in solving a wide range of important tasks.

One of the main cloud technologies disadvantages, that significantly restricts their use, is the possibility of unauthorized access to user data during their processing the data on the remote computing capacities. A significant number of works deal with the problem of user data protecting on distance processing systems in recent years [7, 8, 9]. The fact of the matter is that there is no single approach to data protection in the procedure of their processing. Accordingly, there can’t be a single homomorphic algorithm for data encryption, on which calculations on remote smoked capacities are performed.

This means that the homomorphic signal encryption algorithm should depend on the nature of their remote processing operations.

Accordingly, almost all homomorphic encryption researches solve the problem of data protection only for certain classes of computing problems [10].

The above analysis of the computational procedures that is fundamental for the discrete Fourier transform showed that for the algebraic basis of the homomorphic signal encryption can be used additive or multiplied operations, and operation of modulatory arithmetic in a traditional algebra.

When using modular arithmetic [11], each of the signal X samples: x0,x1,…,xn-1 is added to the additive mask r0,r1,…,rn-1 in such a way that the remote system is transmitted to the set of additive disguised samples: x0+r0, x0+r1,…,xn-1+rn-1. When this happens the values of masks are chosen in order that (r0+r1+…+rn-1) mod Z = 0, where Z is the private key of homomorphic encryption.

Then, with the remote execution of a discrete Fourier transform according to formula (4), the obtained results can be represented in the form:

$$\forall i \in \{0,1,...,n-1\}: g_i' = \sum_{l=0}^{n-1}(x_{il}+r_l)\cdot a_{il} = \sum_{l=0}^{n-1} x_{il}\cdot a_{il} + \sum_{l=0}^{n-1} r_l \cdot a_{il} =$$
$$= g_i + \beta\cdot Z; \quad q_i' = \sum_{l=0}^{n-1}(x_{il}+r_l)\cdot c_{il} = \sum_{l=0}^{n-1} x_{il}\cdot c_{il} + \sum_{l=0}^{n-1} r_l\cdot c_{il} = q_i + \gamma\cdot Z \quad (8)$$
,

where β and γ are an integer. In other words, from the formula (8) it follows that the result of the remote processing is the sum of the real component of the spectral representation and some number that is targeted is divided without a remainder into the private key Z. Accordingly, the decryption of the received data is carried out in the following form:

$$\forall i \in \{1,2,...,n\}: g_i = g_i' \bmod Z,\ q_i = q_i' \bmod Z \quad (9)$$
.

The disadvantage of the described well-known method is the high complexity of the decrypting implementation. In fact, for each of this spectral representation component there must be performed a division operation for finding a remainder according to the formula (9).

The work [12] introduces the method for encryption of signal, which remotely performs the discrete Fourier transform. In this case, the data encryption uses replication operations based on modular exponential derivation. This allows to flexibility in adjust the safety level of remote discrete Fourier transform implementation. In addition, in the development proposed the effective mechanisms of controlling the functional correctness of the remote discrete Fourier transform in the uncontrolled computing platforms. At the same time, using modular exponential as a mechanism for encryption markedly complicates the choice of keys and causes the considerable computing complexity of discrete Fourier transform realization, which is several times greater than the complexity of discrete Fourier transform realization with formula (4). Generations of keys is significant problem in the proposed method, which also requires significant computing resources. Supposing that the keys are used repeatedly, it results a security level decreasing, due to the fact that it opens to the attacker wider abilities to crack the cipher code. Usage of single use key in the proposed method leads to an increase in computing resources costs for discrete Fourier transform.

3. Purpose and objectives of research

The aim of the research is to increase the efficiency of the protected implementation of the discrete Fourier transform on remote computer systems.

Research objectives are determined by the aim and are as follows:

- Investigation of possible options for homomorphic encryption of signal samples before their transmission to the cloud, as well as decryption of the results obtained. Choosing the most effective option.
- Development of a method for secure implementation of the discrete Fourier transform on remote computer systems based on additive homomorphic encryption of signal samples.
- Theoretical and experimental study of the effectiveness of the method of secure implementation of discrete transformations in the cloud both in terms of the level of security and in terms of the achievable processing acceleration.

4. The method of the protected discrete Fourier transform implementation on remote multiprocessor computer systems

Since the operations that make up the discrete Fourier transform can be reduced to addition operations, the most natural way to hide signals by the sample is to add the components of the secret key to them. So, the basic idea of the proposed method is that each encrypted signal sample δj, j∈{0, 1,…,n-1} can be presented as a sum or deviation of two values: real signal sample and components of the secret key:

$$\forall j \in \{0,1,...,n-1\}: \delta_j = x_j + b_j, \tag{10}$$

where B= {b0,b1,…,bn-1} is the value of the carrier signal, whose components make up the secret key. To increase the level of protection of samples of real signals during their remote processing, it is proposed to use several carrier signals, which form a set Ω of κ elements Ω = {B1, B2, …, Bκ}. This carrier signal is correlated with the values of the results of the Fourier transform of: Θ = {V1, V2, …,Vκ} and Ψ = {W1, W2, …, Wκ}. In this way, the numerical values of sets items Θ and Ψ are calculated according to the following formulas:

$$\forall i \in \{0,1,...,n-1\},\ l \in \{1,2,...,\kappa\}: v_{li} = \sum_{j=0}^{n-1} a_{ij} \cdot b_{lj},\ w_{li} = \sum_{j=0}^{n-1} c_{lj} \cdot b_{lj}. \tag{11}$$

The values of the bias signals that make a set Ω are chosen in such a way for the most distort of the dominant character type of useful signal X. For each area of practical usage of digital signal processing, the characteristic properties of the signal spectrum can be identified. Accordingly, the carrier signals must be selected in such a way as to distort the spectrum of these signals as much as possible. After selecting the set Ω the components of sets Θ and Ψ are calculated using formulas (11) with the image of the results in the memory. It should be noted that these calculations are realized in a non-critical time mode using the user's computing platform.

The proposed method of homomorphic encryption assumes the following sequence of actions

When the DFT is carried out over the signal X, set by the n values of its simples (x1, x2, …, xn) follows out the next action order:

Randomly, one of the carrier signals is selected from a set of Ω. Without losing the generalization, you can assume that the number of the chosen carrier signal in set Ω is h.

Bias values are calculated δ0, δ1, …, δn-1 of the X signal simples relatively to the selected Bh carrier signal according to the following formula (10).

The resulting values δ0, δ1, …, δn-1 are sent to a remote computer system.

A remote computer system performs a DFT for over a set of simples δ0, δ1, …, δn-1 according to formulas:

$$\forall i \in \{0,1,2,...,n-1\}: s_j = \sum_{j=0}^{n-1} a_{i,j} \cdot \delta_j,\ z_j = \sum_{j=0}^{n-1} c_{i,j} \cdot \delta_j. \tag{12}$$

The remote computer system returns to the user the calculated values of s0, s1,…, sn-1 and z0, z1, …,zn-1.

The user restores the real values of the X signal spectral representation components: g0, g1, …,gn-1 and q0, q1, …,qn-1 with the following transformations:

$$\forall j \in \{0,1,...,n-1\}: g_j = s_j - v_{hj},\quad q_j = z_j - w_{hj}. \tag{13}$$

The correctness of the results can be proved by the following calculations:

$$\forall i \in \{0,1,\ldots,n-1\}:$$

$$\begin{aligned} g_i &= \sum_{j=0}^{n-1} \alpha_{ij} \cdot x_j = \sum_{j=0}^{n-1} a_{ij} \cdot (\delta_j - b_{hj}) = \sum_{j=0}^{n-1} a_{ij} \cdot \delta_j - \sum_{j=0}^{n-1} a_{ij} \cdot b_{hj} = s_i - v_{hi} \\ q_i &= \sum_{j=0}^{n-1} c_{ij} \cdot x_j = \sum_{j=0}^{n-1} c_{ij} \cdot (\delta_j - b_{hj}) = \sum_{j=0}^{n-1} c_{ij} \cdot \delta_j - \sum_{j=0}^{n-1} c_{ij} \cdot b_{hj} = z_i - w_{hj} \end{aligned} \tag{14}$$

.

It can be seen from the transformations (14) that the term-by-term results as a result of the proposed computational procedure correspond to the results of the discrete Fourier transform over the original signal samples.

When using the developed procedure on the side of the remote computer system there is only a set of signals X (δ0, δ1, …, δn-1) relatively to the selected carrier signal which without knowledges of a given carrier signal cannot be recovered. Because the Bq carrier signal counts are calculated and stored on the user's side, signal X is practically impossible to recover on the side of the system that performs the remote DFT procedure.

The security level of the signal, which is transmitted in the form of an array of samples to the remote computer system, is largely determined by the choice of the carrier signal. The choice of the carrier signal should be carried out in such a way that the selection of the signal samples x0, x1, x2, ..., xn-1 from the samples δ0,δ1,δ2,…,δn-1 of the mixture of signals and the spectrum components s0,s1,s2, …,sn-1 and z0,z1,z2, …,zn-1 required the greatest possible resources.

With a random selection of samples b0,b1,b2, …,bn-1 of the carrier signal, its spectral components have approximately the same values (white noise). This makes it easier to fit the spectrum of the real signal x0,x1,x2, …,xn-1 by removing white noise.

Experimental studies have shown that more efficient is the random generation of the components s0,s1,s2, …,sn-1 of the real component and z0,z1,z2, …,zn-1 of the imaginary component of the spectrum. In this case, the samples of the carrier signal are calculated by the inverse discrete Fourier transform.

5. Evaluation of the developed method effectiveness

Efficiency assessment should include two conditions:

- Security level against illegal access attempts on signal reports, which is determined by the volume of resources required to violate the protection.
- The calculation acceleration coefficient ξ, which is determined by the formula (7). In this formula (7), the value of T0 is replaced by T0′ - the total time for performing the discrete Fourier transform on a remote computer system, which is determined by formula (5) and the time for performing encryption and decryption operations - Tcd: T0′= T0 + Tcd.

As a result of the proposed remote DFT procedure on the user's side, in the process of implementing p.2, we get only n additional operations, as part of formula calculations (10), as well as in the implementation of p. 6 we get 2·n subtractions operations according to the formula (13). All other DFT operations are carried out on a remote computer system. Thus, the numerical value of the time spent on the performing encryption and decryption operations on the user platform Tcd is determined as Tcd = 3·n·ta. Correspondently T0′= T0 + Tcd = tm + ta·(3·n +log2n).

In accordance with formulas (7) the proposed method acceleration factor ξ′ will be calculated as follows:

$$\xi' = \frac{T_1}{T_0'} = \frac{t_m \cdot n^2 + t_a \cdot n \cdot (n-1)}{t_m + t_a \cdot (3 \cdot n + \log_2 n)} \tag{15}$$

.

For example, if we take into account that the execution time of the multiplication is about 30 times longer than the execution time of the addition, the numerical value of the acceleration factor ξ' with the proposed method of remotely protected DFT, calculated by formula (15), is $\xi'= 34.6$ for n=8 and $\xi'= 1206$ for n=128.

The real value of the acceleration coefficient ξ'' for the implementation of the discrete Fourier transform when using the proposed method is somewhat lower. This is because the user can apply the Fast Fourier Transform technology. This allows you to significantly reduce the value of T1. The formula for calculating the acceleration coefficient is as follows:

$$\xi'' = \frac{T_1'}{T_0'} = \frac{t_m \cdot n \cdot \log_2 n + t_a \cdot n}{t_m + t_a \cdot (3 \cdot n + \log_2 n)} . \tag{16}$$

Calculated by formula (16) the acceleration factor ξ'' with the proposed method is $\xi''= 26$ for n=8 and $\xi''= 64$ for n=128.

Formulas (15,16) did not take into account the time of data delivery to remote computing systems and the time of returning the results to the user. Full consideration of the corresponding times reduces the real value of the acceleration coefficient ξ. However, in practice it is usually not a single signal that is processed, but a stream of signals. At the same situation, the time of transporting data over the network practically does not affect the rate of signal processing and the real data are close to theoretical estimates by (15,16).

The level of data protection provided by the proposed method is determined by the ability of an attacker to reconstruct them at the place of their processing, that is, on a remote computer system. This means that from the set of samples δ0,δ1,…,δn-1 it is necessary to restore set of values of the true signal samples x0,x1,…,xn-1. To do this, he needs to know the set of carrier signal samples b0,b1,…,bn-1. If a different carrier signal are used for each processed signal, then the task of reconstructing the useful signal requires of the resources, the costs of which make this process economically unfeasible for the vast majority of practical applications.

However, in reality, the number of reference signals is limited by the value κ. That can do possible the situation in which the same carrier signal was used for homomorphic encryption of several signals. If the attacker does not know any set of samples of the true signal, then it is practically impractical to reconstruct the rest of the signals.

The situation changes if an attacker in one way or another can obtain information about the value of the X0 signal samples. This possibility can arise if an attacker illegally connects to one of the terminal devices that generate the signal. For example, an attacker can illegally connect to the output of one of the surveillance cameras. In this case, an attacker using set δ0,δ1,…,δn-1 can restore the samples of one of the carrier signals: b0,b1,…,bn-1.

Accordingly, an attacker will be able to recover that part of the signals that is encrypted using the specified carrier signal. This attack will be successful only if the attacker is able to match the samples x0,x1,…,xn-1 of the signal X0, which he became aware of with the encrypted sample δ0,δ1,…,δn-1 .

However, when using modern cloud technologies that redistribute processing tasks across different computer systems, this is difficult to do. Therefore, the attack described above will be more effective if the attacker has control over the data transmission channel. To avoid this situation, additional encryption of the sample streams using streaming or symmetric block ciphers can be recommended.

An effective way to increase the level of data security, over which the discrete Fourier transform is performed remotely, is the use of linear combinations of reference signals for homomorphic encryption.

6. Conclusion

As a result of the research, a new method of the protected discrete Fourier transform implementation on remote computer systems has been proposed. The peculiarity of the developed method is the use of additive encryption. A sequence of pre-formed carrier signals is used as an additive mask. Real and imaginary components that are calculated on the user’s computer are used to decrypt the values obtained from the remote system.

As a result of the research, it was proved that parallelization of the calculations of the discrete Fourier transform reduces the processing time of information by one-two oders. Theoretically and experimentally proved the effectiveness of encryption: the resources of modern computer systems do not correspond to the level that is needed to perform the selection of additive mask.

The proposed method can be used effectively in systems that use real-time signal processing.

References

1. Armbrust M. A view of cloud computing / M. Armbrust, A. Fox, R. Griffith, R. Katz, A.A. Konwinski // International Journal Computer Technology.-2013.- No.4.- PP.50-58.
2. Bianchi T. On the Implementation of the Discrete Fourier Transform in the Encrypted Domain / T. Bianchi, A. Piva, and M. Barni // IEEE Transactions on Information Forensics and Security,-2009.-Vol. 4, - no.1, - PP. 86–97.
3. Nakonechny A.J. Signal processing using modern cloud technologies / A.J. Nakonechny, P.G. Pazan // Visnik of the National University "Lviv Polytechnic", series Automation, measurement and control.-2015.- Vol. 821.- PP.8-16.
4. Texas Instruments. TMS320F2812 Digital Signal Processor. Implementation Tutorial.-2013.- 122 P.
5. Markovskyi O.P. Secure Modular Exponentiation in Cloud Systems/ O.P. Markovskyi, N. Bardis, S.J. Kirilenko // Proceeding of the Congress on Information Technology. Computationnal and Experimental Physics (CITCEP 2015), 18-20 December 2015, Krakow. Poland. – PP.266-269.
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. Guduguntla Sandeep, S.P.V.Subba Rao. Radix 4 Fast Fourier Transform Using New Distributive Arithmetic// International Journal of Recent Technology and Engineering.- 2019.- vol. 8.- pp.11-15.
8. Xia Z.. Towards privacy-preserving content-based image retrieval in cloud computing / Z. Xia, Z, Y. Zhu, X. Sun, Z.Qin, K. Ren // IEEE Trans. Cloud Comput. – 2018.- No.6,- PP. 276–286.
9. Hamdi Hassen. Distributed Fast Fourier Transform (DFFT) on MapReduce Model for Arabic Handwriting Feature Extraction Technique via Cloud Computing Technologies / Hamdi Hassen, Khemakhem Maher // IEEE Jordan Conference on Applied Electrical Engineering and Computing Technologies (AEECT).- 2014.- PP.33-39.
10. Li L. Separable Data-Hiding Scheme for Encrypted Image to Protect Privacy of User in Cloud / Lin Li, W.Lifang, S. Tun-Qing, C. Chin-Chen // Symmetry.- 2019,- No.11.- PP. 1-14.
11. Markovskyi O.P. The method of accelerated secure image filtering on remote computer systems / O.P. Markovskyi, I.O.Gymenuk, Alireza Mirataei, J.I. Turoshanko, M.O. Voloshuk // Telecommunication and information technology.- 2019,- Vol.65.-no.4.- PP.99-110.
12. Bujbarova M.F. Method for protected Fourier transforms on remote distributed computer systems / M.F.Bujbarova, Y.M. Vynogradov, V.Y. Priymak // Visnik of National Technical University of Ukraine "KPI" Informatics, Control and Computer Engineering.- 2016.- Vol. 65,- PP.64-71.

UDC 004.934

MARKOVSKYI O.,
SHEVCHENKO O.,
HUMENIUK I.

HASH SEARCH ORGANIZATION IN E-DICTIONARIES USING BLOCK CIPHERS

The article is devoted to the problem of developing high-speed electronic dictionaries for systems computer translation.

The method of organizing a high-speed electronic dictionary based on an ideal hash addressing, where the cryptographic cipher block acts as a hash transformation is proposed. This method was developed taking into account the multilevel memory structure of modern computer systems.

It was testified theoretically and experimentally that the proposed organization of electronic dictionaries guarantees at least twice higher search rate compared to known technologies.

Keywords: hash-search, e-dictionary, contextual search, perfect hash-addressing, computer translation.

1. Introduction

Over the past two decades, the significant changes in research and production work centers in the world has bees occurred. Until recently, there was the tendency of transfers of production strenth to the countries of the Far East. But nowadays these countries are achieving main positions in various fields of science and, above all, in technological researches.

This rules to a further increase of exchange scientific and technical information of East and West. But, there is one significant obstacle to the flow of further increasing information sharing. This obstacle is the language barrier between East and West. Mainly, it is caused by the linguistic differences between East and West languages [1].

Experience has shown that the most promising way to overcome the language barrier in the question of scientific and technical information exchange is the usage of highly-efficient computer translation technologies.

Progress in the field of computer technologies made the background for the successful realization of this approach [2]. The recent progress in artificial intelligence make possible to reach the semantic relevancy of translation, made by computer translation systems. These technologies are based on the analysis of a vast amount of translation options, which make necessary the widespread usage of electronic dictionaries. In such order, as more efficient electronic dictionaries are, as more efficient work of machine translation tools provides. Thus, is necessary to develop some new methods of contextual search in electronic dictionaries.

Therefore, the scientific task of increasing the speed of searching for words in electronic dictionaries is relevant at the present stage of developing information technologies.

2. Problem statement

To reach this aim, an analysis of electronic dictionaries` models has proceeded. The model of context-search dictionary requires the storage of sets of contextual language constructs to each search key-word of the dictionary. In this order, there are two types of information that could be stored in electronic dictionary: keywords used for the search, and the associated with them data. The size stored keywords is totally much smaller than the size of the associated data.

The performance of electronic dictionaries is highly dependent on a compromise between search speed rate and the size of the number of search key-words is an electronic dictionary [3].

Analysis of modern translation systems shows that the semantic adequacy of translation hugely depends on the amount of context information [4]. On the other hand, modern translation systems require fast search speed electronic dictionaries. As a result, the improvement of computer translation requires finding the compromise between search speed and the size of the dictionaries.

3. The purpose and objectives of the study

The purpose of the study is to enhance the word search rate in e-dictionaries, as parts of computer-assisted translation systems, by reducing the number of swapping cycles proceeded while accessing to the dictionary stored data.

Therefore, the main objective of the study is finding the way to reduce the number of swaps while accessing e-dictionary stored data.

4. Organization of electronic dictionary based on perfect hash-addressing

Potentially, hash addressing is the fastest key-search technology. Hash addressing provide access to any keyword data in a fixed time. This time is approximately equal to time of one hash transformation.

When hashing the position of the element corresponding to the keyword k is calculated as h(k), where h is some hash-function.

The number h(k) is the hash value of the keyword k. The hash values of two different keys can match. This means that a collision has occurred [5].

Collisions are the most significant disadvantage against using hashing in electronic dictionaries organization. This situation is due to the fact that conflict resolution requires a large number of resources and is time consuming, which is unacceptable in the context of using hashing as a technological solution for high-speed dictionaries. It is well known fact that the probability of collisions and the effectiveness of their resolution largely depend on the coefficient φ of memory filling [6].

As already mentioned, the information corresponding to the keyword k is placed at the address, which is calculated as a hash of the keyword h(k). Thus, the coefficient under the coefficient φ means the ratio of the cells of the address hash keywords to the total number of memory cells allocated for the organization of the dictionary.

Another feature of hashing is that the result of a hash function for two close keys will be different. In the context of using hashing in dictionaries, this means that came-root words will be addressed to different parts of memory [7]. This can reduce the efficiency of using hashes in dictionaries.

This problem can be solved by preliminary analysis of the searched word or language construction. This analysis can occur with the involvement of means of stemming, morphosyntactic analysis, root extraction [8]. Thus, the search keys will be generalized matches of the initial input data.

The analysis of the considered features of work of electronic dictionaries, use of hashing and work of systems of computer translation has led to the decision in which it is offered to divide process of search of relevant translation of a keyword into two stages. The first stage is to actually search for contextual keyword data. The second stage is the selection of the most successful translation performed by the computer translation system.

To solve the problem of collisions, it is proposed to use perfect hash addressing. as perfect hash transformation, it is proposed to use a symmetric encryption algorithms. Thus, the searched keyword is fed to the input of the cipher block, and the corresponding ciphertext acts as a hash address.

This solution is due to the possibility of using hardware implementation of symmetric encryption algorithms for faster execution of hash transforms that will speed up the work of the dictionary in general. Using this advantage is easy on a practical level because almost all modern computer systems have built-in cryptoprocessors that implement standardized symmetric cipher.

