Simulation Modelling Practice and Theory 114 (2022) 102402

Contents lists available at ScienceDirect

Simulation Modelling Practice and Theory

journal homepage: www.elsevier.com/locate/simpat

Flexible architecture for deployment of edge computing applications

Abdukodir Khakimov [a], Ibrahim A. Elgendy [b,c], Ammar Muthanna [d], Evgeny Mokrov [a], Konstantin Samouylov [a,e], Yassine Maleh [f,*], Ahmed A. Abd El-Latif [g]

[a] Peoples' Friendship University of Russia (RUDN University), Miklukho-MaklayaSt, 6, Moscow, 117198, Russia
[b] School of Computer Science and Technology, Harbin Institute of Technology, Harbin 150001, China
[c] Department of Computer Science, Faculty of Computers and Information, Menoufia University, Shibin el Kom 32511, Egypt
[d] The Bonch-Bruevich Saint-Petersburg State University of Telecommunications, Pr. Bolshevikov, 22, St.Petersburg 193232, Russia
[e] Federal Research Center "Computer Science and Control" of the Russian Academy of Sciences (FRC CSC RAS), 44-2 Vavilov St, Moscow, 119333, Russia
[f] LaSTI Laboratory, ENSAK, Sultan Moulay Slimane University, Morocco
[g] Department of Mathematics and Computer Science, Faculty of Science, Menoufia University, Al Minufiyah 32511, Egypt

ARTICLE INFO

Keywords:
5G
Wireless networks
Distributed architecture
Edge computing
Modeling

ABSTRACT

Recently, a great development has occurred in the communication networks, in which the trend of development is towards automation and flexible network structure. In addition, the stages of telecommunication development are conventionally divided from manual switching to flexible virtualized architecture, to which telecommunications operators have resorted. Furthermore, in the context of the 3rd generation partnership project (3GPP) standards framework, the network function virtualization (NFV) with might and main has been evaluated by telecom operators and shows that the efficiency of the topology can be significantly improved. However, it is difficult to organize ultra-reliable low-latency communication (URLLC) and massive machine type communications (mMTC) services within 5G, due to the topology centralization. Moreover, network monitoring, in particular of carrier-class equipment, reveals that there is no ideal static network topology in which the network and its elements have been uniformly loaded over a long service time. Therefore, in this study, a dynamic network topology and service placement is proposed to analyze and predict services, in which Genetic Algorithm (GA) is utilized. In addition, an efficient forecasting and live migration methods of service as an application to edge computing systems are introduced, where this approach can be used in the systems with an intelligent allocation of operator equipment resources for providing flexibility and high-quality topological organization. Finally, simulation results proved that the network equipment efficiency can significantly be increased by more than 30%.

1. Introduction

Nowadays, 5G communication networks standardization has been completed and commercially launched in some countries such as South Korea, China, and the United States [1]. Consequently, the conceptualization and planning of 6G communications have been started which aims to provide communications services while satisfying the future requirements of 2030 [2–4]. Moreover, in future

* Corresponding author.
E-mail addresses: khakimov-aa@rudn.ru (A. Khakimov), ibrahim.elgendy@hit.edu.cn (I.A. Elgendy), ammarexpress@gmail.com (A. Muthanna), mokrov-ev@rudn.ru (E. Mokrov), samuylov-ke@rudn.ru (K. Samouylov), yassine.maleh@ieee.org (Y. Maleh), a.rahiem@gmail.com (A.A.A. El-Latif).

https://doi.org/10.1016/j.simpat.2021.102402
Received 6 May 2021; Received in revised form 11 August 2021; Accepted 27 August 2021
Available online 4 September 2021

networks, new requirements will be needed for the applications with ultra-low latency. For example, the tactile internet [5,6] has contributed to many more important developments in the field of building communication networks, as in this case it was required to transmit information with only one ms delay, which is actually 100 times less than in the existing networks. Therefore, it is worth noting that the concept of the tactile Internet leads to the decentralization of the network, since the fundamental limitations on the speed of light transmission are naturally insurmountable. Another fundamental change in the development of communication networks was the unification of the flying and ground network segments into a single network [7,8]. It should be noted that with the low flight height of UAVs in such networks, measured in tens of meters, makes them suitable for use in communication networks with ultra-low delays.

6G means 6th generation wireless mobile technology, it will be integrated version of 5G wireless mobile technology. 6G will also deals with satellite network for the global coverage. There are three type of satellite network telecommunication satellite network, Navigation satellite network and Earth imaging satellite. In this technology wireless broadband will use to connect device to internet. Data speed of 6G devices will be 1 GB or even more.

One of the promising applications for the services of the 2030 networks will be Telepresence [9,10], the presence of digital avatars on the network. It provides for the possibility of several avatars of one person. To implement this type of network, it is necessary to build networks with ultra-low delays. At the same time, telepresence services are not limited only to reproducing user actions. When using holographic applications and avatars, it will be possible, for example, to watch a football match not on TV, but as a holographic model from any angle. In 2030, various applications of nano-grids and nano-things are likely to be widely used [11]. At least in medicine. Also, one of the directions in nanonets is molecular networks.

The characteristics of 2030 networks will be driven by emerging new technologies that will accelerate the universal implementation of such networks, such as artificial intelligence, genetic algorithms and new approaches to device design, using micro-services and live migration [12]. Micro-service architecture is considered recently as an evolved concept in the area of software models that can split a monolithic application into separate processes that implement a single function [13,14]. This approach can provide many benefits over the traditional monolithic architectures, such as reducing the complexity by using small apps, easier deployment and removal of service components, flexibility in using different structures and tools, predictable scalability and improving the fault tolerance of the service.

Furthermore, with a view to efficient resource management and the potential to scale a single component to boost the performance and failure tolerance, the micro-service architecture is important for the deployment of edge computing networks [15–19]. Additionally, the scalability can be enhanced by various deployment options for small-scale components. As a result, micro-services are considered as a promising paradigm for designing, implementing, and deploying 2030 network applications. Besides, human-centered mobile communications will continue to be the most critical focus for 6G development. However, high security, reliability and privacy should be the key characteristics of 6G as well as the focus of the wireless research community. Therefore, motivated by such consideration and to solve the problems of the optimal distribution of tasks in the MEC infrastructure of a network operator and short-term lease of computing resources, we propose a platform, called a Resource Exchange platform (REx), that could guide research in the post-5G era and looks at potential 6G application scenarios, as shown in Fig. 1. This platform is a universal one that allows you to place an application of service providers for a short time on the infrastructure of a network operator, to improve the quality of the service provided by bringing it closer to end-users. It further explores issues beyond communications technology that could hinder 6G research and deployment [2,20,21]. Thanks to the platform, we can reduce the distance between subscribers and the application to a minimum, reduce latency, reduce the load on the central cloud of the service provider and, of course, reduce the load on the operator's peer connections. The distinct features of this paper over the current state-of-art are as follows:

- Firstly, we use the computing resource of another carrier by another participant as a service provider.
- Secondly, the proposed platform functions is considered as the MEC orchestrator of the network operator and the cloud infrastructure of the service provider as well as can interact based on API.
- Finally, in our work, the network operator and service provider are independent of each other.

In summary, the main contributions of this paper include:

- A novel platform is proposed that could guide research in the post-5G era and looks at potential 6G application scenarios. In addition, an application of service providers are allowed to be placed for a short time on the infrastructure of a network operator, to improve the quality of the service provided by bringing it closer to end-users.
- Subsequently, key potential features of 6G are identified and the necessary communication technologies discussed.
- Furthermore, it explores issues beyond communications technology that could hinder 6G research and deployment.
- Finally, simulation results proved that the proposed platform can significantly increase the network equipment efficiency by more than 30%.

The remainder of this paper is organized as follows. Related works are introduced in Section 2. In Section 3, the network architecture is presented. Then, simulation composition are conducted in Section 5. Finally, Section 7 concludes the paper.

2. Related work

Recently, significant efforts have been made to address the development of distributed computing systems using MEC. For example, Guo et al. [22] considered the analysis of MEC resources using ARBP, autoregressive integrated moving average (ARIMA)

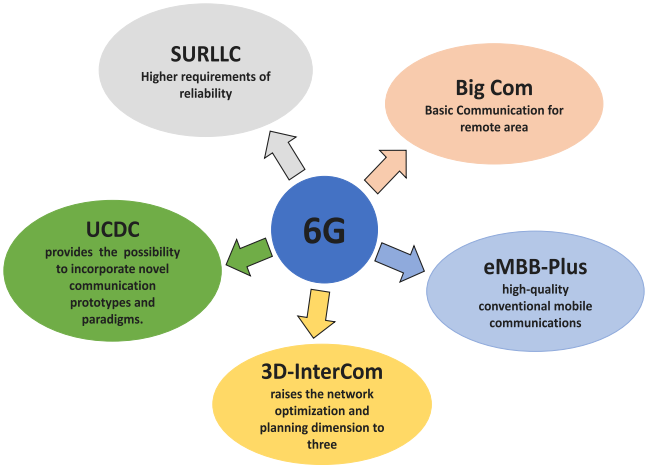

Fig. 1. Applications of 6G.

