Scheduling and allocation of tasks and virtual machines in cloud computing

Raissa K. Uskenbayeva[1], Abu A. Kuandykov[1], Zhyldyz B. Kalpeyeva[2], Sabina B. Rakhmetulayeva[3], Nurzhan K. Mukazhanov[1]

[1]The International Information Technologies University, Manas Str./Zhandosov Str., 34 «A»/8 «A», Almaty, 050040, Kazakhstan
[2]Kazakh National Technical University after K.I.Satpayev, Satpaev Str., 22a, Almaty, 050013, Kazakhstan
[3]Kazakh Economic University after T. Ryskulov, Zhandosov Str., 55, Almaty, 050035, Kazakhstan

Abstract. This work enunciates the task of input stream optimisation of user task and virtual machines, installed on physical servers in cloud data-centre. The characteristic feature of the task setting consists of the fact that the deceleration of physical servers computing is taken into account so far as their resources are filled with virtual machines. Basing on heuristics, offered in this work, in future one can investigate and build effective algorithms for facilities assignment in computer systems, built according to "cloud computing" technology.

[Uskenbayeva R.K., Kuandykov A.A., Kalpeyeva Z. B., Rakhmetulayeva S.B., Mukazhanov N.K. **Scheduling and allocation of tasks and virtual machines in cloud computing.** *Life Sci J* 2014;11(8):532-538] (ISSN:1097-8135). http://www.lifesciencesite.com. 73

Keywords: cloud computing, virtual machine allocation, task allocation, scheduling

Introduction

Recently cloud computing systems, which provide computing resources on demand on a rental basis, become more and more popular. Cloud computing provide infrastructure, platform and software as a service which is provided for consumers on a rental basis [1-3]. A user can transparently and flexibly change volumes of consumable resources in the case of changing of his/her demands, increase or decrease server capacity along with corresponding change of terms of payment [4].

Generally the architecture of cloud environments consists of 4 layers [2]: hardware layer, infrastructure layer, platform layer and applications layer. In this work we have focused on the "Infrastructure as a Service" layer (IaaS) as the layer which provide a user with possibilities of creation of a needed infrastructure to the fullest extent possible. Often the basis for hardware component building of cloud infrastructures is the technology of server virtualization which allows increasing the effectiveness of use and consolidation of computing resources on account of deployment of several virtual machines (VM) on one physical node (server).

For all distributed systems there is a general scientific-technical problem which consists in finding out ways for effective allocation of heterogeneous resources among diverse tasks [5]. While the problem of resource management contains a lot of aspects including those of necessity of hardware solutions, scheduling, tasks allocation and load balancing, this work refers mainly to general theoretical aspects connected with the logic scheme of management organisation in cloud computing and possibility of mathematical methods application for development of allocation algorithms of task flows and virtual machines among physical servers.

The problem statement

One of the important questions for IaaS is an effective scheduling of virtual resources and cloudlets on virtual machines. Effective allocation of virtual machines inside the data centre is very important in cloud computing environment for increasing of effective use of data centre computing resources. High price of data centre infrastructure, significant expenditures for maintenance and energy supply cause the necessity to develop methods and algorithms of effective use of computing resources. Today significant financial losses of data centres are conditioned by the irrational allocation of virtual machines among physical servers (PS). At the same time, maximum heavy coverage of servers with virtual machines leads to the shortage of expenditures for energy supply and maintenance at cost of shut-down of idle servers or allows maintaining more customers by the same server farm [6].

Structure and architecture of computing environment

Computing environment built on the basis of distributed cloud computing technology represents a heterogeneous set of computing resources in the form of both same- and diverse-type of physical servers and virtual machines. Let us given computing system (CS) consisting of physical servers (PS). Organisation of computing process consists in scheduling and allocation of users' tasks among executing servers. For this case every PS should receive a task, load necessary software packages, link informational resources, or in other words to load data store and necessary preliminary information from data storage on PS. The example of this class of CS is a data-centre of cloud services' provider.

