iJRASET
International Journal For Research in Applied Science and Engineering Technology

INTERNATIONAL JOURNAL FOR RESEARCH

IN APPLIED SCIENCE & ENGINEERING TECHNOLOGY

Volume: 5 **Issue:** V **Month of publication:** May 2017

DOI:

www.ijraset.com

Call: 08813907089 | E-mail ID: ijraset@gmail.com

International Journal for Research in Applied Science & Engineering Technology (IJRASET)

Network Aware Virtual Machine Migration by PSO Optimization

Sarthak Tandon[1], Chandan Kesarwani[2], Paridhi Srivastava[3], Abhishek Suryan[4], Swathi J N[5]
[1,2,3,4,5]*School of Computer Science and Engineering, VIT University, Vellore, India*

Abstract: ***A good VM migration algorithm can greatly improve network performance and scalability. Only a few studies presently focus on the network-aware VM migration (NetVMM) problem. The NetVMM problem is a type of a multiple-knapsack problem. Thus, finding an optimal solution in polynomial time is not practical. Our goal is to find an approximation solution to this NP-complete problem which is energy efficient. In this era of technology, some energy efficient techniques are needed. Hence we were motivated to carry out this project and work for energy efficient environment. Our objective to find is an approximate optimal solution which is energy efficient through repeated iterations to make it a good solution for the VM migration problem. We will make a Matlab program for showing the output of the PSO optimization algorithms used. In computing and research, a genetic algorithm (GA) may be a meta-heuristic galvanized by the method of natural action that belongs to the larger category of organic process algorithms. Genetic algorithms are unremarkably accustomed generate high-quality solutions to improvement and search issues by hoping on bio-inspired operators like mutation, crossover and choice. In a genetic algorithm, a population of candidate solutions to an optimization problem evolves toward better solutions. Each candidate solution has a set of properties. Particle swarm optimization (PSO) solves a optimization scenario by having a number of candidate solutions, here virtual machines, and migrating these virtual machines around in the search-space according to some derived formulae over the virtual machines' position and velocity. We will be implementing the PSO optimization algorithm in Matlab environment to solve the problem of cost optimization of the data centers, so that optimal number of tasks are divided in each virtual machine, hence balancing the cost and the network.***

Keywords: network aware, live migration, virtual, virtualization, network virtual machine migration, particle swarm optimization.

I. INTRODUCTION

A server farm expends the power that can generally be utilized to power a large number of homes. The immense levels of force utilization is the thing that makes server farms and preservationists search for approaches to diminish the power utilized and make server farms much more vitality productive than they right now are. Thinking on it, is virtualization the response for decreasing force utilization by server farms? It most certainly is. The most important objective of virtualization is to make the most proficient utilization of accessible framework assets. Virtualization brings about significantly more productive utilization of assets, including vitality. Characterizing virtualization just, to virtualize is to make a solitary bit of equipment capacity as numerous parts. Distinctive UIs seclude diverse parts of the equipment, in this way making every one act and capacity as an individual, isolate element. With regards to a server farm, introducing virtual framework permits a few working frameworks and applications to keep running on a lesser number of servers, lessening the general vitality utilized for the server farm and for its cooling.

With the expanding pattern towards more correspondence concentrated applications in the Cloud server farms, the between VM arrange data transfer capacity utilization is developing quickly. This circumstance is disturbed by the sharp ascent in the measure of the information that are taken care of, prepared, and exchanged by the Cloud applications. Moreover, the general application execution very relies on upon the hidden system assets and administrations. As an outcome, the system conditions have coordinate effect on the Service Level Agreements (SLAs) and incomes earned by the Cloud suppliers. Late progression in virtualization advances develops as an exceptionally encouraging instrument to address the previously mentioned issues and difficulties. Ordinarily, VM administration choices are made by utilizing different scope organization apparatuses, for example, VMware Capacity Planner and their destinations are set to combine VMs for higher use of figure assets (e.g., CPU and memory) and minimization of force utilization, while disregarding the system asset utilization and conceivable prospects of advancement. Thus, this frequently prompts to circumstances where VM sets with high common movement burdens are put on physical servers with expansive system cost between them. Such VM position choices put weight on the system joins and affect the application execution. Enhancement of VM arrangement and movement choices has been turned out to be down to earth and viable in the field of physical

server asset use and vitality utilization diminishment, and a plenty of research commitments have as of now been made tending to such issues. As of not long ago, a modest bunch of research endeavors are made to address the VM situation and relocation issue concentrating on between server organize separate, run-time between VM movement qualities, server load and asset requirements, register and system asset requests of VMs, information stockpiling areas, et cetera. These works not just contrast in the tended to framework suspicions and demonstrating procedures, additionally fluctuate extensively in the proposed arrangement approaches and the directed execution assessment strategies and situations. As an outcome, there is a quickly developing requirement for expound scientific classification, overview, and near investigation of the current works in this rising examination region. Keeping in mind the end goal to dissect and evaluate these works in a uniform design, this part shows a review of the parts of Cloud server farm administration as foundation data, trailed by different cutting edge server farm organize models, between VM movement designs saw underway server farms took after by an intricate scientific classification and study of remarkable research commitments.