& GA. However, importing the applications from third-party services is considered the main drawback of this work. Meanwhile, in [23], an efficient method for redirecting service channels during the migration process is proposed, which also is used in our work when developing our REx platform. Whereas, in [24] and [25], the computation resources have been managed for multi-access edge computing. In [24], Zhou et al. derived a technique to boost the overall delay-aware efficiency of heterogeneous networks with MEC processing and storage capacities as well as user association choices. In addition, A coalition-game-based algorithm is therefore proposed to shape user coalitions for the strategy of association and resource sharing. While, Zakarya et al. considered the resource management with respect to both parties. Besides, they proposed a game-theoretic resource management strategies to reduce infrastructure energy consumption and costs while ensuring application efficiency. However, as in any system of interaction of various external applications in the field of telecommunications, it is necessary to apply mechanisms to ensure SLA between the service provider and the customer.

Elsewhere, [26] proposed a model for creating SLA using TOSCA technology. However, this type of proposal could cause a number of technical complications in terms of synchronization between the service provider and the network operators.

Furthermore, migrating process and maintaining SLA conditions for containers and virtual machines are addressed in [27] and [28], in which they have studied the analysis of system resources in an efficient way. However, the authors of these works did not consider the resource during the installation (during migration) process and the latent costs. In addition, most of the studies that work on the application of mobile edge computing (MEC) hosting with external services are not proposed a platform which could combine the estimation cost of the resource, digital SLA mechanism, the mechanism for forecasting service demand and single migration mechanism with a standardized API.

Recently, the placement problem in a MEC system is addressed in [29] and [30]. In [29], Bahreini et al. handled the multi-component placement problem for edge computing system, in which they formulate a Mixed Integer Linear Program which considers the capabilities of the network and the dynamic of the user. In addition, an efficient allocations algorithm is presented to solve this problem. Whereas in [30], dealt with a non-cloud boundary environment, in which data blocks placement and tasks scheduling are optimally combined with the goal of minimizing the delay of computation as well as the response time for the submitted tasks. Additionally, they improved the user experience in the edge computing system. Table 1 presents a succinct summary of the literature reviewed in terms of types of design objective, solution and their advantages.

Motivated by the problems of the above review of related work, we proposed a model of on-demand resource delivery based on service costs. Compared to the studies above, both explicit and implicit costs are jointly considered in the cost of services. In addition, the nodes are sorted by the remaining time of the resource occupied.

3. Network architecture

In this work, we consider an architecture of three main components, which are cloud computing, network operator and REx platform, as shown in Fig. 2. Cloud computing is a kind of space in which the service can be located and provided. Whereas the network operator is the backbone component, which is composed of a set of different nodes where the edge server can be installed. In our work, only seven zones are selected for emulation. Finally, it is an independent platform where the data can be exchanged between the service provider and the provider takes place. These data can be categorized into three different types, which are cost of hosting service on edge cloud, Time of service placement on edge cloud and Resources occupied by the service within one edge cloud unit such as RAM, CPU, bandwidth and ROM.

In our booth, each service is provided via a docker container and all the containers, by default, are launched on the cloud application, where they look like a centralized circulation. Afterward, the requests for each service are evaluated and then the clones of our services are placed on the required cloud edge. This scenario can give us two solutions which are:

Table 1
Comparison between existing studies.

Reference	Design objective	Solution	Advantages
[22]	Workload prediction and service cost	• ARIMA • Neural network	Migration cost is taken into account
[27]	Efficiency and fairness	• Fairness on Dominant • Shares and Generalized • Fairness on Jobs	Taking into account the resource utilization of CPU and memory
[31]	Load balancing	• Graph theory	Solving the problem of resource provision
[29]	Multi-component application placement	• Graph theory	Time delay optimization
[32]	System cost	• Queueing theory	System cost calculation
[33]	Backhaul cost	• Heuristic solution	Backhaul cost calculation
Our work	System, application and migration costs	• Genetic algorithm • (Ongoing Neural network and ARIMA)	Optimization of the application exchange mechanism

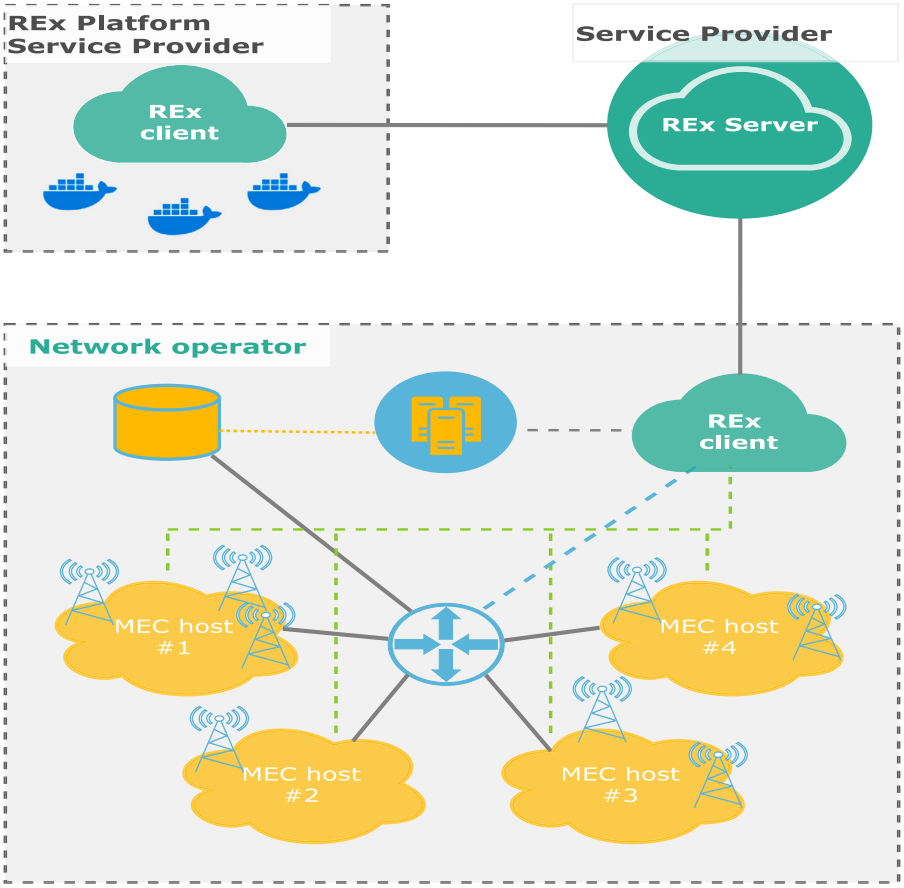

Fig. 2. Network architecture.

1. Unloading the backbone
2. Reducing latency

3.1. REx platform

In this subsection, REx platform is described in more detail in which its main roles are first introduced. Then, the main components of its architecture are provided, followed by how it can work. Afterward, different operation modes are presented.

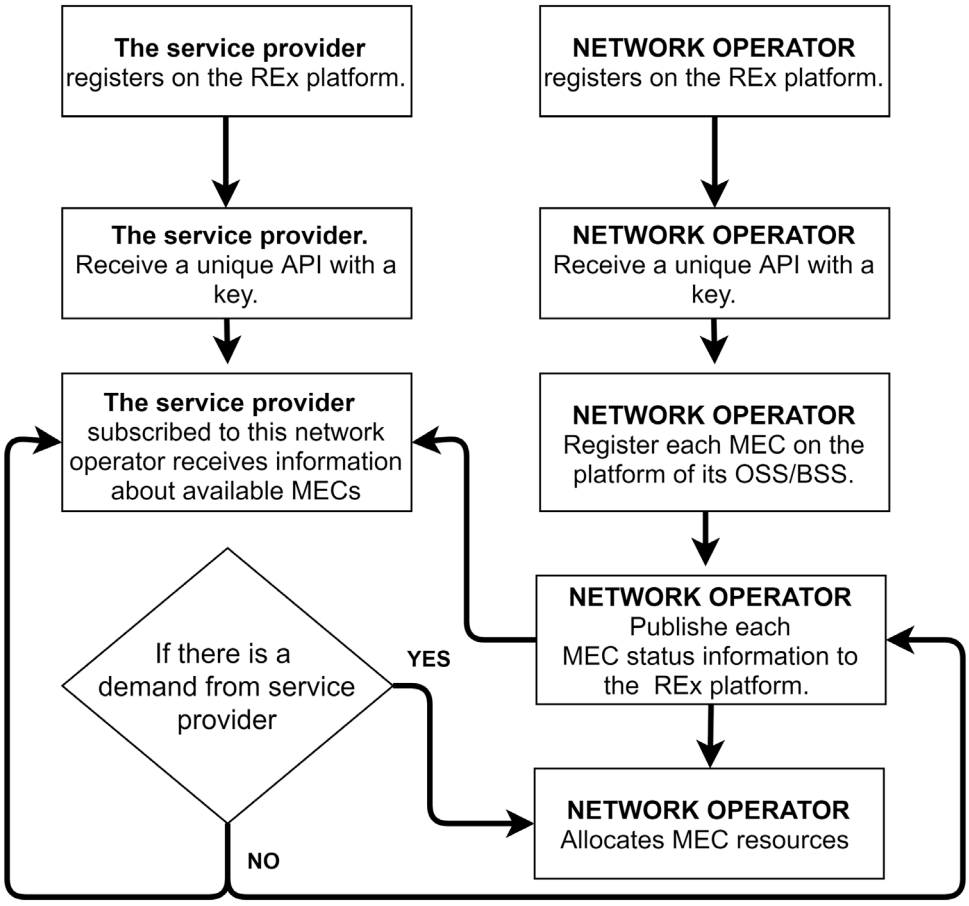

Fig. 3. Flow diagram of connections.

