An Efficient Approaches on VM Consolidation and VM Allocation in Data Centers using Cloud Computing

RUBAN E
COMPUTER SCIENCE AND ENGINEERING
SRM ISTITUTE SCIENCE AND TECHNOLOGY
CHENNAI,TAMILNADU

VIGNESH M
COMPUTER SCIENCE AND ENGINEERING
SRM ISTITUTE SCIENCE AND TECHNOLOGY
CHENNAI,TAMILNADU

NAVEEN KUMAR R
COMPUTER SCIENCE AND ENGINEERING
SRM ISTITUTE SCIENCE AND TECHNOLOGY

ANWAR BASHA H
COMPUTER SCIENCE AND ENGINEERING
SRM INSTITUE OF SCIENCE AND TECHNOLOGY
CHENNAI, TAMILNADU

***Abstract*— Cloud computing data centres are rapidly growing in numbers and capacity to meet the huge demands for large storage and high performance computing storage. Such data centres produces high carbon dioxide and operating costs because it consumes huge amount of electrical energy. Because of the power inefficiency of hardware, quantity of computing resources and the inefficient usage of these resource there is extremely high energy is consumed. Virtual Machine [VM] consolidation has the transferring capability of VM between physical servers with a close to zero down time and also involves VMs live migration. To increase the energy efficiency and improve the resource utilization in data centres, it is an efficient way. Host overload/underload detection, VM selection and VM placement are there in the VM consolidation. In Our Proposed Model We are going to use Roulette-Wheel Selection Strategy, Where the VM selects the Instance type and Physical Machine [PM] using Roulette-Wheel Selection Mechanism.**

I. Introduction

An effective technique for energy footprint reduction and improve the utilization of resources in cloud data centres is Virtual Machine (VM) consolidation. In a distributed fashion or a centralized way it can be implemented. Virtual Machine (VM) consolidation is one of the key mechanisms for energy efficient dynamic cloud resource management system designing is Virtual machine consolidation. It is based on the premise that migrating VMs into fewer number of Physical Machines (PMs) can achieve both optimization objectives, increasing the utilization of Cloud servers while concomitantly reducing the energy consumption of the Cloud data centre. However, poor Quality of Service (QoS) can be caused due to more packing of VMs into a single server, since VMs shares the PMs underlying physical resources. To select dynamically the VMs for migration and the impact on QoS is considered in addition to the above mentioned objectives of optimization. In VMC a wide range of meta-heuristic and heuristic VMC algorithms is proposed to achieve near-optimality because it is a NP-Hard problem. Cloudsim Setup: - CloudSim is a Java Programming language written framework. It is a toolkit for evaluating of provisioning algorithms resources and simulation of cloud computing environment and modeling it. CloudSim developed in the year 2009 by buyya in the Grid Computing and Distributed Systems laboratory. (GRIDS) University of Melbourne. CloudSim has classes that serve as a simulator of cloud computing components such as broker, CIS (Cloud Information Service), VM, cloudlet (job or task in cloud computing), datacentre and PMs.

Scheduling the cloudlet on VMs and in datacentre is based on two policies
1. Time Shared

2. Space Shared

In Time-Shared scheduling policy, one or more processing elements are allocated to a VM and sharing of elements processing by multiple VMs are allowed. This can be explained as the executing powers are shared such as logical processor and CPU etc. It processes many requests at a time and computing power of that machine is shared, so it results in performance degradation because each other's processing time is affected.
In Space Shared policy one or more elements processing are allocated to a VM, and sharing of elements processing are not allowed .The allocation fails if there is no free elements processing. In Space Sharing refers to memory space sharing such as RAM and Hard Disk etc.

II. Literature Survey

Yousefipour A et al [1] proposed a mathematical model aimed at reducing power consumption and costs by employing an effective VM consolidation in the cloud data center. Subsequently, they proposed a genetic algorithm–based meta-heuristic algorithm, namely, energy and cost-aware VM consolidation for resolving the problem.