In practice, the main limitation is the time of selection of such hash transformation, which for a given array of input keywords will be perfect hash transformation. The estimate of the time T required to find the ideal hash transformation can be expressed by the following formula (1):

$$T = t \cdot \varpi \cdot M \, , \tag{1}$$

where t — is the time of performing h(k), ϖ - mathematical expectation of number of samples, M - number of samples.

It is known that mathematical expectation ϖ depends on the probability that collisions will not occur before current key selection stage and the probability that collision will occur at the current key selection stage [9].

The results of the estimation of coefficient φ that could be achived by selecting the perfect hash-transformation for T (hours) are shown in the Table 1. These results were got by using formula 4 and they are based on the fact that DES algorithm was used as hash transformation and it was performed on the Cortex-M4. According to Cortex-M4 documentation [10], time t = 7.6·10-8 sec. The T values are given for estimation and can be reduced by γ times by using γ processors.

Table 1
The dependence of coefficient φ on time T and number α of keywords

T, hours	Coefficient φ for α of keywords			
	1000	10000	50000	100000
10	0.2210	0.0594	0.0271	0.0184
50	0.2308	0.0615	0.0265	0.0189
100	0.2361	0.0642	0.0281	0.0196
500	0.2462	0.0675	0.0289	0.0210

Based on results presented in Table 1, it can be concluded that the memory efficiency is small if perfect hash addressing is used in real-size dictionaries. To overcome this drawback, it is proposed to place contextual information related to keywords in the hash memory.

It is proposed to fill the hash memory in two stages. Firstly, memory is allocated for links, at addresses formed by perfect hash-transformation of keywords. After that, the context information is placed in the gaps between the primary links. Thus primary links and the context data can be read into the cache memory in minimum number of swapping cycles.

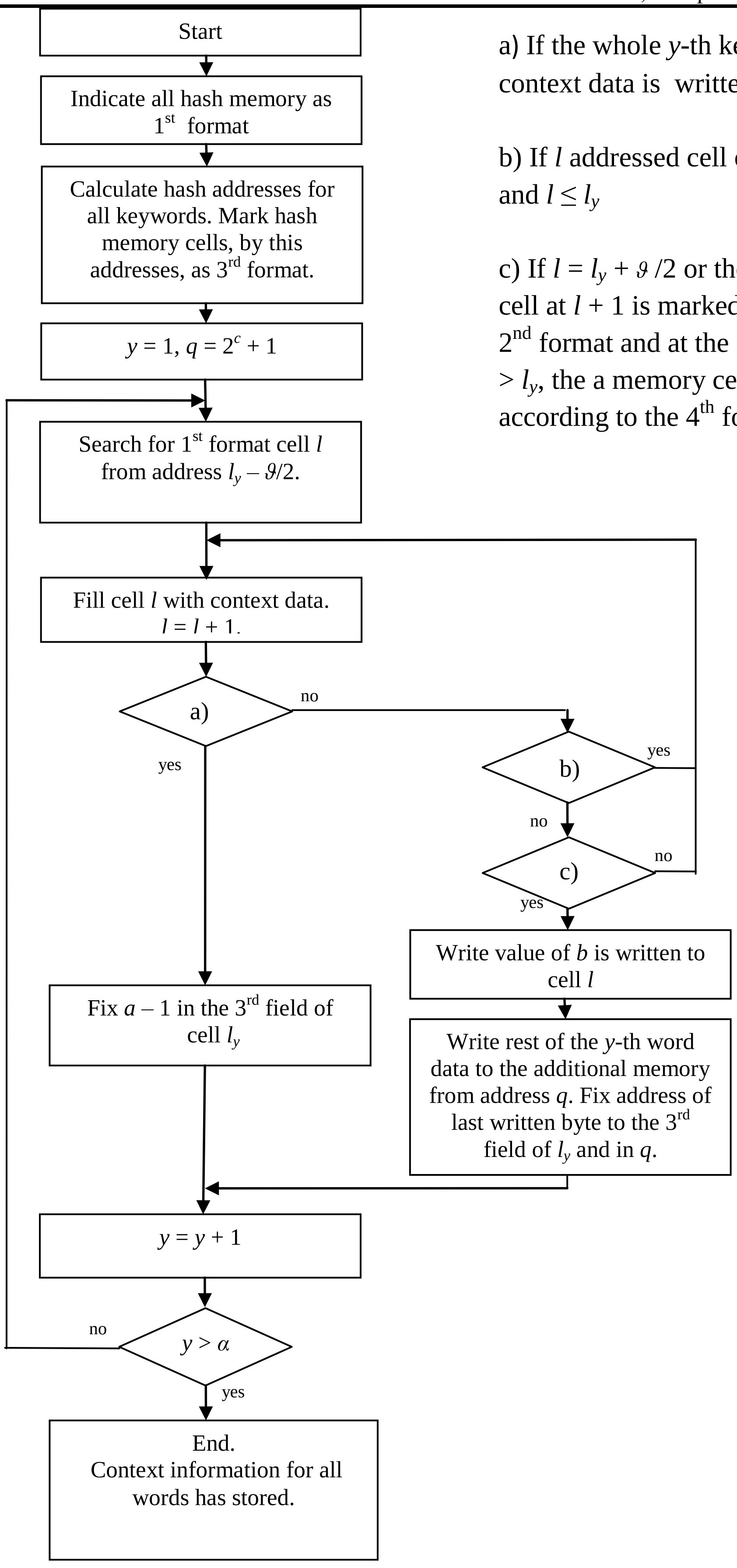
It is proposed to divide memory into two zones: a hash memory and an additional memory. The first zone stores the primary address links and context data that can be read into the cache in a single swap. The additional memory is provided for storing rest of context information.

It is proposed to realize this idea by using four formats of organizing data in the memory cell. The 1st format is proposed to be used to indicate free memory cells. Cells of these formats have the marker S1 in the first byte. The 2nd format has no markers and it is the format of cells that contain some payload data. The 3rd format is used for storage the store primary references. The cells of this format contain three fields. The first field is marker S2 of size 1 byte. The second field holds an address link to the beginning of the contextual information of a certain word; the third field is the address of the last memory cell of the corresponding context information. The 4th format of memory cells has marker S3 and are used for storage an address reference to the continuation of the context data of a keyword in the additional memory.

Every memory cell contains c/4 + 1 bytes, where c is size of address in bits. Markers can be implemented as symbols, which are not used in dictionaries.

The dictionary is filled with keywords and their corresponding contextual data according to the algorithm, the graphical representation of which is shown in Fig.1. Primary links of α keywords are l1, l2,…, lα, ϑ – swapping buffer size.

Search for contextual data of a given keyword is performed according to the algorithm, the graphical representation of which is shown in Fig.2 It should be noted that the algorithm involves recovering contextual data stage of analysis which is performed by the computer translation system.

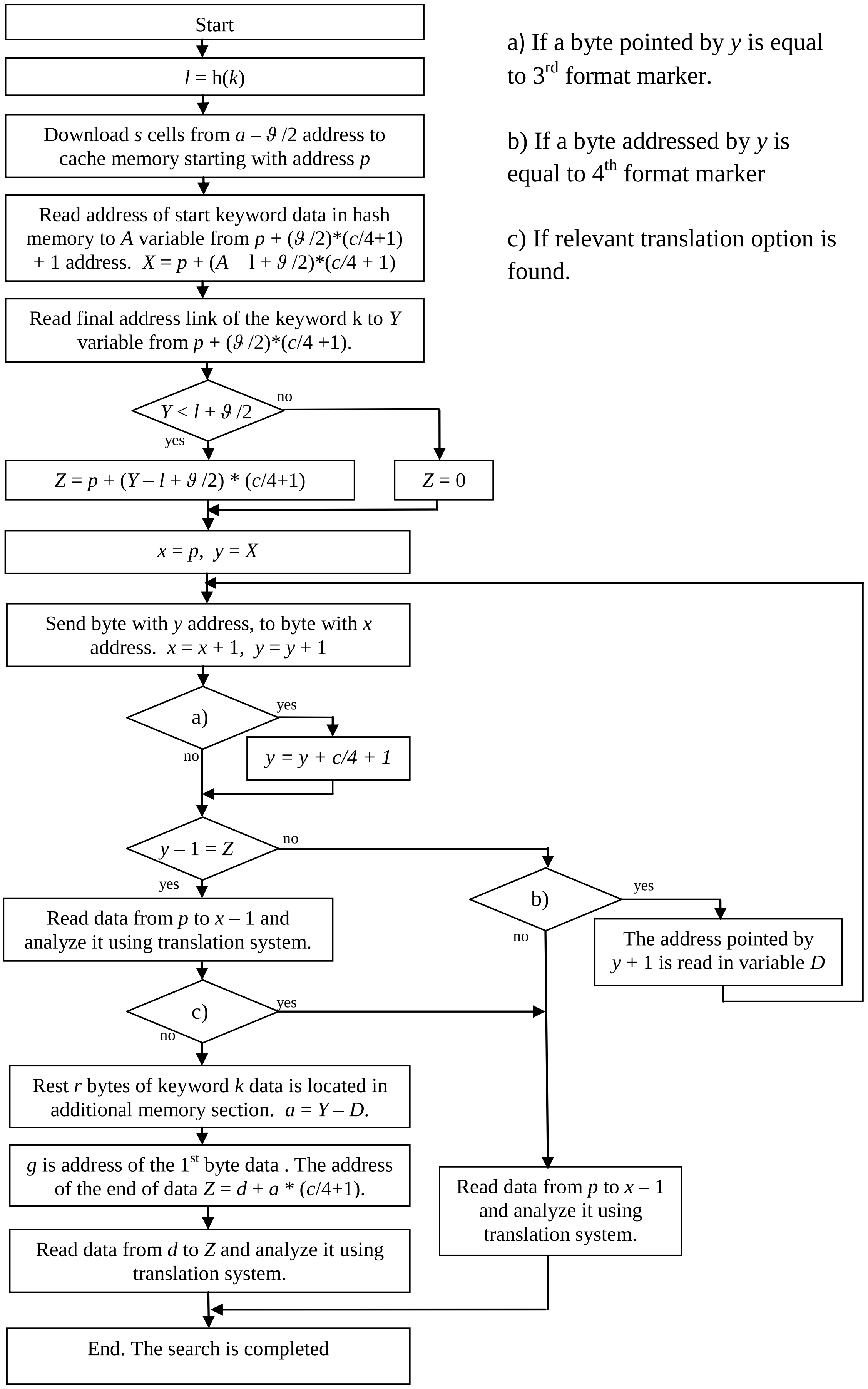

Start
Indicate all hash memory as 1st format
Calculate hash addresses for all keywords. Mark hash memory cells, by this addresses, as 3rd format.
$y = 1$, $q = 2^c + 1$
Search for 1st format cell l from address $l_y - \vartheta/2$.
Fill cell l with context data. $l = l + 1$.
a)
no
yes
b)
yes
no
c)
no
yes
Write value of b is written to cell l
Write rest of the y-th word data to the additional memory from address q. Fix address of last written byte to the 3rd field of l_y and in q.
Fix $a - 1$ in the 3rd field of cell l_y
$y = y + 1$
$y > \alpha$
no
yes
End. Context information for all words has stored.

a) If the whole y-th key-word context data is written

b) If l addressed cell of 2nd format and $l \le l_y$

c) If $l = l_y + \vartheta/2$ or the memory cell at $l + 1$ is marked as cell of 2nd format and at the same time $l > l_y$, the a memory cell is marked according to the 4th format.

Fig.1. Algorithm of organization and allocation data in a dictionary.


Start
$l = \mathrm{h}(k)$
Download s cells from $a - \vartheta/2$ address to cache memory starting with address p
Read address of start keyword data in hash memory to A variable from $p + (\vartheta/2)*(c/4+1) + 1$ address. $X = p + (A - 1 + \vartheta/2)*(c/4 + 1)$
Read final address link of the keyword k to Y variable from $p + (\vartheta/2)*(c/4 + 1)$.
$Y < l + \vartheta/2$
no
yes
$Z = p + (Y - l + \vartheta/2) * (c/4+1)$
$Z = 0$
$x = p,\ y = X$
Send byte with y address, to byte with x address. $x = x + 1,\ y = y + 1$
a)
yes
no
$y = y + c/4 + 1$
$y - 1 = Z$
no
yes
b)
yes
no
Read data from p to $x - 1$ and analyze it using translation system.
The address pointed by $y + 1$ is read in variable D
c)
yes
no
Rest r bytes of keyword k data is located in additional memory section. $a = Y - D$.
g is address of the 1st byte data . The address of the end of data $Z = d + a * (c/4+1)$.
Read data from p to $x - 1$ and analyze it using translation system.
Read data from d to Z and analyze it using translation system.
End. The search is completed

a) If a byte pointed by y is equal to 3rd format marker.

b) If a byte addressed by y is equal to 4th format marker

c) If relevant translation option is found.

Fig.2. Algorithm of the search for keywords context data in a dictionary.

5. Results

The main criteria for estimating the performance of an electronic dictionary, in terms of usage it as an element of computer translation systems, are the speed of context search in the electronic dictionary and the level of memory usage.

In the modern computer systems, the search time Tv is calculated using the following formula (2):

$$T_v = \kappa \cdot \tau_\eta + \tau_n, \qquad (2)$$

where κ — is the number of swapping cycles, τη — is the of one performing one swapping cycle , τп — is the time of the context search.

The time τп depends on the complexity of translation algorithm and can be different for different translation systems. Time τη of one swapping cycle is proportional to buffer size ϑ.

Analysis of the modern electronic dictionaries shows that the time τη is much longer than the time τn required to find the most appropriate translation option from stored data. As a result, the speed of the electronic dictionary search is determined by the number of swapping cycles.

On this basis, time τη in can be estimated in terms of the average number η of swap cycles required to access to the keyword context information.

The level of redundancy in memory use can be estimated as follows. If μ is the average amount of contextual information of one key word, then the general amount of linguistic information in electronic dictionary is product μ and α.

All electronic dictionaries contain a some service data. This means that the size of real dictionaries is always greater than the size of linguistic information itself.

The index Ω of the redundancy of memory use in electronic dictionaries can be determined according to the following formula (3):

$$\Omega = \frac{\varsigma}{\mu \cdot \alpha}, \qquad (3)$$

where ζ - is size of whole dictionary.

For the experimental study of the efficiency of the proposed electronic dictionary, a statistical modeling software complex has been developed.

In the framework of the simulation, the number α = 10000, the average μ = 400 bytes. To determine these parameters, a statistical study of the translated and interpreted dictionaries of computer terms [4] was carried out.

The first part of experimental study was aimed to determine influence of coefficient φ on main performing criteria. Results of experimental study of dependence between coefficient φ and average amount η of swapping cycles needed to access to all context data of keyword are shown on Fig.3.

The fig.3 clearly proves that the amount η of swapping cycles directly depends on coefficient φ.

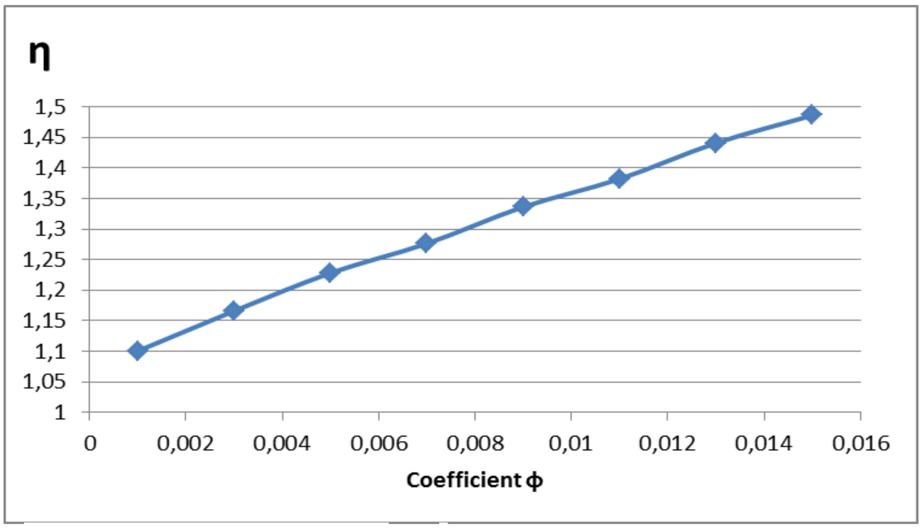


Fig.3. The dependence of amount η of swapping cycles on coefficient φ.

The second part of experimental study was aimed to determine influence of coefficient φ on value of load factor λ . Load factor λ is equal to the ratio of the filled cells of the hash memory to the total hash memory. Results of this study are shown on Fig.4.

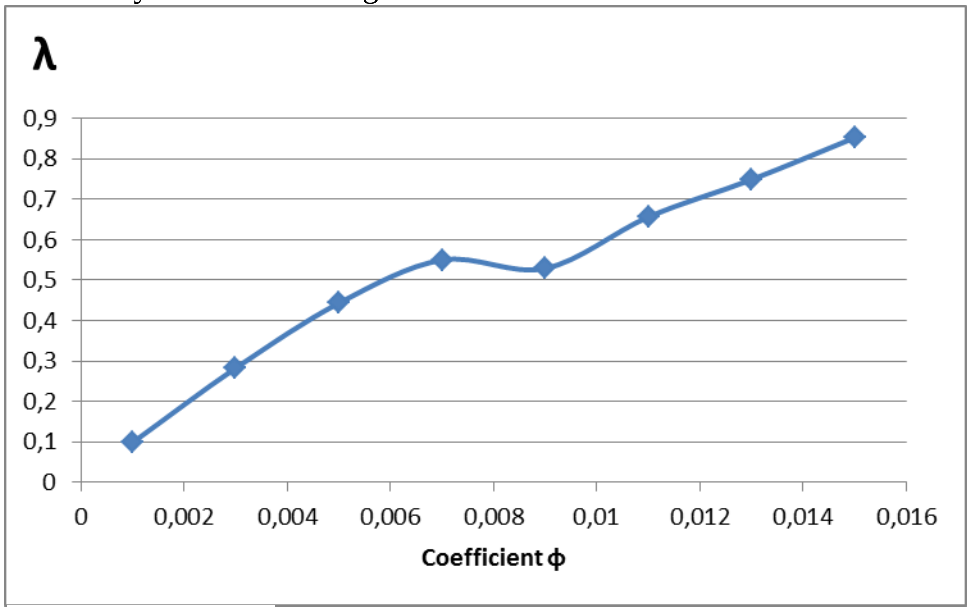


Fig.4. The dependence of the hash memory full load factor λ on coefficient φ.

Fig. 3 and Fig.4 demonstrate that with an increase in the coefficient φ, the redundancy of memory use decreases. But, it increases the number η of swapping cycles during which all the context information of a certain word can be loaded into cache memory.

For instance, if coefficient φ = 0.012, hash memory is totally filled on 65% and the amount of swaps which must be executed to load whole context data is 1.38. The indicator Ω is about 1.6.

The main advantage of the proposed electronic dictionaries is speed rate of access to contextual information of the key-word.

It is advisable to evaluate the work of such an organization of the electronic dictionary by comparing it with known developments according to the selected efficiency criteria.

Comparative performance analysis requires determining the number of η swap cycles for which certain contextual information is accessed.

For tree-based electronic dictionaries, this number depends largely on the type of tree. For balanced binary trees, the average number of memory accesses is $\log_2 \alpha$. For other types of trees, this value is greater.

When searching in the contextual information tree for a certain word, a chain of memory accesses is performed. In particular, for an electronic dictionary based on a balanced binary tree with the storage of contextual information in its nodes, the adjacent addresses of the chain of appeals are stored in memory at a distance exceeding the size of swap buffer. Accordingly, the number η of swap cycles is approximately equal to the number of memory accesses and for α = 10000, is about 13,3. Such kind of dictionaries works at least 9.1598 times slower than the proposed dictionary.

For tree-based e-dictionaries with spaced link tree and contextual information storage, the access chain passes without swapping within a subtree that does not exceed the size swap buffer. In particular, for α = 10000 and the payload data is $4 \cdot 10^6$ bytes, each node stores two links and a word router, which occupy a total of 16 bytes. For the accepted size s of the swap buffer, the subtree contains about 28 nodes, every 4 requests swap the next subtree. Thus, access to information requires 4 swaps during the search and one for direct transportation of contextual information. Compared to the proposed dictionary, it works in 3,45 times slower.

For electronic dictionary based on hash search with collisions resolved by probing and spaced storage of addresses and contextual information, finding and loading in the cache memory of contextual information of a certain word is carried out, on average, in 2,8 cycles of swapping. It is in 2.03 times slower comparing with proposed dictionary.

Evaluation of memory efficiency is based on a comparison of redundancy of memory usage.

For dictionaries based on the tree with the preservation of contextual data in its nodes, the service information consists of address links to the descendants of the node. In this example, when using a balanced binary tree, two address links occupy 6 bytes and, accordingly, the average number of bytes of the node is μ + 6. Thus, the indicator Ω is equal to $\frac{\mu+6}{\mu}=1{,}015$.

In electronic dictionaries with spaced preservation of the link tree and contextual information, in each of its nodes there are a word-router and two address links. In this case, for the given example, the total volume of service information is $\alpha\cdot 16 = 16\cdot 10^3$ bytes. Thus, the indicator is equal to $(16\cdot 10^3+4\cdot 10^6)/4\cdot 10^6 = 1{,}004$.

For electronic dictionaries based on hash search with collisions and spaced storage of addresses and contextual information, hash memory cells containing the keyword and links to its contextual data. In this example, the size of such a cell is 13 bytes, o for φ = 0,7, the total volume of the dictionary is $13\cdot 10000 / 0{,}7 + 4\cdot 10^6$. Thus, the indicator Ω of memory usage is 1,047.

Analysis of this data convincingly shows that the proposed organization of the dictionary provides the highest search speed compared to known methods. The resulting effect is achieved due to slightly lower memory efficiency. The proposed approach is quite justified in the current trends of cheaper hardware memory.

6. Conclusions

As a result of research, the new method based on perfect hash-search has been proposed.

In order to increase the efficiency of using hash-addressing, it is proposed to store data for contextual translation between the hash addresses of key-words. These hash-addresses are followed by address links associated with the data, which are needed for translation. This method of storing data is adapted for the multilevel memory architecture of modern computer systems.

The efficiency of the proposed method was studied both theoretically and experimentally. It has been testified that the proposed method doubles the search speed compared to traditional methods of organizing electronic dictionaries. The cost of increasing the speed of searching for words was paid by increasing the amount of used memory.