Finally, the requirements and recommendations for REx platform are provided. We assume such a scenario for integrating our platform with a network operator. The network operator on his side installs the REx client, connects through the API to his oss/bss and performs the following actions Fig. 3 :

Our proposed platform will save you from creating an SLA every time between the operator and the communication service provider. SLA is a platform on time. And the additional platform performs the necessary rights and reproduces the application permission as needed. It looks like a very rough analogy, it is like a stock exchange but MEC resources are used as a commodity. In addition, we have already started testing on one of the telecom operators with several third-party services. The operator, using our platform, processes netconf packages, displays a forecast. And publishes options for hosting services through REx to service providers. In some sections of the carrier's transport network, during peak periods, the load dropped to 30%. Looking into the future, we can assume that the massive use of such platforms will significantly save on network nodes or increase the efficiency of these nodes. Service providers will also be able to design their services to integrate with platforms such as REx. One of the optimistic outcomes is that they will reduce cloud spending during peak periods of time while increasing the quality of their service delivery to end users through MEC's close range. The network operator installs the rex client. The REx client connects via API to the OSS/BSS network operator and requests the required data. More precisely: MEC_number; ip_pool; NetFlow_packet; ram; bandwidth; cpu; scheduler_list. REx Client includes a pronouncing tool. It is used optionally. It is also important that before integrating the REx into the OSS/BSS, the IP addresses that belong to or another MEC must be properly designated. This will get rid of forecast errors. However, if the service provider chooses to bind their application without relying on prediction (i.e., on-demand mode), they can only request MEC_number; ip_pool; ram; bandwidth; cpu; scheduler_list, which contains the MEC number, subscriber address list, RAM size, interface bandwidth, processor information, and MEC schedule. But for security reasons, the schedule will not be indicated by the name and owner of the application. But only the amount of the occupied resource and the time period. If the service provider decides to start in automatic mode (i.e. traffic prediction), the REx client in the network will process NetFlow packets and then send recommendations to the required service provider. Unlike the on-demand mode, only those service providers who have signed up to receive recommendations will receive recommendations. Service provider, it is security-driven.

3.1.1. REx platform roles

Let us begin with the main roles of REx platform, where network operator, service provider, REx platform provider, REx platform and MEC are the main components. First, the network operator is the owner of the network infrastructure where the MEC hosts are located. Whereas the service provider is an operator which provides with digital services. Furthermore, REx platform provider and REx platform are the operator providing REx platform services and exchanging digital services between the service provider and the network operator, respectively. Finally, MEC is a computing system with Docker software installed at the edge of the network to host digital services.

The migration object is an application installed in a container. As an application, there can be:

- Web serves
- Content Delivery Network
- Data base servers
- Real-time rendering (for cloud gaming)
- and others

3.1.2. REx architecture

As outlined in the preceding section and shown in Fig. 2, the REx architecture consists of two main items which are REx server and REx client. REx server is installed in the data center of the REx service provider, while REx client is installed in the infrastructure of the network operator and service provider as well as connected by standard API to OSS/BSS of the network operator and connected by standard API to the service provider's management system.

3.1.3. How REx platform works?

Regarding the working of REx platform, the network operator has a different number of MEC hosts in their infrastructure in which the REx client is installed on its network. In addition, the network operator can register their MECs on the REx server based on five metrics which are the service area (list of sites), the available amount of random access memory (RAM), the available bandwidth of the transmission channel, the available processor computing resources and the available computing resources of the video card (for services requiring such parameters)

3.1.4. REx operating modes

The REx platform can be operated in two different modes called on demand placement and automatic mode. Regarding the on demand placement mode, the REx client of the network operator firstly sends information about the available computing power of the MEC to the REx server. In addition, the service provider and the network operator enter into an SLA for the provision of the hosting application service, as shown in Algorithm 2. Then, REx client of the service provider as a subscriber reserves the required MEC for a certain period of time. Finally, the service provider sends the Docker image to the network operator's MEC. For example, 3 A cloud gaming provider as a service provider may reserve several hours of MEC to host local competitions within the network operator's network.

For the automatic mode of REx platform, the network operator firstly includes a traffic analysis system. Then, it provides a recommendation for optimal application placement. Afterward, it sends to service providers a recommendation to place applications in the required MEC. Further, the service provider and network operator enter into an SLA for the provision of hosting application service, as shown in Algorithm 2. Finally, the service provider sends the Docker image to the network operator's MEC. More specifically, algorithm 2 describes the mechanism of interaction between the service provider and the network operator through the REx platform. It describes the step-by-step agreement on the parameters for concluding an SLA. The network operator publishes possible MEC resources and service providers, being subscribers, receive a list of resources and, in case of demand, they take and conclude a digital SLA, which is accompanied by a platform, including a temporary key for occupying MEC resources.

3.1.5. Requirements and recommendations of REx platform

When uploading a Docker image, there must be a JSON file with information about the duration of the deployment and the technical requirements for the Docker container.

It is recommended for reliability to create virtual private network (VPN) tunnels between the MEC and the service provider during the service migration.

3.1.6. REx platform

One of the important elements of our architecture is the REx Platform. REx Platform is an independent platform where data is exchanged between the service provider and the provider, as shown in Fig. 4. The need to introduce the element is explained by the fact that organization channels between the operator and the service provider, or rather business to operator (b2o), are not always possible to organize. Especially if the service provider is located in another country. REx Platform allows operators to place information about their services (communication channel, edge cloud lease …) on their site. And service providers can book the operator's resources for a certain time in real through the REx Platform. For example: The operator predicts that the resource is available from 17: 00–19: 00 in edge cloud 1 with RAM-16 GB, CPU-2.00 GHz, Bandwidth-140 Mbps and Cost-50 dollars. Subsequently, REx sends this information to all service providers. And if the service provider decides to use the service, it issues a digital contract. And it starts migrating its application to edge cloud 1.

Sequence diagram of REx operation is described in Fig. 5, in which the whole process of interaction between the operator and the service provider is performed automatically.

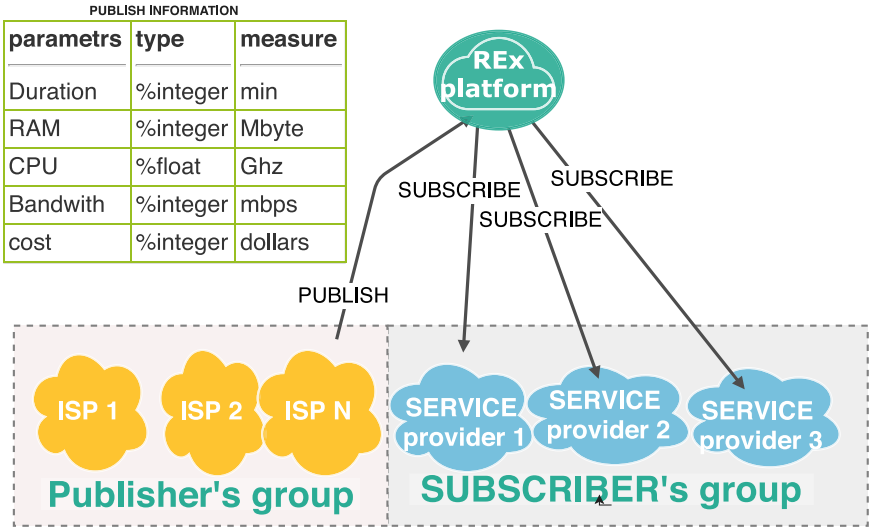

Fig. 4. REx Platform.

