A model of cloud application assignments in software-defined storages

Irina P Bolodurina¹, Denis I Parfenov¹, Petr N Polezhaev¹, Alexander E Shukhmán¹

¹ Orenburg State University, 13, Pobedy ave., Orenburg, 460018, Russia

E-mail: parfenovdi@mail.ru

Abstract. The aim of this study is to analyze the structure and mechanisms of interaction of typical cloud applications and to suggest the approaches to optimize their placement in storage systems. In this paper, we describe a generalized model of cloud applications including the three basic layers: a model of application, a model of service, and a model of resource. The distinctive feature of the model suggested implies analyzing cloud resources from the user point of view and from the point of view of a software-defined infrastructure of the virtual data center (DC). The innovation character of this model is in describing at the same time the application data placements, as well as the state of the virtual environment, taking into account the network topology. The model of software-defined storage has been developed as a submodel within the resource model. This model allows implementing the algorithm for control of cloud application assignments in software-defined storages. Experimental researches returned this algorithm decreases in cloud application response time and performance growth in user request processes. The use of software-defined data storages allows the decrease in the number of physical store devices, which demonstrates the efficiency of our algorithm.

1. Introduction

In recent years, cloud computing has become a popular approach to provide an access to services and applications for operation of business processes [1]. There are three main models of cloud computing services: Infrastructure as a Service (IaaS), Platform as a Service (PaaS) and Software as a Service (SaaS). The use of these approaches in deploying cloud computing platforms has many advantages, such as reliability and quality of service [2]. At the same time, there are some limitations, caused both by consumers and providers of cloud services. For consumers, the cloud resources are endless in terms of scalability. However, if we consider economic aspects of their consumption, then their ability to scale is significantly narrower. From the side of cloud service providers, a set of services and computing powers is limited. In order to maximize the economic effect of cloud services by increasing the number of users, providers have to apply policies for flexible usage of allocated resources, while minimizing operation costs. So in today’s virtual and physical data centers (DC) the problem of resources and cloud applications management is an important issue, because it has a direct impact on the operation costs [3-5].

In the past few years, large IT corporations (such as Amazon, Google, Salesforce, IBM, Microsoft, Oracle) have developed a renewed approach to management of the resources and objects in DC using cloud applications. The main trend in this sphere is the optimization of the DC resource consumption. In our recent researches, we have developed the approaches to the storage optimization of cloud
application data and to improvement in efficiency of the access to cloud resources [6, 7]. However, they do not solve the assignment problem of cloud application instances in cloud environment. In addition, the review of researches in this field has shown that the problem of optimizing resource selection for specific types of cloud applications is insufficiently investigated [8-10].

We suggest the approach to solve the above-mentioned problems. It is based on the management of cloud application assignment to appropriate resources for its work.

2. Structural model of cloud datacenters

For understanding the operation principles of a cloud application, we need to define its place in the infrastructure of the virtual DC. DC is a dynamic object, changing in time $t$, its state can be formalized as a directed graph of the following form:

$$\text{VirtDC} = \{ \text{Node}(t), \text{Connect}(t), \text{CloudAppl}(t) \},$$

where vertices $\text{Node}(t) = \{ \text{Node}_1(t), \text{Node}_2(t), \ldots, \text{Node}_n(t) \}$ are active elements of the virtual DC cloud infrastructure (computing nodes, storage systems and others);
directed edges $\text{Connect}(t) = \{ \text{Connect}_1(t), \text{Connect}_2(t), \ldots, \text{Connect}_v(t) \}$ are active user connections to the virtualized applications;
$\text{CloudAppl}(t) = \{ \text{CloudAppl}_1(t), \text{CloudAppl}_2(t), \ldots, \text{CloudAppl}_v(t) \}$ are active instances of the cloud applications, launched on the basis of the virtual resources.

The major feature of cloud applications is the approach, in which users have access to them and to their services, and they do not know anything about their actual location. Most often, users know only the address of the aggregation node and the application name. The cloud system automatically selects the optimal virtual machine for the request, on which it is to be processed.

Before we talk about the resource allocation for the cloud applications, it is necessary to determine their structure, the basic parameters and the key characteristics of their operation, affecting the efficiency of their use. For this purpose, we have developed a generalized model of cloud application.