The developed method of organizing data in electronic dictionaries can be used as a component of highly efficient computer translation systems.

References

1. Pastor V. Searching Techniques in Electronic Dictionaries: A Classification for Translators / V.Pastor, A. Ampago //International Journal of Lexicography.- 2010.- Vol.23.- № 23.- P.307-354.
2. Marchuk U.N. Computational linguistics. –AST, Vostok-Zapad, 2001.-165
3. Agapova N.A. On the principles of creating an electronic dictionary of the linguocultorological type: to the problem statement / N.A. Agapova , N.F.Kartofeleva // Vestnik Tomskogo gosudarstvenogo universiteta. № 382 -2014.- .6-10.
4. Jongejan B. Automatic training of lemmatization rules that handle morphological changes in pre-, in- and suffixes alike. / B. Jongejan B. and H.Dalianis // Proceeding of the ACL-2009, Joint conference of the 47th Annual Meeting of the Association for Computational Linguistics of the Asian Federation of Natural Language Processing, Singapore. – 2009. - P. 145-153
5. Johnson R. Using finite state transducers for making efficient reading compression dictionary / R Johnson, L Antonsen, T Trosterud // Proceeding of 19-th Nordic Conference of Computational Linguistics- NODALIDA 2013.- Olso, Norway.- 2013.- P.75-87.
6. Fuertes-Olivera P.A. E- Lexicography: the internet, digital initiatives and lexicography / P.A. Fuertes-Olivera, H. Bergenholtz.- London. Continuum International Published Group.- 2011 – 282 P.
7. Markovskyi A.P. Interactive template method of computer translation of scientific and technical publications / A.P Markovskyi, O.M. Mykolayivna, Fan Chunlei // Visnik of NTUU “Igor Sykorsky KPI” Informatika, upravlinnia ta obchisliuvalna tekhnika - 59. - 2013 - 86-97.

8. Vidrin D.V., Polyakov V.N. Implementation of an electronic dictionary using n grams/ D.V. Vidrin, V.N. Polyakov//Shtuchnyi intellect. № 4 - 2002. -.180-183.
9. Fuertes-Olivera P.A. Wiktionary as a prototape of collective free multiple-language internet dictionary/ P.A. Fuertes-Olivera // The functional theory of lexicography and electronic dictionary.- 2009.- P.99-108.
10. Ball L.H. Heuristic evaluation of e-dictionary / L.H. Ball, Bothma T.J.D. // Library Hi Tech, - 2018.- Vol. 36. - № 2.- P. 319-339.

UDC 004.72

DOLYNNYI O.,
NIKOLSKIY S.,
KULAKOV Y.

THE METHOD OF SDN CLUSTERING FOR CONTROLLER LOAD BALANCING

The paper proposes the method of SDN clustering using connections density and controller load distribution that solves the problem of controller load balancing. Clustering efficiency criteria have been considered, including fault tolerance, controller-to-switch and intercontroller latency and network limitations. Review of the key clustering methods has been performed, and the base algorithm for modification has been chosen. Density-based controller placement algorithm is modified to solve the problem of multicontroller placement. Metric of the node boundary index is introduced to advance the efficiency of proposed algorithm. A software implementation of the developed algorithm has been created, and its performance has been tested. The algorithm's modelling results have been compared with those of the other algorithms using the metrics of distribution of service traffic propagation delay and computational speed in relation to network topology size.

Keywords: SDN clustering, network, connections density, controller placement

1. Introduction

The problem of controller placement is a critical part of implementing the Software Defined Networks (SDN) technology. The concept of SDN, which has emerged in recent years, is increasingly becoming a vital technology in the development of large-scale datacenter networks. The essence of the SDN concept is to separate the control layer from the data layer, which simplifies network management, improves network flexibility and scalability.

In large networks, the good placement of the controllers makes the best use of the existing structure of links between the network nodes. One controller is not enough to control all the nodes in a large SDN, as its performance is limited, and the signal propagation delays between the controller and all nodes in the network is very large; moreover, a single controller on the network cannot meet fault tolerance requirements. Therefore, it is logical to place many controllers at different points in the network, which together control the entire data layer. The critical question that emerges is how to cluster the network and where it will be most efficient to place the controllers. It is known as the controller placement problem.

2. Clustering efficiency factors

The problem of controller placement is defined as a multifactor complexity optimization problem [1] [2], in which the position of a controller should be decided upon with the following characteristics taken into consideration:

(1) Fault tolerance. The switches are dependent on their controllers providing them with instructions on how to forward packets. If the connection between a switch and its controller is disrupted, the switch becomes unable to forward packets and, thus, becomes essentially nonfunctional in the network [3]. Possible reactions to such failures must be taken into account when designing a network.

(2) Controller-to-switch latency. Controllers send instructions to switches, according to which the latter forward packets. Consequently, the transmission delays of those control signals influence the latency metric of the entire network. The network-wide latency cannot be minimized without minimizing local controller-to-switch latencies [4].

(3) Inter-controller communication. Each controller has a number of switches to which it sends commands. To exchange messages at a network-wide level, i.e. between remote switches managed by

different controllers, controller-to-controller communication must be implemented [5]. So the total communication delay between remote switches in the network depends on the delay of controller-to-controller information exchange.

(4) The density of controller placement. There is no predefined number of controllers that would ensure the optimal performance of an SDN network. The desired number of controllers is especially hard to decide on in large networks with advanced structure. It has to be determined using assorted methods, some of them complex and time-consuming, as they consider a large number of placement variants [6].

(5) Constraints set by controller capacity. Hardware limitations of a controller make it possible to manage only a certain number of switches without getting overloaded. Controller overload, i.e. its processing queue becoming overflown by the tasks, is a scenario that has a negative effect on the performance of the SDN as a whole [7].

3. Related works.

Some works have proposed heuristic algorithms to solve the controller placement problem and defined this as multi-objective combinatory optimization task. However, for a large-scale network such algorithms demonstrate an unreasonably high solution time [3].

On the other hand, there exists an approach in which controller placement is determined by certain defined metrics.

K-means method, developed by B.Heller in 2012, focuses on minimizing the communication latency in a network [5]. This method doesn't take into account the reliability aspect, as well as controller load when solving the clustering problem.

Another popular method is the density based controller placement (DBCP) introduced by J Liao in 2016, which uses a nodes clustering algorithm based on density distribution [6]. These nodes have higher quantity of connectivity inside the cluster and lower quantity of connectivity with nodes from other networks, thus there is one and only one controller in each subnetwork. To minimize costs for intercontoller communication, the proposed algorithm advises to place them at the closest possible distance. However, every controller's placement is chosen independently in every cluster as well as sequential order of cluster choosing for nodes, therefore controller position and cluster's load is not optimal as far as all clusters are concerned.

Pareto-Optimal Resilient Controller Placement method (POCO), introduced by D. Hock in 2013, aims to guarantee the best possible controller placement using Pareto distribution of network metrics for different nodes. This algorithm also considers such parameters as inter-controller communication and load balancing for cluster controllers, but has the drawback of low computational speed due to a large amount of calculations.

Focusing on equal load balancing goal and considering the positive and negative sides of each method, this study uses DBCP method as the base algorithm for the following modification.

4. Aim and objectives

In this study, we propose a method of network clustering, aimed at improving the efficiency and equal load balancing of an SDN network by analyzing the density distribution of links. This requirement is substantial to consider due to existence of physical restrictions for cluster size, it cannot have extremely high value due to central controller load limits.

In order to achieve the aim of our study, the following objectives are proposed:

1) to identify the factors that affect the efficiency of the clustering process;

2) to analyse the clustering methods that can be used in software-configured networks;

3) to develop a method of SDN clustering, taking into account the distribution of link density and data flows;

4) to model and analyse the effectiveness of the proposed method of SDN clustering.

5. The proposed method

5.1. General principles of clustering method construction

In this section, the basic steps for clustering method construction are observed.

Let us assume that the network topology $G(S, L)$consists of a set of routers Sand a set of bidirectional connections between themL. Note that if there is a direct connection between the nodes of the network, the length of the connection is equal to one, otherwise it is equal to zero.

Unlike the other methods, this algorithm analyzes the network topology, which is then divided into subnets. Nodes, that have the most common connectivity, have a relatively larger number of connections inside the subnet, and so have fewer connections to other subnets. Therefore, they can be considered basic nodes for separate subnets. After following the described method during the aggregation of nodes, the constructed subnet has the feature of higher fault tolerance. Delay in the subnet is also reduced due to fine-grained clustering.

In general, the clustering method consists of three main steps:

1) to analyze the network topology for the distribution of connection density between routers;

2) to distribute network routers in clusters according to the found values of density, distance from the current node to the node with a higher value of density, and the node boundary index;

3) to solve the problem of controller placement for each subnet according to a given criterion (in this case - the metrics of the average distance to other cluster nodes).

5.2. Network graph analysis of deviations in the density distribution

In the algorithm, the division of the whole network occurs due to cluster allocation. Two values are calculated for each routers_i: local densityρ_i and distance to a node with a higher local density value δ_i (node priority for the clustering task). These values depend only on the connections between the routers.

The local density ρ_i for each router is calculated by the formula (1), which uses the coupling factor k_{cl} and the distance between the routers d_{ij}:

$$\rho_i = \sum_j k_{cl}(d_{ij} - d_c) \tag{1}$$

The maximum distance value to a nearby node d_c is the limit of the distance that the routers assigned to the current node can have. The coupling factor k_{cl} is taken as one in the case of a positive difference between the reference distance limit and the distance between the routers, and zero otherwise. For distributed networks with more than 50 nodes, it is effective to set d_c as 30% of the network diameter.

The distance to the node with higher value of local density δ_i is calculated as the shortest distance to the node with higher value of local density:

$$\rho_i = \sum_j k_{cl}(d_{ij} - d_c) \tag{2}$$

For nodes with the highest local density in a cluster, the distance to the node with the even higher value of local density is taken as the maximum value; such nodes will be selected first as cluster centers.

As a result, we get the following pseudocode of the network graph analysis of deviations in the density distribution:

```
int analyzeDensity(G = (S, L), dc) {
   int k = 0;
   for (s: S) {
      ro[s] = getNumberOfNodesWithinDistance(s, dc, G);
   }
   for (s: S) {
```

```
            delta[s] = minDistanceToHigherDensityNode(s, ro, G);
            delta_Avg = sum(delta[0], ..., delta[S]) / S;
            if delta[s] > delta_Avg {
                k++;
            }
        }
        return k;
    }
```

5.3. Algorithm of network graph clustering

To ensure that the even load on the cluster controllers is reached, it is necessary to introduce the node boundary index (distance to the cluster boundary), in addition to following the formulas 1 and 2.

Let us assume that the load of each router equals l(s). Then the maximum possible controller load L(Θ) should not be less than the sum of the traffic loads of each cluster node:

$$L(\Theta) \geq \sum_{s \in S(\Theta)} l(s) \tag{3)}$$

Let us consider the case of a cluster already containing enough nodes to control, so it cannot be expanded with new nodes; so it is needed to join those nodes to other clusters or create a new cluster. Maintaining the compliance with the maximum load requirement on the cluster controller, it is necessary to distinguish between nodes that are closer to the center of the cluster and nodes that are closer to the boundaries of the cluster. The latter nodes can be considered as candidates for joining expandable clusters.

Let us denote total local density of current node as the sum of local densities for such nodes that have higher density value than current node:

$$td_i = \sum_{j}^{s_j \in UL(s_i)} \rho_j \tag{4}$$

To find the nodes in the cluster that are on the border, we calculate a new value for each router, namely the node boundary index σ_i. The node boundary index is a value that indicates the uncertainty of the node membership in the current cluster. If the local density of the neighboring router, which is greater or equal, differs by no more than the limit value of the density difference, then this router can be considered as the border node of the cluster.

To find the node boundary index σ_i, we use the formulas from the theory of information entropy:

$$\sigma_i = \sum_{j}^{s_j \in UL(s_i)} \frac{\rho_j}{td_i} \log_{|UL(s_i)|} \frac{\rho_j}{td_i} \tag{5}$$

In the modified algorithm, the selection of a node with a smaller value of the node boundary index has a higher priority during the process of assigning nodes to the cluster. The router is assigned to the same cluster as the nearest neighboring router with a higher local density value, unless the total load on the cluster routers will exceed the maximum possible load on the controller. If there is no cluster to which the current node can be attached, it becomes the center of the new cluster.

As a result, we get the following procedure pseudocode, that performs network graph clustering considering limitations for maximum controller load:

```
    void clusteringWithLoad(Graph G = (S, L), ro, delta, controllerCapacity) {
        for (s: S) {
            delta[s] = getBorderline(s, G);
    }
    S.sortBy(delta);
    for (s: S) {
       n[s] = cluster(s);
    }
   for (s: S) {

       ul = getNeighbors(s, (ro > ro[s]));
       ul.descendingSortBy(ro);
       for (s1: ul) {
          si = union(n[s], n[s1]); //si – all connections, implicit to n[s] or n[s1]
          if (controllerCapacity >= sum(l(si))) {
             n[s1].add(s);
             break;
          }
       }
    }
 }
```

6. The example of the developed clustering method

We will demonstrate the work of the developed algorithm by example.

Let the traffic flows on each router be the same and equal to 1. Then the network clustering task based on the controller load limitations and traffic flows can be reduced to the network clustering task regarding the total number of routers that can be controlled by the controller.

The network topology graph is presented in Fig. 1a. Let us assume there is a limit to the total number of routers that can be controlled by the controller, and this number is 6. We use formulas 1, 2, and 5 in our calculations.

The simulation results of the algorithm for the mentioned network graph are shown in Fig. 1b and Table 1.

34 routers are divided into 7 different clusters, where the routers of a separate cluster are marked with a separate color. It should be noted that the router 22 is allocated to a separate cluster, because all neighboring clusters are already full and cannot be expanded, so the router 22 cannot be connected to any of them.

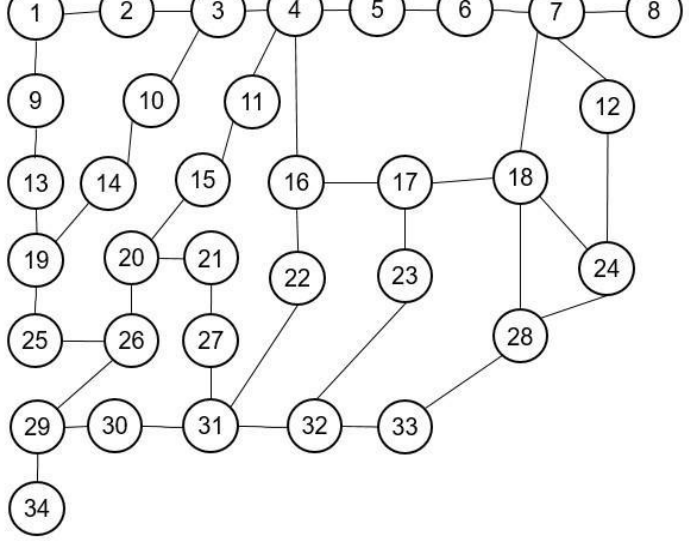


Fig. 1a. Original topology

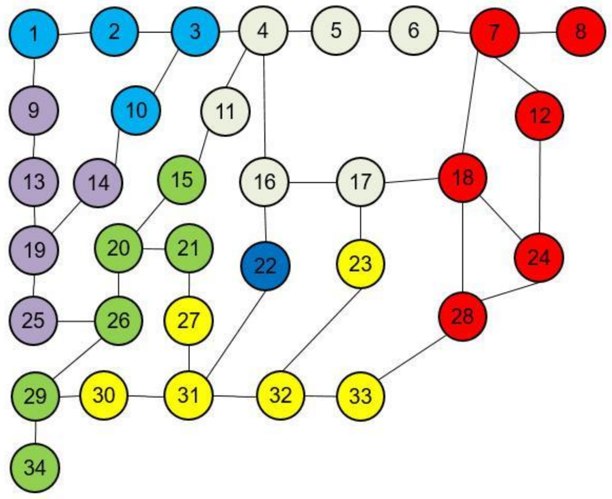


Fig. 1b. Marked topology

7. Results and discussion

The advantages of the developed clustering algorithm include the following aspects. First of all, the proposed method selects the optimal number of controllers for a particular network. Secondly, clustering based on topology link density reduces the likelihood of controllers placement in the nodes with a high risk of a failure. Finally, the problem of multiple controllers placement in the network is reduced to the problem of placing one controller in subnetwork, which reduces the time complexity of solving the problem.

Table 1
The calculation of modelling results

Node id	ρ	δ	σ	Node id	ρ	δ	σ	Node id	ρ	δ	σ
1	4	1	0.91	13	5	1	0.0	24	6	1	0.98
2	8	1	0.0	14	5	1	0.99	25	6	1	0.98
3	8	1	0.99	15	5	1	0.99	26	8	3	0.0
4	10	3	0.0	16	9	1	0.0	27	6	1	0.0
5	6	1	0.95	17	8	1	1.0	28	7	1	0.0
6	6	1	0.98	18	9	3	0.0	29	6	1	0.99
7	8	1	0.0	19	6	2	0.0	30	7	1	0.0
8	4	1	0.0	20	7	1	0.0	31	9	3	0.0
9	4	1	0.99	21	5	1	0.99	32	8	1	0.0
10	5	1	0.96	22	7	1	1.0	33	6	1	0.99
11	6	1	0.0	23	6	1	1.0	34	3	1	0.0
12	6	1	0.98								

During the testing of the obtained algorithm, several metrics were measured for the analysis of the model. A comparison of the developed algorithm with K-means and POCO algorithms was performed. The analysis used a graph generator with a step of 25 nodes, all traffic connections were taken as 1.

Fig. 2 compares the execution time of algorithms. With a relatively small number of nodes (up to 100), there is no big difference in computational speed between algorithms, but the difference begins to grow with higher order polynomial complexity between K-means algorithm and POCO and the developed algorithm.

Fig. 3 presents the results of the analysis of the delay in the propagation of service traffic depending on the number of topology nodes. The considered algorithms have similar delay indicators, differing on different values of the number of nodes, but according to the total efficiency index, the developed method has the best metrics.

The developed method could be applicable for the large-scale networks due to quite good results of modelling metrics. However, it is necessary to note the difficulty of algorithm integration into the existing hardware infrastracture.

Further development of the algorithm could take into consideration the aspects of reliability and minimization of cluster amount. These aspects were not considered in detail in the present paper, as our aim was to guarantee the even controller load. It should be noted there is a possibility of finding other flaws in the presented method, depending on the metrics used for comparison of algorithms.

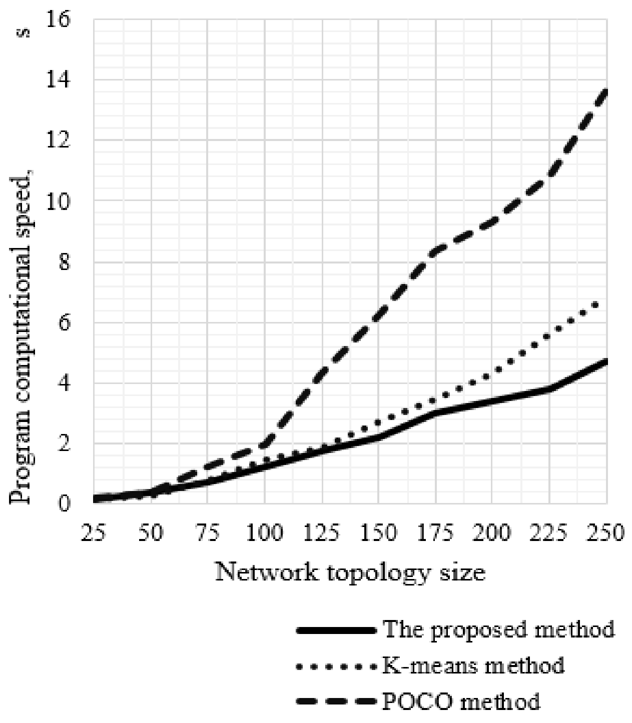


Fig. 2. Comparative graph of computational speed in relation to network topology size

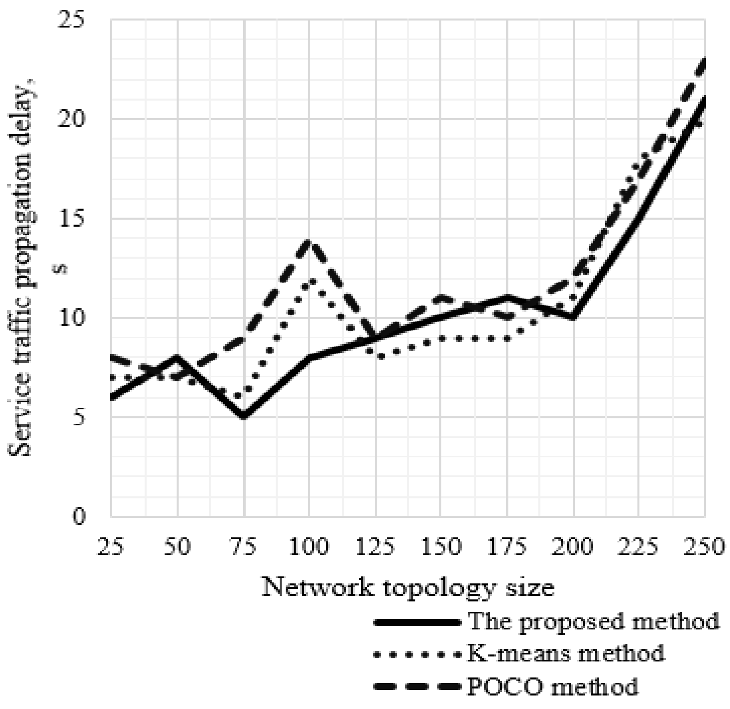


Fig. 3. Comparative graph of distribution of service traffic propagation delay in relation to network topology size

8. Conclusions

The paper considers a network clustering algorithm based on the connections density distribution and the maximum load on the controllers. The example demonstrated the procedures of the algorithm.

As a result of testing on different network topologies and analyzing the obtained results, the developed method of clustering demonstrates high efficiency in comparison with K-means and POCO algorithms regarding the existing metrics of service traffic delay and speed.

From the practical point of view, the results obtained in this paper allow increasing the efficiency of SDN networks management, especially large-scale networks, for which it is important to balance the load on the controllers.

Further development of the algorithm could take into consideration the aspects of reliability and minimization of cluster amount.

