Two Step Algorithm for Virtual Machine Distributed Replication with Limited Bandwidth Problem

Wojciech Bożejko, Piotr Nadybski, Mieczyslaw Wodecki

Two step algorithm for virtual machine distributed replication with limited bandwidth problem

Wojciech Bożejko[1], Piotr Nadybski [2], and Mieczysław Wodecki [3]

[1] Institute of Computer Engineering, Control and Robotics
Wrocław University of Technology
Janiszewskiego 11-17, 50-372 Wrocław, Poland
wojciech.bozejko@pwr.edu.pl
[2] Faculty of Technical and Economic Science
Witelon State University Of Applied Sciences In Legnica
Sejmowa 5A, 59-220 Legnica
nadybskip@pwsz.legnica.edu.pl
[3] Institute of Computer Science, University of Wrocław
Joliot-Curie 15, 50-383 Wrocław, Poland
mieczyslaw.wodecki@uwr.edu.pl

Abstract. This article presents a proposal of solution for the problem of optimization of the virtual machine (VM) backup or replication process in architecture with multiple locations where efficient bandwidth usage and maximal tardiness for single VM are objectives. The two step algorithm is considered, where in the first step a set of tasks is partitioned into smaller subsets for load balancing. In the second step - tabu search algorithm is used to minimize weighted sum of tardiness. The paper contains the results of computational experiments on the scalability of the presented method.

1 Introduction

In the recent years cloud computing model has became one of the most commonly used architectures for modern IT systems. Many software developers or organizational end users decide to migrate from traditional server environment to cloud based solutions. Recent experiences of customers show that products in this group of technology are now mature, reliable and can offer us many benefits: security, demanded performance and reduced costs of exploitation of IT resources. What is more, the use of such a model of IT systems development changes quite radically the way of thinking about IT tools, which, regardless of their complexity and structure, is becoming just a service to the end user instead of being just a piece of hardware and software running on it [1]. Undoubtedly, this simplicity is just a user's point of view. In fact, technical background in hardware, as well as in software layer, is much more complex and is based on many complementary technologies. The term *cloud computing* is not a name for

a specific service or product. It is rather used to describe a certain model, a trend in the area of computer system engineering.

One of the most commonly used technologies connected with cloud computing is *virtualization* as it most often provides technical basis for cloud computing architecture. Virtualization involves separating the software from the physical environment in which it is run. The division may occur at a few levels. Infrastructure-as-a-Service (IaaS) is one of the possibilities [8]. In this model the most important definition is a virtual machine (VM). This is a software implementation of a machine that executes programs operating systems and applications like a physical machine. Each virtual machine has its own set of virtual hardware that implements its functionality through the use of physical hardware. Essentially, many virtual machines can run on one physical computer simultaneously. The most important components of the typical platform for running virtual machines are: hardware components (a single server or a cluster of servers) delivering necessary computing power, memory and disk space for image files of virtual machines and *hypervisor (Virtual Machine Manager - process running on physical machine responsible for realization of VM instruction on physical hardware)*. To ensure reliability and performance in bigger solutions serving dozens or hundreds VMs disk space is delivered usually by dedicated storage systems.

Reliability, in addition to functionality and performance, is the most important requirement of IT systems. Redundancy is one of the best practices in this field. This means that at least two components realize a specific function. If one fails, the other provides the continuous work of the system. For example, in the case of database servers the consistency of the two copies of the data in the both databases is required. In cloud computing infrastructure with virtual machines redundancy in most situation means that there are at least two data stores and hypervisor managing virtual machines on two separated hardware platforms. Undeniably both copies should be identical. Data integrity process is known as replication. There are many strategies for data replication. Among them there are synchronous, asynchronous and *point-in-time* replications. The first two of them assume that data will be distributed to all backup storages immediately (synchronous) or as soon as possible (asynchronous). Undoubtedly this is the best strategy in case of failure, nevertheless it requires a lot of hardware resources and very fast network connection between data stores. In *point-in-time* strategy data (for example periodic *snapshots* - state of a virtual machine at a particular point in time) is copied to backup localization.

With the progress of cloud technology deployment in real business solutions, the problem of workflow scheduling became an important field of interest for researches. Many of them were described in work Wu et al. [9]. Commercial usage requires on-time resource delivery and guaranteed or predictable time of tasks processing, handling of failures, while costs of hardware resources, power consumption etc. should be reduced as much as possible. The most challenging aspect of the workflow scheduling problems solving process is the fact, that they are NP-hard, so finding optimal solution in those cases is usually impossible in

acceptable time period. Problem of tasks scheduling, described in this paper is an example of this class of problems.