The generalized model of cloud application is a multilayer structure formalized in a form of graphs, describing the connections of individual elements. The model can be represented in the form of three basic layers, detailing the connections of the specific objects of virtual cloud infrastructure: applications, related services and allocated resources.

The cloud application is represented as a weighted directed acyclic graph of data dependencies:

$$\text{CloudAppl} = (G, V).$$

Its vertices $G$ are tasks that get information from the sources and process it in accordance with the user requests; its directed edges $V$ between corresponding vertices are the task dependencies on the data sources.

Each vertex $g \in G$ is characterized by the following tuple:

$$g = (\text{Res}, \text{NAppl}, \text{Utime}, \text{SchemeTask}),$$

where $\text{Res}$ are the resource requirements; $\text{NAppl}$ is the number of application instances; $\text{Utime}$ is the estimation of the users request execution time; $\text{SchemeTask}$ is a communication scheme of data transmission between sources and computing nodes.

Each directed edge $v \in V$ connects the application with the required data source. It is characterized by the following tuple:

$$v = (u, v, Tdata, Mdata, Fdata, Vdata, Qdata),$$

where $u$ and $v$ are linked vertices; $\text{Tdata}$ is the type of transmitted data; $\text{Mdata}$ is the access method to the information source (REST, JSON and others); $\text{Fdata}$ is the physical type of the accessed object (file in the storage system, local file, distributed database, data services and so on); $\text{Vdata}$ is the traffic volume estimated by the accessed data (in Mb), $\text{Qdata}$ is the requirements for the QoS (quality of services).

The originality of the model is in the fact that for each application the consolidated assessment of its work with data sources is calculated. It allows predicting the performance of the whole cloud
As mentioned earlier, a cloud service is in one of the key slices in the generalized model of cloud application. The cloud service serves as an autonomous data source for the application, for which it acts as a consolidated data handler. Generally, the cloud service is highly specialized and designed to perform a limited set of functions. The advantage of connecting cloud application to the service is in an isolated data processing, in contrast to the direct access to the raw data, when cloud application does not use a service. The usage of services reduces the execution times of user requests. The cloud service is formalized as a directed graph of data dependencies. The difference lies in the fact that from the user point of view, the cloud service is a closed system.

Cloud service can be formalized as a tuple:

\[ CloudServ = (\text{ArgIP}, \text{NameServ}, \text{Format}), \]

where AgrIP is the address of aggregation computing node; NameServ is the service name; Format is the required format of output data.

The aggregator of a service selects optimal virtual machines, on the basis of which it is executed. In addition, all its applications are distributed between predefined virtual machines or physical servers. Their new instances are scaled dynamically, depending on the number of incoming requests from cloud applications, users or other services.

To describe the assignment of cloud applications and services in the DC infrastructure, we have also implemented the model of cloud resource. Cloud resource represents an object of DC, which describes the behavior and the characteristics of the individual infrastructure elements, depending on its current state and parameters. The objects of DC are disk arrays, including detached storage devices, virtual machines, software-defined storages, databases of various kinds (SQL/NoSQL) and others. In addition, each cloud service or application imposes requirements on the number of computing cores, the RAM and disk sizes, the presence of special libraries on physical or virtual nodes, used to launch their executing environments.

Each cloud resource can be formalized as follows:

\[ CloudRes = (T_{Res}, \text{Param}, \text{State}, \text{Core}, \text{Rmem}, \text{Hmem}, \text{Lib}) , \]

where TRes is the type of resource; Param is the set of parameters; State is the state of resource; Core is the number of computing cores; Rmem is the size of RAM; Hmem is the size of disk; Lib is the libraries requirements.

The distinctive feature of the model suggested implies analyzing cloud resources from the user point of view and from the point of view of a software-defined infrastructure of the virtual data center (DC). The innovation character of this model is in describing at the same time the application data placements, as well as the state of the virtual environment, taking into account the network topology.

