Modeling placement of service-oriented cloud applications in a software-defined infrastructure of virtual data center

I. Bolodurina[a], D. Parfenov[a]

[a] *Orenburg State University, 460018, 13, Pobedy ave., Orenburg, Russia*

Abstract

The paper describes efficient algorithm for optimization technology of containerization cloud applications and services in the virtual data center (VDC) infrastructure. We propose an efficient algorithm for placing applications in the infrastructure of a VDC The problem of optimization of placing services oriented cloud applications based on the template VM and containers with disabilities infrastructure of VDC is reduced to the problem of packing in containers. We also generalize the well renowned heuristic and deterministic algorithms of Karmakar-Karp.

Keywords: software-defined network; virtual data center; cloud applications and services; IT infrastructure; virtualization.

1. Introduction

The cloud computing technology is built on the virtualization base of separate components involved in infrastructure of the data center. The approaches to organization of the virtualization layer are divided according to levels of this technology application. Generally, the following types of virtualization are marked out: operating system, software, memory, data storage, database, and network. The operation system virtualization is the most actively developed level. It helps to create a virtual environment for some users' working space in the scope of one operation system, used for application and services launching inside network environment. The technology of containerization cloud applications and services is the most dynamically developing approach in this regard. The container appears as an object providing the user an access to needed libraries and containing required set of software for launching the environment for creating and finished applications or services. Containerization is solving the problems of depending software on different conflicting versions of libraries used. Besides, comparing containerization and virtualization for program disposal we can notice that the first works on Linux kernel and appears as very light-weight decisions with minimum of productivity costs. In case of virtualization, virtual machines will work under control of hypervisor, which emulates computer hardware. There are also a valid operational system functioning inside every virtual machine and a program working on it. All that leads to extra significant expenses in resource consumption. The inability to provide the due level of dataflow isolation appears as a significant disadvantage of containers. In the scope of this investigation, we suggest the decision that enables to find a solution to the problem of the containers migration. The created decision is founded on the combination of two approaches to resources virtualization (container method and method based on virtual machines) when deploying a software-defined infrastructure of the virtual computation center.

Nowadays the most widespread systems of controlling the containers are Docker. It got the wide spreading because of the simple usage, reliability, availability of an open code and convenient API for using in exterior projects.

With the resources virtualization technology development the number of layers that form infrastructure decisions and are using in the cloud computing technology is steadily increasing. Nowadays can be defined up to six levels of software-defined infrastructure of virtual data center which are using for deploy modern cloud platforms.

Let's describe a model of the virtual data center structure based on the technology of containerized cloud application and services.

2. The level distribution model of software-defined infrastructure with technology of containerization cloud applications and services

Let us introduce the leveled model of software-defined infrastructure of virtual data center, which support the containerization method of application and services disposal in the cloud system. The first level – the hardware component of any data center, which includes computational nodes (Nodes), filing systems (Storages) and physical network units (NetObj). Let us introduce it as the set of decisions: $PhysLayer=\{Nodes, Storages, NetObj\}$.

The next level represents the software-defined layer. This layer consists of the same number of objects as the first level but the main difference is in the fact that all the infrastructure elements are dynamic, easy-transformed and adjusted within the limits of the physical database network environment. The second level can be presented as the next set of connections:

$$SDLayer=\{SDNodes, SDStorages, SDNetwork\}, \qquad (1)$$

where $SDNodes$ – software-defined computing units; $SDStorages$ – software-defined storages; $SDNetwork$ – software-defined network.

Above the layer of software-defined infrastructure there is a level of the specific objects virtualization. The main of them are: computing nodes (VirtNodes), virtual data warehouses and also the elements of the software-defined network that are used in the work of the cloud platform and consolidated in VirtNetwork multitude.

$$VirtLayer=\{VirtNodes, VirtStorages, VirtNetwork\} \quad (2)$$

In the software-defined infrastructure computing nodes and data storages are more often presented as virtual machines that discharge the set of given functions.