The rest of this paper is organized as follows: Section 2 shows the literature survey; Section 3 comprises of the problem formulation; Section 4 comprises of the solution methodology where we show the algorithm and the main concept with the help of diagrams and the flowchart or architecture; Section 5 comprises of the experimental setup in which the system configuration is shown where parameter values and tool kit plays a major role; section 6 concentrates on the results and discussion.

II. LITERATURE REVIEW

Enhancement of VM arrangement and movement choices has been turned out to be down to earth and viable in the field of physical server asset use and vitality utilization diminishment, and a plenty of research commitments have as of now been made tending to such issues. Out of these research topics, we have chosen algorithmic approach to network aware live migration of virtual machines as our area of survey and analysis.

Torantino et.al (2006) pioneered the integration of Virtual Machines with deterministic light-weight path network services across a MAN/WAN. The results provided for a new stage of virtualization, one that computation is not any longer localized inside a knowledge center. Hai et.al (2010) proposed an optimisation scheme for live migration, under that in keeping with pre- copy speed, the VCPU operating frequency might be reduced so at a certain phase of the pre-copy the remaining dirty memory might reach a desired touch. Gang et.al (2015) proposed that we utilize the live migration feature of virtual machine monitors to migrate the job from one sub-cluster to another. Weiwei et.al (2012) proposed the problems of VM placement with 2 distinct optimisation objectives. For each objective presenting the formal definition and proving its NP- hardness. Hai et.al (2013) presented the structure ANd implementation of an innovative VM migration approach that was supported reduction of memory that uses memory compression to implement quick virtual machine migration. Muhammad et.al (2013) used AN rule to improve resource efficiency throughout by decreasing the size of memory image that's stored on source host. Vincenzo et.al (2015) proposed a model for VM migration for the economical energy usage providing fine predictions for para-virtualised VMs running on homogeneous hosts.

Gursharan et.al (2015) proposed 2 strategies to balance the load during a system with multiple virtual machines (VMs) through automated live migration. Umesh and Kate (2015) based the selection of migration techniques on the VMs' network traffic profiles so the direction of migration traffic complements the direction of the most VM application traffic. This approach minimises network contention for migration. Youwei et.al (2015) developed a new VM scheduler to reduce energy value for the cloud service providers in order to deduce that there exists best frequency for a PM to process certain VMs and define the best performance–power magnitude relation to weight the heterogeneous PMs in the cloud. Weizhe et.al (2016) considered a network-aware VM migration in overcommitted cloud and formulated it into a non-deterministic polynomial time-complete problem. Mattias et.al (2014) proposed developing models to seek out performance metrics and a far better serial migration strategy to integrate the previous migration scheme into it. Enzo et.al (2015) gave the minimisation of the migration-induced communication energy under service level agreement induced arduous constraints, on the total migration time, downtime, retardation of the migrating applications and overall available information measure. Raman and B. Annappa (2015) presented AN intelligent decision maker to trigger the migration by predicting the future work and mixing it with predicted performance parameters of migration process.

Xiumin et.al (2016) proposed a heuristic rule for multiple PMs, by finding a series of backpack problems wherein Simulation results demonstrate the effectiveness of the schemes, and show that the proposed rule is able to leverage the spatial variation in the VMs migration for delay and value optimisation.

Yosr et.al (2015) proposed AN automated approach to verify distributed firewalls reconfiguration after migration and elaborate a language that captures distributed stateless and stateful firewalls with their underlying semantics.

Ankita and Shilpa (2016) came up with a technique for optimizing virtual machine placement by live migration exploitation dynamic threshold values ensuring a deadlock free resource allocation. Wang et.al (2016) formulated the quadratic constrained non-

convex 0-1 program and propose to carry the problem to a higher dimensional space by classical linearization, thereby handling the problem in the framework of MIP. Zoha and Shailendra (2016) proposed a comprehensive study of the state-of-the-art VM placement and consolidation techniques used in green cloud that concentrate on rising the energy efficiency.

