VIRTUAL MACHINE CONSOLIDATION USING ROULETTE WHEEL SELECTION STRATEGY

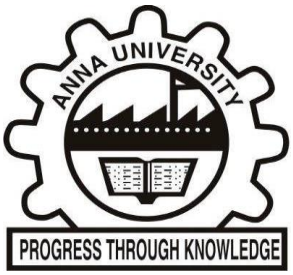
PHASE I REPORT

Submitted by

SAJAN S A

in partial fulfilment for the award of the degree of

MASTER OF ENGINEERING IN COMPUTER SCIENCE



DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

ANNA UNIVERSITY, CHENNAI

NOVEMBER 2018

ANNA UNIVERSITY, CHENNAI

BONAFIDE CERTIFICATE

Certified that this project thesis titled **"VIRTUAL MACHINE CONSOLIDATION USING ROULETTE WHEEL SELECTION STRATEGY"** for the Phase-I of the project is a bonafide work of **SAJAN S A (2017207038)** who carried out the project work under my supervision, for the partial fulfilment of the requirements for the award of the degree of Master of Engineering in Computer Science. Certified further that to the best of my knowledge, the work reported herein does not form part of any other thesis or dissertation based on which a degree or an award was conferred on an earlier occasion on this or any other candidate.

Project Coordinator

Dr. V. MARY ANITA RAJAM

Professor

Department of Computer Science and Engineering

Anna University

Chennai – 600 025

Supervisor

K. LALITHA DEVI

Teaching Fellow

Department of Computer Science and Engineering

Anna University

Chennai – 600 025

Dr. D. Manjula

Professor and Head

Department of Computer Science and Engineering

Anna University, Chennai – 600 025

ஆய்வுசுருக்கம்

கிளவுட் கம்ப்யூட்டிங் தரவு மையம் அதிக எண்ணிக்கையிலான எண்ணிக்கையிலும், உயர்ந்த-பதிலளிக்க கம்ப்யூட்டிங் மற்றும் பாரிய சேமிப்பிற்காக அதிகரித்து வரும் கோரிக்கைகளை பூர்த்தி செய்வதற்கும் வேகமாக வளர்ந்து வருகிறது. இத்தகைய தரவு மையங்கள் அதிக அளவிலான மின்சக்தி எரியூட்டுகின்றன, இதன் விளைவாக அதிக இயக்க செலவுகளும் கார்பன் டை ஆக்சைடு உமிழ்வுகளும் ஏற்படுகின்றன. இந்த மிக உயர்ந்த ஆற்றல் நுகர்வுக்கான காரணம், கணினி வளங்கள் மற்றும் வன்பொருள் திறன் குறைபாடு ஆகியவற்றின் அளவு மட்டுமல்ல, மாறாக இந்த வளங்களின் திறனற்ற பயன்பாட்டில் உள்ளது. மெய்நிகர் இயந்திரம் [VM] ஒருங்கிணைத்தல் என்பது VM இன் நேரடிப் புலம்பெயர்வு, எனவே பூஜ்ஜிய நேரத்திற்கு அருகில் ஒரு இயல்பான சேவையகங்களுக்கிடையே VM ஐ மாற்றியமைக்கும் திறன் உள்ளது. வளங்களைப் பயன்படுத்துவதையும் மேகக்கணி தரவு மையங்களில் ஆற்றல் செயல்திறனை அதிகரிப்பதற்கும் இது ஒரு சிறந்த வழியாகும். VM ஒருங்கிணைப்பு ஹோஸ்ட் ஓவர்லோட் கண்டறிதல், VM தேர்வு மற்றும் VM வேலைவாய்ப்பு ஆகியவற்றைக் கொண்டுள்ளது. எமது முன்மொழிந்த மாதிரியில் நாம் Roulette-Wheel Selection Strategy ஐப் பயன்படுத்தப் போகிறோம், VM ஆனது Roulette-Wheel Selection Mechanism ஐ பயன்படுத்தி Instance வகை மற்றும் பிசிகல் மெஷின் [PM] ஐ தேர்ந்தெடுக்கிறது.