Amany Abdelsamea et al [2] proposed the usage of hybrid factors to enhance VM consolidation. Specifically, they developed a multiple regression algorithm that uses CPU utilization, memory utilization and bandwidth utilization for host overload detection. The proposed algorithm, Multiple Regression Host Overload Detection (MRHOD), significantly reduces energy consumption.

Seyed Saeid Masoumzadeh et al [3] proposed a model to concentrate on the VM selection task and proposed a Fuzzy Qlearning (FQL) technique so as to make optimal decisions to select virtual machines for migration. They validated their approach with the CloudSim toolkit using real world PlanetLab workload.

Giuseppe Portaluri et al [6] compared a set of Virtual Machine (VM) allocators for Cloud Data Centers (DCs) that perform the joint allocation of computing and network resources. Tanasak Janpan et al [4] proposed a VM consolidation framework for Apache CloudStack, which is a popular open source cloud platform software suite.

Md Anit Khan et al [7] proposed a novel heuristic Dynamic VM Consolidation algorithm, RTDVMC, which minimizes the energy consumption of CDC through exploiting CSU provided information.

Zoltan Adam Mann [5] investigated the impact of the choice of cloud simulator on the implementation of the algorithms and on the evaluation results.

III. Design and Implementaion

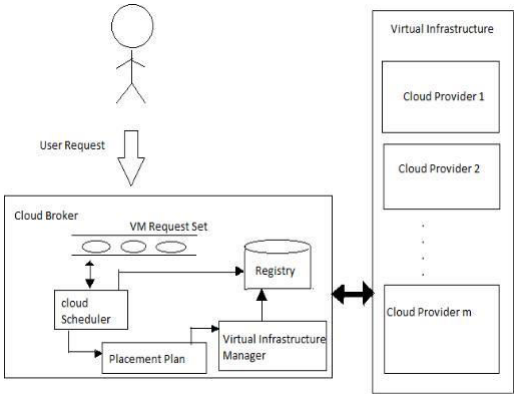
This Chapter discusses the design of the System followed by implementation details. Initially the overall design is discussed followed by the implementation details for each of the modules in the system.

OVERALL DESIGN

The user, cloud provider, and cloud broker are the three major roles in the system model. The cloud broker is the main part of the system model, which handles the delivery and the use of cloud services in-between the user and provider, considering the requirements for the performance. The cloud providers are invisible to the user interface and user with the broker with the help of system transparency provided by the cloud broker instead of communicating with the providers directly.

ARCHITECTURE DIAGRAM

User sends request in form of cloudlets to the broker and the broker assigns it to the cloud provider. The virtual infrastructure manager component asks about any updates on the resource availability to the registry component. From the each cloud provider particulary the registry obtains the needed information. In registry, a new provider registers its own information if it is interested to join the enivornment, which is explained in Figure.



POWER MODEL

The power consumption of data centres is taken from the total power consumption of the physical servers. By considering two states of the server, running and ie static state the power consumption of the each server is calculated. An idle state is represented as the static state where the server is active and no VM exits on the server. The allocation process of VMs is involved in the running state. Therefore, the power consumption of server si in a data centre is Pi, by power function it can be modeled.

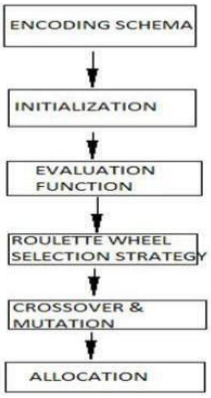
Power = P_idle + P_placement+ P_vm + P_switch

The overall power consumption of server si in a data centre is power, server si in the idle state where the P_idle is the power consumption of it. Power consumed by different VM instances running on the server si is P_placment, P_vm is the outcoming results obtained without processing the workload running a new vm on a server, P_switch is the Consumption of power on system when server switch between on and off states.