Let us describe the structure and architecture of the given CS through the expression:

$$DC = (N, R),$$

where $N = (N_1, N_2, \ldots N)$ is physical servers which can be both same-type and diverse-type, i.e. heterogeneous.

Every physical server N_i is set by the model of the attributes/parameters list and their values/characteristics.

Relationship R sets the system's structure, i.e. relations between physical servers: star, broadcast, ring or more complicated mixed type connections [7].

Cloudlet structure

Let us suppose that the request from a user of CS consists in allocating virtual machine VM_i for fulfilment its tasks Zd_h. The task is described by the list of features and characteristics χ_i which are presented in the task data sheet. Task data sheet may be described using the Job Submission Description Language (JSDL) [8], which is a language for description of computing tasks demands to the resources, especially in Grid and Cloud environment, though it is not limited by these ones.

To perform these tasks it is necessary to have a certain set of programs and data. Program modules can be contained in one package Pt_j (single-module package) or in different modules (multi-module package), they are kept in different places respectively. Similarly, data and information resources or information modules can be kept in one segment or in different segments of data storage system (DSS), which can be built basing on different technologies of SAN, DAS, NAT type.

Mathematical model of virtual machine allocation in cloud computing environments

Let us suppose that the organisation or scheduling of computing processes in CS is performed according to the two-/three-fold allocation scheme in the following way:

1) selection of the physical servers set N_i out of the given structure of a computing system DS;

2) virtual machines allocation VM_i among physical servers N_i;

3) tasks allocation Zd_h among virtual machines VM_i;

4) allocation or assignment of resource packages Pt_j necessary for the task processing Zd_h.

Variants of tasks allocation Zd_h among the set VM_i;

- one VM can perform several tasks;
- one VM performs only one task.

Now basing on the above shown concept of computing processes organisation in cloud environment, i.e. VM and tasks allocation among PS, we can define a task of load allocation and balancing. Let us suppose that CS does not contain previously installed application programs which will be necessary for processing of user's tasks.

Lifecycle of CS work consists of service cycles of request/tasks packages or implementing of tasks:

$$LC = \left\langle Cs_1, Cs_2, Cs_3, \ldots, Cs_i, \ldots, Cs_z, \ldots \right\rangle$$

Every working cycle starts from the time $t = t^n$ of delivery of a new portion of requests $K(Zp) = \{Zp_i\},\ i = 1, h$ from input stream of requests

$$P(Zp) = \left\langle Zp_1, Zp_2, Zp_3, \ldots, Zp_i, \ldots, Zp_z, \ldots \right\rangle$$

up to the time $t = t^k$, when the performing of the last stream of the chosen request portion is finished.

Let us suppose that in the moment of time $t = t^n$ CS includes a variety of physical servers PS:

$$N = \{N_i\},\ i = 1, n$$

We have a variety of virtual machines (VM) which are to be allocated among PS:

$$VM = \{VM_i\},\ i = 1, m$$

Let us suppose that in the moment of time $t = t^n$ there is fulfilled the $n-1$ cycle of requests/tasks services;

in the moment of time $t = t^n$ there is started the performing of service cycle n;

in the moment of time $t = t^{n+1}$ there were chosen requests/tasks from the input pool of requests:

$$Zd = \{Zd_i\},\ i = 1, p$$

Tasks $Zd = \{Zd_i\}$, can be from one user or from several users. Let us consider that all the requests/tasks are from one user. This allows us not to take into account demands and interests of different groups of actors.

Every type of participants or classes of the computing process cycle of requests servicing is characterised by certain properties, attributes, characteristics which can be formally represented in the form of markers:

Physical server N_i is characterised by the following set of markers:

$$\chi(N_i) = (\chi_1(N_i), \chi_2(N_i), ..., \chi_k(N_i), ..., \chi_h(N_i), ..., \chi_m(N_i))$$

where $\chi_1(N_i)$ is a marker of production or operating speed of a N_i server's processor;

$\chi_2(N_i)$ is a marker of capacity/volume of a random access memory of a server N_i, additional definition of this marker will be as $Mr(N_i)$;