Tao et.al (2016) proposed ReSeer—a search-based replay approach for multiprocessor virtual machines consisting of three phases as well as record, search, and replay. ReSeer considerably reduces performance overhead at runtime by searching expected execution ways instead of recording all the operations of accessing shared memory. Zhihua et.al (2016) proposed a hybrid Bayesian network-based VMs consolidation method by conducting a performance evaluation study exploitation CloudSim toolkit, and the tracedriven comparison experiments. Heejun et.al (2016) proposed a threshold-based, wireless link-aware flow placement rule with low complexity and to enhance traffic neck of the woods, suggest a set of virtual machine placement rules under the flow placement algorithm. Umesh and Kate (2016) proposed a traffic-sensitive live VM migration technique to reduce the contention of migration traffic with the VM application traffic employing a combination of pre-copy and post-copy techniques for the migration of the co-located VM's. Benjamin et.al(2016) evaluated the impact of such migrations on the resource allocation process, we use the real traces of a bus transit system to simulate a vehicular network where virtual machines migrate via V2V communications. Santosh and Sunil (2016) engineered a threat and security model for the VMMA system to hold out careful investigation enabling the system administrator and its developer to create organized security requirements and protection mechanism.

It is difficult to find out a solution to the network aware live migration of virtual machines which is optimal and energy efficient at the same time. Thus, we have proposed an approximate optimal solution which is energy efficient through repeated iterations to make it a good solution for the VM migration problem. We have modified the earlier proposed techniques by implementing PSO optimization algorithm for NetVMM in Matlab environment to solve the problem of cost optimization.

III. PROBLEM STATEMENT

There is a set of virtual machines V_1 to V_n and a set of servers S_1 to S_n. Virtual machines V_i and V_j have communication between them. The servers S_i and S_j are the actual physical machines where the data transfer takes place. $W(V_i, V_j)$ is the weight of 2 vertices V_i, V_j Weizhe Zhang et.al (2006) proposed the cost of migrating V_i and V_j to the servers S_i and S_j:

$$Cost(V_i,S_k,V_j,S_l)=Distance(S_k,S_l)*W(V_i,V_j) \quad (1)$$

Distance is defined as the number of hops between the 2 servers. When V_i is placed onto S_k, X_{ik} is 1, otherwise, Xik is 0.

$$X^{jl}_{ik}=X_{ik}*X_{jl} \quad (2)$$

The total communication cost is

$$Cost=\sum Cost(V_i, S_k, V_j, S_l)* X^{jl}_{ik} \quad (3)$$

VM migration also causes additional data transfer overhead, which would also increase the network cost of the data center. Liu et al.(2010) experimentally verified that the migration energy consumption was proportional to the data volume of network traffic caused by VM migration. In this study, we interpret the migration costs incurred by data transferring as the multiplier of the VM size and the distance between the source PM and the destination PM. We assume that the size of Vi is $Size_i$, and it migrates from S_l to S_k. The migration cost is then shown as follows:

$$Cost\_Mig\ (V_i, S_k) = Sizei \times Distance\ (S_l, S_k) \quad (4)$$

The cost of migrating Vi from server S_l is represented by

$$Cost\_Mig(V_i)=\sum(size_i \times D_k \times X_{ik}) \quad (5)$$

S_l is the source server. We consider the situation that Vi migrates to any server S_k. If Xik = 1, then Vi is placed onto Sk. Dlk =Distance(S_l, S_k).

The total migration cost in the data center is:

$$Cost\_Mig(V_i)=\sum\sum(size_i \times D_k \times X_{ik}) \quad (6)$$

The standardization of Cost_Comm is

$$Cost\_Comm\_std= (\sum Cost(V_i,S_k,V_j,S_l) \times X_{ik}^{jl)} \div (e_{Topo\_vm} \times max(cost)) \quad (7)$$

$$Cost\_mig\_std=(size_i \times D_k \times X_{ik}) \div (N_0 \times max(size_i) max(D_{ik})) \quad (8)$$

where eTopo_vm represents the number of VM (VMNum) pairs that have communication demands, i.e., the number of edges in the VM dependency graph. Each edge in the dependency graph has Cost(Vi, S_k, Vj, S_l). Cost_Comm is divided by the number of edges, the maximum communication demand, and the maximum PM distance to standardize the communication costs.

To adjust the weights of communication and migration costs and to ensure scalability and adaptability, we use two weight

Volume 5 Issue V, May 2017
ISSN: 2321-9653
International Journal for Research in Applied Science & Engineering Technology (IJRASET)

coefficients α and β in the optimization function to minimize the total cost Cost_Total as follows:

$$Cost\_Total = \alpha \times Cost\_Comm\_std + \beta \times Cost\_Mig\_std \quad (9)$$

$$F[i] = Fitness[i] = Cost\_Total[i] \quad (10)$$

Hence we will find the minimum value of the fitness function using both the PSO algorithm to find the minimum cost of data center at different values of α for some specified values of PSO parameters. We will also find the maximum number of tasks that could be assigned to the VMs to obtain the minimum cost in the data center.