To control such multi-layer infrastructure the separate orchestration layer is needed (the forth level). There are a number of functions in it and computing nodes and program systems enabled for their execution. The main functions are orchestration of the virtualization objects (virtual machines and data storages) (ONodes, OStorages) and orchestration of the software-defined network (ONetwork). Lately, experimental Network function virtualization is also added to them.

$$OrchLayer=\{ ONodes, OStorages, ONetwork\}. \quad (3)$$

The next level (Service level) represents the services used in the working process either the very cloud platform, or applications distributed there, for example, DBMS, Hadoop, Nginx and others.

$$ServiceLayer =\{Service_1, \ldots, Service_n \}. \quad (4)$$

All the multitude of ServiceLayer cloud services working in the virtual data center infrastructure can be divided into two disjoint subsets $ServVM \cup ServDocker = ServiceLayer$. Services using virtualization on the base of other machines are in the first set (ServVM). In the second set (ServDocker) there are services realized on the base of containers under Docker control.

On the top level there are cloud applications that are exploited by users for flexible scalability providing (AppLayer).

$$AppLayer =\{App_1, \ldots, App_m \}. \quad (5)$$

As on the previous level, cloud applications App_i can be placed in containers, forming the AppVM set or use containerization forming the AppDocker set. At the same time $AppVM \cup AppDocker = AppLayer$.

Thus, the set of object of software-defined infrastructure can be divided into two groups by the methods of placing. Virtual objects using container placing method can be referred to the first group. Let us describe them so:

$$Docker =\{ServDocker, AppDocker \} \quad (6)$$

In the second group there are services and applications using virtual machines as a placing platform:

$$VM=\{ServVM, AppVM\} \quad (7)$$

Let us observe the cloud applications and services virtualization model on containers base.

3. Model of virtualization cloud services and applications on the basis of containers

For describe the model of virtualization with support the containerization method let's describe formalized structure and communications of cloud applications and services.

Every cloud application can be described as a set of components, which are the following union of set:

$$App_i= Lib \cup Qu \cup StgData \cup Service \cup PM \quad (8)$$

where Lib – a set of operating system libraries used in the application; Qu – set of queues formed at the requests of the users accessing the application; StgData – set of storage systems that used for placement data for cloud application; Service – set of service that uses cloud application in the course of their work; PM – set of methods for placing of cloud applications in the and software-defined infrastructure of data center.

In our study we are consider three methods of placing cloud applications. First method is based on using of virtual machines – P_{vm}. The next method involves the use of containers placed on physical compute nodes – P_d. The last method uses hybrid approach based on containerization inside a virtual machine – P_{dvm}.

Service-oriented application (App_i) hosted in a software-defined infrastructure of data center is a set of instances running in the cloud platform $Vapp \in App_i$. Thus, each moment of time the cloud application may be represented as a dynamic weighted directed graph:

$$App_i(t) = (Vapp, Eapp, FlowApp(e_a, t), Iapp(vapp)) \quad (9)$$

where Vapp – set of nodes that represent instances of cloud applications placed in the software-defined infrastructure of data center; Eapp – the maximum total number of network connections forming the graph of the arc in the process of balancing the requests between instances Vapp; FlowApp(e_a,t) – function that determines the number of transmitted data on the arc $e_a \in$ Eapp at time $t \geq 0$. If FlowApp (e_a,t) = 0 , no arc e_a at time t; Iapp(vapp)$_i$ – set of the characteristics of an instance of the cloud applications vapp$\in$Vapp.

In turn, each cloud service in a software-defined infrastructure of data center can be described by the following set of parameters:

$$Service_i = \{AgrIP, NameServ, Ma, PM, Format\} \tag{10}$$

AgrIP – Address of compute node for requests aggregation to cloud service; *NameServ* – name of the cloud service placed in the infrastructure of virtual data center; *Ma* – set of supported methods of access to the service; *PM* – set of methods for placing of cloud service in the and software-defined infrastructure of data center; *Format* – data format.

