UDC 004.312.466.05 **doi: 10.32620/reks.2022.4.12**

Artem PEREPELITSYN, Vitaliy KULANOV, Inna ZARIZENKO

National Aerospace University “Kharkiv Aviation Institute”, Kharkiv, Ukraine

METHOD OF QOS EVALUATION OF FPGA AS A SERVICE

The subject *of study in this article is the evaluation of the performance issues of cloud services implemented using FPGA technology. The* ***goal*** *is to improve the performance of cloud services built on top of multiple FPGA platforms known as FPGA-as-a-Service (FaaS).* ***Task****: to analyze the delays in communications between host computer and FPGA; propose the steps of development to reduce the delay and perform the evaluation of the response time for the FPGA-based accelerator depending on number of involved cards; consider the reliability aspect of such systems implemented using programmable logic. According to the tasks, the following* ***results*** *were obtained. The FPGA-as-a-Service where FPGA resources are provided through a set of hardware/software toolset is considered. The usage of queueing theory for cloud-based services is analyzed. The contribution of the parts of FPGA-as-a-Service to the final delay of the service is discussed. The process of modeling of work the services based on FPGA accelerator cards with use of Jackson's network is analyzed in detail. The model of the delays of FaaS that considers the parameters of accelerator FPGA cards is offered. The formula of the total response time of the service combined based on the response of the components of is obtained. The proposed steps of reduce data processing delays include increase the size of data blocks for processing in FPGA by each kernel, change the communication model with kernel from sequential to pipelined, following timing closure technique and use more FPGA accelerator cards in parallel to divide the enquiring delay. Based on the proposed model the evaluation of response time of FaaS was done. The advantage of the use of many FPGAs in parallel for same data processing task instead of implementation of requests thread for each accelerator card is shown.* ***Conclusions.*** *The main contribution of this study is a step forward to the modeling of FPGA-based services that can be used for FPGA-based artificial intelligence (AI) applications. It helps to improve the performance of the system by means of reducing the delays at different stages of requests processing. Another side of this result is the reliability aspect that is based on modified manner of service operation in case of use the proposed steps of system optimization. It helps to improve the processing of requests to FaaS. The proposed method is the next step after prototyping of such systems because it helps to turn the FaaS from the object for development to the tool for deployment of new technologies like AI applications.*

Keywords: *FPGA; FPGA-as-a-Service; FaaS; cloud infrastructure; queueing theory; performance; reliability.*

Introduction

FPGA accelerators provide high performance, high throughput and predictable time delay with higher design flexibility and relatively low power consumption. FPGA as a service allows applying a new paradigm for addressing issues related to redundancy and parallelization when creating complex safety-critical components, thereby making the transition to the new phase of development of service systems that enhance the reliability of cloud technologies [1].

FPGA-oriented systems are one of the priority areas for development of modern cloud technologies, which, in fact, determines the technology for efficient acceleration of dedicated tasks.

But FPGA resources can support restricted number of kernels running in parallel, every single kernel can run several DPUs (Data Processing Unit) in parallel as well, and multiple FPGA accelerator cards are connected within a single host computer. Communication with kernels adds the response delay due to the specifics of communication framework implementation [2]. Adding more DPUs allows saving time on communication with the host, but it may also decrease the final clock frequency (timings) of a project. But at the same time solving of same computational task with few FPGAs helps improving the total bandwidth.

To find the optimal numbers of DPUs in the kernel, number of kernels in a FPGA and a quantity of FPGAs within a host a developer needs to perform iterative experimental research to find out the results (system clock frequency and a response time) for different configurations.

To avoid this overhead the queueing theory can be used to evaluate the system. The model allows guaranteeing QoS by following one of the important its characteristics which is the response time.

The cloud system is modeled using Jackson's theorem [3], which can be used to determine and measure the QoS, based on the response time for services.

Response time is defined as a sum of latency and service time in a cloud and components’ delays.

The purpose of the study is to improve the performance of services built on top of multiple FPGA platforms known as FPGA-as-a-Service (FaaS).

ISSN 1814-4225 (print)
ISSN 2663-2012 (online)

To achieve the goal, the following **tasks** are going to be performed: to analyze the delays in communications between host computer and FPGA, to propose steps to reduce the delay and to perform the evaluation of the response time for the FPGA-based accelerator depending on cards used and consider the reliability aspect of such systems.

1. Analysis of use of Queueing Theory for building FaaS platform

Many modern FPGA accelerating cards can be considered as a reconfigurable platform for different purposes. Users have an opportunity to implement their own design along with making changes in a predefined project architecture of existing FPGA cores according to the requirements to achieve greater performance compared with products based on fixed architectures [1, 4].