Summary memory capacity of CS servers:

$$Mr(N) = \sum_{i=1}^{n} Mr(N_i)$$

$\chi_3(N_i)$ is a marker of cache memory capacity of server N_i;

$\chi_4(N_i)$ is a marker of storage capacity of server N_i;

$\chi_5(N_i)$ is a marker of accessibility of server N_i;

$\chi_6(N_i)$ is a marker of reliability of server N_i;

$\chi_7(N_i)$ is a marker of security of server N_i;

Virtual machine VM_i is characterised by the following markers:

$$\chi(VM_i) = (\chi_1(VM_i), \chi_2(VM_i), ..., \chi_k(VM_i), ..., \chi_h(VM_i), ..., \chi_m(VM_i))$$

where $\chi_1(VM_i)$ is the number of cores of CS;

$\chi_2(VM_i)$ is a marker of capacity/volume of random access memory of CS VM_i;

$\chi_3(VM_i)$ is a marker of storage capacity of CS VM_i;

$\chi_4(VM_i)$ is a marker of previous installation of operating system of server VM_i;

$\chi_5(VM_i)$ is a marker of reliability of server VM_i;

$\chi_6(VM_i)$ is a marker of security of CS VM_i;

$\chi_7(VM_i)$ is a marker of cost per time unit of CS's work VM_i;

Task Zd_i is characterised by the following markers:

$$Zd_i = (\chi_1(Zd_i), \chi_2(Zd_i), ..., \chi_k(Zd_i), ..., \chi_h(Zd_i), ..., \chi_m(Zd_i))$$

where $\chi_1(Zd_i)$ is a marker of time of receiving of task Zd_i for servicing;

$\chi_2(Zd_i)$ is a marker of the urgency of fulfilment Zd_i, additional definition of an urgency marker of task fulfilment shall be set as: $Sh(Zd_i)$;

$\chi_3(Zd_i)$ is a marker of allowable duration of fulfilment Zd_i;

$\chi_4(Zd_i)$ is a marker of allowable cost of fulfilment Zd_i;

$\chi_5(Zd_i)$ is a marker of necessary continuity level Zd_i;

$\chi_6(Zd_i)$ is a marker of necessary security level Zd_i;

$\chi_7(Zd_i)$ is a marker of necessary memory capacity for fulfilment Zd_i, additional definition of a marker of necessary memory capacity for the task will be as : $Mr(Zd_i)$;

Summary memory capacity for fulfilment of all tasks:

$$Mr(Zd) = \sum_{i=1}^{m} Mr(Zd_i)$$

Value of a characteristic or a marker of fulfilment urgency Zd_i can be as follows:

$$\chi_1(Zd_i) = (\chi_{11}(Zd_i), \chi_{12}(Zd_i), ..., \chi_{1j}(Zd_i), ..., \chi_{1g}(Zd_i), ..., \chi_{1d}(Zd_i))$$

where $\chi_{11}(Zd_i) = 1$ is a marker of fulfilment urgency Zd_i - non-urgent,

$\chi_{12}(Zd_i)$ = 2 is a marker of fulfilment urgency Zd_i - not very urgent,

$\chi_{13}(Zd_i)$ = 3 is a marker of fulfilment urgency Zd_i - a bit urgent,

$\chi_{14}(Zd_i)$ = 4 is a marker of fulfilment urgency Zd_i - urgent,

$\chi_{15}(Zd_i)$ = 5 is a marker of fulfilment urgency Zd_i - very urgent,

It is supposed that among all the PS there is always one or more PS which complies with the requirements of any free-hand task.

Rules of tasks allocation

Previously implement the following assumptions.

Assumption 1. Assessment of tasks homogeneity

Let us suppose that the characteristics of tasks are homogeneous and are included in one portion because of homogeneity of their markers values.