References

1. Scott-Hayward S. Are we ready for SDN? Implementation challenges for software-defined networks? / Scott-Hayward S., Chouhan P.K., Fraser B., Lake D., Finnegan J., Viljoen N., Miller M., Rao N. // Communications Magazine, IEEE . – 2013. – № 51(7) – pp. 36–43.
2. Hakiri, A. Software-defined networking: Challenges and research opportunities for future internet. // Computer Networks. – 2014. – № 75. – 453-471 pp.
3. Hu, F. A survey on software-defined network and openflow: From concept to implementation / Hu F., Hao Q., Bao K. // IEEE Communications Surveys & Tutorials. – 2014. – № 16. – 2181-2206ppc.
4. McKeown, N. OpenFlow: Enabling Innovation in Campus Networks / McKeown N., Anderson T., Balakrishnan H., Parulkar G., Peterson L., Rexford J., Shenker S., Turner J. // SIGCOMM CCR – 2008. – № 38(2).
5. Heller, B. The Controller Placement Problem / Heller B., Sherwood R., McKeown N. // HotSDN. – 2012.
6. Liao, J. Density cluster based approach for controller placement problem in large-scale software defined networkings / Jianxin Liao, Haifeng Sun, Jingyu Wang, Qi Qi, Kai Li, Tonghong Li // Computer Networks. – 2017. – № 112. – 24–35 pp.
7. Hock, D. Pareto-Optimal Resilient Controller Placement in SDN-based Core Networks / Hock D., Hartmann M., Gebert S., Jarschel P., Zinner T., Tran-Gia P. // 25th International Teletraffic Congress (ITC) – 2013.

UDC 004.94: 519.876.5

STETSENKO I.,
DYFUCHYN A.

PETRI-OBJECT SIMULATION: TECHNIQUE AND SOFTWARE

Nowadays information systems tend to be including components for transforming data into information applying modelling and simulation. Combined with real-time data, discrete event simulation could create powerful making decision and control systems. For these purposes, simulation software should be concentrated on creation the model as a code which can be easy integrated with other components of software.

In this regard, Petri-object simulation technique, the main concept of which is to compose the code of model of complicated discrete event system in a fast and flexible way, simultaneously providing fast running the simulation, is requisite. The behaviour description of the model based on stochastic multichannel Petri net while the model composition is grounded on object-oriented technology.

The Petri-object simulation software provides scalable simulation algorithm, graphical editor, correct transformation graphical images into model, correct simulation results. Graphical editor helps to cope with error-prone process of linking elements with each other.

For better understanding the technique, the Petri-object model of web information system has been developed. Investigation of the response time has been conducted. The experiment has revealed system parameters impact on the value of response time. Thus, the model can be useful to avoid long-running request.

Keywords: stochastic Petri net, discrete event simulation, web information system, long-running request, response time.

1. Introduction

Despite a long history of information systems, the vast majority of systems developed stay ‘data rich, information poor’. Huge storage of data can help to find a solution to problems in organizations and business. However, it is often high-effort and time-consuming process for each specific problem. To be useful data should be transformed into information, and the automatization of this process should be advanced considerably. IBM Center for The Business of Government in [1] highlights trends in the information system development, among which providing near-real-time data available to wide range of users and simplifying an interpretation of data to make information more useful.

Discrete event simulation is a powerful technique of the investigation of system behaviour conducting experiments with the model of the system (instead of the real system) that intended to improve the functioning of complicated systems (increasing performance or decreasing risks). Combined with real-time data, simulation components can create a powerful making decision and control systems. However, this requires specific demands to the simulation model: the code of the model should be implemented as a part of software, the model should be able to integrate with data storage and with end-user interface, the model should be able to be run by other components of software (not scientist). As modern simulation tools do not support these requirements, the development of relevant simulation tool is essential.

Particularly, discrete event simulation is useful for information system designing. Performance time, for example, can be estimated. In addition, the impact of parallel and concurrent processes on performance can be researched.

Systems and software engineering standard ISO/IEC 15909-1:2004 highlights following advantages of using Petri in programming systems and software designing: mathematical formalism of Petri net provides unambiguous specifications and descriptions of applications, graphical representation allows visualizing resource flow and control flow to better understanding system behavior, executable technique allows to verify ideas at the earliest and cheapest opportunity [2]. For today, the software implementing the use of Petri net to create relevant models of programming systems does not developed.

This research considers the advantages of Petri-object simulation technique to create models of complicated system. The usage of Petri-object simulation software to simulate information system behaviour is represented.

2. Related works

Discrete event simulation fundamentals are represented in book [3], where the best description of simulation algorithm and the fulfill information about random values generators can be found. However, there is no word about applying Petri nets for the purposes of simulation.

Stochastic Petri net is a widely known mathematical formalism of discrete event processes that provides system behaviour simulation. This type of Petri net is associated with transitions, the time delay of which are determined only by exponentially distributed random variables [4]. Later interpretation of this type includes time delay defined as random value with given probability distribution or zero value [5].

Coloured Petri net is an extension of stochastic Petri net in which simultaneous processing of different types of markers is possible. The mathematical basics of coloured Petri net are presented in book [6]. The number of elements needed for system simulation is significantly less in colored Petri net. However, the firing rule becomes much more complicated. In addition, the model representation has lost its simplicity compared with stochastic Petri net.

One of the latest extensions of Petri net is considered in work [7]. In work [8] the importance of usage of information system simulation to investigate the failure scenarios have been highlighted.

The authors of work [9] considered the development of the software for simulating information systems. The modeling language for information system was proposed as an extension of PNML (Petri Net Markup Language). The description of the model is divided on information model and process model, which manipulates with information model.

3. The purpose and objectives of the study

The aim of this research is to discuss the advantages of Petri-object simulation technique, in particular, in context of information system development. The end objective of the research is to apply the Petri-object software to build an information system model and to obtain a simulation result that is valuable for information system design.

4. Materials and methods

4.1. Petri-object simulation basics

Petri-object simulation basics were stated for the first time in work [10]. During the following years, the software was developed and the experimental research was conducted. In addition, several models were built and investigated using the new technique. The new technique implementation in different fields has revealed its advantages and limitations. Thus, the technique has been improved.

The concept of Petri-object model is grounded in object-oriented technology and at the same time is based on stochastic Petri net. Object-oriented technology states the rule in which elements of the model are created and combined. Stochastic Petri net establishes activities which are being performed by the elements of the model. Provided a specified way of connections between elements, the operation (or behaviour?) of the entire model will be determined by the stochastic Petri net obtained by the union of Petri nets of its elements.

Modeller determines the classes of elements, creates objects of these classes, and the connections between objects. After that, the model is able to simulate the behaviour of a system under investigation. Instead of routine creating a huge number of events the classes of typical model elements should be defined with once determined Petri net in each of them. This makes the construction of the model more convenient, understandable and faster.

Definition 1. Class *PetriSim* is a type of object that specifies the field describing Petri net construction and methods implementing transformations of the net according to the rules of stochastic multichannel Petri net, detailed mathematical description of which has been represented in work [11].

The objects, created by the constructors of the class *PetriSim*, generate a set of objects Θ using data set, corresponding to constructor arguments. The simplest constructor contains the value in which Petri net is determined.

Definition 2. Petri-object is an object of the class *PetriSim* or its descendants (or an object of the type *PetriSim*):

$$O = PetriSim(a) \tag{1}$$

where O is Petri-object, $\mathrm{PetriSim(a)}$ is a constructor of the class, a is a set of values of constructor arguments among which there is the Petri net description.

To link Petri-objects, two types of connection are provided. The first one is to use shared places. It is defined to link pairs of objects by determining one or more places shared (sometimes called places fusion). The second one is to use event initialization. It is defined to link an object with a group of objects by determining additional output arcs for the transition of the object which are directed to the places belonging to the group of objects. Thus, two types of connections could be regarded as 'one-to-one' and 'one-to-many' connections.

Definition 3. Petri-object model is the model composed of the connected Petri-objects:

$$\mathrm{Model} = \bigcup_{\mathrm{j}} \mathrm{O_j}, \qquad \mathrm{O_j} = \mathrm{PetriSim(a_j)}\ , \tag{2}$$

where $\mathrm{O_j} \in \Theta$ is an object of the type *PetriSim*, $\mathrm{a_j}$ is a set of arguments including Petri net.

In terms of UML, the structure of Petri-object model is the aggregation of classes that inherits *PetriSim* class or the class *PetriSim*:

$$Model \diamond\!\!- ((\{C_k, k = 1, \dots L\} \vartriangleleft\!\!- PetriSim) \vee PetriSim), \tag{3}$$

where $\diamond\!\!-$ denotes aggregation, $\vartriangleleft\!\!-$ denotes inheritance, C_k is a class of Petri-objects, L is the number of classes.

In work [Stetsenko, 2011], the following fundamental statements were proved.

Statement 1. The Petri-object model functioning is described by the stochastic Petri net obtained by the union of all nets of Petri-objects the model has been composed:

$$N = \bigcup_j \widetilde{N}_j \tag{4}$$

where the union of nets means the union of places, transitions and arcs sets of these nets, $\widetilde{N}_j$ is the stochastic multichannel Petri net of Petri-object O_j united with additional output arcs in case if 'one-to-many' connection has been used to connect the object. In case of using only 'one-to-one' connection $\widetilde{N}_j = N_j$.

Statement 2. The transformation of the net of Petri-object model is divided on transformations of the nets of Petri-objects the model has been composed:

$$D^-(\boldsymbol{S}) = \{D^-(\tilde{\boldsymbol{S}}_j), j = 1, \dots q\}, \qquad D^+(\boldsymbol{S}) = \{D^+(\tilde{\boldsymbol{S}}_j), j = 1, .. q\}, \tag{5}$$

where $\boldsymbol{S}$ is the vector of model state, containing the state of places $\boldsymbol{M}$ and the state of transitions $\boldsymbol{E}$, D^- and D^+ are the transformations corresponding to tokens input and output in Petri net, that are described by logic-algebraic equations.

Statement 3. The Petri-object model simulation is determined by the state equations, one of which determines the moving of the current time to the nearest event and the other determines the transformation of the state of the Petri net elements:

$$t_n = \min \boldsymbol{E}(t_{n-1}), \quad \boldsymbol{S}(t_n) = (D^-)^m\big(D^+(\boldsymbol{S}(t_{n-1}))\big) \tag{6}$$

where t_n is the current time, $t_n \geq t_{n-1}$, $\boldsymbol{E}(t_{n-1})$ is the state of transitions in the previous moment of time, m is the number of repeated tokens input needed to achieve the state of the net in which all transitions are not firing.

The process of construction of Petri-object model, divided on four steps, is represented in Fig.1 on the widely known model of 'dining philosophers', that demonstrates the using of shared resources by synchronized processes. The model is composed of objects, each of which simulates the behaviour of philosopher trying to catch two sticks to start to eat. Because the total number of sticks is equal to the number of philosophers, one part of philosophers will wait for 'resources' while another part of philosophers will 'process'. In Fig.1, at start moment each philosopher has one stick. Shared places marked with gray color implement the links between objects.

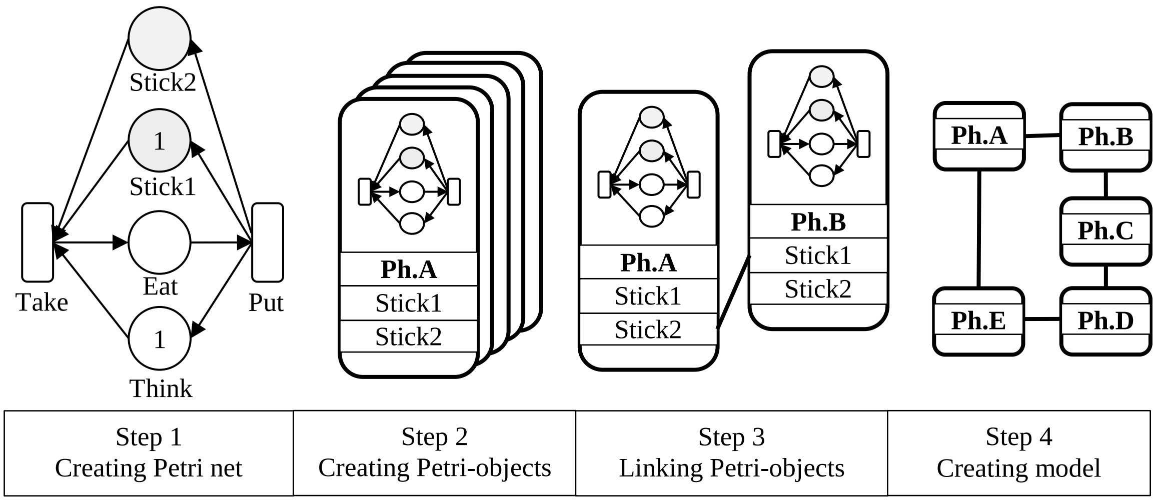


Fig. 1. The dining philosophers model.

4.2. Comparison with known Petri nets

Stochastic Petri net is a very convenient tool to investigate discrete event dynamic systems. However, its modelling power decreases if the number of events significantly increases. Multiple arcs create a confused and incomprehensible net which is difficult to edit and modify. That is why many extensions were developed to overcome this drawback: object-oriented Petri net (place, transition and net is able to be an object) [12], nets within nets (a place is able to contain a net) [13], hierarchical Petri net (a transition is able to contain a net) [14], colored Petri net (use types for markers) [15].

Coloured Petri net was developed to simplify (or reduce) the description of a complicated system which has a lot of elements. Petri-object model has the same aim. CPN Tool combine coloured Petri net with ML programming language while Petri-object simulation combine Petri net with object-oriented programming.

Comparing the representation of the model in Fig.1 with the representation of the same model by colored Petri net [16], it should be noted that the links between elements are created in Petri-object model. It makes the perception of the model structure clearer. Coloured Petri net hides a lot of information about interaction between elements into expressions on arcs and on value declarations that makes model less readable.

Stochastic Petri net does not provide replication of subnets or save information about markers. It does not allow zero value for a time delay or priority value for transition. Colored stochastic Petri net supports subnet replication but does not support subnet replication with given parameters. It provides links between subnets but only by one pair of share places. Simultaneously Petri-object model allows using as much as possible links between objects. In addition, it allows linking between one object and a group of objects.

In addition, object-oriented structure of Petri-object model has multiple advantages. Creating class inherited PetriSim class the user is able to add fields and methods which are specific for the problem under consideration. For example, the methods gathering the information about simulation results are often useful.

4.3. Petri-object model simulation algorithm

Simulation algorithm of Petri-object model should implement the state equation (6). The repeated actions of time moving and transformation the state of Petri net are performed until the simulation time is achieved. Due to the model transformation is divided on the transformations of the Petri-object nets, the complexity of the simulation algorithm of the Petri-object model is significantly less than the simulation algorithm of stochastic Petri net.

5. Results

5.1 Petri-object model simulation software

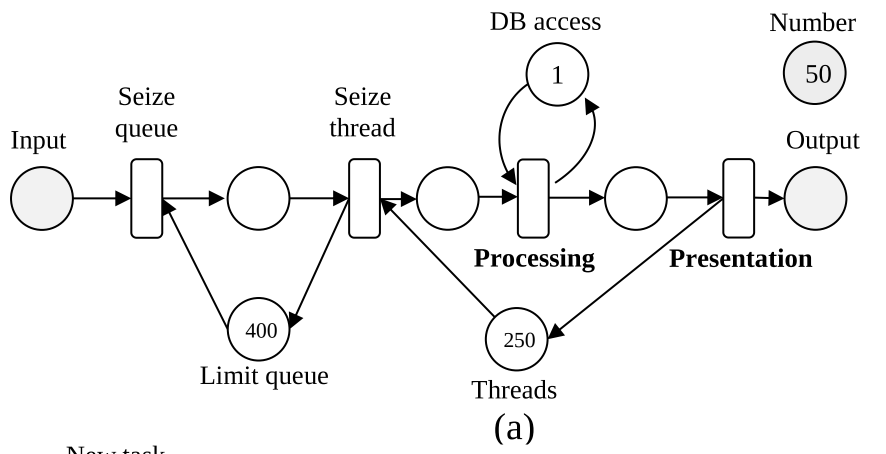
Petri-object simulation software is developed using Java language [17]. It consists of package `PetriObjLib` implementing Petri-object simulation algorithm and the packages providing graphical presentation of nets. The graphical editor helps to build Petri net, save it as a Java method and add method to the `NetLibrary` class. The opening net from file or reproducing net from Java method are supported by the software. In addition, animation of simulation is provided to check the rightness of created Petri net. It should be noted, exception will be generated if Petri net consists a transition which has not input or output places. After successful saving, the method can be used to create Petri-objects. When the list of Petri-objects is prepared and the links between objects are determined, the model can be created using `PetriObjModel` class. The method `go(double time)` of this class run the simulation. Exception will be generated if a time delay generator will produce a negative value.

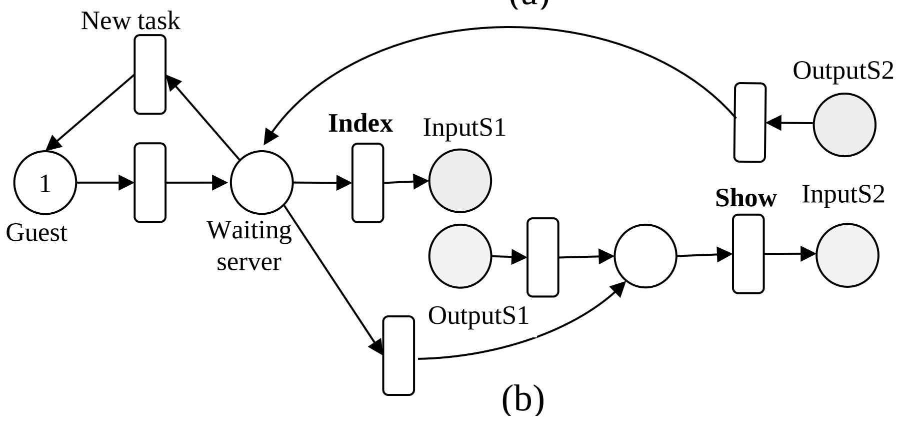
Software main responsibilities are to provide correct simulation algorithm and correct simulation results including mean values of markers in Petri net places, mean value of buffers in transitions and the state of Petri net in the last moment of simulation.

5.2 Web information system Petri-object model creation and simulation.

To consider the example of simulation, the model of web information system based on REST architectural constraints, has been chosen. An ordinary client-server architecture based system includes a client-side application for users and a server-side application for handling user’s requests. There are two types of users: authors and guests. The basic flow for authors consists of following actions: ‘create’, ‘update’, ‘index’, ‘show’, ‘delete’. Guests have only two actions: ‘index’ and ‘show’. Each action is implemented by corresponding request which needs to be handled by the server-side application. Server handles request in concurrent way by processing many requests simultaneously. Request processing consist of following operations: processing data (retrieving data from database, mapping, sorting, calculating, etc.) and rendering response. The common problem of information systems is long-running request, which is influenced by many factors such as number of users, number of simultaneously processed request, data processing algorithms, etc. The server load directly affects the response time which is very critical value for user satisfaction and need to be optimized. It can be a quite complicated task to realize what exactly affects the response time, especially in case of real-world large information systems. With the help of simulation technique, the model can be construct and investigated by making experiments.

Utilizing Petri-object model approach the following classes of objects should be defined: Author, Guest, Server. Corresponding Petri nets are represented in Fig. 2. Share places are used to link objects. The model composing of linked objects is represented in Fig.3.





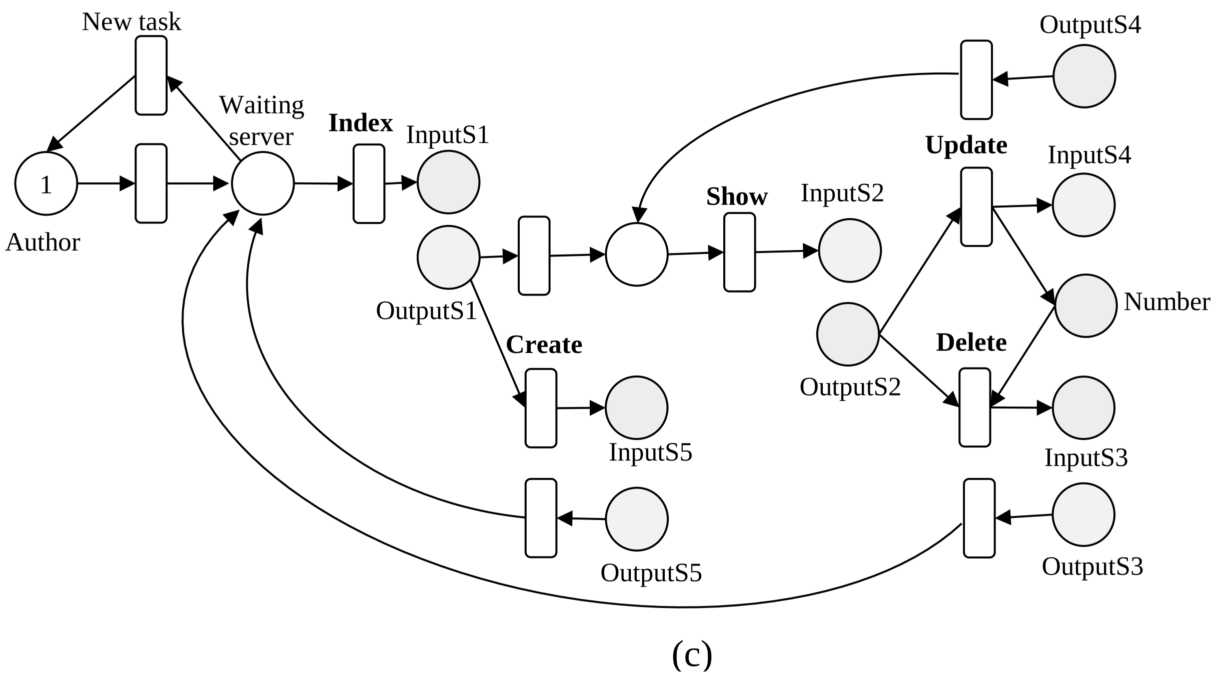


Fig. 2. The nets of Petri-objects Server (a), Guest (b), Author (c).