Providing FPGA resources allows users to allocate an entire FPGA silicon chip for their projects and implement a hardware architecture (platform) according to the business needs. Also, it is possible to allocate a host machine to a user with a set of connected FPGA-boards.

These FPGA accelerating cards allocated remotely and connected to the server using various interfaces. Service-oriented software is used along with the built-in system for processing/controlling the process of loading data and firmware to configure programmable logic [2].

A user interacts with a server to share and collect data; a user can provide a configuration file of a bitstream (firmware) for the programmable device. When managing remote operations with an FPGA board, one must consider distributed access, which consists of setting up permissions as well as setting up and organizing a queue [4].

To initiate interaction with FaaS, clients send requests to rent dedicates resources starting from hardware ones – FPGA boards, RAMs, CPU cores etc. to appropriate software applications and tools. The FaaS platform allocates resources in accordance with client’s needs. The development process is based on modern software development strategy involving CI/CD (Continuous Integration/Continuous Delivery) pipelines [4].

To deploy the FaaS infrastructure, one can use the containerization technology, where each software component is installed in a separate container, isolated from other applications, but executed within the same hardware infrastructure [5].

The performance of cloud services can be modeled using response time metrics, throughput, and network usage [6, 7]. To analyze the quality of service in cloud computing systems in case of processing complex requests with multiple tasks and high system performance requirements, a queuing approach with the fork-join queuing system can be used [6].

The model of use of parallel blocks M/M/1for the average response time was obtained in [7] only for the case of two M/M/1 systems operating in parallel, while in [8], various methods for approximating the average response time were obtained.

The response time distribution of the simulated system can be obtained based on the model [8]. The system model is expressed in an open M/M/m network, assuming an exponential density function for arrival and service times. Using the response time distribution, authors determined the optimal level of service and the relationship between the maximum number of tasks and the minimum number of resources (virtual machines). The response time is considered for both – the Queue waiting time and the Service time.

But at the same time one queue is not enough for modeling of servers. The response time of the system can be treated as the Jackson network. And performance of the server and the network are the significant parameters which are greatly influenced system response time.

The result of the analysis shows that there are examples of modeling of architecture and implementation of a cloud PaaS systems. Similar approach can be applied for FaaS. The optimizing of the location [9] of service-oriented cloud applications based on the Virtual Machine (VM) should be also taken into account [10].

The performed analysis showed that queueing theory is widely used for building models of different service which are used for the evaluation of multiple QoS metrics. Considering FaaS uses the same principle and way of providing service it was suggested to use queuing theory (QT) for its modeling.

2. QoS model of FaaS

The process of request processing in FaaS can be represented as queue that consists of several service units that are service channels. The channel is ready to receive the request. The random nature of requests and the service time leads to the fact that an excessively large number of requests can be accumulated at the input of the queue. They either queue or leave the system unattended, while in other periods, the queue works with underload or stands idle.

Due to the possible increase in the duration of servicing of request, a significant number of subqueries may accumulate in the synchronization buffer, exceeding the allowable volume, which may lead to failures, or decrease the quality of the services provided.

It's offered to apply a queuing system that represents Jackson's open network for modeling the work of FaaS system.

Let consider an open queue network with K nodes satisfying the following conditions:

– each node corresponds to an M/M/n queuing system. The k node has n_k serving devices;

– clients arrive from the external environment to k node according to the Poisson process with λ_k intensity. Requests can also come from other nodes to k node;

– a client can visit the same node several times.

The network has a single IS entry point (Inbound Server). The server acts as a load balancing device, which redirects user requests to the processing server, where i = 1 ... m, namely the nodes of the processing server. The load balancer is modeled by the M/M/1 queue, with the arrival and service rates modeled exponentially with parameters λ and L, where λ < L. The processing server is modeled as an M/M/m queuing system. It has a service speed that is equal to μ; it is μ = μi, i = 1 ... m.

OS is a cloud architecture output server that transmits response data back to client who made request.

CS is a client-server. It sends exponentially distributed queries with the λ parameter to the incoming IS server. It also receives responses from cloud architecture. CS receives files or fragments of files until the request is completely satisfied. Both the OS and CS are also modeled by the M/M/1 queue.

Connecting servers with exponential receipt and distribution of services are independent of each other distribution [11]. Therefore, this is a power distribution for clients leaving the server.

Based on Jackson's theorem for calculating the overall average arrival rate, we must summarize the response time from outside the system and the response time of the internal nodes. Using Jackson's theorem, we obtain γ – the rate of arrival of the exponential distribution at both nodes, γ = λ / (1 - τ).

The response time (T) of cloud architecture is determined by the following formula:

$$T = T_{IS} + T_{PS} + T_{OS} + T_{CS}. \quad (1)$$

Let us consider each term of this equation.