2 Problem description and preliminaries

There is an infrastructure for running clients virtual machines (Figure 1). Analysis of system load statistics shows that there are significant differences in the current use of resources in different parts of the day. In addition to the main data store, there are m backup localizations. Each VM should have one, as actual as possible, backup copy. During the normal system state secondary instance of virtual machine image is immediately synchronized with primary one. Due to big bandwidth usage a situation where users import, restore or create a new virtual machine is more problematic. In such a situation big amount of data is generated and needs to be replicated (Figure 2). To avoid decrease of the system performance, backup copies of new instances are made cyclic, only in the previously mentioned moments when the system usage level is expected to be acceptably low.

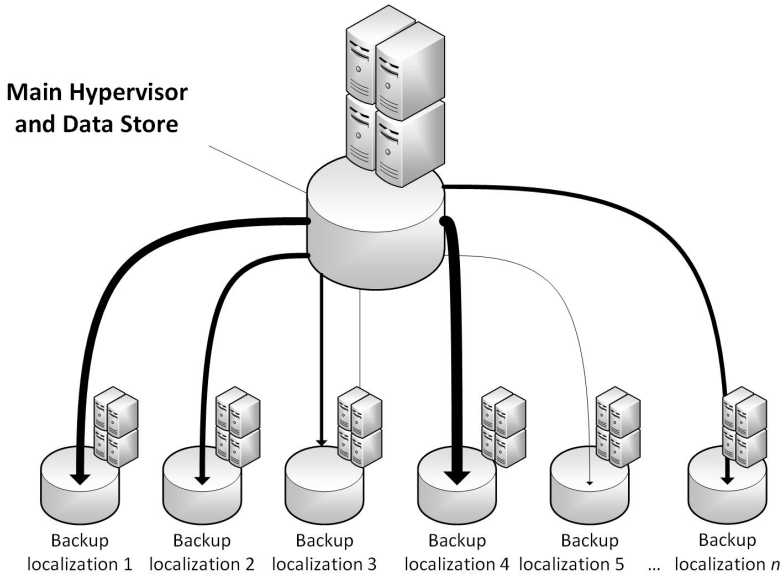

Fig. 1. Cloud architecture.

Basic assumptions:

– There is m backup localization and each of them is connected with primary location by connection of limited bandwidth. It means that in specified time only limited amount of data can be send. Let

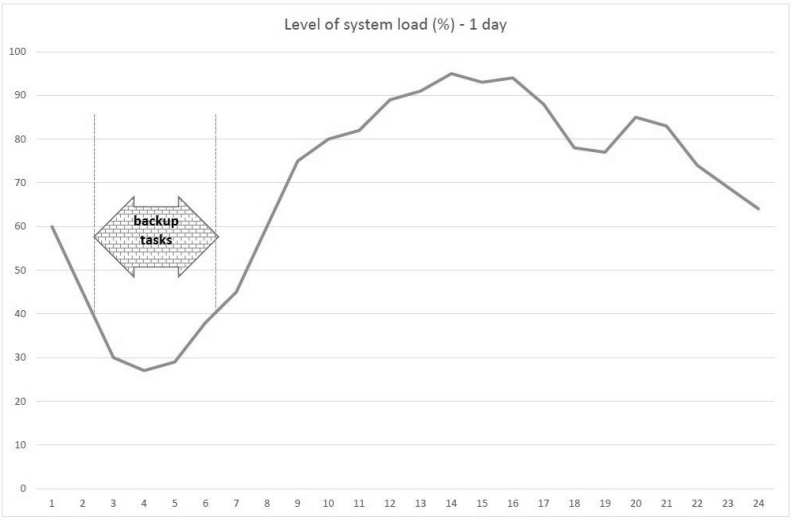

Fig. 2. Daily load condition of the system

$\mathcal{L} = \{1, 2, ..., m\}$ – be a set of backup localizations, whereas
b_l – the amount of data possible to transfer during one cycle for localization $l \in \mathcal{L}$.

– There are n new machines (tasks) to be transferred to secondary location. Each VM has specified size, so transferring it to backup location takes particular period of time, depending on connection bandwidth. Each newly created VM has timer showing the period between creation moment and current time. The timer stops when backup process begins. Each machine should be replicated as soon as possible. There is due time for each new machine to be copied. If not, cost of the tardiness is counted for late replication. What is more, each virtual machine has priority, depending on customer costs plan or administrator's decision.
$\mathcal{V} = \{1, 2, ..., n\}$ - set of tasks - new virtual machines,
s_v - size of machine v ($v \in \mathcal{V}$),
d_v - due time for machine v replication,
w_v - positive weight of the virtual machine v tardiness.