Cloud service presented a set of instances running in the virtual data center infrastructure. Thus, it can be represented in the form of dynamic weighted directed graph:

$$Service_i(t) = (Vserv, Eserv, FlowServ(e_s, t), Iserv(vserv)_i) \tag{11}$$

where Vserv – set of vertices representing the running instances of the cloud service;

Eserv – maximum full set of arcs (network connections) allowed between multiple cloud applications vertices (application-level) and service instances vserv$\in$Vserv deployed in a virtual data center; FlowServ(e_s,t) – function that determines the number of transmitted data on the arc $e_s \in$ Eserv at time $t \geq 0$. If FlowServ (e_s,t) = 0 , no arc e_s at time t; Iserv(vserv) – set of the characteristics of an instance of the cloud service vserv$\in$Vserv.

Every instance of a running cloud application vapp$\in$Vapp or cloud service vserv$\in$Vserv is characterized by the corresponding vectors:

$$Iapp(vapp)_i = n_i(t), m_i(t), u_i(t), \Delta t_i, p_i; \tag{12}$$

$$\mathrm{Iserv}(\mathrm{vserv})_i = n_i(t), m_i(t), u_i(t), \Delta t_i, p_i \tag{13}$$

where $n_i(t)$ – the number of requests flows that are processed instance at the moment time t;

$m_i(t)$ – volume consumed memory for one instance of application for placing on the computing node at time t;

$u_i(t)$ – average load cores computing node for one instance of application at a time t;

Δt – average response time instance to the incoming flow of requests;

$p \in PM$ – method of placing an instance in the virtual data center infrastructure;

The developed model describes the mapping of the service-oriented cloud applications in the virtual data center infrastructure with using different placement methods.

Let us describe the flow of user requests coming to the service-oriented cloud applications by the following functional:

$$Fur = (U, AppLayer, Q) \tag{14}$$

where U - set of users; $AppLayer = \{App_1, \ldots, App_m\}$ - set of service-oriented cloud applications; Q – set of user requests.

To effectively use the resources of the virtual data center and ensure the required quality of service to users we formulate the optimization problem. Flow of user requests must be distributed efficiently between running instances of the service-oriented cloud applications.

$$\mathrm{Fur}(t): Q \to Vapp \tag{15}$$

At the same time copies of applications and services should be optimally placed in the virtual data center infrastructure.

$$\mathrm{App_i}(t): Vapp \to PM \tag{16}$$

It was found that the consumption of basic resources of the virtual data center using different placement method set of service-oriented cloud applications has a different weight. To take into account this feature, we introduce the model of optimize the weighting factors k_1, k_2, k_3 for each type of placement. Then the function of resource consumption will be:

$$Rvapp_i = \sum_{l=1}^{L} k_l Rapp_i \tag{17}$$

$$Rvserv_i = \sum_{l=1}^{L} k_l Rserv_i \tag{18}$$

Rapp$_i$ and Rserv$_i$ - base weight of resource-intensive applications and services, respectively.

Under optimal placement we mean the minimum number of instances running applications and services used by them. This ensures minimal consumption in the virtual data center resources. Also this approach it support response time within the allowable value for service maximum number of users per unit time. This can be formalized as follows:

$$\sum_{i=1}^{N} \sum_{j=m}^{M} vapp_i^j Rvapp_i \to \min$$

$$\sum_{i=1}^{N} \sum_{j=m}^{M} vserv_i^j Rvserv_i \to \min\ , \tag{19}$$

$$\sum_{i=1}^{D} \mathrm{Fur_i} \to \max$$

where $i = 1 \ldots D$ – number of applications received in the interval of time.

This will minimize the number of concurrent computing devices in the virtual data center infrastructure and maximize processing user requests at a given time interval ΔT.

To achieve these requirements is necessary to observe a number of functional limitations.

Response time at the user's request is limited and must not exceed the permissible value. The following restrictions apply to architecture of applications. For distribution users requests in infrastructure of virtual data center using the queue. The residence time of the request queue must be less than the maximum response time to a request to the application. Otherwise the request not will be serviced. Another limitation is the request time of cloud applications. As well as service response time to a request data for application from storage or services.

$$Tu_{resp} \le Tu_{resp}^{\max} \tag{20}$$

$$Tqu_{resp} + Tapp + Tserv < Tu_{resp}^{\max} \tag{21}$$

4. Algorithm optimization of launch and deployment of applications and services in the virtual data center infrastructure using different placement methods