LIST OF MODULES

1. ENCODING SCHEMA
2. INITIALIZATION
3. EVALUATION FUNCTION
4. SELECTION STRATEGY
5. CROSS-OVER OPERATOR

6. MUTATION OPERATOR



ENCODING SCHEMA

In genetic algorithm the chromosome is one of the most important element. The performance gets affected by encoding or representation of the chromosome. In our proposed project, there two segments for each chromosome and 2N + L + M-2 is the length of it. For one instance type each VM has to be assigned and to PM on the last stage of the problem each VM has to assigned, and to define the situation the integer coding is used.

The |N + L – 1| genes is the first segment of the chromosome that is the permutation of integer numbers from 1 to N + L – 1. Delimiters are the numbers larger than N. From the beginning of the first delimeter to the array, and encoding schema of the first segment, It indicates which of the VM selected the first instance type for execution. This method of selecting the instance type continues with respect to delimiter. Here N denotes the total number of VMs and L denotes the total number of instance types. If 2 of the delimiters appear consecutively, it shows that none of the VMs select the specified instance type.

INITIALIZATION

Here it generates the initial population randomly by considering the number of servers, number of virtual machines, and number of virtual machine instance types. To reduce the execution of the computation time of GA, the range of values for each gene can be predetermined based on the Integer encoding schema.

EVALUATION FUNCTION

The generated Chromosome will be evaluated against a power and cost function, the higher performance of the chromosome is implied by the lower objective values. To a placement plan the performance of the each chromosome in the population is proportional, the power consumption is determined by the fitness or objective function and cost of placement plan is based on the received information from the chromosome.

=max(**ΣΣ**=1 =1 * – 1.0), ∀ ∈ **{1,2,…,** }

=max(**ΣΣ** =1 =1 * −1.0), ∀ ∈ **{1,2,…,** }

=max(**ΣΣ**=1 =1 * −1.0), ∀ ∈ **{1,2,…,** }

=**Σ**(++) =1 , ∀ ∈ **{1,2,…,** }

F=(0.5*F2)+(1+ *)

Indices	Notations
i	Server index, **i=1,2,…,M**
j	VM instance type index, **j=1,2,…,L**
k	VM index, **k=1,2,…,N**

Parameters

1, if VM k can be done on VM instance type j, else 0

1, if atleast one VM instance type runs on server i, otherwise 0

CPU requests of each VM instance type j

Memory request of each VM instance type j

Disk request of each VM instance type j

CPU capacity of each server i

Memory capacity of each server i

Disk capacity of each server i

SELECTION STRATEGY

In the proposed model, To assigns the probability of each chromosome selection based on their fitness value, the roulette-wheel selection strategy is used.

CROSSOVER OPERATOR

To produce new individuals from 2 parents, from GA the crossover operator is used. The crossover percentage in each generation controls the number of offspring added to this operator. A segment- based crossover operator was employed which was based on single- point crossover. From each segment of the chromosome the gene is selected randomly, and to produce offspring the fragments of each parent is created and they are combined together.

ALGORITHM

INPUT: two parent chromosomes with q genes, = 1 2 … and = 1 2 …

OUTPUT: two offspring chromosomes, = 1 2 … and = 1 2 …

For each segment in chromosome segments do

1. Randomly generate an integer value between 1 and segment size **'r'**

2. R1 – Intersect(**[1…r],[r+1…Segment** Size]);

3. R2 – Intersect(**[1…r],** **[r+1…Segment** Size]);

4. **Find r1 in [1 … r] and replace the value of this** element with r2;

5. **Find r2 in [1 … r] and replace the value of this** element with r1;

6. **[1…Segment** Size] _ concatenate(**[1…r], [r+1…Segment Size]);**

7. **[1…Segment** Size] – concatenate(**[1…r], [r+1…Segment Size]);**

End For Each

Return ,

MUTATION OPERATOR

Like crossover operator, this operator is is utilized to forestall early combination and find another arrangement. However, unlike the crossover operator, this operator usually changes the value of a gene. In each generation, the number of offspring that are added to the population by this operator is determined using a parameter called mutation percentage (MP). In the proposed calculation, a parent initially choose from population at that point 2 genes of each the segment are chosn.At last, to produce another individual, we trade the quality incentive with one another.