ABSTRACT

Cloud computing data centres are growing rapidly in both number and capacity to meet the increasing demands for highly-responsive computing and massive storage. Such data centres consume enormous amounts of electrical energy resulting in high operating costs and carbon dioxide emissions. The reason for this extremely high energy consumption is not just the quantity of computing resources and the power inefficiency of hardware, but rather lies in the inefficient usage of these resources. Virtual Machine [VM] consolidation involves live migration of VMs hence the capability of transferring a VM between physical servers with a close to zero down time. It is an effective way to improve the utilization of resources and increase energy efficiency in cloud data centres. VM consolidation consists of host overload/underload detection, VM selection and VM placement. 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.

ACKNOWLEDGEMENT

I would like to express my sincere thanks and deep sense of gratitude to my guide, **K. LALITHA DEVI,** Teaching Fellow, Department of Computer Science and Engineering. She has been a constant source of inspiration and I thank her for providing me with the necessary counsel and direction to help me complete this project.

My sincere thanks to **Dr. D. MANJULA,** Professor and Head of Department, Department of Computer Science and Engineering for her kind support and for providing necessary facilities to carry out the work.

I wish to record my sincere thanks to the members of the review panel, **Dr. ARUL SIROMONEY,** Professor, Department of Computer Science and Engineering, **Dr. V. MARY ANITA RAJAM,** Professor, Department of Computer Science and Engineering, **Dr. S. SUDHA,** Assistant Professor (Sl. Gr), Department of Computer Science and Engineering for their valuable suggestions and critical reviews throughout the course.

Finally, I would like to thank my parents and friends for their patience, cooperation and moral support throughout my life and especially during the hardships faced during the phase of this project.

(SAJAN S A)

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

LIST OF ABBREVIATIONS

ABBREVIATIONS	MEANING
Et al.	And others
Etc.	And other things
VM	Virtual Machine
PM	Physical Machine
NP	Non-deterministic Polynomial
VMC	Virtual Machine Consolidation
RAM	Random Access Memory
CIS	Cloud Information Services
CDC	Cloud Data Centre
GA	Genetic Algorithm
CPU	Central Processing Unit
CH	Chromosome
MIPS	Million Instructions per Second
DVFS	Dynamic Voltage Frequency Shifting
CSU	Cloud Service User

CHAPTER 1

INTRODUCTION

This chapter discusses the objective of the project, the problems to be solved, purpose, scope and challenges to be solved by the project.

1.1 OVERVIEW

Virtual Machine (VM) consolidation is an effective technique to improve resource utilization and reduce energy footprint in cloud data centres. It can be implemented in a centralized or a distributed fashion. Virtual Machine (VM) consolidation is one of the key mechanisms of designing an energy-efficient dynamic Cloud resource management system. 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, packing more VMs into a single server may lead to poor Quality of Service (QoS), since VMs share the underlying physical resources of the PM. To address this, VM Consolidation (VMC) algorithms are designed to dynamically select VMs for migration by considering the impact on QoS in addition to the abovementioned optimization objectives. VMC is a NP-Hard problem and hence, a wide range of heuristic and meta-heuristic VMC algorithms have been proposed that aim to achieve near-optimality by Researchers.

Cloudsim Setup: - CloudSim is a framework written in Java Programming language is a toolkit for modelling and simulation of cloud computing environments and evaluation of resource provisioning algorithms. CloudSim developed by Buyya began in 2009 in the laboratory of Grid Computing and Distributed Systems

(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, allocates one or more processing elements to a VM and allows sharing of processing elements by multiple VMs. This can be explained as sharing of executing powers such as CPU, logical processor etc. It processes several requests at a time and shares the computing power of that machine, so they affect each other's processing time which results in performance degradation.