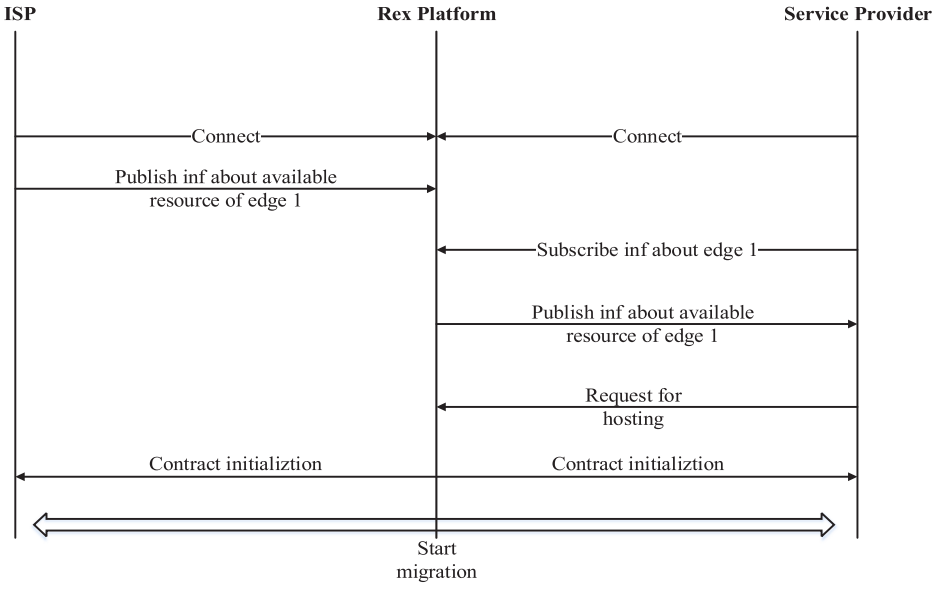

Fig. 5. Sequence diagram of REx operation.

3.2. System model

In case when the system operates under on demand placement mode, it can be modeled as a resource queueing system consisting of two nodes: REx Platform and MEC node. For that purpose, we consider that there are only K service providers in the system and each provider only sends requests to place one application type. In case a provider can send requests for several different application types, it can be modeled as several providers, each with one application type.

Service providers send requests to the REx system to place their applications on MEC at a random time. Thus each service provider can be modeled as an independent arrival flow, i.e., Poisson distributed with rate $\lambda_k, k = 1, \ldots, K$ for kth service provider. All the requests are served at the REx Platform. In this case, the requested service implies that the request is being sent to the corresponding MEC; Thus the service time is random and depends on the application. In the simplest case, it can be modeled as an exponentially distributed random variable. However, the actual service time is given to the system with the request arrival, since at that moment actual application size is given and thus can also be modeled as a constant for each application. It can be calculated as the volume of application data υ_k divided by the channel bandwidth, available for this application C_k for kth service provider, $k = 1, \ldots, K$, in our case $C_1 = \cdots = C_K$.

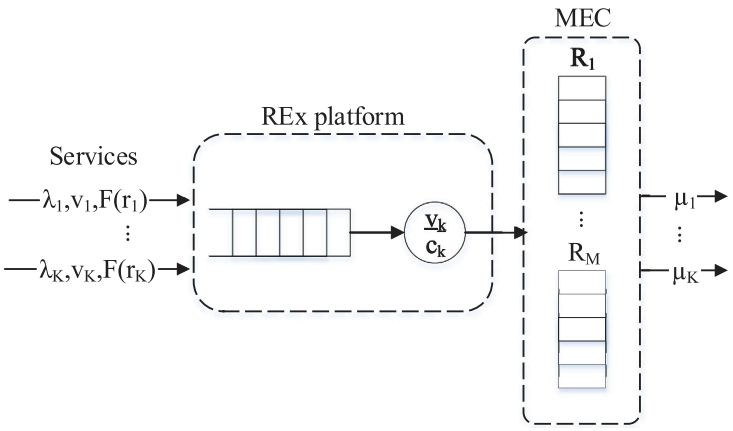

Fig. 6. Queueing model of REx system.

Since the system can send requests to different MECs simultaneously without these requests interfering with each other, we can simplify model by only considering single MEC. In this case, the requests will be serviced sequentially, since each application will occupy the channel while it is being sent to the MEC and no other application can be sent via that channel at that time. Thus the newly arrived requests are placed into the queue before going to the MEC node. The service order, in this case, was chosen to be FIFO (First In First Out) as the default queueing discipline, however, priority queueing can also be considered depending on the system implementation.

On the completion of the service in the REx Platform the request is send to the MEC where it occupies a number of M finite resources $R_1, \ldots, R_M$ distributed with a given function $F_k(\mathbf{x})$, where $\mathbf{x} = \{x_1, \ldots, x_M\}$ for kth service provider according to the application type. In that case, resource can be interpreted as the computational power, memory space and other hardware required by the application. The resources are occupied for a random time. In the model, depicted in Fig. 6 the time is exponentially distributed with mean μ_k for kth service provider.

By modeling the system 6 in this way we can calculate the MEC resources needed to prevent congestion depending on the number of providers and application requirements. Also, it is possible to analyze how the channel bandwidth between the REx Platform and MEC impacts the application download time under high load.

4. Simulation model

In this chapter, we take apart a simulator that was written for the purpose of exploring our proposed platform. AnyLogic software was chosen as the simulator tool. And the proposed system model in Section 3.2 was taken as a source for computer modeling. As a result, the design of the model turned out as shown in Fig. 7.

Elements	Value
Source	Input service
Queue	
SelectOutput5	An element that distributes services across resources
delay, dalay1, delay2, delay3, delay4	Resources of MEC

To simulate the processing request process, we define the arrival process following the Poisson process, which is the average service arrival rate. Moreover, the service time is distributed exponentially at the average service rate. To associate RAM, CPU, storage size, bandwidth and GPU requirements, we randomly generate requirements and associate them with the corresponding request. In this simulation model, user requests are equivalent to requests going into the platform that traverses the Poisson described earlier. All service requirements are "service request arriving", all service requirements and arrival times are inserted into the CPU, RAM and other requirements. The row of this generated matrix represents the service request that arrived at a point in time. Another matrix is created to represent the state of the MEC server platform at a specific point in time.

As a result of the simulation, we got the probability of blocking the placement of services on a different number of MECs with different rates of service arrival, as shown in Fig. 8

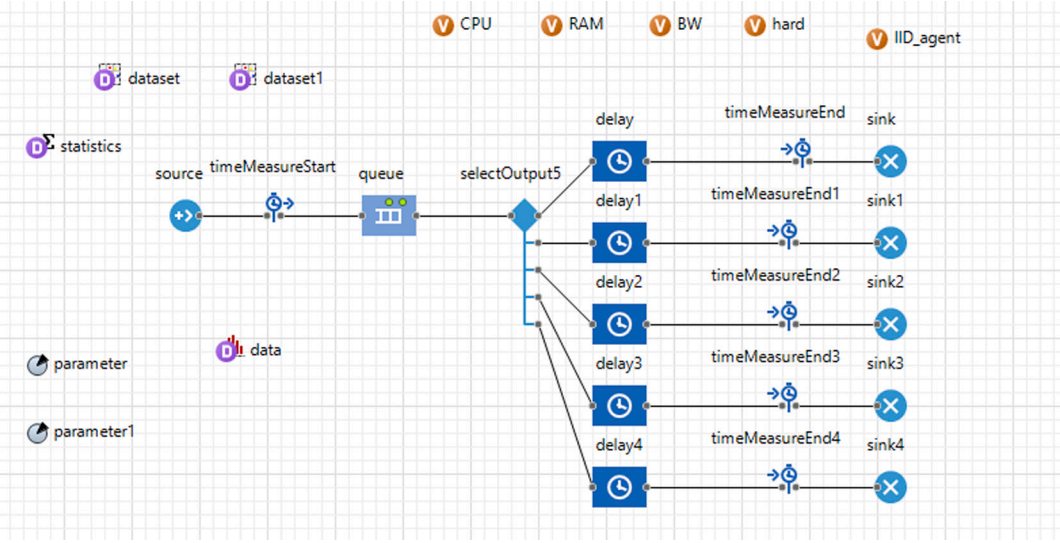

Fig. 7. Simulation model.