ALGORITHM:

INPUT: a chromosome, CH= 1 **2…**

OUTPUT: a mutated chromosome, **′= 1′ 2′… ′** For Each segment in chromosome segments do

1. ′ _ CH;

2. Randomly generate two integer values between 1 and segment size called r1, r2;

3. Exchange the elements in a **′ at indexes r1 and r2** End For Each

Return′;

THE OVERALL PROPOSED ALGORITHM

INPUT: VMList, PMList, InstanceTypeList

OUTPUT: allocation of VMs

1. Initially Pop Size, CP, MP /* num of population, crossover percentage, mutation percentage */

2. 2000 iterations (t>2000) // Termination Condition

3. nc Pop size * CP; // number of offspring

4. nm Pop size * MP; // number of mutants

5. population _ InitializePopulation(Pop Size); /* Generation of initial random population */

6. EvaluatePopulation(population);

7. While the termination condition not true do

8. t_t+1;

9. **FOR i=1…nc /2 // Apply selection and** crossover

10. parents selection(population,2);

11. offsprings Crossover(parents);

12. EvaluatePopulation(offsprings);

13. Population.add(offsprings);

14. End FOR

15. **FOR i=1…nm // Apply** mutation

16. parents selection(population,1);

17. offsprings muatation(parent);

18. EvaluatePopulation(offsprings);

19. Population.add(offsprings);

20. End FOR

21. End While

22. Population.sort(); // sort the individuals according to their fitness

23. Allocation population.get(first); //selection of best individual

24. Return allocation;

EVALUATION METRICS

The objective or fitness function determines the power consumption and cost of placement plan based on information retrieved from the chromosome. The graph is plotted as number of generations verses fitness value, where the fitness value is obtained as Fitness value = Total fitness of the populationnumber of generation (population)

Total fitness of the population = sum of fitness values of all individual chromosomes in a generation