Two tasks (Zd_h, Zd_k) are homogeneous if for their markers the following types of restrictions are performed:

For the set marker j of a pair of tasks (Zd_h, Zd_k):

$$\exists j[\rho(\chi_{hi}(Zd_h)),(\chi_{ki}(Zd_k)) < \Delta\rho_i]$$

For all markers $\forall j$ of tasks (Zd_h, Zd_k):

$$\forall j[\rho(\chi_{hi}(Zd_h)),(\chi_{ki}(Zd_k)) < \Delta\rho_i]$$

For different pairs $\forall h,k$ of tasks according to the set marker j:

$$\forall h,k[\rho(\chi_{hi}(Zd_h)),(\chi_{ki}(Zd_k)) < \Delta\rho_i]$$

For different pairs $\forall h,k$ of tasks and for all markers $\forall j$:

$$\forall h,k,j[\rho(\chi_{hi}(Zd_h)),(\chi_{ki}(Zd_k)) < \Delta\rho_i]$$

Assumption 2. Service strategies

If tasks (Zd_h, Zd_k) are homogeneous for their important (target) markers, for their servicing a concrete service strategy can be applied, for example, strategy $\# j - Ct_j$. The following can serve as target markers: cost of service, time of service, memory capacity taken by tasks [9-10] etc. The service with the following criteria for all tasks is taken as a service strategy, for example:

Strategy #1: minimization of the summary cost of fulfilment of all tasks. Strategy #2: maximization of the summary speedwork of fulfilment of all tasks.

In case of non-homogeneity of tasks $Zd = \{Zd_i\},\ i = 1, p$ it is necessary to make their content homogeneous. One of the methods of making tasks content homogeneous is a grouping of tasks $Zd = \{Zd_i\},\ i = 1, p$ on the basis of adjacency of markers value. We suppose that tasks $Zd = \{Zd_i\},$, $i = 1, p$ are divided into several task groups which have similar values of target markers: cost of task fulfilment, speedwork of task fulfilment etc.

Grouping of task is serviced by a separate cycle of tasks servicing. For servicing of every task we define the order/rule of optimal servicing and criteria of task servicing, which are set by an expert on the basis of the experience, that's why these rules are heuristical.

Heuristics 1

If the fulfilment of group of tasks $\{Zd_i\}$ is not urgent, i.e. $\forall Sh(Zd_i) = 1$, the following can be applied as the criteria for tasks allocation: "minimization of the summary cost of fulfilment of all tasks".

Basing on this requirement let us define the task in the following way:

$$\sum_{i=1}^{n} t_{ij} * s_i * y_i \to \min$$

$$\sum_{i=1}^{m} Mr(Zd_i) < Mr(N_i),$$

$$\sum t_i < \Delta T,$$

Where

s_i is a price of usage, i.e. price for functioning and operation of a server N_i per unit of time;

y_i is a marker of the fact, whether the server N_i is chosen or not. If the server N_i is chosen, $y_i = 1$ otherwise $y_i = 0$;

t_{ij} is a forecasting duration of fulfilment of task Zd_i on server N_i;

$\sum_{i=1}^{m} Mr(Zd_i)$ is a summary capacity of random access memory necessary for fulfilment of tasks allocated on the server N_i, $i = 1, n$;

$Mr(N_i)$ random access memory capacity of server N_i.

ΔT is the upper limit n of service cycle of tasks or maximum time of fulfilment of tasks by servers.

In other words, "Summary memory capacity for fulfilment of all tasks: $\sum_{i=1}^{m} Mr(Zd_i)$ " < "Summary memory capacity of physical servers of CS: $Mr(N') = \sum_{i=1}^{n} Mr(N_i)$ ". So, the following requirement to the emphasising of servers shall be fulfilled for fulfilment of tasks out of the variety of the existed ones:

$$Mr(Zd) = \sum_{i=1}^{m} Mr(Zd_i) < Mr(N') = \sum_{i=1}^{n} Mr(N_i),$$

or $Mr(Zd) < Mr(N')$

Heuristics 2.