IV. SOLUTION METHODOLOGY

We have to determine the overloaded virtual machines from equation 10 and 11 and then determine the candidate host virtual machine. Then we assign the overloaded tasks to the specified virtual machines reducing the total cost of communication between the virtual machines as specified by equation 9.

A. Overview of our Solution Methodology

The Genetic optimization algorithm works in the following way

TABLE I. SYMBOLICAL REPRESENTATION OF THE ALGORITHM PARAMETERS

Sno	Symbolic representation in Algorithm	Actual representation	Variable representation
1	Chromosome	Allocation of VMs to PMs	Solution
2	Chromosome evaluation	Best allocation of Virtual machines	$Solution_{best}$
3	Father chromosome	Allocation a	$Solution_a$
4	Mother chromosome	Allocation b	$Solution_b$
5	Child chromosome	Allocation on Crossover	$Solution_c$ $Solution_d$

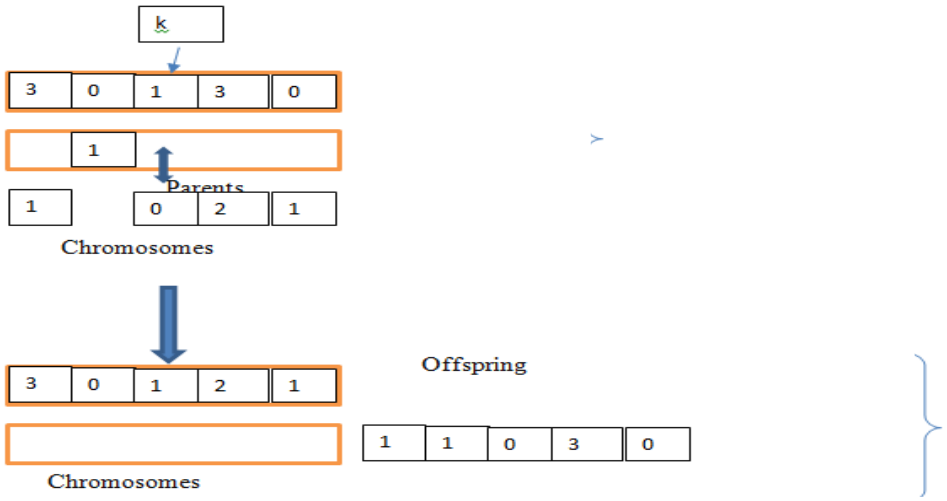


Fig 1. Two point crossover

The genetic algorithm as stated by Weizhe Zhang et.al (2006) is as follows:

Solution <-application-aware()
Solution[1]<-solution
For each Solution[i]
Randomly generate solution
Fitness[i]<-f(solution[i])
If fitness[i]<Best fitness then

```
                    Solution <-solution[i]
                    Best fitness<-fitness[i]
                End if
End for
//main loop
Iter <-0,stag Count<-0
While iter <nIterations and stagCount<nStagCount do
    newSolutions ←NULL
    while new Solutions.size()<popSize do
        Solution_a←RoulettewheelSelection(Solution,popsize)
        Solution_b←RoulettewheelSelection(Selection,popsize)
        (Solution_c ,Solution_d)←Crossover( Solution_c ,Solution_d)
        Solution_c← Mutation(Solution_c)
        Solution_d← Mutation(Solution_d)
        newSolutions←push(Solution_c)
        newSolutions←push(Solution_d)
    end while
    Solution ←new Solutions
    Solution_iterbest← FindWithMinFitness(Solution)
    If fitness(Solution_iterbest) <fitness(Solution_best)then
        Solution_best ← Solution_iterbest
    Else stagCount ←stagCount+1
    End if
    Iter ←iter+1
End while
```

B. Particle Swarm Optimisation

The Particle swarm optimization algorithm on the other hand works in the following way:

TABLE II. SYMBOLIC REPRESENTATION IN PSO ALGORITHM

Sno	Symbolic representation in Algorithm	Actual representation	Variable representation
1	Particle	VM assigned to PMs	S
2	Swarm	PM	G