The presented models allow us to choose the most suitable methods of placing instance of cloud applications and services in the virtual data center infrastructure based on the current load and the incoming flow of requests. The main task of the distribution of cloud applications and services is choice is the number of instances in time interval, that formulate as making a plan. When accessing to the service-oriented cloud applications prepare plan is especially important. Created by the load on the compute nodes may vary widely over relatively short time intervals, and depends on the method of placement of the virtual data center infrastructure. To solve the optimization problem developed an algorithm which monitors the virtual data center infrastructure, scheduling and launching applications and services. It is based on a biased random-key genetic algorithm (BRKGA). But in comparison with BRKGA algorithm uses heuristic analysis queries flows and their classification depending on the application hosted in the virtual data center.

Enlarged algorithm has the following steps.

Step 1. Evaluate incoming flow requests to cloud applications. Group requests by type of application. Ranked type of application by the number of requests.

Step 2. Count the number of running instances of each cloud application and determine the amount of used cloud services. Determine the load on the physical computing nodes. Ranking of cloud applications and services on the load generated by the virtual data center infrastructure.

Step 3. Based on the data obtained in step 1 and 2 to make a comparison of the data and determine the applications and services that require scaling.

Step 4. For applications and services are not involved in the processing to implement the stop and release resources held by the virtual data center. Add the minimum number of instances of using containers.

Step 5. For applications and services that require scalability and creating the maximum load on the infrastructure to evaluate the method of placement.

Step 6. Distribute the most loaded applications and services using a hybrid method of placing (containers deployed in a virtual machine).

Step 7. Less loaded, but requires scaling applications and services, to translate into operation in the virtual machine.

Step 8. Migrate virtual machines on the least-loaded nodes.

The approach used in the proposed algorithm control service-oriented applications, takes into account the way of accommodation and organizes the work of the virtual data center, taking into account incoming flow user requests while adjusting the number of running instances of applications and services.

5. Experimental part

The aim of the experimental research is definition the effectiveness of the use of placement algorithm service-oriented cloud applications in the virtual data center infrastructure.

For each application, and related services created templates container and placing virtual machine images for deployment in a software-controlled data center infrastructure.

On the basis of statistical data using queries created by the flows generator are simulated requests from users to applications. To evaluate application performance used streams of different intensity. In the first case flows create minimum load capacity (to evaluate response times and delays, which makes data center infrastructure) (exp 1). In the second case it was created workload applications hosted in the data center virtual infrastructure, traditional for each application it is possible to evaluate the application response time (exp 2). In the third case it has been applied developed algorithm for load balancing between instances of applications and services. It was established resource consumption by each of the running instances, allowing predicting the required resources for a third computational experiment.

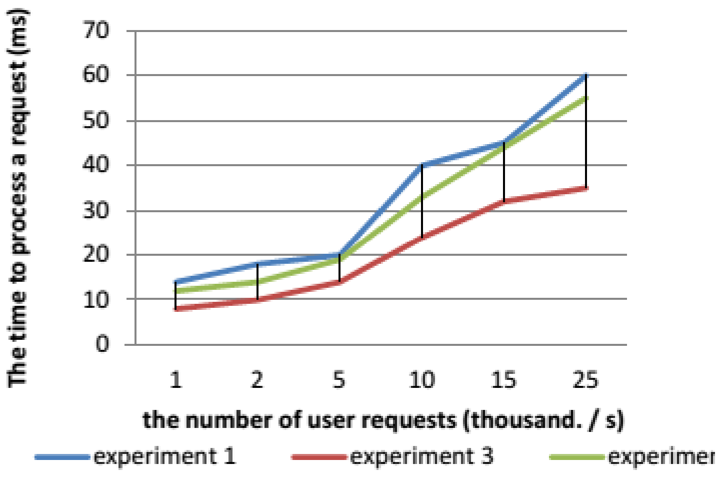

Fig. 1. Computational experiment result.