We developed the model of the software-defined storage, which details the resource model of the virtual DC. It is represented in the form of a directed multigraph, its vertices are the virtual DC elements, which are responsible for applications’ data placement (e.g. virtual disk arrays, DBMS and so on):

\[ Stg_{ki} = \left( \text{MaxV}_{ki}, P^{n}_{ki}, V_{ol_{ki}}(t), \overline{R}_{ki}(t), \overline{W}_{ki}(t), s^{ogr}_{ki}(t) \right), \]

where \( \text{MaxV}_{ki} \in N \) is the maximum storage capacity in Mb; \( P^{n}_{ki} = \{p^{n}_{ki,j}\} \) is the set of network ports; \( V_{ol_{ki}}(t) \in N \cup \{0\} \) is the available storage capacity in Mb; \( \overline{R}_{ki}(t) \) and \( \overline{W}_{ki}(t) \) are read and write speeds; \( s^{ogr}_{ki}(t) \in \{"online", "offline"\} \) is its state.

The data storage system for applications reminds a cake with some layers and uses the principles of self-organization of resources. The basis of self-organization of data storages is an adaptive model of dynamic reconfiguration when resources are changing. The model allows optimizing the organizational structure of the cloud platform based on algorithms for search of optimal control nodes, as well as for allocation of control groups. Our control model assumes two control levels for nodes and resources.
When a software-defined storage is created on each virtual computing node, the software module for exchanging state data between devices is executed. This exchange is carried out within a group of nodes with a single storage method. The least loaded node in the group is selected as the control node. This approach reduces the risk of degradation of the control node.

If the control node is failed, the remaining group of virtual machines has all the information about each other which allows choosing a new control node automatically. Each control node also carries out cooperation with control nodes from other groups to maintain up-to-date information on the state of the entire system.

Thus, the system of software-defined storages is constructed as a hierarchy that includes three basic levels: the level of local access, the level of the controlled group, and the level of data exchange within the whole system. In our model, the description of cloud applications consists of task descriptions and data source descriptions specifying directions and methods of data transfer as well as required resources.

The model has been used for development of our data assignment algorithm for software-defined storages, and our scheduling algorithm for cloud services and applications within the cloud platform.

3. Algorithm implementation

The data assignment algorithm for cloud applications provides heuristic analysis of application requests and traffic classification on data types at the performance time. The flexibility of the algorithm is due to virtualization of data storage. This makes it possible to dynamically change the physical location of the application within the cloud system for providing uninterrupted access to services.

The suggested solution is transparent to the client and scales cloud applications into multiple virtual storage devices. This provides a reduction in the application response time, and also improves the fault tolerance of the whole system.

Creating software-defined self-organizing data storages based on virtual machines and containers reduces the risks associated with data loss or inaccessibility as well as provides intelligent analysis of the use of cloud-based applications, which allows making assignment of virtual machines.

The data assignment algorithm for cloud applications is based on a cloud resource model describing its structure and links between virtual storage devices, machines and cloud applications. The model uses a muti-agent approach in data storage. Agents get information about the system state. This information is analyzed and dynamic assignment maps of virtual machines and devices are created on the basis of read/write events or time intervals. Analyzing the maps, the cloud control system makes decisions on reconfiguring or migrating virtual storage devices as well as data redistributing between physical nodes. Also our data assignment algorithm is used to increase cloud system performance because it provides compact placement of devices [5].

For a user request service, several resources can be used with different access parameters. In this case, the cloud control system has to optimize read time. Our data assignment algorithm forms internal assignment rules and changes them according to resource demands. Such approach allows balancing dynamically the resource load.

Our data assignment algorithm includes three optimization stages for each request. At the first stage, the algorithm analyzes the application type and the data type. At the second stage, the control system determines the ordered list of most valuable resources for the request. At the third stage, the algorithm analyzes the current system state and predicts time of the request execution.

4. Experimental results

We studied the work of the OpenStack based cloud system for efficiency estimation of the suggested algorithm for control of cloud application assignment in software-defined storages of virtual DC.

In experiments, standard OpenStack algorithms for launch control of cloud application and for control of its data placement were used as a reference. To compare the performance of algorithms by the example of different storage systems, we have created three experimental sites, which differ by the
used storage devices: conventional hard disk drives (HDD), solid-state drives (SSD) and virtual software-defined storages (SDS).

The prototype of the cloud system was deployed for experimental research on all experimental sites. It includes basic components and software modules for the developed algorithms, which modify the execution scheme of cloud application for data access in software-defined storages. The module implementing the algorithm for cloud application assignments in software-defined storages of virtual DC has been developed for OpenStack. It is intended to rationally use cloud system computing resources and to efficiently assign virtual machines and related data to physical nodes.

The request stream similar to the real requests to the cloud infrastructure was created for the experiment. This stream is based on logged records of access to the certain types of resources, which were classified by data types and request structures.