Let S be the number of VMs in the particle, each having a position $x_i \in R^n$ in the search-space and a velocity $v_i \in R^n$. Let p_i be the best known position of VM i and let g be the best known position of the entire set of virtual machine. The values b_{lo} and b_{up} are respectively the lower and upper boundaries of the search-space. The algorithm for its optimization is as follows:

```
for each VM i = 1, ..., S do
  //Assign a value to the VM's position with a random vector: pos_i ~ U(b_lo, b_up)
  //Assign a value to the VM's best known position to its initial position: p_i ← pos_i
  if f(p_i) < f(g) then
     Modify the PM's best known  position: g ← p_i
  //Assign a value to the VM's velocity: v_i ~ U(-|b_up-b_lo|, |b_up-b_lo|)
while a exit criteria is not fulfilled do:
  for each VM i = 1, ..., S do
    for each task assigned d = 1, ..., n do
      //Pick random numbers: r_p, r_g ~ U(0,1)
      Modify the VM's velocity: v_i,d ← ω v_i,d + φ_p r_p (p_i,d-pos_i,d) + φ_g r_g (g_d-pos_i,d)
```

Volume 5 Issue V, May 2017
ISSN: 2321-9653
International Journal for Research in Applied Science & Engineering Technology (IJRASET)

//Modify the VM's position: $pos_i \leftarrow pos_i + v_i$
if $f(pos_i) < f(p_i)$ then
  //Modify the VM's best known position: $p_i \leftarrow pos_i$
  if $f(p_i) < f(g)$ then
    //Modify the PHYSICAL MACHINE's best known position: $g \leftarrow p_i$