In Space Shared policy allocates one or more processing elements to a VM, and doesn't allow sharing of processing elements. If there are no free processing elements then the allocation fails. In other words, Space Sharing refers to sharing of memory space such as hard disk, RAM etc.

1.2 PROBLEM STATEMENT

Cloud computing data centres are growing rapidly in both number and capacity to meet the increasing demands for highly-responsive computing and massive storage. Such data centres consume enormous amounts of electrical energy resulting in high operating costs and carbon dioxide emissions. The reason for this extremely high energy consumption is not just the quantity of computing resources and the power inefficiency of hardware, but rather lies in the inefficient usage of these resources. Virtual Machine [VM] consolidation involves live migration of VMs hence the capability of transferring a VM between physical servers with a close to zero down time. It is an effective way to improve the utilization of resources

and increase energy efficiency in cloud data centres. VM consolidation consists of host overload/underload detection, VM selection and VM placement. 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.

1.3 OBJECTIVE OF THE PROJECT

- Is to show how genetic algorithms can be used in solving VM consolidation problem.
- To understand how genetic algorithm works and to find the best algorithm in terms of power and cost utilization.

1.4 ORGANIZATION OF REPORT

Rest of the report is organized as follows, the second chapter deals with literature survey for the thesis which is followed by the third chapter containing detailed design and implementation details. The fourth chapter provides detailed result analysis followed by future work and conclusion in fifth chapter.

CHAPTER 2

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.

CHAPTER 3

DESIGN AND IMPLEMENTATION

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.

3.1 OVERALL DESIGN

There are 3 major roles in the system model including user, cloud provider, and cloud broker. The cloud broker is the main part of the system model, Which acts as an intermediary between the user and providers to handle the use and delivery of cloud services, considering the performance requirements. The cloud broker can provide the system transparency in which the cloud providers are invisible to the users and a user interacts with the broker instead of communicating directly with the providers.

3.2 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 periodically asks the registry component about any updates on the availability of resources. The registry obtains the required information from each particular cloud provider. A new provider interested in joining the environment, registers its own information in the registry, which is explained in Fig 3.1.

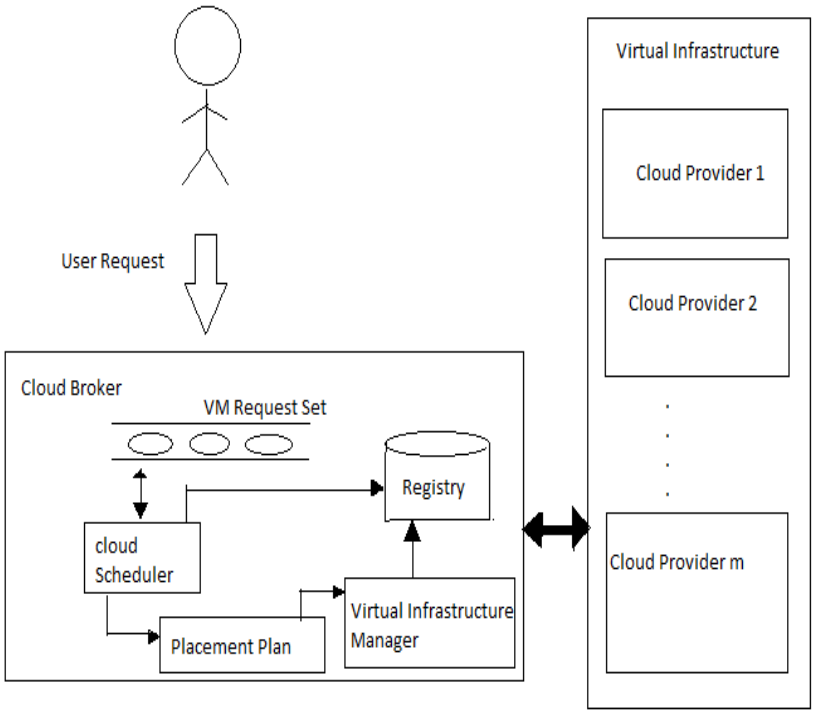


Figure 3.1 Architecture diagram

POWER MODEL

The total power consumption of physical servers is considered as the power consumption of data centres. The power consumption of each server is calculated by considering 2 states of the server, ie static and running state. The static state represents an idle state in which no VM exits on the server and the server is active. The running state involves the allocation process of VMs. Therefore, Pi is the power consumption of a server si at a data centre that can be modelled by a power function as