T_{IS} defines the response time of the input server (IS), which acts as a load-balancing device. The input server is modeled as the M/M/1 queue. Thus, the formula for obtaining the queue model for the cloud system is the response time for the M/M/1 queue:

$$T_{IS} = \frac{\frac{1}{L}}{1-\frac{\lambda}{L}}, \quad (2)$$

where λ is the arrival rate, and L is the input server service speed.

TPS represents the response time of process service nodes that actually process user requests. Process service nodes are modeled as M/M/M queue. The response time of the queue is determined by the formula:

$$T_{PS} = \frac{1}{\mu} + \frac{C(m,\rho)}{m\lambda - y}, \quad (3)$$

where m is the number of processing elements, and γ and μ = μi, i = 1 ... m, are the arrival and service rates of each processing element.

A key point of Jackson's theorem is as follows: each node can be considered independently of all other nodes, and state probabilities can be determined using the Erlang formula. This greatly simplifies the calculation of state-space probabilities.

The term C (m, ρ) represents the Erlang formula, which gives the probability of a new client joining M/M/m queue. The Erlang formula is defined as:

$$C(m,\rho) = \frac{\frac{(m\rho)^m}{m!}\cdot\frac{1}{1-\rho}}{\sum_{k=0}^{m-1}\frac{(m\rho)^k}{k!} + \frac{(m\rho)^m}{m!}\cdot\frac{1}{1-\rho}}, \quad (4)$$

where ρ = γ / μ.

T_{OS} represents the response time of the output server (OS), which transfers data back to a client; its operation is modeled as an M/M/1 queue. The service speed of this node is defined as O/F, where O is the average bandwidth speed (in bytes per second) of the OS, and F is the average size of the system response files. Its response time is determined by the formula:

$$T_{OS} = \frac{\frac{F}{O}}{1-\frac{\gamma}{\left(\frac{O}{F}\right)}}, \quad (5)$$

$$T_{OS} = \frac{F}{O-\gamma F}. \quad (6)$$

TCS is the response time of the client-server (CS) that receives the data; its operation is modeled as an M/M/1 queue.

The service speed, in this case, is defined as C/F, where C is the average bandwidth rate of the client-server in bytes per second. Like in T_{OS}, F is the average size of the system response files in bytes. The response time is defined as:

$$T_{CS} = \frac{\frac{F}{C}}{1-\frac{\gamma}{\left(\frac{C}{F}\right)}}, \quad (7)$$

$$T_{CS} = \frac{F}{C-\gamma F}. \quad (8)$$

The performance of cloud systems must be adapted to the needs of the area to which they are applied.

ISSN 1814-4225 (print)
ISSN 2663-2012 (online)

3. FaaS: proposed steps to reduce data processing delays

The delay of the FPGA-based service includes the delays of its parts. The projects implemented with XRT framework show the delay during launch the kernel. It is based on the way of communication of the kernel with the host application. This communication process is required to pass the data and parameters to the kernel that implements the functions that should be accelerated using FPGA.

Depending on the duration of one kernel iteration the percentage of communication phase from overall kernel execution time can be different. The longer duration of the computation the less contribution of communication to the total kernel running time. Normally this duration is no longer few second. This is based on the nature of system level implementation of the communication with FPGA accelerator card [12].

But Xilinx provides different modes of communication with XRT-managed kernels [13]. It is possible to use the pipelined mode of communication to reduce the impact of enquiring delay in total execution time. But it requires the efforts from the developer.

To reduce the time of computations or running the kernel it is necessary to improve the performance of the FPGA implementation. This parameter depends on the kernel clock frequency that can be assigned as v++ parameter of the project in Vitis [14].

But the final frequency of clk after the compilation depends on the optimization of the project and can be less then specified value.

To improve the timing parameters of FPGA project it is recommended to follow Timing Closure technique [15].

The proposed model of FaaS response time can be used to evaluate the delays in the service implementation to optimize request enquiring and processing.

Considering the results of the delay evaluation in FaaS components the following steps could be applied to improve the timings:

1) increase the size of dataset (batch) for a single data processing iteration to reduce the number of kernel lunches that is one of the most time-consuming operations;

2) use Pipelined Execution Model instead of Sequential Execution Model as XRT Controlled Kernel Execution Model;

3) follow Timing Closure as a basic recommendation of Xilinx (also provided in UG949) to improve the final project clock frequency and as a result the total performance;

4) use more FPGA accelerator cards in parallel for same host-application to distribute data processing and divide the enquiring delay.

4. FaaS response time evaluation

Throughput in data processing is one of the important indicators for the end user of the service. Also, the response time of an individual service element or set of services is important.