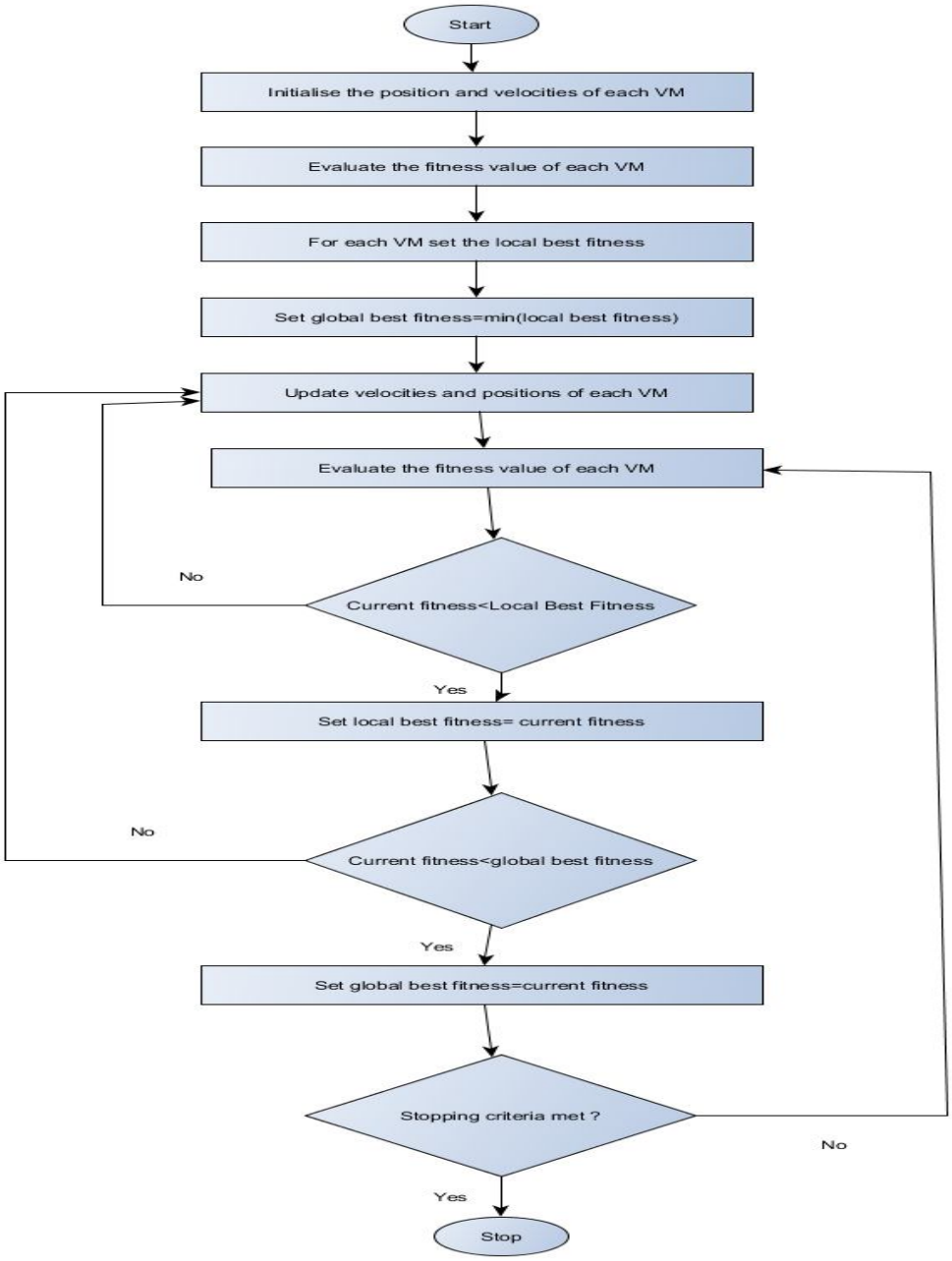


Fig 2. Flowchart of the PSO algorithm for obtaining the minimum cost of data center

V. EXPERIMENTAL SETUP

The algorithm can be run as many number of times. The parameter values will change depending on the values of alpha and beta. The tool required is Dev C++. The minimum system specifications required are:

TABLE III. MINIMUM SPECIFICATIONS REQUIRED

Volume 5 Issue V, May 2017
ISSN: 2321-9653
International Journal for Research in Applied Science & Engineering Technology (IJRASET)

Processor required	Pentium 500 MHz
RAM required	128 MB or More
Disk space required	20 MB Disk Drive
Internet Connection	Nil

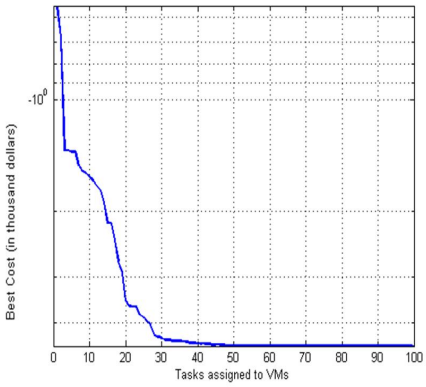
VI. RESULTS AND DISCUSSIONS

We have used the following values of the PSO parameters (assumptions) to execute the algorithm for finding the minimum cost of the data centre for various values of α:

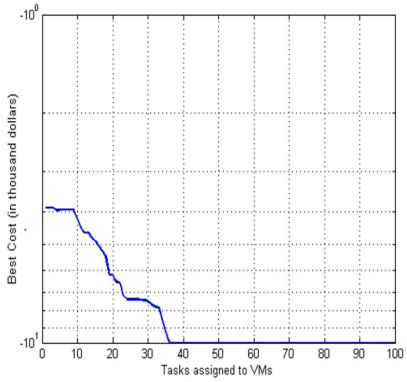
A. Number of virtual machines=10
B. Maximum Number of tasks assigned to each VM=100
C. Population Size (Swarm Size)=5
D. Inertia Mass=1
E. Effective cost of migration=1.5 thousand dollars
F. Effective cost of communication=2.0 thousand dollars
G. Velocity Limits=(-10,2)

We obtain the following output from the given algorithm for 6 different values of α which are 0.1, 0.2, 0.3, 0.4, 0.5 and 1.0 given that β=1- α. Hence the values of β will be 0.9, 0.8, 0.7, 0.6, 0.5 and 0 respectively.

The following graphs in Figure III are drawn in MATLAB environment with the y axis as best cost in thousand dollars and the x axis is the tasks assigned to the virtual machines.

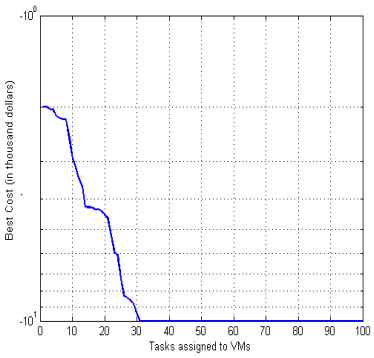


(a)α=0.1

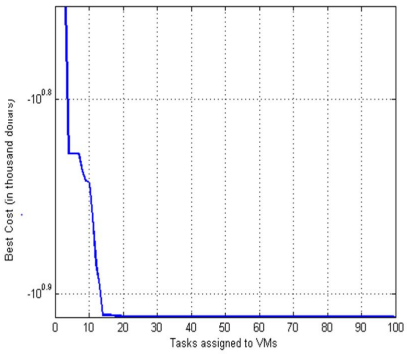


(b)α=0.2

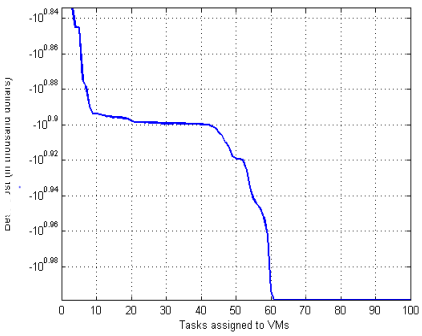
Volume 5 Issue V, May 2017
ISSN: 2321-9653
International Journal for Research in Applied Science & Engineering Technology (IJRASET)