Summary traffic of cost of functioning and operation of all emphasised servers N' shall be minimum:

$$\sum_{i=1}^{L} s_i * y_i \to \min$$

Solving of task

The solving of this task can be implemented directly. However this approach to the solving of task is not reasonable one when a small amount of tasks $Zd = \{Zd_i\}$, $i = 1, p$ are to be allocated among big number of servers, $N = \{N_i\}$, $i = 1, n$ i.e. $|Zd| << |N|$. In this variant of task the process of its solving can be simplified by means of stage by stage way of solving. So, the solving of this task or fulfilment of its requirements in the case of $|Zd| << |N|$ is reached on two stages in the following way:

Stage 1

First of all one should select such number of servers $N' = \{N_i\}$, $i = 1, L$ out of the existing servers $N = \{N_i\}$, $i = 1, n$ where $L \leq n$ in such a way that for the selected servers N' conditions $Mr(Zd) = \sum Mr(Zd_i)$, $i = 1, L$ were performed.

In order to fix the selected servers one should use vector $y = \langle y_i \rangle$, $i = 1, L$ where $y_i = 1$, if on the server N_i there is at least one VM and $y_i = 0$, otherwise.

Cost of unit of work of server N_i shall be defined as s_i, which is a constant value for all the period of work of server, although in general case it can be a variable one and can depend on the value of available space in server's random access memory.

Duration of fulfilment/solving of task shall be defined as $\tau(Zd_i)$, which depends on the value of occupied space (level of fullness) of the random access memory of the server. Duration of fulfilment of task Zd_i - $\tau(Zd_i)$ can approximately be calculated by means of the following correspondence/ratio:

$$\tau(Zd_i) = \begin{vmatrix} \tau_{i1}, \text{if } Mr(N_i) \text{ server's memory is filled from 0\% to 40\%;} \\ \tau_{i2}, \text{if } Mr(N_i) \text{ server's memory is filled from 41\% to 60\%;} \\ \tau_{i3}, \text{if } Mr(N_i) \text{ server's memory is filled from 61\% to 70\%;} \\ \tau_{i4}, \text{if } Mr(N_i) \text{ server's memory is filled from 71\% to 100\%;} \end{vmatrix}$$

New cycle of task servicing starts from the selection of servers N' which will be used in servicing tasks of the current cycle.

1) Select cheap servers for operation in such a way that:

first of all there is enough resources for placement all the VM, i.e. the condition shall be met:

$$Mr(Zd) < Mr(N')$$

where, $Mr(Zd) = \sum_{i=1}^{m} Mr(Zd_i)$, $Mr(N') = \sum_{i=1}^{n} Mr(N_i)$

secondly, summary traffic of cost of functioning and operation of all emphasised servers N' shall be minimum:

$$\sum_{i=1}^{L} s_i * y_i \to \min \qquad (1)$$

where s_i is traffic of usage, i.e. price for functioning and operation of a server N_i per unit of time;

y_i is a marker of the fact, whether the server N_i is chosen or not. If the server N_i is chosen, $y_i = 1$ otherwise $y_i = 0$;

Thus, a domain function N' out of variety N of servers will be chosen and named work domain function. Production servers will be used in further n cycle of service of tasks/requests by means of load and installation of virtual machines and tasks on them.

Stage 2

Tighten the selected servers with tasks in such a way that in new n cycle of task servicing in the environment of the emphasised servers $N' = \{N_i\}, \quad i = 1, L$ it was necessary to place/pack the variety of tasks $Zd = \{Zd_i\}, \; i = 1, p$ in such a way that the summary expenditures for the time of fulfilment, respectively, the cost of the fulfilled tasks $Zd = \{Zd_i\}, \; i = 1, p$ were minimum.

This requirement is performed by means of defining of a task in such a way.

Let: $Zd = \{Zd_i\}, \quad i = 1, p$ and $N' = \{N_i\}, \; i = 1, L$.

The task is that the tasks $Zd = \{Zd_i\}, \; i = 1, p$ shall be allocated among servers $N' = \{N_i\}, \; i = 1, L$ in such a way that the actually needed time for fulfilment of every task Zd_i, is taken into account. The formula (1) will be as follows:

$$\sum_{i=1}^{n} t_{ij} * s_i * y_i \to \min \qquad (2)$$

where t_{ij} is actual time needed for fulfilment of every task Zd_i on server N_j;

Calculation of time t_{ij} of fulfilment of task Zd_i on server N_j is performed in such a way: from the task data sheet Zd_i it is known how much time is needed to fulfil $t^n(Zd_i)$ on "test" computer IC_m on which there were investigated characteristics of a task Zd_i. The speedwork of this computer is $\sigma(IC_m)$.