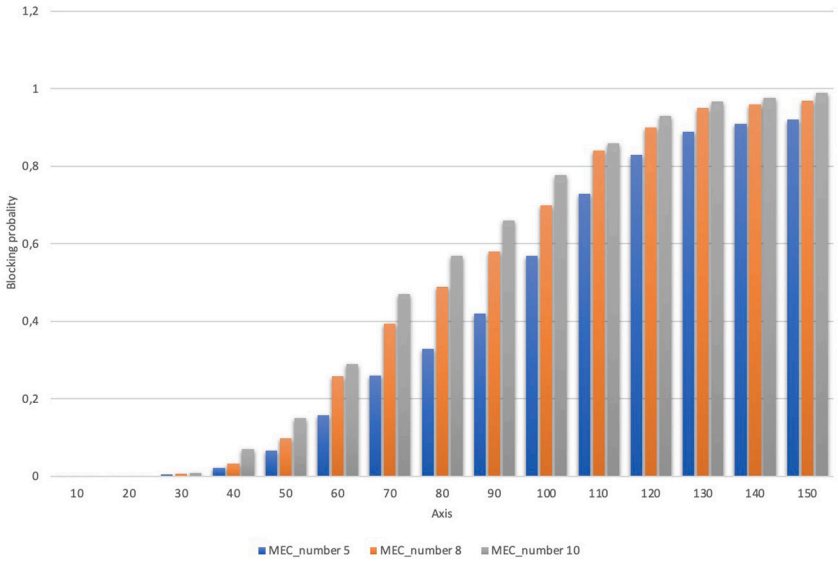

Fig. 8. Simulation results.

5. Experiment composition

As mentioned above, REx platform can be operated in two different modes called on demand placement and automatic mode. Therefore, in the following subsections, we show the effect of using our platform under these modes. Afterward, the accuracy of the genetic algorithm predication is presented. Furthermore, the prediction error in one day is discussed. Finally, the interface of REx platform is introduced.

5.1. Case 1. Migration using REx on demand placement

In this subsection, the process of migration is tested using REx platform under on demand placement operation mode. The on-demand experiment works as follows. Firstly, the local network is spitted into seven segments and the MEC (With the parameters given in Table 2) host is installed for each segment. Then, an IoT application with the MQTT protocol is installed in a remote cloud. Afterward, in the local network, 1000 devices were configured and connected to a remote cloud. In addition, these devices are

Table 2
Notations.

Lanner FW-8894 switch	Description
RAM	32 GB
CPU	Intel® Xeon® CPU E5-2650 v4 @ 2.20 GHz.
SDN controller	
RAM	45 GB
CPU	Intel® Xeon® CPU E5-2620 v4 @ 2.10 GHz.
Node A/B(source/target)	.
RAM	64 GB
CPU	Intel® Xeon® CPU E5-2620 v4 @ 2.10 GHz.

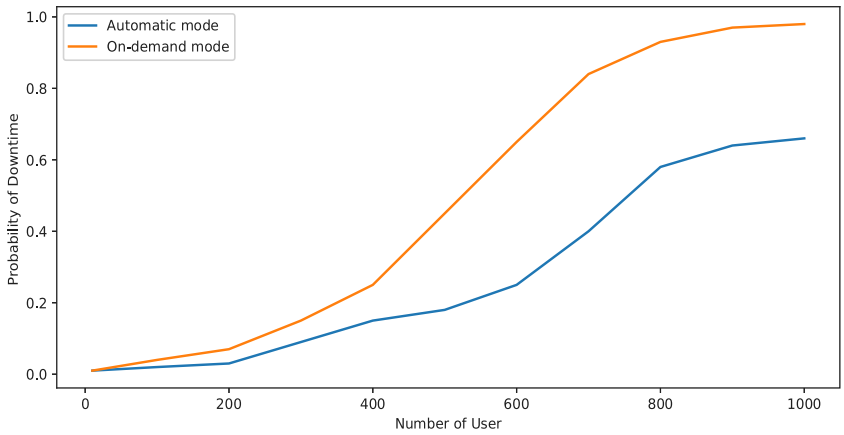

Fig. 9. Case 2 results.

distributed into seven zones with different densities (i.e., Zone 1–50 IoT device, Zone 2–150 IoT devices, Zone 3–250 IoT devices, Zone 4–350 IoT devices, Zone 5–100 IoT devices, Zone 6–70 IoT devices and Zone 7–30 IoT devices).

Assigned different intensity of access to the remote cloud with a step of 1 h. Thus, at different times of the day from different segments, we received different peaks of requests to the cloud. Next, our REx platform was configured for on-demand mode. The platform determined the automatic peak of requests and offloaded a replica of the IoT application to the MEC of a specific zone. In this mode, the platform works on a post factum basis. Begins the migration process only after it detects a peak in zone utilization.

Furthermore, Fig. 9 shows the results of testing the REx platform migration module for the probability of failure was calculated versus a different number of users. In addition, the probability of a session reset during migration in on-demand mode. The algorithm by which the test was performed is given in Algorithm 1. More specifically, this algorithm is responsible for the testing method of platform to assess the quality of the application, in which two hosts are indicated in one of them and there is an application placed in the docker container. Afterward, for migrating the service, the clone should be moved through a dedicated virtual p2p channel. Then, synchronization should be started and transferred of the active session.

5.2. Case 2. Migration using REx automatic mode

In this subsection, the process of migration is tested using REx platform under automatic operation mode. The migration process is based on GA (genetic algorithm) and post factum (come up with a normal word) as well as comparison of the probability of service failure. RTCP + RTP traffic generator service. At the entrance, we submit statistics (in NetFlow format) of the previous days to the GA system. GA determines the best option for placing a service in one of 7 segments based on the number of requests in a given period, the average volume of packets, the required computing and network resources (RAM, RAM, minimum bandwidth) and the volume of the container. From this, we get 6 chromosomes, as shown in Table 3.

As you know, GA works more efficiently and faster with paired parents. Therefore, we divided the chromosomes in pairs of RACE1 $\{1,2\}$, RACE2 $\{3,4\}$, RACE3 $\{5,6\}$ - at each stage of mutation, all three RACEs are checked. Thus, we find the best option. The post-facto process is based on the principle of the current service indicators HERE and NOW.

Dependence of the calculation of the best service placement on the Hurst index. From Case2, we remember that one of the chromosomes of our GA is the intensity (request/h). Since we receive a large number of statistics in the NetFlow format, we can calculate our Hurst coefficient for daily traffic. Then we simulated traffic generation based on existing statistics but with different intensities, leading to the desired coefficient at the output of the generator. In other words, we are engaged in crafting. As a result, we will be able to determine how many generations the GA will calculate the required node for migration. The proposed platform includes the ability to quickly and short-term SLA conclusion between the service provider and the network operator. Having the

Algorithm 1 Service Migration

```
Run docker container on node A.
Input: N (number of session), P (number of migration), j (iteration number)
for all i=0, j<=P do
    stA=container status on node A.
    stB=container status on node B.
    if actual number of session == N & stA ==1 then
        Cloning container from ftp − server to node B
        Sync processes
        Reorganize the p2p channel (restAPI)
        Delay (2s)
        Send to generator command to check loss packets
    else if actual number of session==N & stB==1 then
        Cloning container from ftp − server to node A
        Sync processes
        Reorganize the p2p channel (restAPI)
        Delay (2s)
        Send to generator command to check loss packets
    end if
end for
```

Table 3
Notations.

Chromosome number	Meaning	Variable
1	Number of requests/h	A
2	Average package size	S
3	RAM	V
4	CPU	K
5	Minimum bandwidth	B.
6	Container size	C.

Algorithm 2 The algorithm for concluding an SLA between the service provider and the network operator

```
Procedure: OSS/BSS informs floating form REx about free computing resources on all MECs of the network operator.
while true do
    REx: publishes information on MEC resources for subscribers to hosting services (service providers)
    if REx: received a request from a service provider to host a service then
        OSS / BSS: concludes a digital SLA between the operator and the service provider
    else
        OSS / BSS: updates information about the available resources of the MEC
    end if
    REx: transfers the authorization key to the service provider for the right to use the MEC resource.
end while
```

trusted status and the necessary REx rights, the platform enters into SLA separately and with the service provider and the network operator. SLA negotiation options include

- Duration of MEC hosting
- Start time of the hosting
- Duration of the application migration process
- RAM size
- ROM size
- CPU parameters
- GPU parameters
- Bandwith

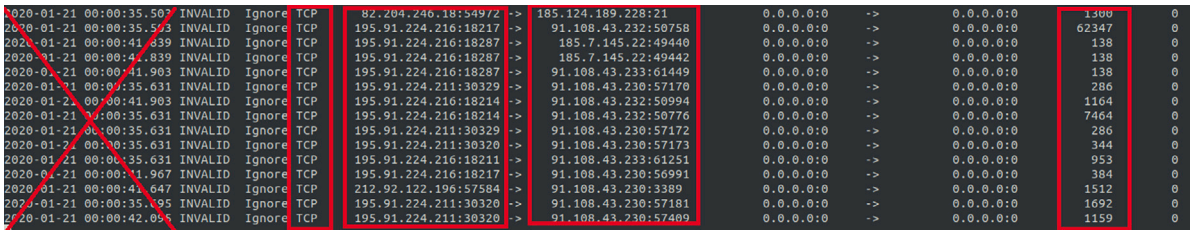

Fig. 10. Node migration.