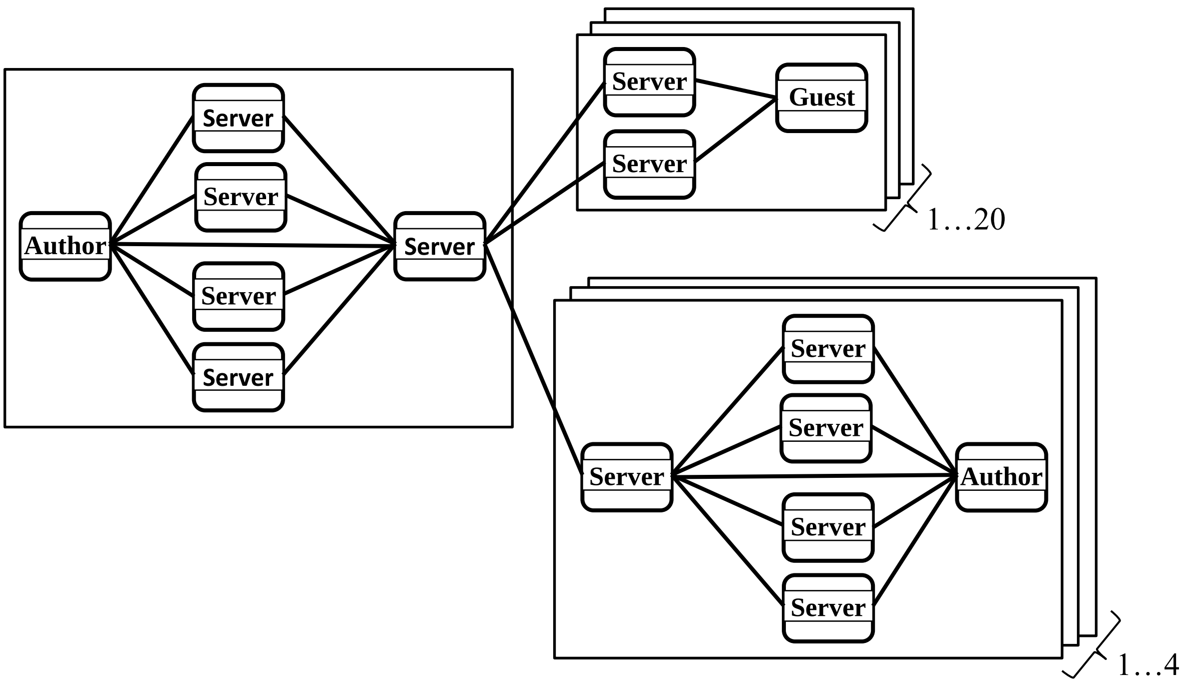


Fig. 3. Composing the information system model of Petri-objects Author, Guest, Server.

The experimental investigation of the information system parameters impact on response time is represented in Fig. 4. In the first experiment the ratio of rendering time to processing time was 10^6. In the second experiment the same ratio was decreased to 10^5. If the rendering time is much more than the processing time, the increasing of the number of threads helps to decrease significantly the response time until the quantity of threads becomes sufficient for the requests flow. Further increasing the quantity of threads leads to unproductive use of server resources. Reduction of the rendering time cause changes in system behavior. In this case the processing time is the main reason for occurring a long-running request because the presentation is fast enough. The increasing of the number of threads leads to increasing the response time because more requests are able to start processing and, as a sequence, the waiting time in the data base queue is grow. When the number of threads is large enough for requests flow, there is not further increasing of response time and it is an unproductive use of resources.

Thus, based on the simulation results, the decision can be made for each set of system parameters. For example, in case of a large number of users with author rights a shorter response time can be achieved if the ratio of rendering time to processing time is large enough and the number of available threads is sufficient. Conversely, when information system is used by a small number of users, a shorter response time can be achieved is the ratio of rendering time to processing time is not so big.

5.3 Algorithm complexity investigation

In case of models with hundreds of elements and connections, traditional simulation techniques cannot be applied because of fast growing computational complexity and, as a sequence, the performance time. However, Petri-object simulation algorithm complexity is polynomial. The experiment conducted on the model of information system confirms this fact (Fig.5).

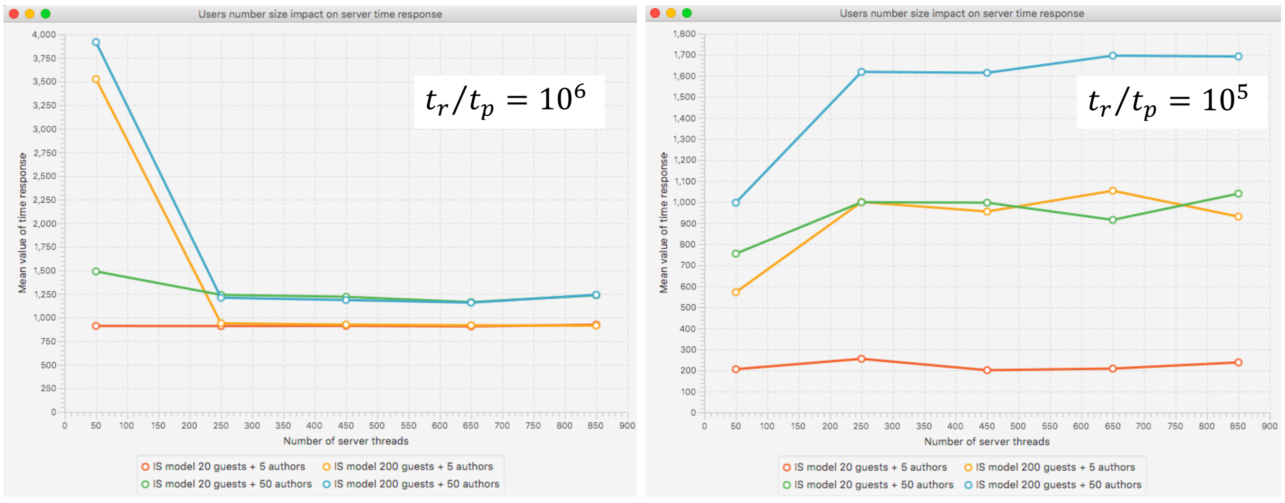


Fig. 4. The information system parameters (the number of users, the ratio of rendering time tr to processing time tp, the number of threads) impact on the response time.

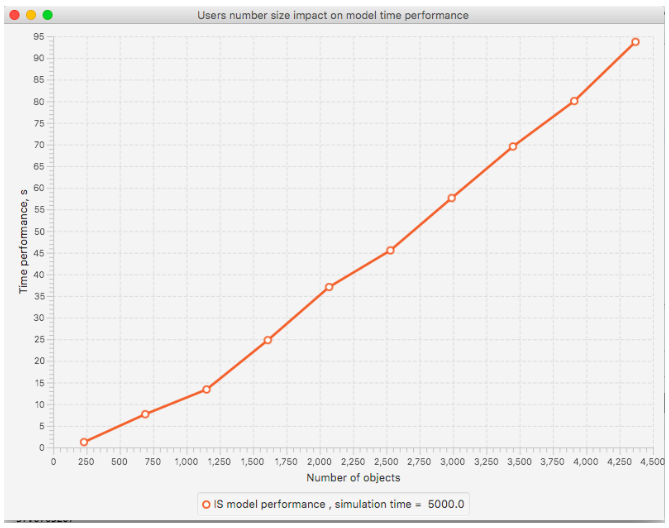


Fig. 5. The model performance time experimental investigation.

6. Conclusions

Thus, Petri-object simulation is a powerful technique to create simulation models of complicated discrete event systems. It provides fast replication the nets of typical model elements with given parameters, construction the model in a way that its behavior is described by the Petri net too, scalable simulation algorithm due to dividing transformation of the model net on transformation of the model elements net. The nets, building with graphical editor, Petri-objects, and the whole model are saved as a Java code that provides the model fast integration with other components of software and its flexible usage.

In particular, the model of web information system, based on REST architectural constraints, is developed by means of Petri-object simulation. Simulation results show the system parameters impact on response time. For example, the large number of threads could not decrease the response time (and avoid long-running request) if the ratio of the rendering time to processing time of data is not sufficiently large. Thus, simulation is able to reveal the drawbacks in information system design.

Further investigation aims to extend Petri-object simulation technology by adding nested Petri-objects which will help to simulate more complicated systems.

References

1. IBM Center for The Business of Government. Kamensky, J.: Data rich, but information poor. (2018). http://www.businessofgovernment.org/blog/data-rich-information-poor, last accessed 2020/08/28.
2. ISO/IEC 15909-1:2004 Systems and software engineering — High-level Petri nets — Part 1: Concepts, definitions and graphical notation. https://www.iso.org/standard/38225.html, last accessed 2020/08/28.
3. Law A. M.: Simulation modelling and analysis. President Averill M. Law & Associates, Inc.Tucson, Arizona, USA, www.averill-law.com. 5th Edition. — New York: McGraw Hill (2015).
4. Haas, P.J.: Stochastic Petri net: modelling, stability, simulation. Springer-Verlag New York (2002).
5. Balbo, G.: Introduction to Generalized Stochastic Petri Nets. In: Bernardo M., Hillston J. (eds) Formal Methods for Performance Evaluation. SFM 2007. Lecture Notes in Computer Science, vol 4486. Springer, Berlin, Heidelberg (2007).
6. Jensen, K.: Coloured Petri Nets. Basic Concepts, Analysis Methods and Practical Use. 2th edition. Springer-Verlag Berlin Heidelberg (1996).
7. Zhao, N., Yu, Y., Wang, J. et al. An extended object-oriented Petri net supporting the description and evolution of components: EOOPN. Cluster Computing 22, 2701–2708 (2019).
8. Silva, P.S. et al.: Simulation in Information Systems: Potential of the Vulnerability Theory. In: Quintela Varajão J.E., Cruz-Cunha M.M., Putnik G.D., Trigo A. (eds) ENTERprise Information Systems. CENTERIS 2010. Communications in Computer and Information Science, vol 109. Springer, Berlin, Heidelberg (2010).
9. van der Werf, J.M.E.M., Polyvyanyy, A.: The Information Systems Modeling Suite. In: Janicki R., Sidorova N., Chatain T. (eds) Application and Theory of Petri Nets and Concurrency. PETRI NETS 2020. Lecture Notes in Computer Science, vol. 12152. Springer, Cham (2020).
10. Stetsenko, I.V.: State equations of stochastic timed petri nets with informational relations. Cybernetics and Systems Analysis 48(5), 784-797 (2012).
11. Stetsenko, I.V.: Theoretical Foundations of Petri-object Modeling of Systems. Mathematical Machines and Systems 4, 136-148 (2011). (In Russian)
12. Miyamoto, T.: A Survey of Object-Oriented Petri Nets and Analysis Methods. IEICE Transactions on Fundamentals of Electronics Communications and Computer Sciences E88–A (11), 2964-2971 (2005).
13. Valk, R.: Object Petri Nets. In: Desel J., Reisig W., Rozenberg G. (eds) Lectures on Concurrency and Petri Nets. ACPN 2003. Lecture Notes in Computer Science, vol 3098. Springer, Berlin, Heidelberg (2004).
14. Fehling, R.: A concept of hierarchical Petri nets with building blocks. In: Rozenberg G. (eds) Advances in Petri Nets 1993. ICATPN 1991. Lecture Notes in Computer Science, vol 674. Springer, Berlin, Heidelberg (1993).
15. Jensen K., Kristensen L. M.: Coloured Petri Nets: Modelling and Validation of Concurrent Systems. Springer-Verlag Berlin Heidelberg (2009).
16. CPN Tools. Dining Philosophers. http://cpntools.org/wp-content/uploads/2018/01/diningphilosophers.pdf, last accessed 2020/08/28.
17. GitHub. https://github.com/StetsenkoInna/PetriObjModelPaint, last accessed 2020/08/28.

UDC 004.67

KOPIIKA A.,
PISKUN R.,
TKACHENKO V.,
KLYMENKO I.

ROAD MONITORING SYSTEM BASED ON IOT TECHNOLOGY FOR SMARTCITY

This work is devoted to the problem of automatic road quality control, which can be used by both road repair services and common drivers. This paper provides a survey of different known techniques and algorithms of finding potholes on the road and describe our own method, using accelerometer. It will be shown our device for detecting potholes on the road, which can be used, as a part of the IoT system of SmartCity. It uses data from an accelerometer for finding road bumps.

Keywords: Internet of Things, SmartCity, Potholes detection, Accelerometer, SPI, STM32, Road surface analysis.

1. Introduction

What do most of big cities have in common? They are all expected to spend enormous sums on various smart city initiatives. Road infrastructure has not stayed away from innovation. About one-third of Pittsburgh's intersections will be equipped with smart traffic signals in the coming years; sensors at the intersections determine traffic volume and adjust stop-and-go times based on the number of vehicles present. Since the project's inception, wait times at intersections are down by 41% and vehicle emissions have been reduced by 21%. The city of Dallas is also currently implementing an IoT-enabled traffic management system in hopes of better managing road congestion.

But despite common opinion, the search for the most convenient route does not always correspond to the search for the shortest route, especially when you are planning to go on a road unknown to you. It may turn out that the road you have chosen is in poor condition, so the trip will be less convenient and the time spent will increase significantly.

All countries around the world suffer enormous losses due to road damage every day. Firstly, millions are spent on repairing and "patching" damaged sections of roads. Secondly, driving on uneven surfaces increases the amount of fuel consumed and more often leads to the need for maintenance. We should also not to forget that potholes on the roads are often one of the causes of accidents and lead to injuries and sometimes even death. And at least, the poor quality of the road lane irritates drivers and creates some discomfort while driving.

Keeping roads in good condition is quite a challenge, due to many factors, which affect the damage. Among them are both natural and man-made. The budget of road repair works directly depends on the speed of their reaction to damage. It is almost impossible to ensure quality control of roads with the help of static sensors or human monitoring. Due to the extremely long length of roads, such systems require funds that could be used for repairs.

Every year the concept of the Internet of Things is gaining popularity. The essence of this principle is the exchange of data between sensors and household items for human safety or simplification of life. Especially popular is the concept of "SmartCity", in which the elements of the infrastructure communicate with each other [1]. As a result, it has been suggested that the ability to control the quality of the road surface as an element of such a system would be an extremely useful addition [2].

Given all the above, it is safe to say that the solution to this problem is the ability of vehicles to find the problem areas while driving. To implement such a system, it is only necessary to obtain the values of linear acceleration, location, and the ability to transmit this data for display in a convenient form.

In this paper we 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 known systems. 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.

2. Related works

Due to survey [2] interest to giving intelligence to vehicles increase rapidly in area of smart applications and algorithms of indication environment around them. Especially, technologies of communication of cars among themselves and with elements of infrastructure extends. So it will be grate, if one car can provide information about quality of road surface to another.

To choose the best way to identify problem areas of the road, we considered the existing analogs of the systems, which also dealt with this issue, namely, what characteristics they relied on to decide on the quality of the road surface.

The costliest way to determine the quality of roads is to use road labs, mobile data, and roadside data collection centers. Depending on the set purposes, such cars equipped by a large number of sensors (several video cameras, GPS receiver, three-dimensional laser scanner, soil analysis system, positioning unit). The main part of the analysis is non-autonomous and proceed after driving. Such a method is more scientific but requires a special person to process data.

Another option presented in [3] is determining potholes and cracks on the roads by finding them using data in video format. The basis of the search for potholes and patches in this method is the use of the method of active circuits. The essence of the method is to select the contour of the object at a given point, which exactly belongs to the object. The intensity dispersion in its inner region is considered as the initiator for determining the damage. This method has some critical shortcomings. Firstly, already repaired areas that differ in color or covered with a shadow are also can be detected as potholes. Also, the original video gives a high error due to problems with perspective distortion of shooting.

The next option for solving this problem described in [4] - a mobile service to assess the road surface. To join this program, just install the application on your mobile phone, register, allow access to GPS and the accelerometer built into each phone and place the phone on the desk of your car. All modern smartphones can measure the force of "shaking" the phone with an accelerometer. This allows you to turn the machine into a mobile laboratory without any additional spending. Collected information is processed using mathematical algorithms. Also, as an option, the possibility of using artificial intelligence for search is considered [5]. The GPS sensor finds the current location and displays the quality on the map. Such a method is available to all car owners, but this option of data collection will quickly discharge your mobile phone because the long use of the GPS sensor is very resource-intensive. It is also worth noting that moving the phone will give false results, so you can’t use your phone while traveling, for example, as a navigator or for other purposes.

The closest technology was used in BusNet [6] system developed at the University of Colombo and in Pothole Patrol system [7] developed at Massachusetts Institute of Technology. They create their own environment using a 3-axes accelerator and GPS module to store data and predict potholes in the traffic system of their hometown using social transport (buses/taxi). Unlike others, the system in [6] doesn’t work in real-time. The data is stored on local memory and transmitted to the database at the bus station for later processing. The main difference of Pothole Patrol [7] system is using machine learning algorithms consisting of 5 different filters for getting more truthful information and rejecting not related occasions such as railway crossings etc.

3. The purpose and objectives of the research

After analysis of related works, we decided to create our own device which finds potholes on a road surface, as a part of IoT road monitoring system. The main point of our work is to make pure automatic road quality control system with high accuracy of gained result for decent price which can be built in any car. There are a few steps which we made to reach our goal.

First of all, we choose hardware and peripherals which would meets our needs to create our system and device for finding potholes in particular and developed device and system architecture.

The next step was to develop an algorithm for spotting road bumps. On this phase we implemented methods for reading data from sensors, data processing, communication between device and cloud storage, processing the result.

The final stage of the work was the implementation of the system of finding pits on the roads using a linear acceleration sensor with the following functions: removal of indicators from the sensor, filtering noise and errors, sending data to cloud storage with further processing. According to the results of the created system it was carried out testing.

4. Materials and methods of reserch

4.1. IoT system architecture

In the book Designing the Internet of Things [12], the elements of the IoT are presented as a simple equation:

Physical Objects + Controllers, Sensors, Actuators + Internet = IoT.

This equation clearly explains the nature of the Internet of Things. Standard IoT system consists of a collection of physical objects, each of which:

Contains a microcontroller that provides intelligence;

Contains a sensor that measures some physical parameter and/or an actuator that acts on some physical parameter;

Provides a means of communicating via the Internet or some other network.

The role of the object in our system is played by the car. It houses our device, which is a microcontroller with an accelerometer attached to it. The microcontroller exchanges information with server using Ethernet technology. The general architecture of our system consists of hardware and server parts, as shown in Figure 1.

The microcontroller reads data from the accelerometer and performs the primary processing of the received data. During processing, extra data is removed and a data packet is prepared for transmission to the server. The data received by the server undergoes a formatting process. The values obtained from the accelerometer are reduced to units of measurement g = 9.8m / s2. After processing, the results go through a pit search algorithm that identifies possible problem areas of the road surface.

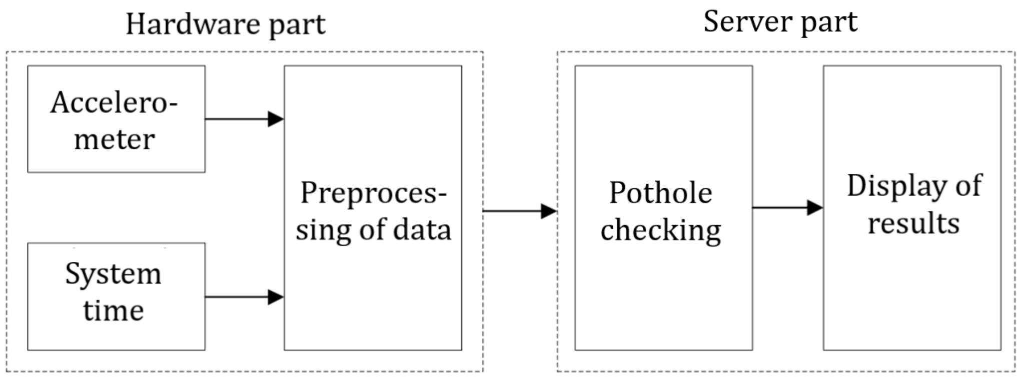


Figure 1. System architecture

The general hardware diagram of the device is presented in figure 2. The reason for this choice is the rapid popularization of the development of SmartCity systems. Dozens of different sensors work in such systems at once, so it is extremely valuable to implement the system so that the developed device can be used as part of a larger project.

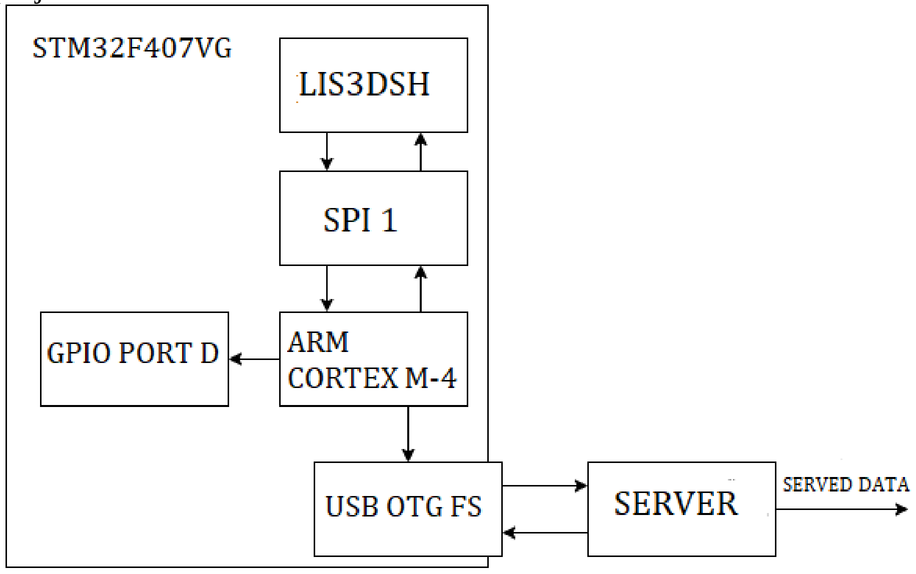


Figure 2. Hardware architecture

First of all, the developed device is implemented on the STM32F4 Discovery microcontroller. STM boards have developed most of the projects related to IoT and work with peripherals in general. The choice of the STM32F4 Discovery board is explained by the fact that it has all the necessary elements to solve the problem. It has low power consumption, high performance, has several interfaces for information exchange, has a large number of free I / O ports. However, there is no hard link to work with this particular board. Project transfer is possible by reconfiguring the I / O ports, clocking, and using the built-in libraries for another microcontroller.