– Tardiness is a time between the moment of completion the replication and due time:

$$T_v = max\{0, C_v - d_v\} \ - \ \text{tardiness, and}$$

$$w_v \cdot T_v \ - \ \text{cost of } v \text{ machine replication.}$$

– The main goal is to make a backup copy of each new virtual machine and minimize the summary cost of the replication process. Summary cost is given

by formula

$$\sum_{v=1}^{n} w_v \cdot T_v.$$

As mentioned earlier in this document, finding global optimal solution due to computational complexity is not possible in most real situations where a set of solutions is too big to check all possibilities. For problem described above there is the two step algorithm proposed

Step 1: for each $v \in \mathcal{V}$ choose $l \in \mathcal{L}$ where $\sum V_j < b_l$ (this condition is true when the total bandwidth is sufficient) and j is a number true of machine planned to be replicated to location l (see **Algorithm 1**);

Step 2: sort the tasks for each location l to minimize cost.

General conception is shown in Figure 3.

2.1 Assign Virtual Machine to backup location

The main goal of the first step is to provide optimal use of all available bandwidth. This is a typical problem for issues related to the load balancing in computer networks. In algorithm described in this paper *Weighted Least-Used* strategy is implemented. It means that each time when location for the single Virtual Machine is chosen, the least heavily loaded backup storage is preferred [6]. Level of load (CL) is calculated as

$$CL = \frac{\sum_{i=1}^{k+1} s_i}{b_l},$$

where k is a number of virtual machines already assigned to location l, the *k+1* machine is the actually considered one and b_l is maximal amount of data to be transferred during one cycle without overload to location l and s_i is a size of machine i.

2.2 Tabu Search algorithm for The Single Machine Total Weighted Tardiness Problem

As a result of *Step 1* all Virtual Machines are assigned for replication to one of the available backup locations. From this moment each channel can be considered separately and is known as The Single Machine Total Weighted Tardiness Problem ($1||\sum w_i T_i$, [10], [11], see also [7], [5], [4]). As the problem is *NP-hard*, the heuristics algorithms are used instead of classic method. As the problem has applications in many fields of science and industry, this problem was the subject of many studies. For the purpose of this experiment classic form of the tabu search (TS) algorithm has been implemented. For neighborhood permutations generating *s-moves* there are used ([2], [3]). After preliminary calculations, tabu list length was set to 7.