Suppose that the task Zd_i is to be fulfilled on server N_j its speedwork is known and fixed in the marker $\chi_1(N_i)$ and is used for recalculation of duration of time of fulfilment of this task Zd_i on server N_i in the following way:

$$\tau(Zd_i, N_i) = t^n(Zd_i) * (\chi_1(N_i) / \sigma(IC_m))$$

where $\tau(Zd_i, N_i)$ is the duration of time of fulfilment of task Zd_i on server N_j;

$t^n(Zd_i)$ is the duration of time of fulfilment of Zd_i on the "test" computer IC_m;

$\chi_1(N_i)$ is a marker capacity, productivity, speed of work of the processor of server N_i;

$\sigma(IC_m)$ is the speedwork of the "test" computer IC_m;

$(\chi_1(N_i))$ is the value (numeric value) of a marker $\chi_1(N_i)$;

Thus we can define the value t_{ij} and it is: $\tau(Zd_i, N_i)$.

On this stage we finish the solving of the task of the task allocation among servers in such a way that the price for fulfilment of all the tasks is minimized.

Conclusion and future plans

Considering that this task can be solved under the conditions of uncertainties, irregularities, disbalance of tasks fulfilment can occur in the process of tasks performance. Consequently, in the process of tasks performance it is necessary to keep track of regularity on the servers input and it is also necessary to finish loading of those servers which work queue on input is finished quicker from the list of input tasks of those servers which are late to fulfil tasks along with other servers. Thus, minimization of performance time of all the tasks by all the servers is reached both for the account of optimal allocation of tasks among servers and for the account of constant balancing of servers' load on input of task performance by these servers.

In the further works we plan to implement a genetic algorithm to solve the second part of the above mentioned task.

Corresponding Author:
Dr. Uskenbayeva Raissa K.
The International Information Technologies University, Manas Str./Zhandosov Str., 34 «A»/8 «A», Almaty, 050040, Kazakhstan

References

1. NIST Special Publication, Sept. 2011. A NIST Definition of Cloud Computing. pp: 800-145.
2. Zh, Lu Ch and R. Boutaba, 2010. Cloud computing: state-of-the-art and research challenges. Journal of Internet Services and Applications, 1 (1): 7-18.
3. Rajkumar, B., R. Ranjan and R.N. Calheiros, 2009. Modelling and Simulation of Scalable Cloud Computing Environments and the CloudSim Toolkit: Challenges and Opportunities. Proceedings of the 7th High Performance Computing and Simulation Conference. IEEE Press NY, USA.
4. Grebnev, E., 2011. Cloud services. View from Russia. Moscow: CNews, pp: 282.
5. Golosov, P.E., M.V. Kozlov and Yu.E. Malashenko, 2012. Analysis of specialized computing systems management under the conditions of uncertainties, RAS, 1: 50-66.
6. Telenik, S.F., A.I. Rolik, P.S. Savchenko and I.U. Bodanyuk 2011. Managed genetic algorithm in problems of virtual machines distribution in the data center. Herald of CSTU, 2, pp: 104-113.
7. Uskenbayeva, R.K., A.A. Kuandykov, C. Y. Im, Zh.B. Kalpeyeva and D.K. Kozhamzharova, 2013. Organization of computing processes in the large heterogeneous distributed systems. In the Proceedings of the 44th International Symposium on Robotics (ISR), pp: 1,4; 24-26.
8. Job Submission Description Language (JSDL) Specification. Date Views 20.03.2014 www.ogf.org/documents/GFD.136.pdf.
9. Mohialdeen, I. A., 2013. Comparative study of scheduling algorithms in cloud computing environment. Journal of Computer Science, 9 (2): 252-263.
10. Khan, S., 2013. A survey on scheduling based resource allocation in cloud computing. International Journal For Technological Research In Engineering, 1 (1).

5/18/2014