The ability to estimate response latency indicators for a service and throughput indicators allows to predict the operation of such a system depending on the intensity of requests. Such assessment allows to detect bottlenecks in advance and to use this information when building or optimizing such a service.

The proposed FaaS model allows to carry out such evaluation with the assignment of the actual values of the parameters of individual components of the service.

The architecture deployed for this test case consisted of servers created both using virtual and host machines containing an inbound server (IS), as well as output server (OS) and client-server (CS). Virtual machines were provided by VMAccel with the same U50 cards as installed at the host machine.

According to the simulation results, the most significant improvement was obtained in the transition from one to two virtual machines (fig. 1 and fig. 2). Adding of one server allows to obtain faster response.

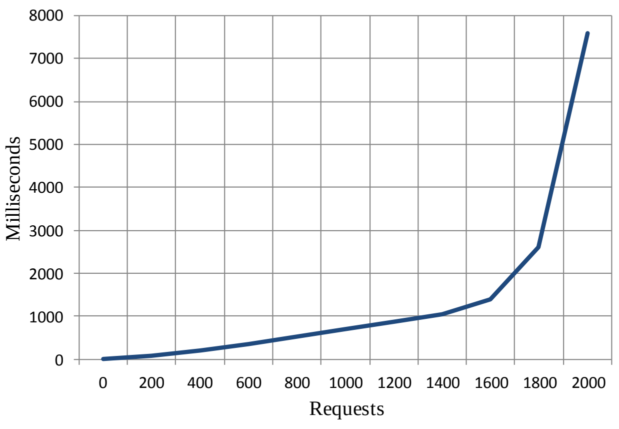

Fig. 1. Response time (T) with one server

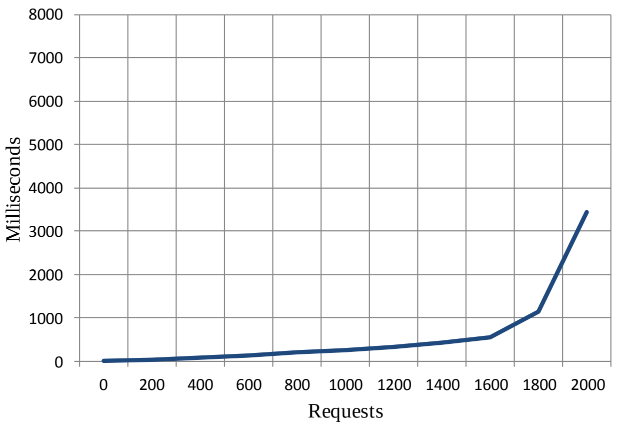

Fig. 2. Response time (T) with two servers

Based on the model the response time depends on different parameters including the delay of communication of XRT-managed kernel with the host application. This delay can be shorted using the change of manner of communication with kernel or by means of penalization of the acceleration with use of few cards for same task.

If the host application enquires the task with use of few FPGAs in parallel the delay for user would be the same as for one chip. But in this case the amount of processed datasets would be multiplied by the number of cards running in parallel. It means that the contribution of kernel communication delay to the total delay of data processing depends on the number of working in parallel FPGA accelerator cards. In this case to obtain the exact value of the delay per dataset it is necessary to use the coefficient. This coefficient is the result of the division of 1 by the number of involved cards.

The performed modeling of value of this coefficient shows that for 2 cards in parallel the value of the kernel communication delay per one dataset is a half of its value for 1 card. Adding of more cards improves the FaaS timing and reduces the role and contribution of kernel communication delay (fig. 3).

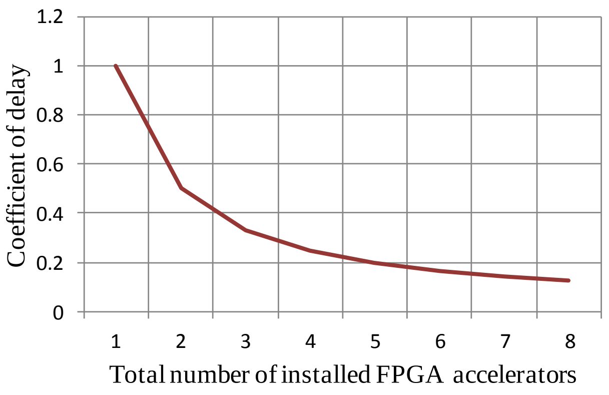

Fig. 3. Dependence of the data transferring time as the component of entire duration of data processing in FPGA depending on number of installed cards

This result shows that it is important to organize the parallel processing of the same task at all available FPGA accelerator cards to reduce the impact of the communication delay. It means that if there are n cards and n tasks it is better to organize only one processing thread with n accelerators instead of creation of n parallel threads. The rest tasks wait in the queue while current is processing on n cards in parallel. This concept is based and valid for XRT-managed FPGA projects.