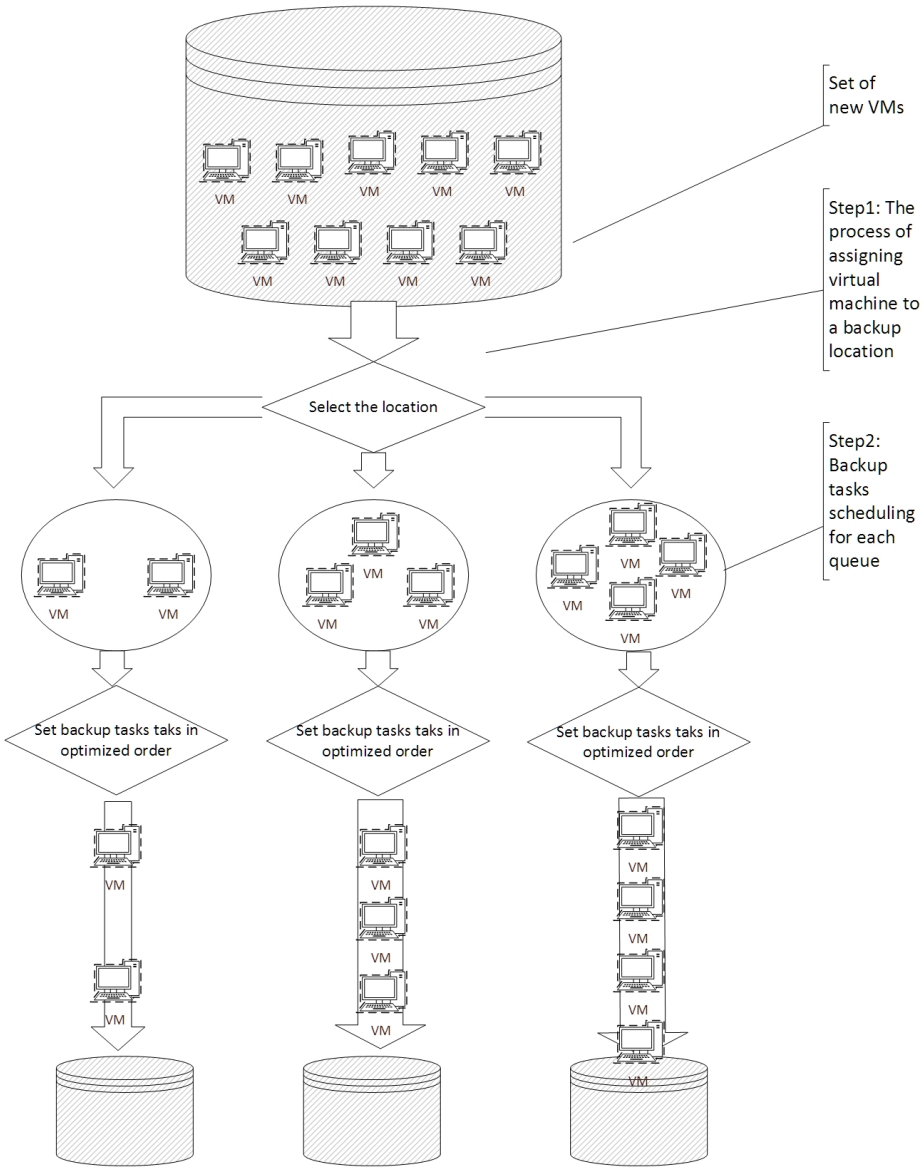

Fig. 3. Conception of backup data flow.

Algorithm 1 Pseudocode for least-used method.

```
sort 𝒱 in descending order
for all v in 𝒱 do
    x ← 0                                   ▷ index of least loaded location
    y ← 0                                   ▷ index of least loaded and not overloaded location
    for all l in L do
        calculate CL
        if CL_l < CL_x then
            x ← l
        end if
        if CL_l < CL_y then
            y ← l
        end if
    end for
end for
if y <> 0 then
    v ← y
else
    v ← x
end if
```

3 Computational Experiments

Algorithm was implemented in *C#* language and tested on Intel i7 3537U (3.1 GHz) processor. In the first run the number of backup locations was set to 5. The set of VMs was randomly generated five times and the algorithm was run for each of them. Then the average time was calculated. The number of iterations for tabu search was set to 50. The operations were repeated for 100, 200 and 500 VMs. In the second run the number of backup location was set to 10. The rest of the procedure was unmodified. The results are presented in the Table 1 and Figure 2.

On the test platform, sets of few hundreds were computed in acceptable time of few minutes, especially when the number of locations was also 10 or more. The performed experiment proved that algorithm is more efficient in cases where more backup location is available. The reason for this is the fact that in such cases the whole set is subdivided into more smaller subsets that are optimized by tabu search.

As expected, with increasing amount of virtual machines to backup, the number of operations to be performed consequently increases very fast. It is worth mentioning that increasing number of backup location causes shorter completion time.

Table 1. Computing time in (ms) for increasing number of VMs).

VMs	storages	iterations	computing time
50	5	50	294
100	5	50	2578
200	5	50	21538
500	5	50	508956
1000	5	50	6431293
50	10	50	97
100	10	50	930
200	10	50	6331
500	10	50	137865
1000	10	50	1804413
50	5	100	562
100	5	100	3320
200	5	100	45944
500	5	100	860553
1000	5	100	11504165

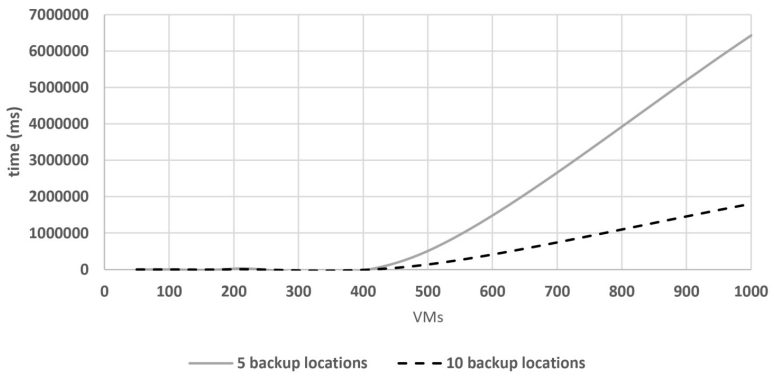

Fig. 4. Computing time in (ms) for increasing number of VMs (50 iterations).