$$Power = P\_idle + P\_placement + P\_vm + P\_switch$$

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

3.3 LIST OF MODULES

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

3.3.1 ENCODING SCHEMA

One of the most important elements of the Genetic Algorithm is the chromosome. Representation or encoding of the chromosome affects the performance. In our proposed project, each chromosome consists of 2 segments and the length of it is 2N + L + M-2. Since each VM has to be assigned to one instance type and each VM has to be assigned to PM on the last stage of the problem, integer encoding was used to define this situation.

The first segment of the chromosome is comprised of |N + L – 1| genes that is the permutation of integer numbers from 1 to N + L – 1. Numbers larger than N are used as delimiters. Hence the decoding schema of the first segment, from the beginning of the array to first delimiter, 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.

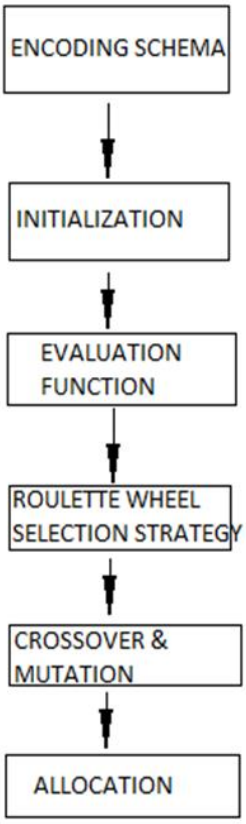


Figure 3.2 List of Modules

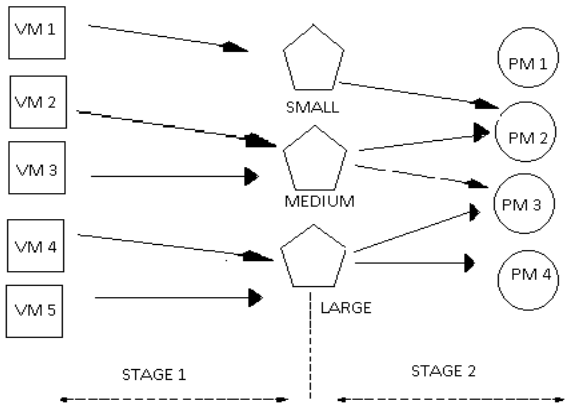


Figure 3.3 A sample of VM allocation problem

STAGE 1: VM CHOOSE THE INSTANCE TYPE

STAGE 2: VM CHOOSE THE PHYSICAL MACHINE [PM]

3.3.2 INITIALIZATION

The initial population is randomly generated by considering the number of servers, number of virtual machines, and number of virtual machine instance types. In order to reduce the computation time of GA execution, the range of values for each gene can be predetermined based on the Integer encoding schema.

3.3.3 EVALUATION FUNCTION

The generated Chromosome will be evaluated against a power and cost function, lower objective value implies the higher performance of the chromosome.