The LIS3DSH linear acceleration sensor is used in the work. First of all, the choice was based on it because this sensor is built into the selected board, but this is not its only advantage. It is highly specialized and therefore does not spend extra energy in its work, which is very important for microprocessor systems. It has high sensitivity and adjusts well to different measurement ranges depending on the tasks.

To receive data from the sensor was selected using bus SPI, as one of the most convenient for receiving data. This choice is due to the simplification of data processing, as this method organizes fast synchronous communication, and can simultaneously implement serial communication between many peripherals and a single microcontroller. Transition of data between device and server occurs using Ethernet. After receiving information server performs bump detection algorithm and displays the result of road testing.

4.2. Bump detection algorithm

During the review of the literature, many ways to identify potholes on the roads were found. In works [8] and [9] were described method which offers to mark the quality of segments of the road but the main point of our work is detecting potholes, not specifying types of road. Therefore, the methods presented in [10] were modified and an own algorithm was created on their basis. Before making an algorithm we should determine the right positioning of the device to gain clear results. Because the main characteristic to define pits is acceleration on Z-axes, the device must be placed parallel to the road.

The algorithm we offer performs a two-level test. First of all, when the car hits a pothole, the car significantly changes its acceleration along the vertical axis. With this in mind, the first method in testing which we named Threshold method is to compare the acceleration value with the threshold value. Since entering the pit of acceleration takes the form of a sinusoid - sharply accelerating, and then just as sharply slowing down like it’s shown on figure 3, it is necessary to set both the upper and lower thresholds of acceleration.

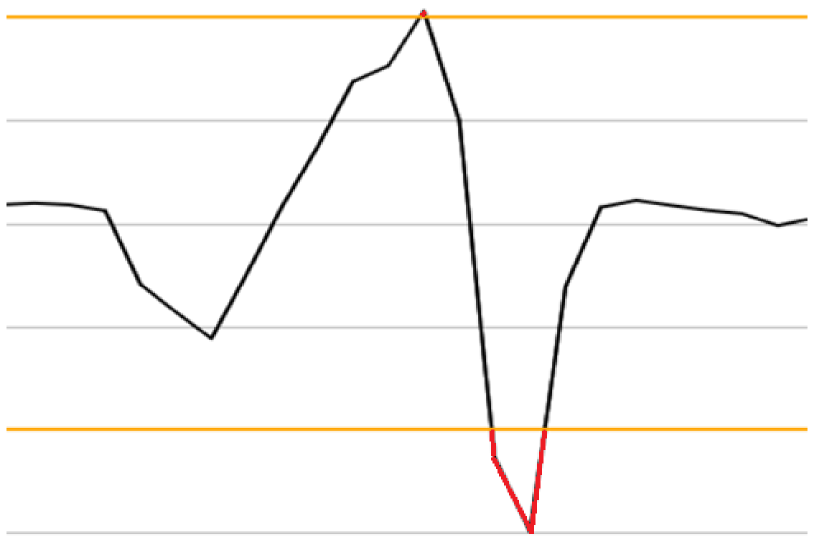


Figure 3. Threshold method

Secondly, the acceleration when entering the pit increases very sharply, so it is possible to detect cracks that may not lead to high performance, but the change between successive measurements will be significant to assume the presence of a defect on the road. Therefore, the second method to check the indicators, which we named Difference method, will be to compare neighboring values in the indicator buffer.

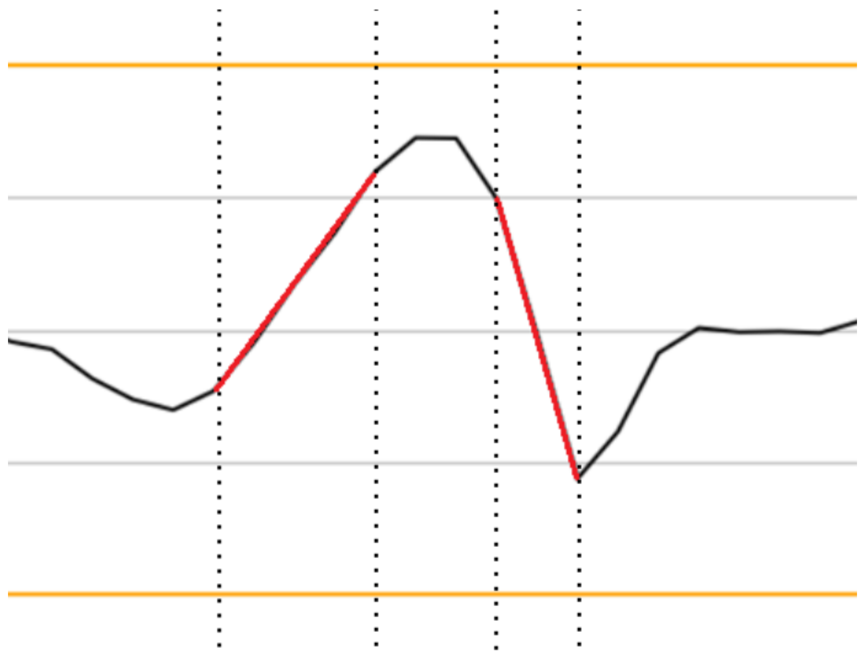


Figure 4 Difference method

5. 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), like in [11]. It helps to find profits and disadvantages of both methods we used.

To get results of work road monitoring device firstly we must found the optimal comparative indicators for pothole search algorithms. Both methods were tested with different comparative values (threshold method – 0.1-1g; difference method - 0.1-0.5g). Threshold method shows the best results using 0.4g as indicator with 76% found bumps. Talking about difference method we find out when indicator is equal to 0.2g algorithm finds 92% of potholes, but it has more false results because it finds pavements and rail crossings as a holes on the road as shown on the table 2. Detailed information about testing described at table 1. All tests were performed at a speed of 60 km per hour. At different speeds the accuracy of operation may vary. For further usage of the system it is necessary to add elements of machine learning to analyze the obtained indicators in the presence of a large sample of results on the same section of the road.

Table 1
Percentage of true founded road roughness

Method	**Threshold method**	**Difference method**
Large potholes	100%	100%
Small potholes	82%	91%
Pothole clusters	80%	90%
Gaps	42%	88%
Total	**76%**	**92%**

Table 2
Percentage of false founded road roughness on “fake gaps”

Method	**Threshold method**	**Difference method**
Pavements	22%	81%
Rail crossings	25%	75%

Given this, it can be assumed that double-checking helps to eliminate possible errors. At the same time, applying multiple inspections of one section of the road each time will improve the accuracy of testing.

The sample of checking is the graph shown in figure 5. The graph is a set of two static and one dynamic lines. The two lines are placed parallel to the X-axis and indicate the maximum allowable values of the deviation of acceleration, which can be considered pits. The third curve shows the processed acceleration indicators along the Z-axis. In the case of finding a sharp acceleration, the algorithm considers such a section as a potential and the graph of the third curve on this segment becomes red.

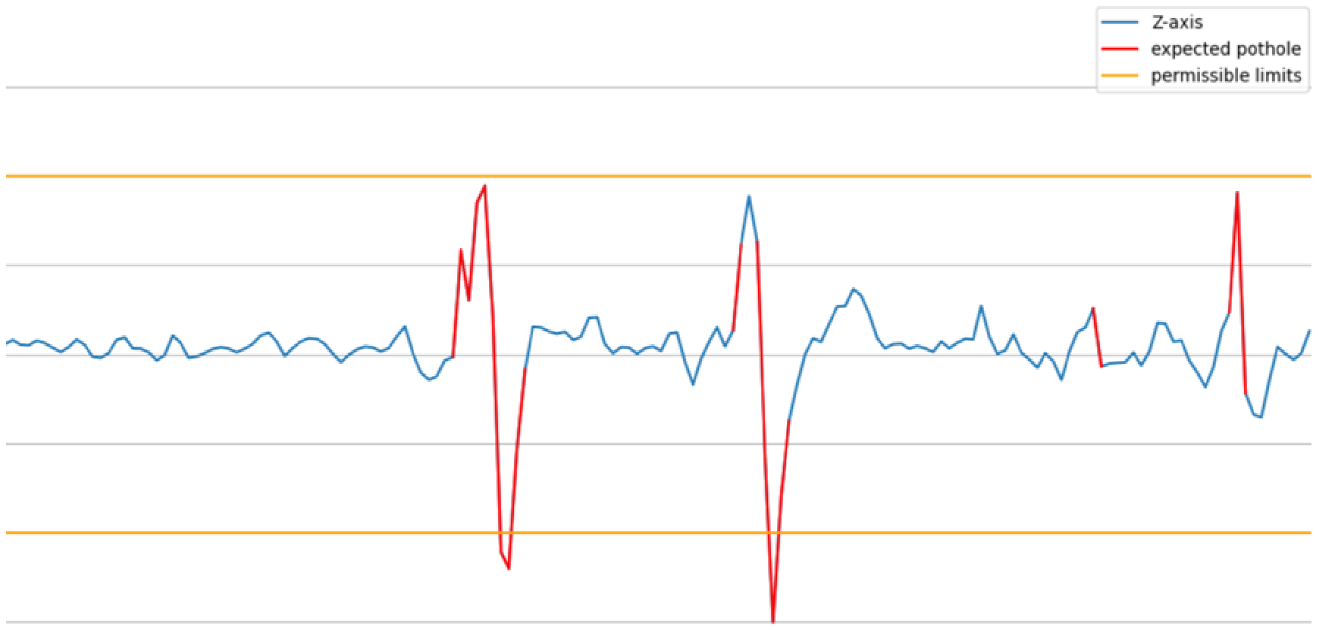


Figure 5. Pothole detection

6. Conclusion

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. The hole search algorithm is a combination of several algorithms to obtain a more accurate result. The essence of the device is the processing of data read from the linear acceleration sensor to identify potholes on the roads and present this information in a convenient form. At this stage, the basic part of the system is developed, which correctly responds to changes in the position and movement of the device. Such hardware 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 complicating the algorithm or using machine learning methods.

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.
9. Chen, K.; Lu, M.; Fan, X.; Wei, M.; Wu, J. Road condition monitoring using on-board three-axis accelerometer and GPS sensor. In Proceedings of the 2011 6th International ICST Conference on Communications and Networking in China (CHINACOM), Harbin, China, 17–19 August 2011; pp. 1032–1037
10. Prof.B.Lanjewar, Jyoti Khedkar, Rahul Sagar, Rasika Pawar, Kunal Gosavi. Survey of Road Bump and Intensity Detection algorithms using Smartphone Sensors. B.Lanjewar et al, / (IJCSIT) International Journal of Computer Science and Information Technologies, Vol. 6 (6) , 2015, pp. 5133-5136.
11. Mednis, A.; Strazdins, G.; Zviedris, R.; Kanonirs, G.; Selavo, L. Real time pothole detection using android smartphones with accelerometers. In Proceedings of the 2011 International Conference on Distributed Computing in Sensor Systems and Workshops (DCOSS), Barcelona, Spain, 27–29 June 2011; pp. 1–6.
12. Adrian McEwen Designing the Internet of Things / Adrian McEwen, Hakim Cassimally; John Wiley & Sons, 2013.- 336.

UDC 004.35 004.78 004.454

MURAVIOV I.,
TARANIUK V.,
KLYMENKO I.

MAKING AN IOT DEVELOPMENT PLATFORM FROM A SIMPLE MICROCONTROLLER DEMONSTRATION BOARD

The IOT is now in trend. Because of its huge popularity and business interest to it, IOT is only now coming massively to universities worldwide as a separate study. This article provides an IOT solution based on embedded technologies that were not specifically designed for IOT. The core of this solution - board is developed for studying peripherals of a particular general-purpose MCU. But we successfully adapted it to use in IOT and for study of IOT systems.

Keywords: Internet of things, wireless connectivity, Wi-Fi, embedded platform, AT commands.

1 Introduction

Thing in IOT (Internet of things) is a device, that interacts with outer world using physical processes and connectivity with other devices. In other words, thing is an embedded system, connected to the Internet and has a unique way of identification and can be addressed, IOT gives an opportunity to have two forms of communication: human-thing and thing-thing (machine to machine) [1]. Development of IOT today is development of such things, specific hardware architectures those include wireless connectivity, development of connection protocols used for interaction between such things, development of back-end, cloud systems for managing things remotely, and front-end for connecting things to humans. Things can control, as examples of such things can be any remote sensors: gas sensor, remote measuring instruments, such as thermometer, barometer. The examples of things those can be controlled are switchers, relays, lamps.

Of course, the connection between two things can be of any nature, but if a particular thing is declared by manufacturer as part of IOT, it is usually supposed that it supports wireless connectivity. Wireless connectivity is much more flexible, than wired and usually cheaper. But it was confirmed, that huge infrastructures with things, connected to their hubs, those are connected to clouds, clients are efficient if constructed as mutually complementary network systems between wireless and wired communications [2]. The wireless connectivity used in IOT is divided to 2 types of communication protocols: short range protocols and low power wide area networks [3].

Embedded world is huge of different manufacturers of embedded systems, such as single board computers, MCUs (microcontroller units). A typical embedded developer is given with such set, is studying their production, learns programming for this specific production, for example special purpose wireless microcontroller, then develops application – solution to a given problem. The IOT is a little bit different to embedded. There is less documentation available to the hardware and it is more closed. Part of information is usually for security purpose or because of commercial considerations. Sometimes only a communication protocol with it is provided, accordingly, this embedded system is then called a black box, into which signals can be sent and response signals at the output can be received [4]. In this article given solution uses two distinctive parts: embedded system, which is provided with full available documentation how it works, which is connected to another embedded system, which is used as black box, but it gives an ability to connect solution to the IOT. The solution used in this article is built with most popular technologies and platforms. It also shows that it is not so difficult to connect already developed embedded solution to the internet even if it wasn’t intended for this initially.

2 Related work

Today we have many of different development boards from different manufacturers those push their production to the market. Usually each development kit comes with demonstration solution, code base, documentation for very specific solution. There are numbers of examples of embedded systems with host controller and slave device, for example wireless connectivity module with single board computer [5] or MCU [6] as host. There exist some IOT development boards, based on general-purpose microcontrollers as hosts [7].

Any MCU with enough resources can be used as host that controls wireless connectivity module. The most popular choise of general–purpose MCUs those work well as hosts are ARM (Ashton Raggatt McDougall) architecture industry-standard core based STM32 MCU family [6] [8] [9].

Because of its popularity this family supports wide variety of RTOSes(real-time operating systems), a particular one is open source and one of most popular on different architecuters. It is FreeRTOS, that is used in variety of different MCU projects, there exists port of FreeRTOS, that supports Amazon cloud services, it is Amazon FreeRTOS, that is also used in a variety of IOT projects [9]. In this article no cloud services were used, so the mainstream FreeRTOS was used as host MCU RTOS.

3 The purpose and objectives of the study

The purpose of this study is to find out a way to simplify the connection of an arbitrary embedded system to IOT. This article shows one of ways of studying the world of IOT using just one MCU based embedded system and how to make their conception and code more portable.

The developed software solution can be easily ported to any other MCU-based sensor.

There exist devices with sensors and display inside. The ported solution to any particular of such devices can give remote web interface to this particular device and remote control of its sensors and display via web interface via local network.

The solution may be used for remote notification boards with displays, for example in railway stations for train arrival notifications via web interface.

4 Materials and methods of research

A Global Logic starter kit was used as a device with embedded symbolic display and sensors, such as temperature sensor, accelerometer etc. It is connected with STM32F407-DISC1 Discovery board with STM32F407VGT6 microcontroller inside. This was an embedded solution with no software and hardware examples provided for connecting it to the IOT.

In this solution we used Wi-Fi networking technology, as it can be used in applications with varying levels of power consumption and signal range and now is one of the most popular [10] networking technologies in IOT.

To implement Wi-Fi connectivity, one of the most popular transceivers was used, it is ESP8266 wireless module [11]. we used its factory firmware, that supports opening TCP/IP (Transmission Control Protocol/Internet Protocol) connection.

With the standard firmware ESP8266 and ESP32 modules communicate via AT (Attention) commands. AT commands are one of simple interfaces used for communication with host and slave devices. ESP8266 uses UART (Universal Asynchronous Receiver Transmitter) to communicate with its host. Any MCU with UART support can interface ESP8266 connection [12]. This command set is also known as Hayes command set, that was originally developed for smart modems and today is still used in various modems [4] [13]. AT command set was chosen to communicate with ESP8266 because it is easy to parse expressions of this language and generate them.

The code base for solution was written in the C programming language that is de-facto standard for programming embedded systems.

5 Design hardware of IOT solution

So, we take a particular device with microcontroller or microcomputer inside and place support for AT commands for a particular module. So, the Global Logic Starter Kit, based on STM32F407-DISC1 Discovery [14] board was used as sensors board with the main MCU which was connected by the UART interface to ESP8266 Wi-Fi module.

The MCU (microcontroller unit) - STM32F407VGT6 is connected with ESP8266 module via UART interface. Schematic for connection between starter kit and ESP8266 you can see in figure 1. The MCU with its initialized pins PD8, PD9 for UART number 3 is connected to default UART pins of ESP8266. One wire of UART bus is for transfer data from MCU to slave and another for reverse direction. MCU can send AT commands as string of characters via UART and receive responses via interrupts from UART.

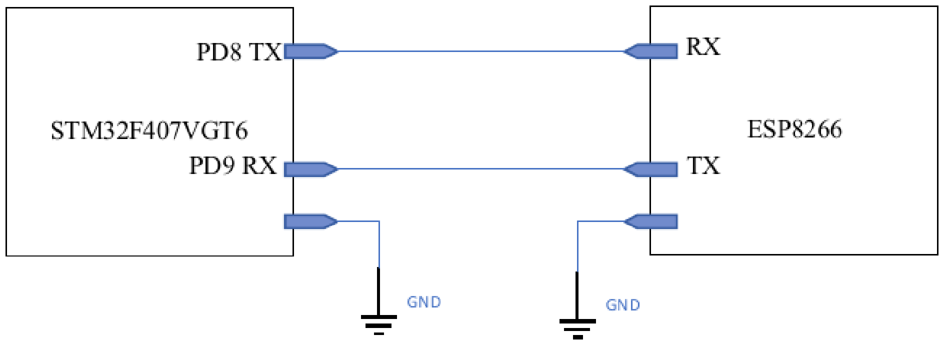


Figure 1. The connection diagram between MCU and ESP8266

6 Design software of the IOT solution

The task, that interfaces host and slave device is made as FreeRTOS thread. This thread uses the library that can generate AT commands, send them, parse responses from ESP8266, that was also written by us. After startup main task initializes ESP8266 using AT command set, connects to the Wi-Fi given in settings of firmware and starts server mode. After that it waits for connection from client. If connected client is a web browser, it sends HTTP requests, in particular to receive the web page, they are received by the interrupt handler of MCU from UART of ESP8266 and given via a message passing interface from interrupt to the ESP8266 interfacing thread in MCU. This thread gets identifier of connected client and sends it a generated web page. The library that has an ability to statically generate HTML (hypertext markup language) web pages on MCU with given parameters and settings in header files is also written by us. After generating it places web page as character string in statically defined RAM (random access memory) buffer. After that thread sends this string to ESP8266. The web browser can request to edit input form if it is provided in web page and user added or modified data in this form or pressed a button given in this page. It sends HTTP request with identification of that form and the value result. We added the ability to retrieve this result and save the value to the variable associated with the input form.

The FreeRTOS port for STM32 is built upon HAL (hardware abstraction layer) and CMSIS (cortex microcontroller software interface standard). These layers are also used for creating drivers for STM32F4 Discovery board and Global Logic Starter Kit peripherals. Drivers for symbolic display, ADC (analog to digital converter) thermometer, accelerometer were created and connected with the built web server via its configuration. The given schematic for software solution is shown on figure 2. Drivers collect data to a given structure, which contains also state machine of the web server. Then data is copied to a formed web page – text on RAM buffer. And another task configures ESP8266 for initializing server and connecting to the internet. If ESP8266 is successfully connected, task, handling HTTP and any other requests from ESP8266, may receive connection from the other device. When it comes true, connection is established, and the web page is updated by new values and transferred to the client.

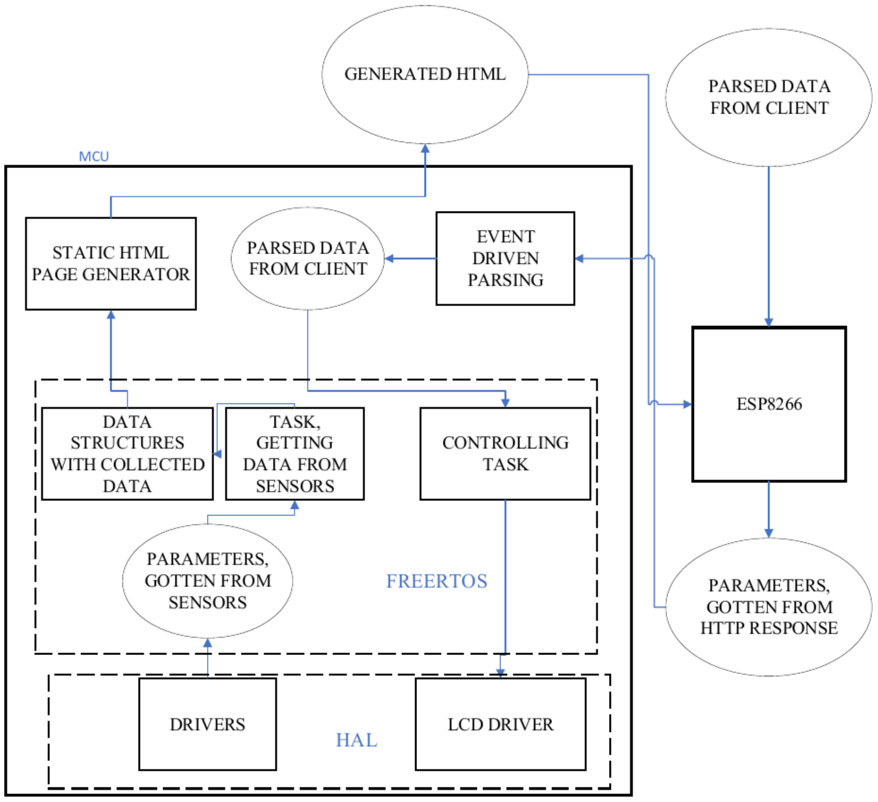

GENERATED HTML
PARSED DATA FROM CLIENT
MCU
STATIC HTML PAGE GENERATOR
PARSED DATA FROM CLIENT
EVENT DRIVEN PARSING
ESP8266
DATA STRUCTURES WITH COLLECTED DATA
TASK, GETTING DATA FROM SENSORS
CONTROLLING TASK
PARAMETERS, GOTTEN FROM SENSORS
FREERTOS
PARAMETERS, GOTTEN FROM HTTP RESPONSE
DRIVERS
HAL
LCD DRIVER

Figure 2. Schematic for software solution

7 Research results and future work

Research and experiments with given embedded system showed that it is possible to study IOT with standard embedded systems and general-purpose development boards. The platform for IOT creating was developed with software library for building firmware. The software solution can be easily ported to another platform.