Table 2 contains average relative change between greedy algorithm (GREE) and tabu search optimized results for set of data used in this experiment. Due to greedy algorithm tasks are firstly ordered by d_i (due time) from the shortest time to the longest and then, one by one, assigned to remote backup location where cost for this machine would be the lowest. The method presented in this paper, compared with GREE, allowed for a 40 to 50 percent reduction of the $w_i \cdot T_i$ cost, depending on the test set of data.

Algorithm 2 Pseudocode for GREE algorithm.

sort $\mathcal{V}$ in ascending order by due time
for all v in $\mathcal{V}$ **do**
  $x_v \leftarrow 1$ ▷ index of optimal already found backup location for this VM
  $y \leftarrow \sum_{i=1}^{n_1}(w_i * T_i) + (w_v * T_v)$ ▷ n_1 - number of backup tasks assigned already to location 1
  for all l in $\mathcal{L}$ **do**
    calculate $C_l = \sum_{i=1}^{n_l}(w_i * T_i) + (w_v * T_v)$ ▷ Cost of tasks (vm) assigned to l location and task v
    if $C_l < y$ **then**
      $x_v \leftarrow l$
    end if
  end for
end for

Table 2. Relative change (GREE and TS algorithms)

VMs/tasks	GREE cost	TS cost	Relative change
50	1803	872	0.52
100	8911	4493	0.50
200	43318	24885	0.43
500	253636	148099	0.42
1000	1051256	624702	0.41

4 Conclusion

Strategy for scheduling backup or replication tasks in cloud computing, virtualization environment presented in this paper is based on two methods widely used for solving problems in industry and computer networks. Both of them used together as the two-step algorithm allow to improve performance and reduce time needed for operations. In the first step the size of problem is reduced by quick method, where each task is assigned to destination based on amount of data to be transferred and bandwidth usage of each available network connection. This operation reduces the size of a set processed by a more precise but also more complex algorithm used in the second step. The proposed method can produce less accurate results than in case, when classic heuristic like Tabu Search is only used, but should reduce processing delay. Further research is also possible for tabu search for or example more sophisticated method in the first step can be tested. The discussed method can be used for problems in cloud computing like non on-line replication, synchronization where many network connections, data storages or virtualization platforms are available. As the parameters of cost function are freely definable other use is also possible.

References

1. Armbrust M., Fox A., Griffith R., Joseph A. D., Katz R., Konwinski A., Zaharia M., A view of cloud computing, Communications of the ACM, 53(4), 2010, 50–58.
2. Bożejko W., Wodecki M., On the theoretical properties of swap multimoves, Operations Research Letters, Elsevier Science Ltd., 35/2, 2006, 227–231.
3. Bożejko W., Wodecki M., Solving Permutational Routing Problems by Population-Based Metaheuristics, Computers & Industrial Enginering, 57, 2009, 269–276.
4. Bożejko W., Wodecki M., Parallel genetic algorithm for minimizing total weighted completion time, Lecture Notes in Artificial Intelligence, 3070, 2004, 400-405.
5. Bulbul K., A hybrid shifting bottleneck-tabu search heuristic for the job shop total weighted tardiness problem, Computers and Operations Research, 38, 2011, 967 – 983.
6. Ellrod C., Load Balancing – Least Connections, *https://www.citrix.com/blogs/2010/09/02/load-balancing-least-connections/*, (Access: 2016.04.10).
7. Lin Y.K., Chong C.S., A tabu search algorithm to minimize total weighted tardiness for the job shop scheduling problem, Journal of industrial and management optimization, Vol. 12, Nr 2, 2016, 703 – 713.
8. Mell P., Grance T., The NIST definition of cloud computing, Recommendations of the National Institute of Standards and Technology, *http:// faculty.winthrop.edu/domanm/csci411/Handouts/NIST.pdf* (Access: 2016.04.03).
9. Wu F., Wu Q., Yousong T. Workflow scheduling in cloud: a survey, The Journal of Supercomputing 71(9), 2015, 3373–3418
10. Wodecki M., A Branch-and-Bound Parallel Algorithm for Single-Machine Total Weighted Tardiness Problem, International Journal on Advanced Manufacturing Technology, Vol. 37, Nr 9-10 , 2008, 996–1004.
11. Wodecki M., A block approach to earliness-tardiness scheduling problems, International Journal on Advanced Manufacturing Technology, 40, 2009, 797–807.