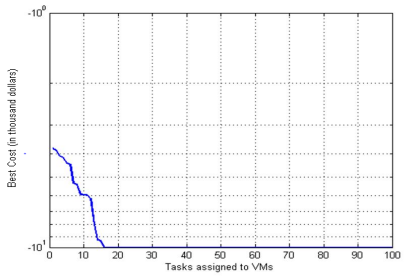
(c)α=0.3



(d)α=0.4



(e) α=0.5



(f)α=1.0 (f) α=1.0

Fig 2. Graphs obtained through MATLAB

VII. CONCLUSIONS

The following inference is drawn from the graphs obtained in Fig 3 for finding the maximum number of tasks assigned to the Virtual machines to achieve minimum cost of data center:

TABLE IV. INFERENCE DRAWN

Value of α	Value of β	Minimum cost (in dollars)	Maximum tasks assigned to achieve minimum cost
0.1	0.9	5699	60
0.2	0.8	6085	32
0.3	0.7	4903	15
0.4	0.6	4156	35
0.5	0.5	9629	55
1.0	0.0	8542	15

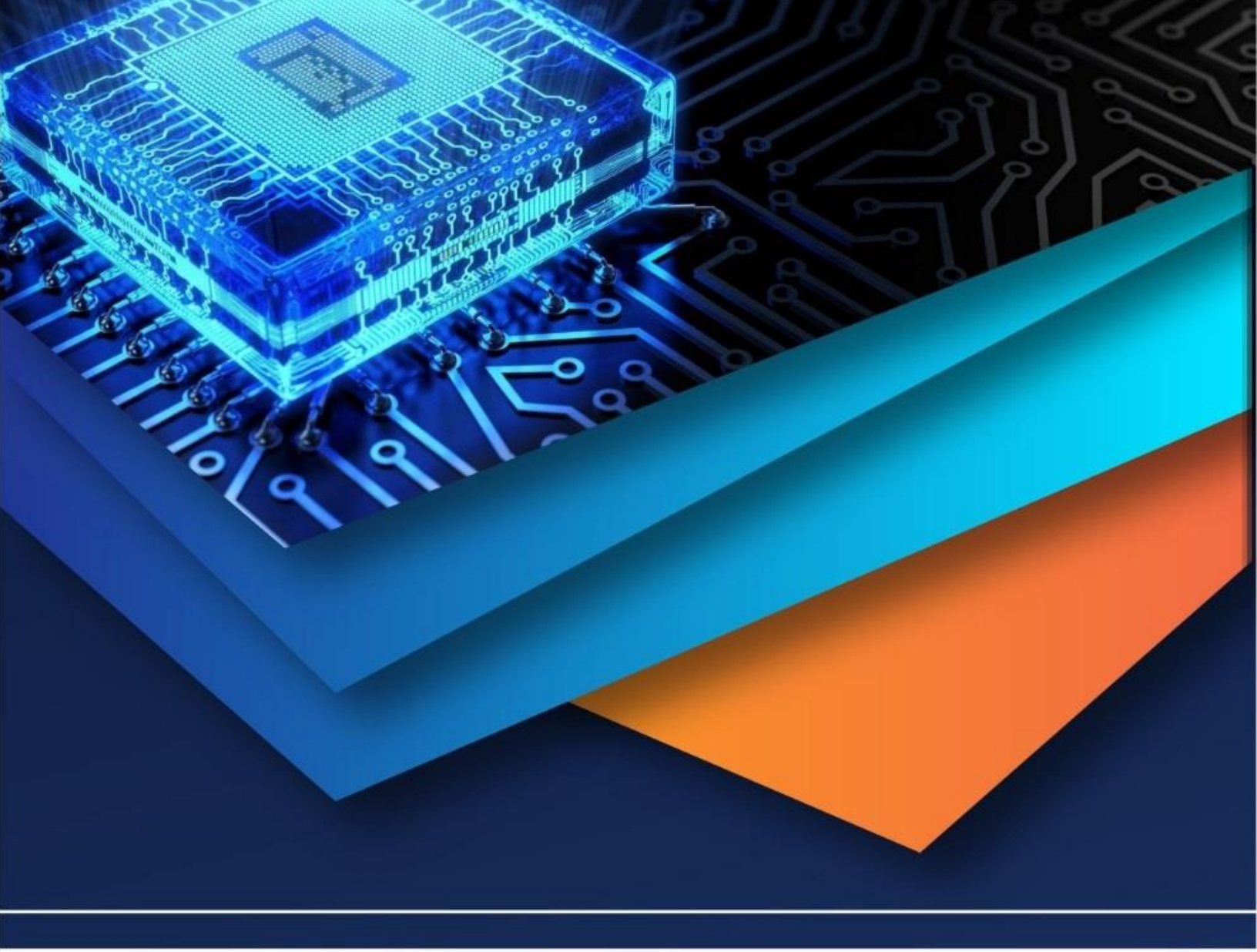
Hence we infer that under the given conditions or assumptions for a data center scenario we achieve the best cost of approximately 5700 dollars in case of having the maximum tasks assigned to the virtual machines to be 60, with the value of α to be 0.1 and that of β to be 0.9. As we have previously stated, because migration energy consumption was proportional to the data volume of network traffic caused by VM migration, hence our scenario will be called network aware virtual machine migration.