These, the iterative evaluation of the delay and throughput results for the service allows to change the values of the parameters of individual components in accordance with the proposed set of steps. The optimization of the system with values of parameters can reduce service response times and increase throughput.

5. Discussion

This study considers the FPGA-as-a-Service in which FPGA resources are provided through a set of hardware/software toolset. The use of the proposed approach for modeling services based on FPGA allows describing the processes of processing requests in such systems at the level of the mathematical apparatus. This allows you to find bottlenecks and influence the final throughput of FPGA-based services.

It is shown that the total delay in the implementation of such systems is the sum of two delay factors inside the chip, the delay in the process of communication between individual cores and the host computer, the total delay in the server rack with a set of accelerator cards, and the additional delay introduced by network.

The proposed sequence of steps to reduce overall latency and increase system throughput results allows the predictable reduction of latency at both: the chip and the host levels.

These steps also include the process of increasing the frequency of system clocking by optimizing the design itself inside the accelerator. This is important for implementation of resource-intensive AI applications based on different kinds of neural networks [16, 17].

Conclusions

The concept of developing cloud services using FPGA-based systems has been described and discussed to improve their productivity. The performed analysis showed that FPGA implementation of acceleration for the dedicated task ensured better performance then CPU and even GPU implementations with less power consumption.

The analysis showed that the process of programming the FPGA solutions with accelerator cards adds the delays caused by different reasons, including the delay of data transferring between kernel and host computer, limited clock frequency of FPGA project and fixed number of cards running in parallel.

To improve the timing of FaaS implementations it was proposed to evaluate it using modeling. It was proposed to use models based on the queueing theory. They allow describe analytically the process of functioning of FPGA as a service.

Different parameters of service organization were analyzed including the speed and number of requests from users, speed of server maintenance and internal service performance.

Based on the proposed model the evaluation of response time of FaaS was done to determine possibilities of improving productivity. It was showed the difference of the response time and internal delays for different number of installed FPGA accelerator cards.

To improve the timing of FPGA-based services it was proposed to use the set of optimization steps that can be applied for both: system at the development stage and for existing projects that allow modification and optimization.

The proposed steps include increase the size of data blocks for processing in FPGA by each kernels, change the communication model with kernel from sequential to pipelined, following timing closure technique and use more FPGA accelerator cards in parallel to divide the enquiring delay.

The model for designing cloud computing architectures for FaaS is the main contribution of the research. We consider the possibility of using the elements of the queueing theory for analyzing performance and reliability of the FaaS and FPGA technology. The queuing theory and Jackson's networks chosen to evaluate performance in terms of response time to allow formulating recommendations to improve technical characteristics.

The methodology for calculating the performance parameters of cloud services proposed in the article can be used to simplify the search of the bottlenecks in the implementation of the service. It helps to improve the reliability of such systems.

It helps to improve the performance of the system by means of reducing the delays at different stages of requests processing. Another side of this result is the reliability aspect that is based on modified manner of service operation in case of use the proposed steps of system optimization. It helps to improve the processing of requests to FaaS. The proposed method is the next step after prototyping of such systems because it helps to turn the FaaS from the object for development to the tool for deployment of new technologies like AI applications.

The contribution of the research is that this is a step forward to the modeling of FPGA-based services that can be used for FPGA-based AI applications. The development of this methodological and technological basis is the direction of further research and development.

Contribution of authors: tasks formulation, proposing of steps to reduce data processing delays in FaaS, description of FaaS evaluation and elements of QoS evaluation method – **Artem Perepelitsyn**; formulation the title, goal and part of the research tasks, preparing of the introduction and the analysis sections – **Vitaliy Kulanov**; review and analysis of references on service modeling, use Jackson's network for description of the model, calculation and visualization the service response time – **Inna Zarizenko**.

All the authors have read and agreed to the published version of the manuscript.

References (GOST 7.1:2006)