Research has shown that static placement of containers on the physical nodes is not effective because it does not allow redistributing the load quickly. In addition, movement of the container on another computing unit leads to a loss of the current connections. When placing applications based on virtual machines due to the flexibility of load balancing showed better results, but the load on the compute nodes has increased considerably due to the additional overhead associated with the use of virtual machines. The most effective placement of the study was the use of containers inside the virtual machines. It is possible to increase the density of application hosting and managed services and software within the data center, as well as allowed to place containers and data services and network applications in close proximity to each other thereby reducing the response time of applications on users' queries and thus increase the efficiency of the system.

6. Discussion

Traditional approaches to route traffic based on load-balancing and are reactive by its nature. They use simple classical algorithms for distributed computing tasks First Fit or Best Fit. Also popular are the following algorithms [5-8]: First Come First Served Scan, Most Processors First Served Scan, Shortest Job First Scan. Their main disadvantage - poor utilization of the computer system due to the presence of a large number of windows in the launch schedule of tasks and "hanging up" problems, when their service is postponed for an indefinite period due to admission to all higher priority tasks. As an alternative method of load distribution between nodes generally used solution Argonne National Laboratory, proposed by D. Lifka, based on the aggressive variant Backfill algorithm, which has two conflicting goals - more efficient use of computing resources by filling the empty windows schedule and prevent "hang" of problems due to redundancy mechanism. Feytelson D. and A. Weil offered a conservative variant Backfill algorithm; various modifications have been created in the future: B. Lawson and E. Smyrni [8], D. and P. Perkovic Keleherom. The main drawback of these algorithms, the time lag that occurs in the process of calculation, which is not acceptable for critical services at the time of failure.

In addition to the traditional reactive fault-tolerant technology, such as replication and redundancy to ensure reliability of networked storage cloud platforms group of scientists Nankai University Ying Lee, Lee Mingze Ganges Schang, Hiaoguang Liu Zhongschei Lee Huiyun Tang proposed an approach based on a Markov model, which provides secure storage of data without excessive redundancy. However, a significant drawback of the proposed model is the lack of classification and analysis of the types and sources of data to be placed in their consumption. Nevertheless, the model demonstrates a proactive approach that gives certain advantages to achieve the desired resiliency of cloud storage.

Reliability and availability of applications and services play an important role in the evaluation of its performance cloud platform. A major shortcoming of existing software reliability solutions in the data center infrastructure is the use of traditional data flow routing methods. As part of this work on the basis of technology and software-defined network is proposed to adjust the network to the current load of the applications and services that are hosted in cloud platform before they start using pre-computed and installation routes of transmission (if known oriented acyclic graph task dependencies and communication schemes). The principles of software-configurable network emerged for the first time in research laboratories at Stanford and Berkeley, and are currently being developed by a consortium of Open Network Foundation, GENI project, the European project OFELIA and the Russian "University Consortium for the development of technology and software-defined network", which is a member of the Orenburg State University.

Centralized decision on the organization of the data center, as proposed in Articles heterogeneous infrastructure has a number of deficiencies, including by ensuring the reliability, cost of obtaining a complete and current network conditions, low scalability. The best option is seen the development of fully decentralized solutions, but in this case there is a problem of interaction between the controllers of autonomous systems. As part of this research, it will be decided via SDX technology, which will be extended to exchange not only information about the network, but also distributed sections, able to cloud services and applications.

Published in scientific sources algorithms for routing data streams in software-defined network when selecting tracks does not take into account the need to ensure the QoS parameters for the previously installed and routed data streams, it is planned in the framework of the developed adaptive network communications routing methods.

Existing QoS algorithms to provide software-defined network is also not sufficiently effective. In [21] describes an approach to dynamic routing routes of transmission of multimedia streams that provide a guaranteed maximum delay via LARAC algorithm (Lagrangian Relaxation Based Aggregated). However, the author consider only the case of unit delays on each network connection and does not take into account the minimum guaranteed bandwidth. A similar approach is described in, the authors pose and solve the optimization problem for the transfer of multimedia traffic without losses on alternative routes, leaving the shortcuts for common data.

At Stanford, an algorithm for adaptive control of QoS Shortest Span First, which allows you to mathematically calculate the optimal priorities for each stream, to minimize crosstalk flows on delay dynamically manage priorities, depending on the current situation, as well as pave the flow of data transmission through a specific line ports.