5.3. Accuracy of GA prediction

To integrate the prediction of the genetic algorithm, we used the network of a real operator in St. Petersburg (Russia) as daily data which imported from the NetFlow collector for 8 months. In addition, based on these input packages, we created a forecasting model. GA creates a function for several arguments, and based on this function, a forecasting model is created. The forecast arguments, shown in Fig. 10, were ip src, ip dst, port, time, duration, intensity. Accordingly, the larger the dataset, the more accurate the forecast. Each MEC is tied to a specific zone, and a group of ip subscribers is tied to each zone. This is how the service areas of the MECs can be determined. The next day was used as the verification data. For example, we predict the behavior of user service requests for September 1. And when September 1 hits, the platform checks the forecast accuracy. This is how the forecasting model is adjusted.

GA is used to determine the hourly forecast error. Set 2-day (Mon–Tue) NetFlow statistics. GA determines the forecast for the next week, and calculates the deviation after the first 2 days have passed. The largest number of errors occurs in the interval (13–16) hours shown in Fig. 12. We assume that such an output result is obtained due to the unevenness of user traffic during the working day and the small amount of NetFlow data as a dataset. We also do not discard the situation with the coronavirus, which influenced the course of the experiment. In the middle of the research, a lockdown was introduced. And that influenced the change in service requests. However, in the future, we modify the forecasting tool of the REx platform

To predict services, the forecasting module is used, as input data, the impersonal NetFlow data of the telecom operator was used. The source ip, service ip, protocol, request time and packet size are indicated. NetFlow data must be pre-processed. For example, in our work, we have excluded constant private transmissions of packets that take place on a strictly defined schedule, such as a backup. Since these strictly persistent packets can introduce errors in the forecasting of services.

Genetic algorithms (GA) use concepts from the principle of natural selection to solve a wide range of problems, in particular optimization. They are reliable, versatile and adaptable. Inspired by how Darwinism explains the evolutionary process of species, GAs are broken down into the following stages: initiation, evaluation, selection, crossover, mutation, renewal, and completion. Fig. 11 describes the general structure of a simple GA.

Individuals are the fundamental unit of the genetic algorithm: they encode possible solutions to a problem that needs to be solved, and the answers to them are found precisely through its manipulation (in the process of evolution). The selection of representation for individuals is the most important step in the development of the GA, as it will have primary responsibility for the implementation of the program. In this study, each individual is represented by a set of five real numbers: *source ip, dist ip, packet size,time,date* Each of these numbers is called a gene.Each gene is a time series forecasting model. Each individual is made up of a combination of genes representing each of the predictive models used.

$$individual = [src.ip,\ dist.ip,\ P.size,time,date]$$

5.4. Prediction error in one day

In this subsection, the prediction error in one day is discussed, in which the evaluation of efficiency distribution using genetic algorithm with and without the addition of forecasting is presented. In the course of the study, experiments were reproduced in which they compared the efficiency of using the network infrastructure using prediction by the genetic algorithm and the distribution of services without prediction.

5.4.1. Without forecasting

The distribution method without prediction is shown in the following algorithm.

Our infrastructure starts to migrate the service to the nearest group of users as soon as the controller notices that 30% + of users start using the same service.

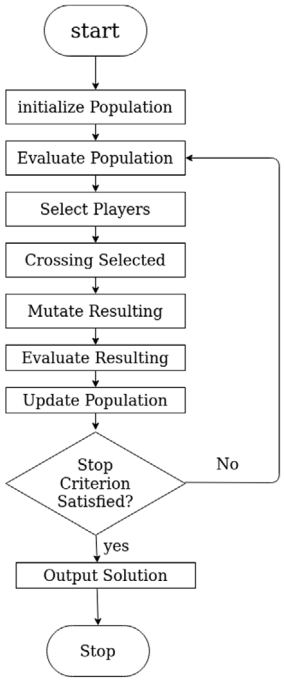

Fig. 11. The overall structure of genetic algorithm.

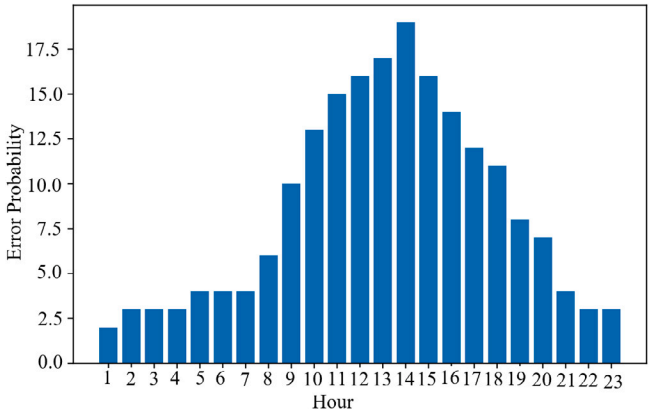

Fig. 12. Estimation error.

5.4.2. Forecasting using genetic algorithm

The network controller receives NetFlow statistics over a long period of time and divides network segments by day of the week. Then, every day begins to conduct analytic on a frequent basis with predicting user traffic. And at the end of the day, it compares the result with the reference one and displays an error. Afterwards, at the moment when the error reaches less than 20%, the early migration process begins.

For this case, the sequence of the platform operation is shown in Fig. 13. The network operator loads NetFlow packets through the client's REx into a genetic algorithm. it calculates the forecast and gives the best recommendation for the placement of service provider applications in the MEC infrastructure. Next, the REx platform publishes the results of the required service provider recommendation. In case of an agreement between the parties (service provider and network operator), the migration of the application docker image to the required MEC node begins.

The experiment uses 2-day (Mon–Tue) NetFlow statistics. GA determines the forecast for the next week, and calculates the deviation after the first two days have passed. The deviation is the ratio of the difference in the number of predicted requests to the real number of requests. $E_{error} = \frac{\sum_{i=1}^{n} \frac{|F-R|}{R}}{n}$

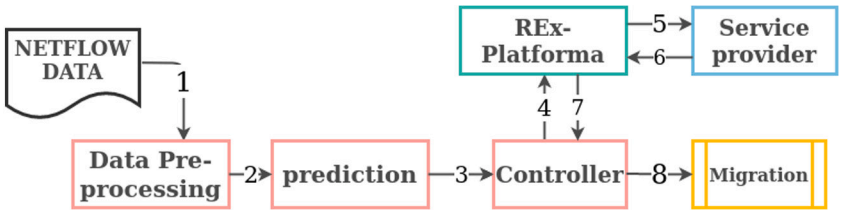

Fig. 13. Connection diagram.

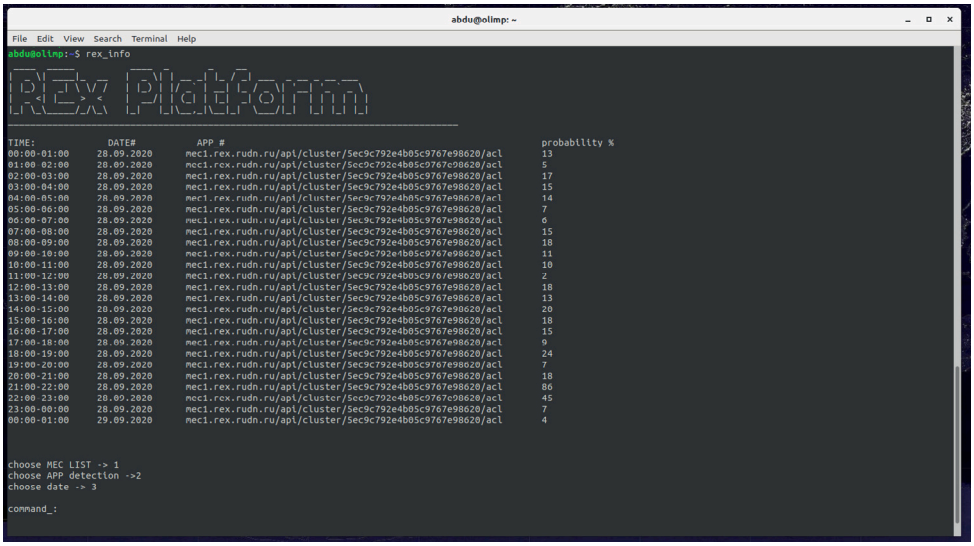

Fig. 14. REx platform interface.

5.5. REx interface

Its observed form Fig. 14 that through the interface of REx platform we can directly monitor the status and probability of using a particular application on MEC devices in real-time.

6. Case2

Another experiment was used to test the platform. One hundred users were created, each of them was randomly linked to 7 different MECs. Within 5 weeks, on Mondays, Thursdays and Saturdays, within an hour, a request for the same content was launched from different groups of subscribers. The requests were normally distributed with the average displayed on Fig. 15. The content was a video stream with a VLC application.