VIII. ACKNOWLEDGMENT

I would like to thank the people who have helped me most throughout my project. I am grateful to my teacher Prof. Swathi J.N. for her nonstop support for the project. A special thanks of mine goes to my colleagues who helped me out in completing the project, where they all exchanged their own thoughts and made this possible to complete the paper with all accurate information.

REFERENCES

[1] Wezhe Zhang, Shuo Han, Hui He : Network-aware virtual cloud, 2016
[2] Rochweger, B., Breitgand, D., Epstein Computer, Reservoir-when one cloud is not enough 44(3), 44–51 (2011)
[3] Poli, R., Kenedy, J., Blackwell, T **1**(1), 33–57 (2007),Particle swarm optimization.
[4] Ranjan, R., Buyya, R.: Decentralized overlay for fedeation of enterprise clouds (2008)
[5] Gori, I., Guitart, J., Torres, J. Characterizing cloud federation for enhancing providers: pp. 123–130 (2010)
[6] Kennedy, J. and Eberhart, R. C. Particle swarm optimization Vol. IV, pp. 1942-1948 Piscataway, NJ, 1995.
[7] Eberhart, R. C. and Kennedy, JPiscataway, NJ, Nagoya, Japan, 1995.
[8] Eberhart, R. C. and Shi, Y. Particle swarm optimization: developmentsPiscataway, NJ., Seoul, Korea., 2001.
[9] Eberhart, R. C. and Shi, Y. Evolving artificial neural networks.
[10] Eberhart, R. C. and Shi, Y. Comparison between genetic algorithmsSpringer-Verlag, Berlin, San Diego, CA., 1998.
[11] Shi, Y. 591-600. Springer-Verlag, New York, 199 Parameter selection in particle swarm optimization. Evolutionary Programming
[12] Shi, Y. and Eberhart, R. C. A modified particle swarm optimizer 1998
[13] Eberhart, R.C.; "Comparing inertia weights and constriction factors in particle swarm optimization".
[14] Carlisle, A.; Dozier, G. (2001). "An Off-The-Shelf PSO" (PDF). Proceedings of the Particle Swarm Optimization Workshop
[15] van den Bergh, F. (2001). An Analysis of Particle Swarm Optimizers (PhD thesis)
[16] Clerc, M.; Kennedy, J. (20026 (1): 58–73.). "The particle swarm - explosion, stability, and convergence in a multidimensional complex space".
[17] Trelea, I.C. (2003). The Particle Swarm Optimization
[18] Bratton, D.; Blackwell, T. (2008) PSO". Journal of Artificial Evolution and Applications.
[19] Evers, G. (2009)The University of Texas - Pan American, Department of Electrical Engineering.
[20] Meissner, M.; Schmuker,:OPSO and its application to artificial neural network training.
[21] Pedersen, M.E.H. (2010). Tuning & Simplifying Heuristical Optimization (PhD thesis)
[22] Pedersen, M.E.H.; Chipperfield, A.J. (2010)."Proactive particles in swarm optimization: a self-tuning algorithm based on fuzzy logic"
[23] Nobile, M.S:; Pasi, G.; Cazzaniga, P.; Besozzi, D.; Colombo, R.; Mauri, G. (2015)
[24] Cazzaniga, P.; Nobile, M.S.; Besozzi, D. (2015), The impact of particles initialization in PSO: parameter estimation as a case in point
[25] Pedersen, M.E.H. (2010), "Good parameters for particle swarm optimization"



10.22214/IJRASET

45.98

IMPACT FACTOR: 7.129

IMPACT FACTOR: 7.429

INTERNATIONAL JOURNAL FOR RESEARCH

IN APPLIED SCIENCE & ENGINEERING TECHNOLOGY