1. *Perepelitsyn, A. FPGA as a Service Solutions Development Strategy [Text] / A. Perepelitsyn, I. Zarizenko, V. Kulanov // Proceedings 2020 IEEE 11th International Conference on Dependable Systems, Services and Technologies, DESSERT 2020. – 2020. – P. 376-380. DOI: 10.1109/DESSERT50317.2020.9125017.*
2. *Zarizenko, I. Analysis of tools and technologies of FaaS development [Text] / I. Zarizenko, A. Perepelitsyn // Radioelectronic and Computer Systems. – 2019. – No. 4. – P. 88–93. DOI: 10.32620/reks.2019.4.10.*
3. *Statistical Characterization of Containerized IP Multimedia Subsystem through Queueing Networks [Text] / M. Di Mauro, A. Liotta, M. Longo, F. Postiglione // 2020 6th IEEE Conference on Network Softwarization, NetSoft 2020. – 2020. – P. 100-105. DOI: 10.1109/NetSoft48620.2020.9165357.*
4. *Kulanov, V. Method of development and deployment of reconfigurable FPGA-based projects in cloud infrastructure [Text] / V. Kulanov, A. Perepelitsyn, I. Zarizenko // Proceedings of 2018 IEEE 9th International Conference on Dependable Systems, Services and Technologies, DESSERT 2018. – 2018. – P. 103-106. DOI: 10.1109/DESSERT.2018.8409108.*
5. *Containerizing Alveo Accelerated Applications with Docker [Online]. – Available at: https://xilinx.com/developer/articles/containerizing-alveo-accelerated-application-with-docker.html. – 27.01.2020.*
6. *Tsimashenka, I. Reduction of Subtask Dispersion in Fork-Join Systems [Text] / I. Tsimashenka, W. J. Knottenbelt // Computer Performance Engineering. EPEW 2013. Lecture Notes in Computer Science. – Springer, Berlin, Heidelberg, 2013. – Vol. 8168. – P. 325–336. DOI: 10.1007/978-3-642-40725-3_25.*
7. *Samuilov, K. Analysis of the response time of a cloud computing system [Text] / K. Samuilov, I. Zaryadov, A. Gorbunova // IX International industry scientific and technical conference "Information Society Technologies". – 2015. – P. 29–30.*
8. *Gaidamaka, Yu. A Simplified model for performance analysis of cloud computing systems with dynamic scaling [Text] / Yu. Gaidamaka, E. Sopin, M. Talanova // Proc. of the 18th International Scientific Conference "Distributed Computer and Communication Networks: Control, Computation, Communications", DCCN 2015. – 2015. – P. 75–86.*
9. *Vats, S. A Switch Based Resource Management Method for Energy Optimization in Cloud Data Center [Text] / S. Vats, S. Kumar Sharma, S. Kumar // International Journal of Computing. – 2021. – Vol. 20, iss. 1. – P. 85-91. DOI: 10.47839/ijc.20.1.2103.*
10. *Genetic-Based Task Scheduling Algorithm with Dynamic Virtual Machine Generation in Cloud Computing [Text] / A. A. A. Gad-Elrab, T. A. Alzohairy, K. R.*

Raslan, F. A. Emara // International Journal of Computing. – 2021. – Vol. 20, iss. 2. – P. 165–174. DOI: 10.47839/ijc.20.2.2163.

11. *The queueing theory in cloud computing to reduce the waiting time [Text] / T. Sai Sowjanya, D. Praveen, K. Satish, A. Rahiman // International Journal of Computer Science Engineering and Technology. – 2011. – Vol. 1, iss. 3. – P. 110–112.*

12. *Alveo U280 Data Center Accelerator Card User Guide, Xilinx, UG1314 [Online]. – Available at: sandycast.com/support/documentation/boards_and_kits/accelerator-cards/ug1314-u280-reconfig-accel.pdf. – 27.02.2020.*

13. *XRT Controlled Kernel Execution Models [Online]. – Available at: https://xilinx.github.io/XRT/master/html/xrt_kernel_executions.html. – 07.10.2022.*

14. *Vitis Unified Software Platform Documentation: Application Acceleration Development, Xilinx, UG1393 (v2019.2) [Online]. – Available at: https://docs.xilinx.com/r/en-US/ug1393-vitis-application-acceleration. – 28.02.2020.*

15. *UltraFast Design Methodology Guide for the Vivado Design Siute, Xilinx, UG949 (v2019.2) [Online]. – Available at: https://docs.xilinx.com/v/u/2019.2-English/ug949-vivado-design-methodology. – 6.12.2019.*

16. *Neural network model of hetero-associative memory for the classification task [Text] / T. Martyniuk, B. Krukivskyi, L. Kupershtein, V. Lukichov // Radioelectronic and Computer Systems. – 2022. – No. 2. – P. 108–117. DOI: 10.32620/reks.2022.2.09.*

17. *Moskalenko, V. Neural network based image classifier resilient to destructive perturbation influences – architecture and training method [Text] / V. Moskalenko, A. Moskalenko // Radioelectronic and Computer Systems. – 2022. – No. 3. – P. 95-109. DOI: 10.32620/reks.2022.3.07.*

References (BSI)

1. Perepelitsyn, A., Zarizenko, I., Kulanov, V. FPGA as a Service Solutions Development Strategy. *Proceedings 2020 IEEE 11th International Conference on Dependable Systems, Services and Technologies, DESSERT 2020,* 2020, pp. 376-380. DOI: 10.1109/DESSERT50317.2020.9125017.