In the framework developed in this research adaptive routing methods of network communications cloud services and applications is planned to formulate optimization problems for laying routes with constraints on QoS, load balancing. In their decision heuristics can be used, similar to the algorithm Shortest Span First. It will also take into account the distributed nature of cloud platform.

Analysis of the problem of scientific and research information sources showed that:

a) at the moment no effective algorithmic solutions for planning of virtual machines, cloud services, application-oriented accounting topological topology of the computer system, as well as communication tasks schemes;

b) the existing solutions for controlling distributed multi cloud platforms to carry out planning of computing tasks without subsequent adjustment network for their communication schemes - using traditional routing methods;

c) the existing methods of routing the data streams can be enhanced by taking into account the QoS requirements of heterogeneous and distributed nature of cloud platform.

This demonstrates the novelty of the solutions offered by the project.

Thus, the development of new methods and algorithms to improve the efficiency of cloud computing, implemented using heterogeneous cloud platform, it is an actual fundamental task.

7. Conclusion

We propose an efficient algorithm for placing applications and services in the infrastructure of a virtual data center. The problem of optimization of placing services oriented cloud applications based on the template VM and containers with disabilities infrastructure virtual data center is reduced to the problem of packing in containers We also generalize the well renown heuristic and deterministic algorithms of Karmakar-Karp. We have developed an efficient algorithm to placing VM by neural network optimization. Comparing with the exact algorithm developed algorithm, we find that its approximate solutions do not differ much from the exact solutions.

Thus, assessing the overall result of the algorithm performance gain can be obtained from 12 to 15% compared with conventional tools that are extremely effective at high intensities requests.

Acknowledgements

The research has been supported by the Russian Foundation of Basic Research (grants 16-37-60086 mol_a_dk, 16-07-01004 a), and the President of the Russian Federation within the grant for state support of young Russian scientists (MK-1624.2017.9).

References

[1] Bein, D. Cloud Storage and Online Bin Packing / D.Bein, W. Bein, S. Venigella // Proc. of the 5th Intern. Symp. on Intelligent Distributed Computing. – 2011. – P. 63-68.

[2] Nagendram, S. Efficient Resource Scheduling in Data Centers using MRIS / S. Nagendram, J.V. Lakshmi, D.V. Rao// Indian J. of Computer Science and Engineering. – 2011. – Vol. 2. Issue 5. – P. 764-769.

[3] Arzuaga, E. Quantifying load imbalance on virtualized enterprise servers / E. Arzuaga, D.R. Kaeli//Proc. of the first joint WOSP/SIPEW international conference on Performance engineering. – 2010. – P. 235-242.

[4] Mishra, M. On theory of VM placement: Anomalies in existing methodologies and their mitigation using a novel vector based approach / M. Mishra, A.Sahoo// IEEE International Conference Cloud Computing. – 2011. – P. 275-282.

[5] Bolodurina, I. Development and research of models of organization storages based on the software-defined infrastructure / I. Bolodurina, D. Parfenov // Proc. 39th International Conference on Telecommunications and Signal Processing. – 2016. – P. 1-6.

[6] Singh, A. Server-storage virtualization: integration and load balancing in Data Centers / Singh A., Korupolu M., Mohapatra D. // Proc. of the ACM/IEEE Conf. on Supercomputing. – 2012. – P. 1-12.

[7] Plakunov, A. Data center resource mapping algorithm based on the ant colony optimization / A. Plakunov, V. Kostenko // Proc. of Science and Technology Conference (Modern Networking Technologies) (MoNeTeC) . – 2014. – P. 1- 6.

[8] Darabseh, A. SDStorage: A Software Defined Storage Experimental Framework / A. Darabseh, M. Al-Ayyoub,Y. Jararweh, E. Benkhelifa, M. Vouk, A. Rindos //Proc. of Cloud Engineering (IC2E), Tempe: IEEE Press, 2015. p.341- 346.

[9] Bolodurina, I. Approaches to the effective use of limited computing resources in multimedia applications in the educational institutions / I. Bolodurina, D. Parfenov // WCSE 2015-IPCE. – 2015.