Where,

Individual chromosome fitness value is calculated based on formula’s

=max(**ΣΣ**=1 =1 * **−1.0),**

∀ ∈ {1,2,…, }

=max(**ΣΣ**=1 =1 *

−1.0), **∀ ∈ {1,2,…,** }

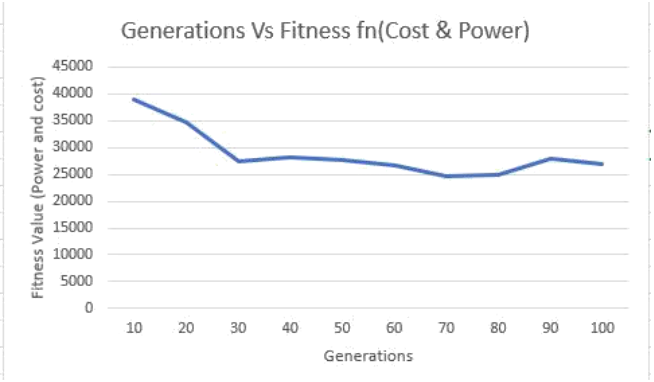
=max(**ΣΣ**=1 =1* **−1.0**

), **∀ ∈ {1,2,…,** }

=**Σ**(++)=1, **∀ ∈**

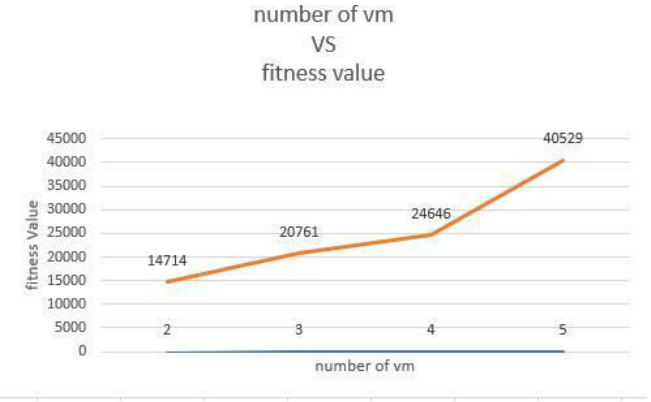
{1,2,…, }

F=(0.5*F2)+(1+ *)



The graph represents the number of generations (population) and its impact on fitness value. X-axis in the graph represents the Number of generation and y-axis represents the fitness value. The number of VM and cloudlet is set to be constant as 4 VM and 4 Cloudlets. From the graph we could infer as the generation (population) increases the fitness value that is the power and cost is reduced.

GRAPH REPRESENTING NUMBER OF VIRTUAL MACHINE [VM] VERSES FITNESS VALUE (POWER & COST)



Here the number of cloudlets is kept to be constant value of 4 and number of generations(population) is kept to be constant of 100, and the system is evaluated varying the number of VM counts, and result was found that the fitness value is increasing in increase of VM count. Image shows the increasing fitness value with respect to increasing VM count.

III. CONCLUSION

In cloud computing, power the board has been concentrated totally with the goal of limiting the absolute force utilization in server farms. To this point, specialists have proposed a variety of approaches

on Green Communications and Networking, 1(4), pp.541-550.

Maede Yavari, Akbar Ghaffarpour Rahbar, Mohammad Hadi Fathi, (2019). Temperature and Energy-aware Consolidation Algorithms in Cloud Computing. Springer, Journal of Cloud Computing: Advances, Systems and Applications.

fluctuate from programming based methods (such as server solidification) to equipment-based solutions (such as DVFS). In this paper, we have focused on software-based optimizations using genetic-algorithm based technique and, in specific, on active Virtual machine placement techniques for the virtual machine solidification. The proposed algorithm uses objective function based on fitness value to evaluate power and cost, which was implemented using the help of CloudSim simulation platform

IV. REFERENCES

Yousefipour, A., Rahmani, A.M. and Jahanshahi, M., (2018). Energy and cost - aware virtual machine consolidation in cloud computing. Software: Practice and Experience, 48(10), pp.1758-1774.

Abdelsamea, A., El-Moursy, A.A., Hemayed, E.E. and Eldeeb, H., (2017). Virtual machine consolidation enhancement using hybrid regression algorithms. Egyptian Informatics Journal, 18(3), pp.161-170.

Masoumzadeh, S.S. and Hlavacs, H., (2013), October. Integrating VM selection criteria in distributed dynamic VM consolidation using Fuzzy Q-Learning. In Network and Service Management (CNSM), 2013 9th International Conference on (pp. 332-338). IEEE.

Janpan, T., Visoottiviseth, V. and Takano, R., (2014), February. A virtual machine consolidation framework for CloudStack platforms. In 2014 International Conference on Information Networking (ICOIN) (pp. 28-33). IEEE.

Mann, Z.Á., (2018). Cloud simulators in the implementation and evaluation of virtual machine placement algorithms. Software: Practice and Experience, 48(7), pp.1368-1389.

Portaluri, G., Adami, D., Gabbrielli, A., Giordano, S. and Pagano, M., (2017). Power consumption-aware virtual machine placement in cloud data center. IEEE Transactions on Green Communications and Networking, 1(4), pp.541-550.

Khan, M.A., Paplinski, A.P., Khan, A.M., Murshed, M. and Buyya, R., (2018, April). Exploiting user provided information in dynamic consolidation of virtual machines to minimize energy consumption of cloud data centers. In Fog and Mobile Edge Computing (FMEC), 2018 Third International Conference on (pp. 105-114). IEEE.