The genetic algorithm, getting statistics from NetFlow for 5 weeks, was able to determine the close value of the average requests for content from different MECs. Based on this, in the 6th week, REx platform was able to predict the time when there will be a similar request. And in advance, migrated the application with the necessary content to the necessary MECs. In the experimental stand, this gave the effect of reducing the occupancy of the uplink channel.

6.1. Result

We assume such a scenario for integrating our platform with a network operator. The network operator on his side installs the REx client, connects through the API to his oss/bss and performs the following actions:

1. The service provider registers on the REx platform.
2. Receive a unique API with a key.
3. Register each MEC on the platform of its OSS/BSS.
4. Publish its MEC status information to the platform.
5. Service provider subscribed to this carrier receives information about available MECs.
6. In case of interest, rents the required MEC for the required time.

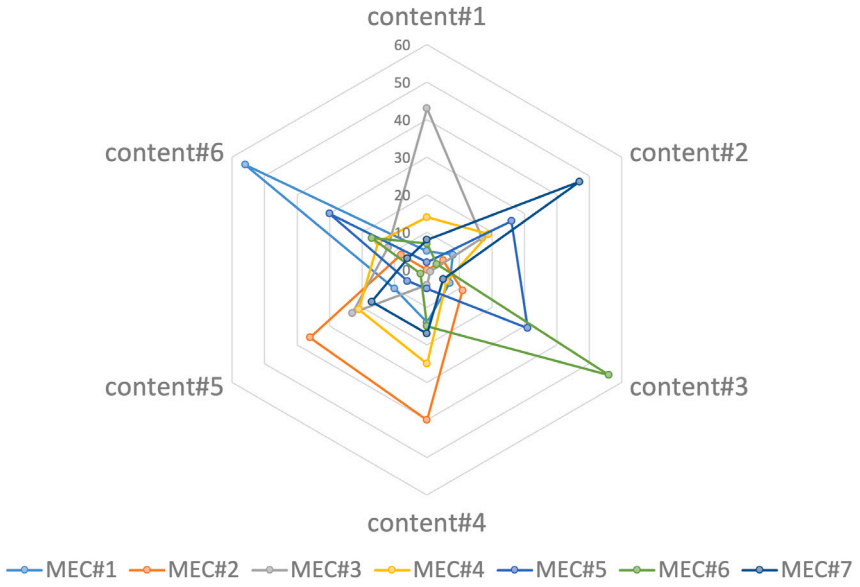

Fig. 15. Distribution of content by zones.

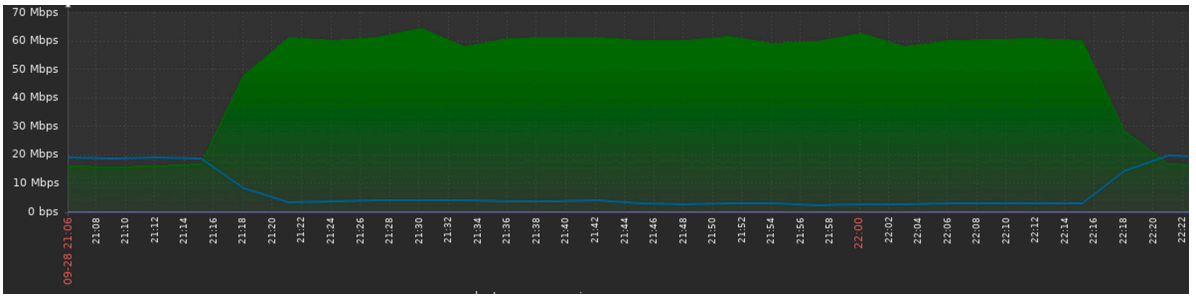

Fig. 16. Loaded channel.

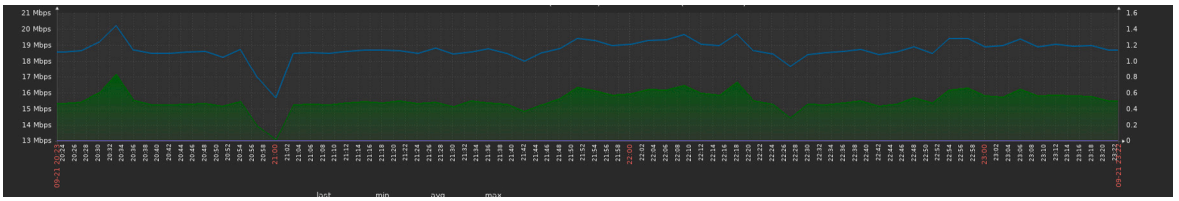

Fig. 17. Unloaded channel.

Moreover, our proposed platform will save you from creating an SLA every time between the operator and the communication service provider. SLA is a platform one time. And the additional platform performs the necessary rights and reproduces the application permission as needed. It looks like a very rough analogy, it is like a stock exchange but MEC resources are used as a commodity.

As a result, we managed to unload the transport interface from mass viewing of groups of subscribers in the network section. As shown in Figs. 16 and 17.

The REx platform was able to predict content requests in lab conditions. And install the application in advance. Such a case would be well suited for optimization during the broadcast of TV series or youtube videos with high user expectations. Since recently, users are increasingly using online cinemas. And, accordingly, multicast broadcasting is not suitable for this. In general, such a platform is a plus for

1. Telecom operator. Service provider and subscriber.
2. Network operator — saves transport network resources for similar traffic.

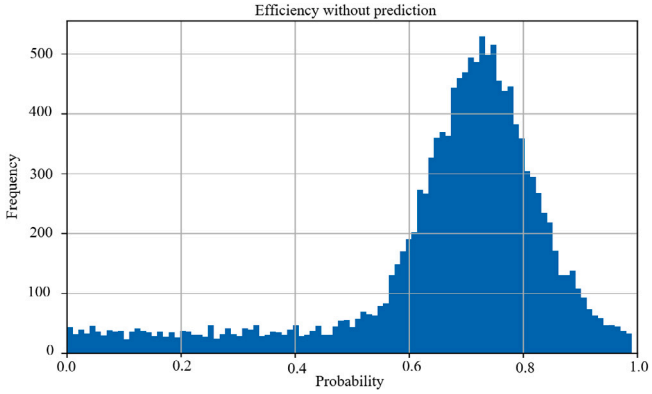

Fig. 18. Efficiency without prediction.

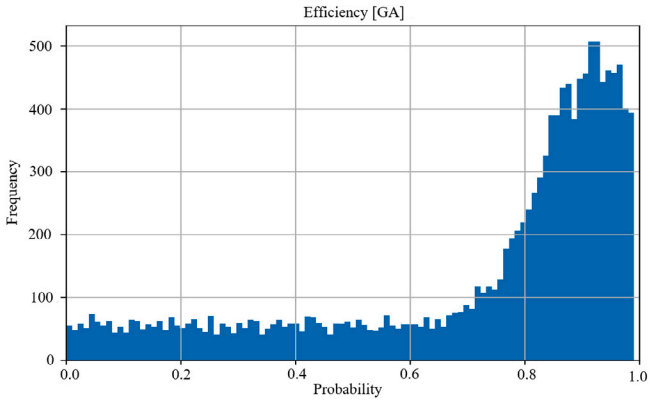

Fig. 19. Efficiency [GA].

3. Service provider reduces the load of requests from central clouds. Thus, it does not rent additional computing power during peak hours.
4. Customers receive services with low latency.

In the final stage, we compared the effectiveness of the two modes of operation of the REX platform. And we calculated the distribution of efficiency. Where a value of 1 means complete efficiency. By efficiency, we have defined a certain value that determines how much the uplink of the presenter in MEC is freed from the traffic leading to the service provider. And defined by the expression:

$$K_{efficiency} = \frac{PACKETS_{MEC}[i]}{PACKETS_{Total}[i]} \tag{1}$$

where $PACKETS_{MEC}$ is the number of packets of the ith application accessing MEC and $PACKETS_{Total}$ is the number of Total packets of the ith application.

In case 1 (on request) operating mode, we migrated services where forecasting tools were not used and the service provider was directly responsible for making a decision. And the second case where forecasting was used with the GA tool. As a result, the following values were obtained, where the use of GA (Fig. 19.) shows greater efficiency due to the high repetition rate of successful shuffled application images. And in Fig. 18, where the person analyzed the effectiveness on his own, it is somewhat worse.

7. Conclusion

With the development of networks and a smooth transition to 5G & 6G networks, the number of subscribers' traffic will grow exponentially. Standard methods for improving communication channels are no longer working. Therefore, it is necessary to vector development of networks towards dynamically configurable systems. Thanks to our proposed platform, it will be possible to Reconnect different telecommunication services into a single ex-system. In this article, we divided the roles of the participants: Network operator, service provider, and REx platform. In addition, we assumed that the proposed platform will not only improve the technical system of the telecommunications industry, but also the business model. Further, we observe that in the future, a

universal model of communication networks, where many network operators and service providers will connect to such platforms, and service providers, as needed, place their applications on the network operator's MEC infrastructure. This model of our vision of the networks of the future can satisfy all participants such as communication operator (optimization of the internal transport network) and service provider (reduces the load of requests from the central cloud). Thus, it does not rent additional computing power during peak hours. Furthermore, a single point of resource exchange in the face of the REx platform will improve the system of interaction between telecom operators and service providers by automating the SLA agreement system. Finally, the work of the REx platform in an automated mode will optimize the interaction of all parties.