2. Zarizenko, I., Perepelitsyn, A. Analysis of tools and technologies of FaaS development. *Radioelectronic and Computer Systems*, 2019, no. 4, pp. 88-93. DOI: 10.32620/reks.2019.4.10.

3. Di Mauro, M., Liotta, A., Longo, M., Postiglione, F. Statistical Characterization of Containerized IP Multimedia Subsystem through Queueing Networks. *Proceedings of 2020 6th IEEE Conference on Network Softwarization, NetSoft 2020*, 2020, pp. 100–105. DOI: 10.1109/NetSoft48620.2020.9165357.

4. Kulanov, V., Perepelitsyn, A., Zarizenko, I. Method of development and deployment of reconfigurable FPGA-based projects in cloud infrastructure. *Proceedings of 2018 IEEE 9th International Conference on Dependable Systems, Services and Technologies, DESSERT 2018*, 2018, pp. 103-106. DOI: 10.1109/DESSERT.2018.8409108.

5. *Containerizing Alveo Accelerated Applications with Docker.* Available: https://xilinx.com/developer/articles/containerizing-alveo-accelerated-application-with-docker.html. (accessed January 27, 2020).

6. Tsimashenka, I., Knottenbelt, W. J. Reduction of Subtask Dispersion in Fork-Join Systems. *Computer Performance Engineering. EPEW 2013. Lecture Notes in Computer Science.* Springer, Berlin, Heidelberg, 2013, vol. 8168, pp. 325–336. DOI: 10.1007/978-3-642-40725-3_25.

7. Samuilov, K., Zaryadov, I., Gorbunova, A. Analysis of the response time of a cloud computing system. *IX International industry scientific and technical conference "Information Society Technologies"*, 2015. pp. 29–30.

8. Gaidamaka, Yu., Sopin, E., Talanova, M. A Simplified model for performance analysis of cloud computing systems with dynamic scaling. *Proc. of the 18th International Scientific Conference "Distributed Computer and Communication Networks: Control, Computation, Communications", DCCN 2015*, 2015, pp. 75–86.

9. Vats, S., Kumar Sharma, S., Kumar, S. A Switch Based Resource Management Method for Energy Optimization in Cloud Data Center. *International Journal of Computing*, 2021, vol. 20, iss. 1, pp. 85–91. DOI: 10.47839/ijc.20.1.2103.

10. Gad-Elrab, A. A. A., Alzohairy, T. A., Raslan, K. R., Emara, F. A. Genetic-Based Task Scheduling Algorithm with Dynamic Virtual Machine Generation in Cloud Computing. *International Journal of Computing*, 2021. vol. 20, iss. 2, pp. 165–174. DOI: 10.47839/ijc.20.2.2163.

11. Sai Sowjanya, T., Praveen, D., Satish, K., Rahiman, A. The queueing theory in cloud computing to reduce the waiting time. *International Journal of Computer Science Engineering and Technology*, 2011, vol. 1, iss. 3, pp. 110–112.

12. *Alveo U280 Data Center Accelerator Card User Guide, Xilinx, UG1314 (v1.3).* Available at: https://www.sandycast.com/support/documentation/boards_and_kits/accelerator-cards/ug1314-u280-reconfig-accel.pdf. (accessed February 27, 2020).

13. *XRT Controlled Kernel Execution Models.* Available: https://xilinx.github.io/XRT/master/html/xrt_kernel_executions.html. (accessed October 7, 2022).

14. *Vitis Unified Software Platform Documentation: Application Acceleration Development, Xilinx, UG1393 (v2019.2).* Available at: https://docs.xilinx.com/r/en-US/ug1393-vitis-application-acceleration. (accessed February 28, 2020).

15. *UltraFast Design Methodology Guide for the Vivado Design Siute, Xilinx, UG949 (v2019.2).* Available at: https://docs.xilinx.com/v/u/2019.2-English/ug94

9-vivado-design-methodology. (accessed December 6, 2019).

16. Martyniuk, T., Krukivskyi, B., Kupershtein, L., Lukichov, V. Neural network model of hetero-associative memory for the classification task. *Radioelectronic and Computer Systems,* 2022, no. 2, pp. 108–117. DOI: 10.32620/reks.2022.2.09.

17. Moskalenko, V., Moskalenko, A. Neural network based image classifier resilient to destructive perturbation influences – architecture and training method. *Radioelectronic and Computer Systems,* 2022, no. 3, pp. 95–109. DOI: 10.32620/reks.2022.3.07.