The time period of the reproduced requests was three years. The averaged data have been used for load experiments. The maximum number of concurrent income requests was 100000, which corresponds to the maximum count of the cloud system potential users.

All created streams were sequentially reproduced on three experimental sites. The aim was to compare the results from physical storage devices, which are not capable to reconfigure, and software-defined storages. All the three groups of experiments for efficiency estimation were carried out on all the experimental sites. Those groups are: intensive data read operations (see Figure 1a for experiment 1), intensive data write operations (Figure 1b for experiment 2), and concurrent read and write operations (Figure 1c for experiment 3).

The duration time of each experiment was one hour. It corresponds to the longest time period of peak system load found in a real traffic. The results of experimental research show that the suggested algorithm for control of cloud application assignments is more efficient than standard OpenStack solution regardless of the used type of physical storage devices. In addition, they demonstrate the possibility to apply the developed algorithm to provide efficient access to applications and services of
cloud systems. The experiments show the decrease by 20-25% of failures in data placements in software-defined storages. In addition, the algorithm for the control of cloud application assignments in software-defined storages may release 20-30% of computing node resources. Thus, this algorithm can be used to build architectures of cloud computing systems with the heterogeneous configuration of physical nodes and virtual machines.

5. Conclusion
The result of research was the development of a model for cloud applications based on software-defined infrastructures. Also, we suggest approaches to optimize cloud application and service assignments within a storage system built on the virtual DC software-defined data storage. This model allows implementing the algorithm for control of cloud application assignments in software-defined storages. Experimental researches proved that this algorithm decreases in cloud application response time and performance growth in user request processes. The use of software-defined data storages allows the decrease in the number of physical store devices. This demonstrates the efficiency of our solution.

6. Acknowledgments
The research has been supported by the Ministry of Education of Orenburg region (project no. 37 on 30 June 2016), the Russian Foundation of Basic Research (grants 16-37-60086 mol_a_dk, 16-07-01004 a, 16-47-560335 r_a), and the President of the Russian Federation within the grant for young scientists and PhD students (SP-2179.2015.5)

References
[1] Kolb S 2016 Unified cloud application management Proc. of t World Congress on Services (San Francisco: IEEE Press) pp 1-8
[2] Guisheng F, Huiqun Y, Liqiong C, Dongmei L 2016 Modeling and analyzing cost and utilization based task scheduling for cloud application Proc. of the 40th Annual Computer Software and Applications Conference (Atlanta: IEEE Press) pp 245-250
[3] Raschinger M, Kipouridis O, Gunthner W A service-oriented cloud application for a collaborative tool management system 2016 Proc. of Cloud Engineering (IC2E) (Jeju: IEEE Press) pp 1-5
[4] Chamorro L, López-Pires F, Barán B A genetic algorithm for dynamic cloud application broker age 2016 Proc. of Cloud Engineering (IC2E) (Berlin: IEEE Press) pp 131-134
[5] Singh A, Korupolu M, Mohapatra D 2008 Server-storage virtualization: integration and load balancing in data centers Proc. of the 2008 ACM/IEEE Conf. on Supercomputing (Austin: IEEE Press) pp 1-12
[6] Bolodurina I, Parfenov D Development and research of models of organization storages based on the software-defined infrastructure 2016 Proc. 39th International Conference on Telecommunications and Signal Processing (Viena: IEEE Press) pp 1-6
[7] Parfenov D, Bolodurina I, Shukhman A Approach to the effective controlling cloud computing resources in data centers for providing multimedia services 2015 Proc. International Siberian Conference on Control and Communications (Omsk: IEEE Press)
[8] Wuhib F, Stadler R, Lindgren H 2012 Dynamic resource allocation with management objectives - implementation for an OpenStack cloud Proc. 8th international conference on Network and service management (Las Vegas: IEEE Press) pp 309-315
[9] Bocchi E, Drago I, Mellia M 2015 Personal cloud storage benchmarks and comparison IEEE Trans. on Cloud Computing 99
[10] Darabshe A, Al-Ayyoub M, Jararweh Y, Benkelifa E, Vouk M, Rindos A 2015 A SDStorage: a software defined storage experimental framework Proc. of Cloud Engineering (IC2E) (Tempe: IEEE Press) pp 341-346