In ongoing and future work, we plan to integrate other methods of predicting machine learning, and ARIMA method. Thus, to reduce errors in the forecast. Also assessed by other performance indicators.

Acknowledgment

This paper has been supported by the RUDN University Strategic Academic Leadership Program.

References

[1] S. Forge, K. Vu, Forming a 5G strategy for developing countries: A note for policy makers, Telecommun. Policy 44 (7) (2020) 101975.
[2] S. Dang, O. Amin, B. Shihada, M.-S. Alouini, What should 6G be? Nat. Electron. 3 (1) (2020) 20–29.
[3] Y. Lu, X. Zheng, 6G: A survey on technologies, scenarios, challenges, and the related issues, J. Ind. Inf. Integr. (2020) 100158.
[4] M. Khayyat, A. Alshahrani, S. Alharbi, I. Elgendy, A. Paramonov, A. Koucheryavy, Multilevel service-provisioning-based autonomous vehicle applications, Sustainability 12 (6) (2020) 2497.
[5] A.A. Ateya, A. Vybornova, A. Muthanna, E. Markova, I. Gudkova, A. Gogol, A. Koucheryavy, Key solutions for light limitations: toward tactile internet system realization, in: Proceedings of the 2nd International Conference on Future Networks and Distributed Systems, 2018, pp. 1–6.
[6] S.K. Sharma, I. Woungang, A. Anpalagan, S. Chatzinotas, Toward tactile internet in beyond 5G era: Recent advances, current issues, and future directions, IEEE Access 8 (2020) 56948–56991.
[7] O.S. Oubbati, A. Lakas, F. Zhou, M. Güneş, M.B. Yagoubi, A survey on position-based routing protocols for flying ad hoc networks (FANETs), Veh. Commun. 10 (2017) 29–56.
[8] M.S.A. Muthanna, P. Wang, M. Wei, A.A. Ateya, A. Muthanna, Toward an ultra-low latency and energy efficient lorawan, in: O. Galinina, S. Andreev, S. Balandin, Y. Koucheryavy (Eds.), Internet of Things, Smart Spaces, and Next Generation Networks and Systems, Springer International Publishing, Cham, 2019, pp. 233–242.
[9] J. Lee, R. Balachandran, Y.S. Sarkisov, M. De Stefano, A. Coelho, K. Shinde, M.J. Kim, R. Triebel, K. Kondak, Visual-inertial telepresence for aerial manipulation, 2020, arXiv preprint arXiv:2003.11509.
[10] A. Sau, R.D. Roychoudhury, H.B. Barua, C. Sarkar, S. Paul, B. Bhowmick, A. Pal, et al., Edge-centric telepresence avatar robot for geographically distributed environment, 2020, arXiv preprint arXiv:2007.12990.
[11] K. Samouylov, E. Popov, K. Semyachkov, Institutional support of a smart city, Monten. J. Econ. 15 (2019) 87–98, http://dx.doi.org/10.14254/1800-5845/2019.15-4.7.
[12] S. Al-Sarawi, M. Anbar, R. Abdullah, A.B. Al Hawari, Internet of things market analysis forecasts, 2020–2030, in: 2020 Fourth World Conference on Smart Trends in Systems, Security and Sustainability (WorldS4), IEEE, 2020, pp. 449–453.
[13] R.V. O'Connor, P. Elger, P.M. Clarke, Continuous software engineering—A microservices architecture perspective, J. Softw.: Evol. Process 29 (11) (2017) e1866.
[14] S. Shorgin, K. Samouylov, I. Gudkova, O. Galinina, S. Andreev, On the benefits of 5G wireless technology for future mobile cloud computing, 2014, http://dx.doi.org/10.1109/MoNeTeC.2014.6995601.
[15] M. Alam, J. Rufino, J. Ferreira, S.H. Ahmed, N. Shah, Y. Chen, Orchestration of microservices for iot using docker and edge computing, IEEE Commun. Mag. 56 (9) (2018) 118–123.
[16] I.A. Elgendy, W. Zhang, Y.-C. Tian, K. Li, Resource allocation and computation offloading with data security for mobile edge computing, Future Gener. Comput. Syst. 100 (2019) 531–541.
[17] L. Yan, S. Cao, Y. Gong, H. Han, J. Wei, Y. Zhao, S. Yang, Satec: A 5G satellite edge computing framework based on microservice architecture, Sensors 19 (4) (2019) 831.
[18] I.A. Elgendy, W.-Z. Zhang, Y. Zeng, H. He, Y.-C. Tian, Y. Yang, Efficient and secure multi-user multi-task computation offloading for mobile-edge computing in mobile IoT networks, IEEE Trans. Netw. Serv. Manag. (2020).
[19] W.Z. Zhang, I.A. Elgendy, M. Hammad, A.M. Iliyasu, X. Du, M. Guizani, A.A.A. El-Latif, Secure and optimized load balancing for multi-tier IoT and edge-cloud computing systems, IEEE Internet Things J. (2020) 1, http://dx.doi.org/10.1109/JIOT.2020.3042433.
[20] Z. Zhang, Y. Xiao, Z. Ma, M. Xiao, Z. Ding, X. Lei, G.K. Karagiannidis, P. Fan, 6G wireless networks: Vision, requirements, architecture, and key technologies, IEEE Veh. Technol. Mag. 14 (3) (2019) 28–41.
[21] A. Paramonov, A. Muthanna, O.I. Aboulola, I.A. Elgendy, R. Alharbey, E. Tonkikh, A. Koucheryavy, Beyond 5G network architecture study: Fractal properties of access network, Appl. Sci. 10 (20) (2020) 7191.
[22] J. Guo, C. Li, Y. Chen, Y. Luo, On-demand resource provision based on load estimation and service expenditure in edge cloud environment, J. Netw. Comput. Appl. 151 (2020) 102506.
[23] V. Kopparty, V. Manjunath, K. Subramaniam, P. Jain, V.M. Gadag, Efficient methods for bearer traffic flow migration in NG-RAN, in: 2018 IEEE Wireless Communications and Networking Conference, WCNC, IEEE, 2018, pp. 1–6.
[24] J. Zhou, X. Zhang, W. Wang, Joint resource allocation and user association for heterogeneous services in multi-access edge computing networks, IEEE Access 7 (2019) 12272–12282.
[25] M. Zakarya, L. Gillam, H. Ali, I. Rahman, K. Salah, R. Khan, O. Rana, R. Buyya, Epcaware: a game-based, energy, performance and cost efficient resource management technique for multi-access edge computing, IEEE Trans. Serv. Comput. (2020).
[26] J. Carrasco, F. Durán, E. Pimentel, Live migration of trans-cloud applications, Comput. Stand. Interfaces 69 (2020) 103392.
[27] J. Yao, Q. Lu, H.-A. Jacobsen, H. Guan, Robust multi-resource allocation with demand uncertainties in cloud scheduler, in: 2017 IEEE 36th Symposium on Reliable Distributed Systems, SRDS, IEEE, 2017, pp. 34–43.
[28] S.Y.Z. Fard, M.R. Ahmadi, S. Adabi, A dynamic VM consolidation technique for QoS and energy consumption in cloud environment, J. Supercomput. 73 (10) (2017) 4347–4368.
[29] T. Bahreini, D. Grosu, Efficient placement of multi-component applications in edge computing systems, 2017, pp. 1–11, http://dx.doi.org/10.1145/3132211.3134454.

[30] C. Li, J. Bai, J. Tang, Joint optimization of data placement and scheduling for improving user experience in edge computing, J. Parallel Distrib. Comput. 125 (2019) 93–105.
[31] L.I.C. Vidyasagar, D. Pal, P. Khethavath, Optimized resource allocation and load balancing in distributed cloud using graph theory, in: 2018 International Conference on Advances in Computing, Communications and Informatics, ICACCI, 2018, pp. 2054–2058, http://dx.doi.org/10.1109/ICACCI.2018.8554929.
[32] X. Ma, S. Wang, S. Zhang, P. Yang, C. Lin, X. Shen, Cost-efficient resource provisioning for dynamic requests in cloud assisted mobile edge computing, IEEE Trans. Cloud Comput. (2019) 1.
[33] Q. Vo, D.A. Tran, Probabilistic partitioning for edge server assignment with time-varying workload, in: 2019 28th International Conference on Computer Communication and Networks, ICCCN, 2019, pp. 1–8, http://dx.doi.org/10.1109/ICCCN.2019.8846932.