Надійшла до редакції 10.09.2022, розглянута на редколегії 20.11.2022

МЕТОД ОЦІНЮВАННЯ ЯКОСТІ ОБСЛУГОВУВАННЯ FPGA ЯК СЕРВІС

Артем Перепелицин, Віталій Куланов, Інна Зарізенко

Предметом вивчення в даній статті є сучасні технології, інструменти та способи побудови апаратних прискорювачів для спеціалізованих обчислень і оцінка проблем продуктивності сервісів, реалізованих з використанням FPGA технології. **Метою** роботи є покращення продуктивності сервісів, створених на основі набору FPGA прискорювачів, відомих як FPGA-as-a-Service (FaaS). **Завдання:** проаналізувати затримки зв'язку між хост-комп'ютером і FPGA; запропонувати кроки розробки для зменшення затримки та провести оцінку часу відгуку для прискорювача на основі FPGA залежно від кількості задіяних карт FPGA прискорювачів; розглянути аспект надійності таких систем, реалізованих за допомогою програмованої логіки. Відповідно до поставлених завдань, були отримані наступні **результати**. Розглянуто побудову FPGA як сервіс, де ресурси FPGA надаються через набір апаратних/програмних засобів. Проаналізовано використання теорії масового обслуговування для хмарних сервісів. Обговорюється внесок складових частин FPGA в кінцеву затримку сервісу. Детально проаналізовано процес моделювання роботи сервісів на основі карт прискорювача FPGA з використанням теорії масового обслуговування. Запропоновано модель затримок FaaS, що враховує параметри карт прискорювачів FPGA. Отримано формулу сумарного часу відгуку сервісу, отриманого з урахуванням відгуку компонентів. Запропоновані кроки для зменшення затримок обробки даних включають збільшення розміру блоків даних для обробки в FPGA кожним ядром, зміну моделі зв’язку з ядром з послідовної на конвеєрну, дотримуючись техніки закриття часу, і паралельно використовуйте більше карт прискорювача FPGA для розподілу затримки запиту. На основі запропонованої моделі проведено оцінку часу відгуку реалізації FaaS. Показано перевагу використання багатьох FPGA паралельно для вирішення однієї задачі обробки даних замість реалізації потоку запитів для кожної карти прискорювача окремо. **Висновки:** Головний внесок цього дослідження полягає в тому, що це крок вперед до моделювання сервісів на основі FPGA, які можна використовувати для побудови штучного інтелекту (ШІ) на основі FPGA. Це допомагає підвищити продуктивність системи за рахунок зменшення затримок на різних етапах обробки запитів. Іншою стороною цього результату є аспект надійності, який базується на модифікованому способі роботи служби у разі використання запропонованих кроків оптимізації системи. Це допомагає покращити обробку запитів до FaaS. Запропонований метод є наступним кроком після прототипування таких систем, оскільки він допомагає перетворити FaaS з об’єкта для розробки в інструмент для розгортання нових технологій, таких як сервіси ШІ.

Ключові слова: FPGA, FPGA-as-a-Service, FaaS, хмарна інфраструктура, теорія масового обслуговування, продуктивність, надійність.

Перепелицин Артем Євгенович – канд. техн. наук, доц., доц. каф. комп’ютерних систем, мереж і кібербезпеки, Національний аерокосмічний університет ім. М. Є. Жуковського «Харківський авіаційний інститут», Харків, Україна.

Куланов Віталій Олександрович – канд. техн. наук, доц., доц. каф. комп’ютерних систем, мереж і кібербезпеки, Національний аерокосмічний університет ім. М. Є. Жуковського «Харківський авіаційний інститут», Харків, Україна.

Зарізенко Інна Миколаївна – асп. каф. комп’ютерних систем, мереж і кібербезпеки, Національний аерокосмічний університет ім. М. Є. Жуковського «Харківський авіаційний інститут», Харків, Україна.

Artem Perepelitsyn – PhD, Associate Professor of Computer Systems, Networks and Cybersecurity Department, National Aerospace University «Kharkiv Aviation Institute», Kharkiv, Ukraine,
e-mail: a.perepelitsyn@csn.khai.edu, ORCID: 0000-0002-5463-7889, Scopus Author ID: 56332607800.

Vitaliy Kulanov – PhD, Associate Professor of Computer Systems, Networks and Cybersecurity Department, National Aerospace University «Kharkiv Aviation Institute», Kharkiv, Ukraine,
e-mail: v.kulanov@csn.khai.edu, ORCID: 0000-0002-9312-0735, Scopus Author ID: 54911941800.

Inna Zarizenko – PhD student, Computer Systems, Networks and Cybersecurity Department, National Aerospace University «Kharkiv Aviation Institute», Kharkiv, Ukraine,
e-mail: i.kolesnyk@csn.khai.edu, ORCID: 0000-0003-4649-5793, Scopus Author ID: 57194777356.