The structure of connection of developed solution is provided in figure 3.

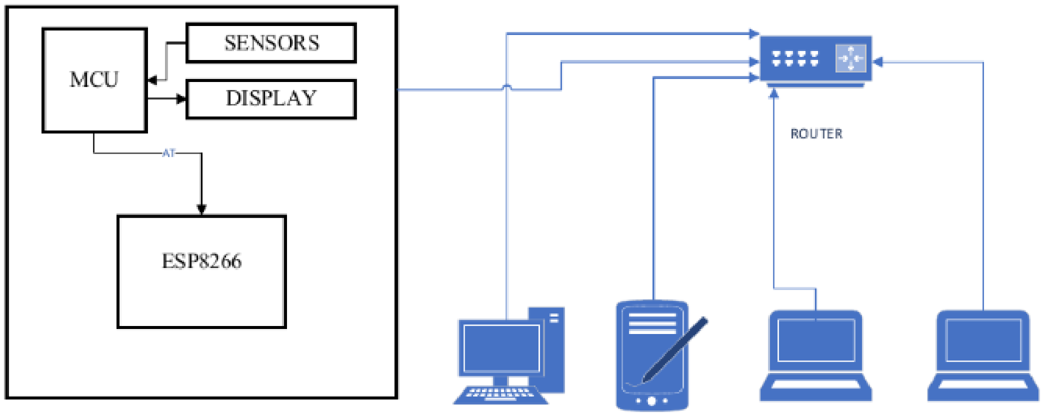


Figure 3. The structure of connection of developed solution

This solution does not have a cloud, it is local system. And the host MCU is server because it handles HTTP requests from any other device in local network.

The MCU is host and it sends AT commands, waits for answers and for other notifications from ESP8266. MCU via ESP8266 is automatically connected to the working network with specified SSID (Service Set Identifier) and password in the settings applied in firmware, and becomes a web server. Now other hosts from network can connect to this server and get statically created web page with real-time measures from sensors. The photo of working solution is provided in figure 4.

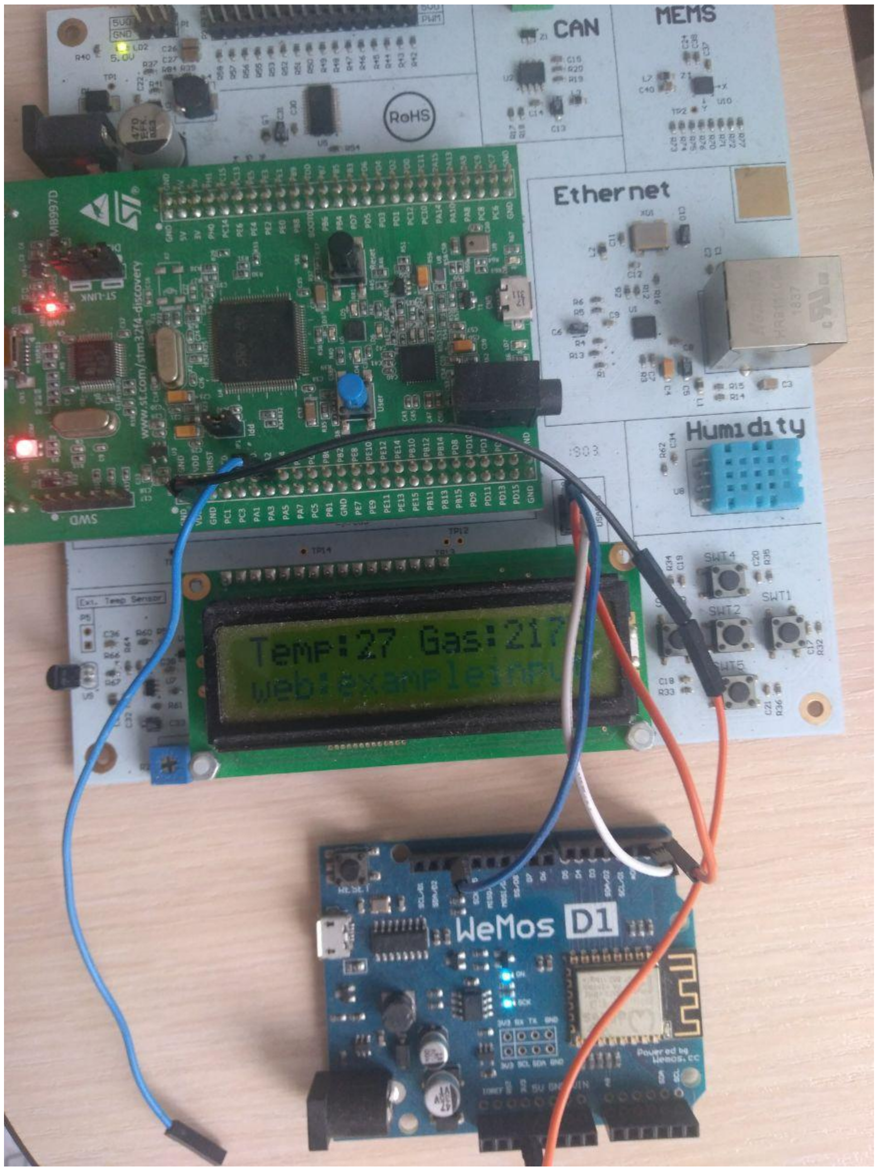
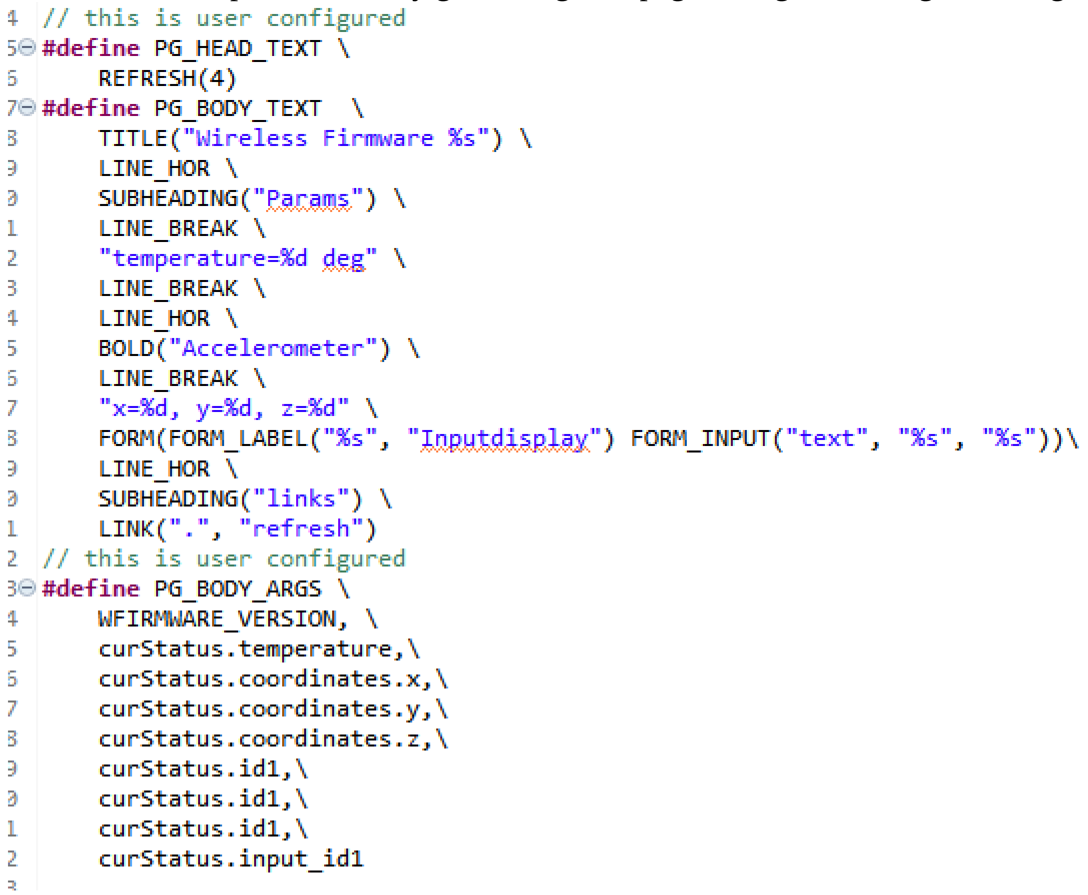


Figure 4. photo of the working solution

The web user interface of embedded system is developed in firmware of MCU. Any mobile or desktop device with the web browser can be connected to the solution provided and get the web page. The software solution gives an ability to give a web interface for sensors and displays. Information from peripherals drivers can be configured to be output to web page, the input field on web page can be configured to be output to display of embedded system.

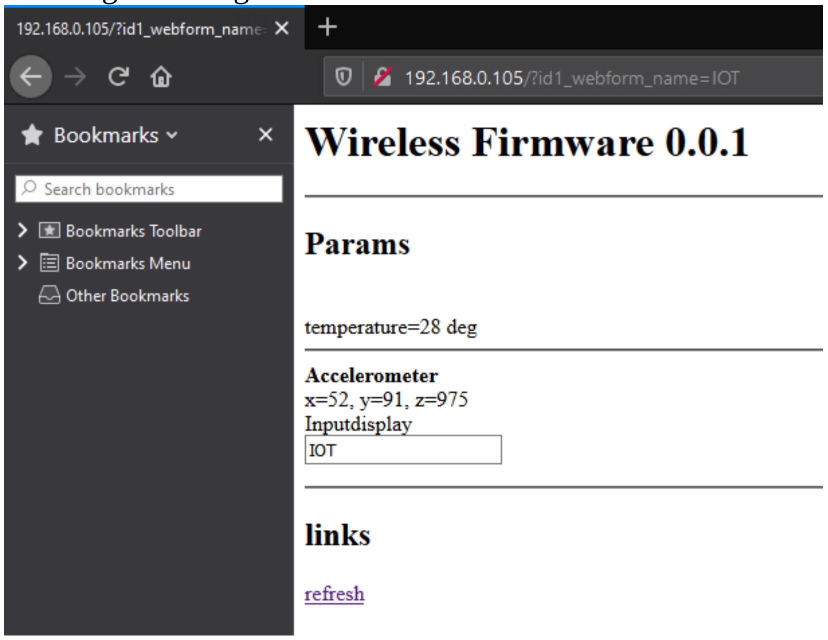
The example of statically generating web page configuration is given in figure 5.



```
// this is user configured
#define PG_HEAD_TEXT \
    REFRESH(4)
#define PG_BODY_TEXT  \
    TITLE("Wireless Firmware %s") \
    LINE_HOR \
    SUBHEADING("Params") \
    LINE_BREAK \
    "temperature=%d deg" \
    LINE_BREAK \
    LINE_HOR \
    BOLD("Accelerometer") \
    LINE_BREAK \
    "x=%d, y=%d, z=%d" \
    FORM(FORM_LABEL("%s", "Inputdisplay") FORM_INPUT("text", "%s", "%s"))\
    LINE_HOR \
    SUBHEADING("links") \
    LINK(".", "refresh")
// this is user configured
#define PG_BODY_ARGS \
    WFIRMWARE_VERSION, \
    curStatus.temperature,\
    curStatus.coordinates.x,\
    curStatus.coordinates.y,\
    curStatus.coordinates.z,\
    curStatus.id1,\
    curStatus.id1,\
    curStatus.id1,\
    curStatus.input_id1
```

Figure 5. Structure of web page

And how it looks like is given in figure 6.



Wireless Firmware 0.0.1

Params

temperature=28 deg

Accelerometer
x=52, y=91, z=975
Inputdisplay
IOT

links

refresh

Figure 6. Web page

The page is configured to refresh automatically in 4 seconds, it gives temperature, accelerometer axis output in real time. And it is possible to input text to display using “Inputdisplay” form, the result of input text “IOT” is given in figure 7.

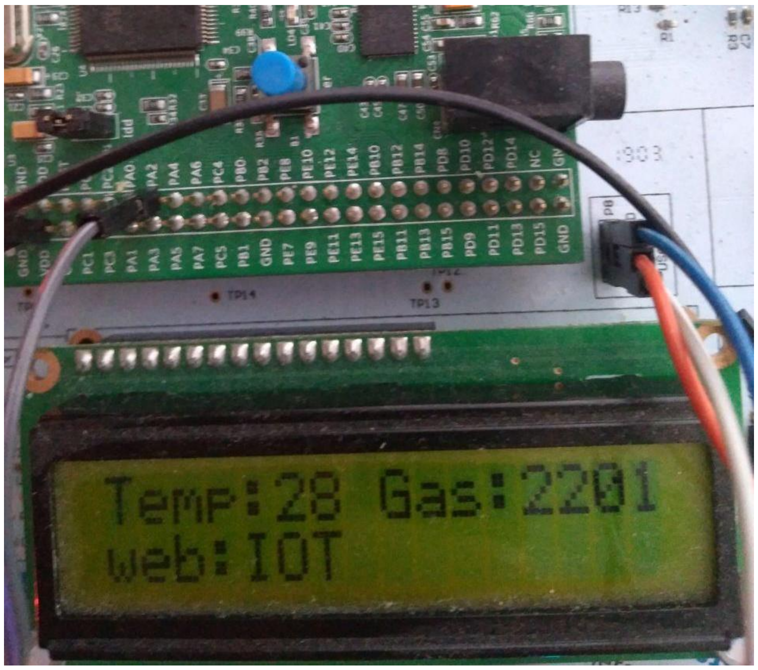


Figure 1. The result of inputted text to web page “IOT”

This software solution makes a web server from the given development board for local networks. IOT without cloud technologies is used with small number of interconnected devices, smart home solutions, those usually come with cloud technologies, are the opposite to this software solution. Also, cloud can benefit from IOT by increasing its scope to deal with existent world things in a more distributed and lively manner, and for bringing new services in a large number for real life scenario [15]. But the idea of local cloudless web server is alive because it has its own benefits, like it is cheaper and more reliable, and standalone solution.

The future improvement of solution assumes code portability and flexibility improvements and support of another MCU families, support of other wireless modules those support another networking technologies.

6 Conclusions

Existing special-purpose MCU-based IOT solutions are restricted to their specific hardware architecture and usually can’t be expanded to another use cases using only their own peripherals. Unlike them embedded systems have much more use cases because of their more universal architectures. The ESP8266 has a little number of peripherals itself.

The constructed solution is Wi-Fi supported board, that has a huge number of peripherals, so it has huge amount of capabilities, those are unavailable on most of other IOT boards. All these peripherals can be controlled by web interface provided. This solution can be used for studying IOT and for developing any particular IOT device for a particular use case. We expanded ESP8266s capabilities using general-purpose MCU board as host computer, that can control and can be controlled via ESP8266 from external devices. For this purpose, we used ESP8266s factory firmware with AT commands, that is usually used for verifying correct functionality of ESP8266.

The embedded board for studying is adapted for use in IOT. The adaptation process showed that it is possible and not so difficult to use general-purpose microcontrollers for IOT and it is possible to modify existing embedded platforms and embedded system to use in IOT and to study IOT. It also showed that the embedded world and IOT world are interconnected. Any embedded platform can be used to study IOT.

The software part of solution shows that it is possible using standard interfaces, such as AT commands, to create portable software solutions and make them work in other platforms with minimal changes of code.

The use of RTOS showed that it makes code much better structured than without it. It makes code more flexible and gives ability to divide software solution into different tasks and modules.

The web server creation and creation of web page using general-purpose MCU has showed that it is possible to use general-purpose MCUs to construct modern user interface inside thing and it is not necessary to have a cloud for a small IOT system that can provide user interface itself. It is not necessary to have a cloud to study IOT. And if thing provides a user interface itself, such IOT can`t be scalable enough but it fits well local usage, when only one thing or a few ones needed and is much cheaper, is less resource intensive and is much easier to set up.

The creation of statically generating web page library showed, that user interface for thing can be highly configurable and output or input any information provided.

The creation of hardware solution that can be controlled because it has display and web interface to fill it, and can control, because it has sensors and interfaces to collect data from them in real time, showed that this platform is well adapted to study IOT, because it shows different aspects of creating requests and responses.

References

1. L. Tan, "Future Internet: The Internet of Things," in 3rd International Conference on Advanced Computer Theory and Engineering(ICACTE) , Shanghai, 2010.
2. M. K. K. Y. Satoshi Kawaguchi, "Mutually Complementary Network System between Wired and Wireless Communication for Home Appliances," The 13th IEEE International Symposium on Consumer Electronics (ISCE2009), 2009.
3. M. A. K. A. M. A. Shadi Al-Sarawi, "Internet of Things (IoT) Communication Protocols : Review," in 2017 8th International Conference on Information Technology (ICIT) , Penang, 2017.
4. SIMCom, "SIM800c," 16 10 2017. [Online]. Available: https://simcom.ee/documents/SIM800C/SIM800C_Hardware_Design_V1.05.pdf. [Accessed 8 August 2020].
5. K. G. R. Johari, "IOT based Electrical Device Surveillance and Control System," pp. 2-3, 2019.
6. J. Hu, S. Zeng and Z. Zhang, "The Design of Wireless Data Acquisition System," in 2012 8th International Conference on Wireless Communications, Networking and Mobile Computing, Guilin, 2012.
7. Microchip Technology Inc, "AVR-IoT WG Development Board User Guide," 2020. [Online]. Available: http://ww1.microchip.com/downloads/en/DeviceDoc/AVR-IoT-WG-Development-Board-User-Guide-DS50002809C.pdf. [Accessed 25 July 2020].
8. B. Yang, "Design and Implementation of Intelligent Home," in 4th International Conference on Information Science and Control Engineering, Guilin, 2017.
9. N. Nikolov and O. Nakov, "Research of Communication between STM32L475 and Private Cloud realized by using Amazon FreeRTOS and MQTT," in 27-th National Conference with International Participation "TELECOM 2019", Sofia, 2019.
10. "The Role of WiFi in IoT," Maravedis, 31 March 2020. [Online]. Available: https://www.iotforall.com/wifi-role-iot/. [Accessed 15 July 2020].
11. Espressif Inc, "ESP8266," 2020. [Online]. Available: https://www.espressif.com/sites/default/files/documentation/0a-esp8266ex_datasheet_en.pdf. [Accessed 8 August 2020].

12. S. S. Pushkar Singh, "Arduino-Based Smart Irrigation Using Water Flow Sensor, Soil Moisture Sensor, Temperature Sensor and ESP8266 WiFi Module," in IEEE Region 10 Humanitarian Technology Conference (R10-HTC), Noida, 2016.
13. Telegesis (UK) Ltd, "ETRX2 ZigBee Modules AT-Command Dictionary," June 2009. [Online]. Available: http://wless.ru/files/ZigBee/ETRX2/TG-ETRX-R302-Commands.pdf. [Accessed 25 July 2020].
14. STMicroelectronics, "Discovery kit with STM32F407VG MCU," May 2017. [Online]. Available: https://www.st.com/resource/en/user_manual/dm00039084-discovery-kit-with-stm32f407vg-mcu-stmicroelectronics.pdf. [Accessed 25 July 2020].
15. N. K. Walia, "An IOT by Information Retrieval approach:Smart Lights controlled using WiFi," in 2016 6th International Conference - Cloud System and Big Data Engineering (Confluence), Noida, 2016.

ABSTRACT

UDC 004.056.57:032.26
DOI: https://doi.org/10.20535/2708-4930.1.2020.216042

USAK METHOD FOR THE REINFORCEMENT LEARNING
(p. 4 – 14)

Valentyn Kuzmych
National Technical University of Ukraine
“Igor Sikorsky Kyiv Polytechnic Institute”, Kyiv, Ukraine
ORCID: http://orcid.org/0000-0002-6077-3609

Mykhailo Novotarskyi
National Technical University of Ukraine
“Igor Sikorsky Kyiv Polytechnic Institute”, Kyiv, Ukraine
ORCID: http://orcid.org/0000-0002-5653-8518

In the field of reinforcement learning, tabular methods have become widespread. There are many important scientific results, which significantly improve their performance in specific applications. However, the application of tabular methods is limited due to the large amount of resources required to store value functions in tabular form under high-dimensional state spaces. A natural solution to the memory problem is to use parameterized function approximations. However, conventional approaches to function approximations, in most cases, have ceased to give the desired result of memory reduction in solving real-world problems. This fact became the basis for the application of new approaches, one of which is the use of Sparse Distributed Memory (SDM) based on Kanerva coding. A further development of this direction was the method of Similarity-Aware Kanerva (SAK). In this paper, a modification of the SAK method is proposed, the Uniform Similarity-Aware Kanerva (USAK) method, which is based on the uniform distribution of prototypes in the state space. This approach has reduced the use of RAM required to store prototypes. In addition, reducing the receptive distance of each of the prototypes made it possible to increase the learning speed by reducing the number of calculations in the linear approximator.

Keywords**:** reinforcement learning, Kanerva coding, function approximation, prototype, value function

UDC 004.4’24
DOI: https://doi.org/10.20535/2708-4930.1.2020.216044

LOOP PARALLELIZATION AUTOMATION FOR GRAPHICS PROCESSING UNITS
(p. 15 – 25)

Anatoliy Doroshenko
National Technical University of Ukraine
“Igor Sikorsky Kyiv Polytechnic Institute”, Kyiv, Ukraine
ORCID: https://orcid.org/0000-0002-8435-1451

Olexii Beketov
National Academy of Sciences of Ukraine
Institute of Software Systems, Kyiv, Ukraine
ORCID: https://orcid.org/0000-0003-4715-5053

Olena Yatsenko
National Academy of Sciences of Ukraine
Institute of Software Systems, Kyiv, Ukraine
ORCID: https://orcid.org/0000-0002-4700-6704

A technology that allows extending GPU capabilities to deal with data volumes that outfit internal GPU’s memory capacity is proposed. It involves loop tiling and data serialization and can be applied to utilize clusters consisting of several GPUs. Applicability criterion is specified and a semi-automatic proof-of-concept software tool is implemented. The experiment to demonstrate the feasibility of the proposed technology is described.

Keywords: CUDA, general-purpose computing on graphics processing units, loop optimization, parallelization methods.

UDC 004.383.3, 004.056
DOI: https://doi.org/10.20535/2708-4930.1.2020.216050

PROTECTED DISCRETE FOURIER TRANSFORM IMPLEMENTATION ON REMOTE COMPUTER SYSTEMS

(p. 26 – 33)

Alireza Mirataei
National Technical University of Ukraine
“Igor Sikorsky Kyiv Polytechnic Institute”, Kyiv, Ukraine
ORCID: http://orcid.org/0000-0002-4732-7030

Hana Khalil
National Technical University of Ukraine
“Igor Sikorsky Kyiv Politechnic Institute” Kyiv, Ukraine
ORCID: https://orcid.org/0000-0001-5275-8305

Olexander Markovskyi
National Technical University of Ukraine
“Igor Sikorsky Kyiv Politechnic Institute” Kyiv, Ukraine
ORCID: https://orcid.org/0000-0003-3483-4233

Effective method of the discrete Fourier transform acceleration with the use of cloud computing is theoretically substantiated and developed. The reigning feature of the suggested method is homomorphic encryption of the input signals, which provides efficient protection during the remote calculation. It has been shown theoretically and experimentally that the proposed method allows one to 1-2 orders of magnitude to speed up the processing of signals while maintaining their confidentiality. The proposed method can be applied for effective signal stream processing in clouds.

Keywords: discrete Fourier transform, cloud computer systems, remote signal processing, homomorphic encryption.

UDC 004.934
DOI: https://doi.org/10.20535/2708-4930.1.2020.216053

HASH SEARCH ORGANIZATION IN E-DICTIONARIES USING BLOCK CIPHERS

(p. 34 – 42)

Olexander Markovskyi
National Technical University of Ukraine
“Igor Sikorsky Kyiv Politechnic Institute” Kyiv, Ukraine
ORCID: https://orcid.org/0000-0003-3483-4233

Inna Humeniuk
National Technical University of Ukraine
“Igor Sikorsky Kyiv Politechnic Institute” Kyiv, Ukraine
ORCID: https://orcid.org/0000-0002-7004-3271

Olha Shevchenko
National Technical University of Ukraine
“Igor Sikorsky Kyiv Politechnic Institute” Kyiv, Ukraine
ORCID: https://orcid.org/0000-0002-0532-7315

The article is devoted to the problem of developing high-speed electronic dictionaries for systems computer translation. The method of organizing a high-speed electronic dictionary based on an ideal hash addressing, where the cryptographic cipher block acts as a hash transformation is proposed. This method was developed taking into account the multilevel memory structure of modern computer systems. It was testified theoretically and experimentally that the proposed organization of electronic dictionaries guarantees at least twice higher search rate compared to known technologies.

Keywords**:** hash-search, e-dictionary, contextual search, perfect hash-addressing, computer translation

UDC 004.72
DOI: https://doi.org/10.20535/2708-4930.1.2020.216056

THE METHOD OF SDN CLUSTERING FOR CONTROLLER LOAD BALANCING
(p. 43 – 50)

Oleksandr Dolynnyi
National Technical University of Ukraine
“Igor Sikorsky Kyiv Politechnic Institute” Kyiv, Ukraine
ORCID: https://orcid.org/0000-0002-2887-5090

Sergiy Nikolsky
Department of Computer Engineering National Technical University of Ukraine
“Igor Sikorsky Kyiv Politechnic Institute” Kyiv, Ukraine, Ukraine
ORCID: https://orcid.org/0000-0003-4893-3339

Yurii Kulakov
National Technical University of Ukraine
“Igor Sikorsky Kyiv Politechnic Institute” Kyiv, Ukraine
ORCID: orcid.org/0000-0002-8981-5649

The paper proposes the method of SDN clustering using connections density and controller load distribution that solves the problem of controller load balancing. Clustering efficiency criteria have been considered, including fault tolerance, controller-to-switch and intercontroller latency and network limitations. Review of the key clustering methods has been performed, and the base algorithm for modification has been chosen. Density-based controller placement algorithm is modified to solve the problem of multicontroller placement. Metric of the node boundary index is introduced to advance the efficiency of proposed algorithm. A software implementation of the developed algorithm has been created, and its performance has been tested. The algorithm’s modelling results have been compared with those of the other algorithms using the metrics of distribution of service traffic propagation delay and computational speed in relation to network topology size.

Keywords**:** SDN clustering, network, connections density, controller placement

UDC 004.94: 519.876.5
DOI: https://doi.org/10.20535/2708-4930.1.2020.216057

PETRI-OBJECT SIMULATION: TECHNIQUE AND SOFTWARE
(p. 51 – 59)

Inna Stetsenko
National Technical University of Ukraine
“Igor Sikorsky Kyiv Politechnic Institute” Kyiv, Ukraine
ORCID: https://orcid.org/0000-0002-4601-0058

Anton Dyfuchyn
National Technical University of Ukraine
“Igor Sikorsky Kyiv Politechnic Institute” Kyiv, Ukraine
ORCID: https://orcid.org/0000-0002-1722-8840

Nowadays information systems tend to be including components for transforming data into information applying modelling and simulation. Combined with real-time data, discrete event simulation could create powerful making decision and control systems. For these purposes, simulation software should be concentrated on creation the model as a code which can be easy integrated with other components of software. In this regard, Petri-object simulation technique, the main concept of which is to compose the code of model of complicated discrete event system in a fast and flexible way, simultaneously providing fast running the simulation, is requisite. The behaviour description of the model based on stochastic multichannel Petri net while the model composition is grounded on object-oriented technology. The Petri-object simulation software provides scalable simulation algorithm, graphical editor, correct transformation graphical images into model, correct simulation results. Graphical editor helps to cope with error-prone process of linking elements with each other. For better understanding the technique, the Petri-object model of web information system has been developed. Investigation of the response time has been conducted. The experiment has revealed system parameters impact on the value of response time. Thus, the model can be useful to avoid long-running request.

***Keywords*:** stochastic Petri net, discrete event simulation, web information system, long-running request, response time.

UDC 004.67

DOI: https://doi.org/10.20535/2708-4930.1.2020.216061

ROAD MONITORING SYSTEM BASED ON IOT TECHNOLOGY FOR SMARTCITY

(p. 60 – 67)

Anton Kopiika
National Technical University of Ukraine
“Igor Sikorsky Kyiv Politechnic Institute” Kyiv, Ukraine
ORCID: https://orcid.org/0000-0003-4090-7507

Roman Piskun
Global Logic Ukraine, Kyiv, Ukraine
ORCID: https://orcid.org/0000-0002-7242-9063

Valentyna Tkachenko
National Technical University of Ukraine
“Igor Sikorsky Kyiv Politechnic Institute” Kyiv, Ukraine
ORCID: https://orcid.org/0000-0002-1080-5932

Iryna Klymenko
National Technical University of Ukraine
“Igor Sikorsky Kyiv Politechnic Institute” Kyiv, Ukraine
ORCID: https://orcid.org/0000-0001-5345-8806

This work is devoted to the problem of automatic road quality control, which can be used by both road repair services and common drivers. This paper provides a survey of different known techniques and algorithms of finding potholes on the road and describe our own method, using accelerometer. It will be shown our device for detecting potholes on the road, which can be used, as a part of the IoT system of SmartCity. It uses data from an accelerometer for finding road bumps.

Keywords: Internet of Things, SmartCity, Potholes detection, Accelerometer, SPI, STM32, Road surface analysis.

UDK 004.35 004.78 004.454

DOI: https://doi.org/10.20535/2708-4930.1.2020.216065

MAKING AN IOT DEVELOPMENT PLATFORM FROM A SIMPLE MICROCONTROLLER DEMONSTRATION BOARD

(p. 68 – 76)

Ihor Muraviov
National Technical University of Ukraine
“Igor Sikorsky Kyiv Politechnic Institute” Kyiv, Ukraine
ORCID: https://orcid.org/0000-0001-7965-8638

Viktoriia Taraniuk
Global Logic Ukraine, Kyiv, Ukraine
ORCID: https://orcid.org/0000-0001-9044-1499

Iryna Klymenko
National Technical University of Ukraine
“Igor Sikorsky Kyiv Politechnic Institute” Kyiv, Ukraine
ORCID: https://orcid.org/0000-0001-5345-8806

The IOT is now in trend. Because of its huge popularity and business interest to it, IOT is only now coming massively to universities worldwide as a separate study. This article provides an IOT solution based on embedded technologies that were not specifically designed for IOT. The core of this solution - board is developed for studying peripherals of a particular general-purpose MCU. But we successfully adapted it to use in IOT and for study of IOT systems.

Keywords**:** Internet of things, wireless connectivity, Wi-Fi, embedded platform, AT commands.

АНОТАЦІЇ

UDC 004.056.57:032.26
DOI: https://doi.org/10.20535/2708-4930.1.2020.216042

МЕТОД УСАКА ДЛЯ ЗАКРІПЛЕННЯ ЗНАНЬ
(стор. 4 – 14)

Кузьмич Валентин
Національний технічний університет України
«Київський політехнічний інститут імені Ігоря Сікорського», Київ, Україна
ORCID: http://orcid.org/0000-0002-6077-3609

Новотарський Михайло
Національний технічний університет України
«Київський політехнічний інститут імені Ігоря Сікорського», Київ, Україна
ORCID: http://orcid.org/0000-0002-5653-8518

У сфері навчання з підкріпленням набули поширення табличні методи. Є багато важливих наукових результатів, які значно покращують їх ефективність у конкретних застосуваннях. Однак застосування табличних методів обмежене через велику кількість ресурсів, необхідних для зберігання функцій значень у табличній формі в просторах станів великої розмірності. Природним вирішенням проблеми пам'яті є використання наближень параметризованих функцій. Однак звичайні підходи до апроксимації функцій у більшості випадків перестали давати бажаний результат зменшення пам’яті при вирішенні реальних задач. Цей факт став основою для застосування нових підходів, одним з яких є використання Sparse Distributed Memory (SDM) на основі кодування Kanerva. Подальшим розвитком цього напрямку став метод Similarity-Aware Kanerva (SAK). У цій статті запропоновано модифікацію методу SAK — метод Канерви з урахуванням подібності (USAK), який базується на рівномірному розподілі прототипів у просторі станів. Цей підхід зменшив використання оперативної пам’яті, необхідної для зберігання прототипів. Крім того, зменшення рецептивної відстані кожного з прототипів дозволило підвищити швидкість навчання за рахунок зменшення кількості обчислень у лінійному апроксиматорі.

Ключові слова: навчання з підкріпленням, кодування Канерви, апроксимація функції, прототип, функція значення.

UDC 004.4’24
DOI: https://doi.org/10.20535/2708-4930.1.2020.216044

АВТОМАТИЗАЦІЯ ПАРАЛЕЛІЗАЦІЇ ЦИКЛУ ДЛЯ БЛОКІВ ОБРОБКИ ГРАФІКИ
(стор. 15 – 25)

Анатолій Дорошенко
Національний технічний університет України
«Київський політехнічний інститут імені Ігоря Сікорського», Київ, Україна
ORCID: https://orcid.org/0000-0002-8435-1451

Олексій Бекетов
Інститут програмних систем
Національної академії наук України, Київ, Україна
ORCID: https://orcid.org/0000-0003-4715-5053

Олена Яценко
Інститут програмних систем
Національної академії наук України, Київ, Україна
ORCID: https://orcid.org/0000-0002-4700-6704

Пропонується технологія, яка дозволяє розширити можливості графічного процесора для роботи з обсягами даних, які переповнюють ємність внутрішньої пам’яті графічного процесора. Він передбачає розбиття циклів і серіалізацію даних і може застосовуватися для використання кластерів, що складаються з кількох графічних процесорів. Визначено критерій застосовності та реалізовано напівавтоматичний програмний засіб для підтвердження концепції. Описано експеримент для демонстрації здійсненності запропонованої технології.

Ключові слова: CUDA, обчислення загального призначення на графічних процесорах, оптимізація циклу, методи розпаралелювання.

УДК 004.383.3, 004.056
DOI: https://doi.org/10.20535/2708-4930.1.2020.216050

РЕАЛІЗАЦІЯ ЗАХИЩЕНОГО ДИСКРЕТНОГО ПЕРЕТВОРЕННЯ ФУР’Є НА ВІДДАЛЕНИХ КОМП’ЮТЕРНИХ СИСТЕМАХ

(стор. 26 – 33)

Аліреза Міратаеї
Національний технічний університет України
«Київський політехнічний інститут імені Ігоря Сікорського», Київ, Україна
ORCID: http://orcid.org/0000-0002-4732-7030

Хана Халіл
Національний технічний університет України
«Київський політехнічний інститут імені Ігоря Сікорського», Київ, Україна
ORCID: https://orcid.org/0000-0001-5275-8305

Олександр Марковський
Національний технічний університет України
«Київський політехнічний інститут імені Ігоря Сікорського», Київ, Україна
ORCID: http://orcid.org/0000-0003-3483-4233

Теоретично обґрунтовано та розроблено ефективний метод прискорення дискретного перетворення Фур’є з використанням хмарних обчислень. Головною особливістю запропонованого методу є гомоморфне шифрування вхідних сигналів, що забезпечує ефективний захист при дистанційному обчисленні. Теоретично та експериментально показано, що запропонований спосіб дозволяє на 1-2 порядки прискорити обробку сигналів при збереженні їх конфіденційності. Запропонований метод може бути застосований для ефективної обробки потоку сигналів у хмарах.

Ключові слова: дискретне перетворення Фур'є, хмарні комп'ютерні системи, віддалена обробка сигналів, гомоморфне шифрування.

УДК 004.934
DOI: https://doi.org/10.20535/2708-4930.1.2020.216053

ОРГАНІЗАЦІЯ ХЕШ-ПОШУКУ В ЕЛЕКТРОННИХ СЛОВНИКАХ З ВИКОРИСТАННЯМ БЛОЧНИХ ШИФРОВ

(стор. 34 – 42)

Олександр Марковський
Національний технічний університет України
«Київський політехнічний інститут імені Ігоря Сікорського», Київ, Україна
ORCID: http://orcid.org/0000-0003-3483-4233

Інна Гуменюк
Національний технічний університет України
«Київський політехнічний інститут імені Ігоря Сікорського», Київ, Україна
https://orcid.org/0000-0002-7004-3271

Ольга Шевченко
Національний технічний університет України
«Київський політехнічний інститут імені Ігоря Сікорського», Київ, Україна
https://orcid.org/0000-0002-0532-7315

Стаття присвячена проблемі розробки швидкісних електронних словників для систем комп'ютерного перекладу. Запропоновано метод організації високошвидкісного електронного словника на основі ідеальної хеш-адресації, де блок криптографічного шифру виконує роль хеш-перетворення. Цей метод розроблено з урахуванням багаторівневої структури пам'яті сучасних комп'ютерних систем. Теоретично та експериментально доведено, що запропонована організація електронних словників гарантує принаймні вдвічі вищу швидкість пошуку порівняно з відомими технологіями.

Ключові слова: хеш-пошук, електронний словник, контекстний пошук, ідеальна хеш-адресація, комп'ютерний переклад

УДК 004.72
DOI: https://doi.org/10.20535/2708-4930.1.2020.216056

МЕТОД КЛАСТЕРИЗАЦІЇ SDN ДЛЯ БАЛАНСУВАННЯ НАВАНТАЖЕННЯ КОНТРОЛЕРА
(стор. 43 – 50)

Олександр Долинний
Національний технічний університет України
«Київський політехнічний інститут імені Ігоря Сікорського», Київ, Україна
ihttps://orcid.org/0000-0002-2887-5090

Сергій Нікольський
Національний технічний університет України
«Київський політехнічний інститут імені Ігоря Сікорського», Київ, Україна
ORCID: https://orcid.org/0000-0003-4893-3339

Юрій Кулаков
Національний технічний університет України
«Київський політехнічний інститут імені Ігоря Сікорського», Київ, Україна
ORCID: orcid.org/0000-0002-8981-5649

У статті запропоновано метод кластеризації SDN з використанням щільності з’єднань і розподілу навантаження на контролер, який вирішує проблему балансування навантаження на контролер. Були розглянуті критерії ефективності кластеризаці, включаючи відмовостійкість, затримку між контролерами та між контролерами та мережеві обмеження. Проведено огляд основних методів кластеризації та обрано базовий алгоритм для модифікації. Алгоритм розміщення контролерів на основі щільності модифіковано для вирішення проблеми розміщення мультиконтролерів. Для підвищення ефективності запропонованого алгоритму введено метрику індексу межі вузла. Створено програмну реалізацію розробленого алгоритму та перевірено її працездатність. Результати моделювання алгоритму порівнювали з результатами інших алгоритмів, використовуючи метрики розподілу затримки поширення трафіку служби та швидкості обчислення по відношенню до розміру топології мережі.

Ключові слова: Кластеризація SDN, мережа, щільність підключень, розміщення контролерів.

УДК 004.94: 519.876.5
DOI: https://doi.org/10.20535/2708-4930.2.2021.244189

АПАРАТНІ КОМПРЕСОРИ ЗОБРАЖЕНЬ GIF
(стор. 51 – 59)

Інна Вячеслававна
Національний технічний університет України
«Київський політехнічний інститут імені Ігоря Сікорського», Київ, Україна
ORCID: https://orcid.org/0000-0002-4601-0058

Антон Дифучін
Національний технічний університет України
«Київський політехнічний інститут імені Ігоря Сікорського», Київ, Україна
ORCID: https://orcid.org/0000-0002-1722-8840

В даний час інформаційні системи мають тенденцію включати компоненти для перетворення даних в інформацію за допомогою моделювання та симуляції. У поєднанні з даними в реальному часі симуляція дискретних подій може створити потужні системи прийняття рішень і контролю. Для цих цілей програмне забезпечення моделювання повинно бути зосереджено на створенні моделі як коду, який можна легко інтегрувати з іншими компонентами програмного забезпечення. У зв’язку з цим необхідна методика моделювання об’єктів Петрі, основною концепцією якої є створення коду моделі складної системи дискретних подій швидким і гнучким способом, одночасно забезпечуючи швидке виконання моделювання. Опис поведінки моделі на основі стохастичної багатоканальної мережі Петрі, а композиція моделі базується на об'єктно-орієнтованій технології. Програмне забезпечення моделювання об’єктів Петрі забезпечує масштабований алгоритм моделювання, графічний редактор, коректне перетворення графічних зображень у моделі, правильні результати моделювання. Графічний редактор допомагає впоратися з помилковим процесом зв'язування елементів один з одним. Для кращого розуміння методики була розроблена Петрі-об'єктна модель веб-інформаційної системи. Проведено дослідження часу реагування. Експеримент виявив вплив параметрів системи на значення часу відгуку. Таким чином, модель може бути корисною для уникнення тривалого запиту.

Ключові слова: стохастична мережа Петрі, моделювання дискретних подій, веб-інформаційна система, довгостроковий запит, час відповіді.

УДК 004.67
DOI: https://doi.org/10.20535/2708-4930.1.2020.216061

СИСТЕМА МОНІТОРИНГУ ДОРОГ НА ОСНОВІ ТЕХНОЛОГІЇ IOT ДЛЯ SMARTCITY

(стор. 60 – 67)

Антон Копійка
Національний технічний університет України
«Київський політехнічний інститут імені Ігоря Сікорського», Київ, Україна
ORCID: https://orcid.org/0000-0003-4090-7507

Роман Піскун
Глобал Лоджик Україна, Київ, Україна
ORCID: https://orcid.org/0000-0002-7242-9063

Валентина Ткаченко
Національний технічний університет України
«Київський політехнічний інститут імені Ігоря Сікорського», Київ, Україна
ORCID: https://orcid.org/0000-0002-1080-5932

Ірина Клименко
Національний технічний університет України
«Київський політехнічний інститут імені Ігоря Сікорського», Київ, Україна
ORCID: https://orcid.org/0000-0001-5345-8806

Дана робота присвячена проблемі автоматичного контролю якості доріг, яким можуть користуватися як дорожні ремонтні служби, так і звичайні водії. У цій статті представлено огляд різних відомих методів і алгоритмів пошуку вибоїн на дорозі та описано наш власний метод із використанням акселерометра. Буде показано наш пристрій для виявлення ям на дорозі, який можна використовувати, як частину системи IoT SmartCity. Він використовує дані акселерометра для пошуку нерівностей.

Ключові слова: Інтернет речей, SmartCity, виявлення вибоїн, акселерометр, SPI, STM32, аналіз дорожнього покриття.

УДК 004.35 004.78 004.454
DOI: https://doi.org/10.20535/2708-4930.1.2020.216065

СТВОРЕННЯ ПЛАТФОРМИ РОЗРОБКИ IOT З ПРОСТОЇ ДЕМОНСТРАЦІЙНОЇ ПЛАТИ МІКРОКОНТРОЛЕРА

(стор. 68 – 76)

Ігор Муравьов
Національний технічний університет України
«Київський політехнічний інститут імені Ігоря Сікорського», Київ, Україна
ORCID: https://orcid.org/0000-0001-7965-8638

Вікторія Таранюк
Глобал Лоджик Україна, Київ, Україна
ORCID: https://orcid.org/0000-0001-9044-1499

Ірина Клименко
Національний технічний університет України
«Київський політехнічний інститут імені Ігоря Сікорського», Київ, Україна
ORCID: https://orcid.org/0000-0001-5345-8806

Технологія IOT зараз в тренді. Через свою величезну популярність і бізнес-цікавість IOT тільки зараз масово надходить в університети по всьому світу як окреме дослідження. У цій статті представлено рішення IOT на основі вбудованих технологій, які не були спеціально розроблені для IOT. Ядро цього рішення - плата розроблена для вивчення периферійних пристроїв окремого мікроконтролера загального призначення. Але ми успішно адаптували його для використання в IOT і для вивчення систем IOT.

Ключові слова: Інтернет речей, бездротове підключення, Wi-Fi, вбудована платформа, AT-команди.