The performance of each chromosome in the population is proportional to a placement plan, the objective or fitness function determines the power consumption and cost of placement plan based on information received from the chromosome.

$$v_i^{cpu}=\max\left(\frac{\sum_{j=1}^{L}\sum_{k=1}^{N} R_j^{CPU} Xijk}{Yi * C_i^{CPU}} - 1.0\right), \qquad \forall i \in \{1,2,\dots,M\}$$

$$v_i^{Memory}=\max\left(\frac{\sum_{j=1}^{L}\sum_{k=1}^{N} R_j^{Memory} Xijk}{Yi * C_i^{Memory}} - 1.0\right), \qquad \forall i \in \{1,2,\dots,M\}$$

$$v_i^{Disk}=\max\left(\frac{\sum_{j=1}^{L}\sum_{k=1}^{N} R_j^{Disk} Xijk}{Yi * C_i^{Disk}} - 1.0\right), \qquad \forall i \in \{1,2,\dots,M\}$$

$$v_T=\frac{\sum_{i=1}^{M}\left(v_i^{CPU}+v_i^{Memory}+v_i^{Disk}\right)}{M}, \qquad \forall i \in \{1,2,\dots,M\}$$

F= (0.5 * F2) + (1 + β * v_T)

Table 3.1 Notations used in the model

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	
$Xijk$	1, if VM k can be done on VM instance type j, else 0
Yi	1, if atleast one VM instance type runs on server i, otherwise 0
R_j^{CPU}	CPU requests of each VM instance type j
R_j^{Memory}	Memory request of each VM instance type j
R_j^{Disk}	Disk request of each VM instance type j
C_i^{CPU}	CPU capacity of each server i
C_i^{Mem}	Memory capacity of each server i
C_i^{Disk}	Disk capacity of each server i

3.3.4 SELECTION STRATEGY

In the proposed model, we use the roulette-wheel selection strategy that assigns the probability of selecting each chromosome according to their fitness value.

3.3.5 CROSSOVER OPERATOR

The crossover operator in GA is used to produce new individuals from 2 parents. In each generation, the number of offspring that are added to the population by this operator is controlled by a crossover percentage. We employed a segment-based crossover operator that was based on a single- point crossover. At first a gene is selected randomly in each segment of the chromosome, and then the created fragments of each parent to produce offspring are combined together.

ALGORITHM

INPUT: two parent chromosomes with q genes, $CH^b = a_1^b a_2^b \ldots a_q^b$ and $CH^c = a_1^c a_2^c \ldots a_q^c$

OUTPUT: two offspring chromosomes, $CH^d = a_1^d a_2^d \ldots a_q^d$ and $CH^e = a_1^e a_2^e \ldots a_q^e$

For each segment in chromosome segments do

1. Randomly generate an integer value between 1 and segment size ‘r’
2. R1← Intersect(CH^b[1…r], CH^c[r+1…Segment Size]);
3. R2← Intersect(CH^c[1…r], CH^b[r+1…Segment Size]);
4. Find r1 in CH^b[1 … r] and replace the value of this element with r2;
5. Find r2 in CH^c[1 … r] and replace the value of this element with r1;

6. CH^d[1…Segment Size]← concatenate(CH^b[1…r], CH^c[r+1…Segment Size]);
7. CH^e[1…Segment Size]← concatenate(CH^c[1…r], CH^b[r+1…Segment Size]);

End For Each

Return CH^d, CH^e

3.3.6 MUTATION OPERATOR

Like the crossover operator, this operator is used to prevent early convergence and discover a new solution. 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 algorithm, a parent first chooses from population randomly, then 2 genes of each segment are randomly selected. Finally, to generate a new individual, we exchange the gene value with each other.

ALGORITHM:

INPUT: a chromosome, CH=$a_1 a_2 \ldots a_q$

OUTPUT: a mutated chromosome, CH'=$a'_1 a'_2 \ldots a'_q$

For Each segment in chromosome segments do

1. CH'← CH;

2. Randomly generate two integer values between 1 and segment size called r1, r2;
3. Exchange the elements in a CH' at indexes r1 and r2

End For Each

Return CH';

3.3.7 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;
```

3.4 TEST CASES

Table 3.2 Test Cases

S.No	Module Name	Input	Expected Output
1	Crossover	Two parent chromosomes A and B	Two offspring's with gene get shifted from one chromosome to another after segment size ‘r’.
2	Mutation	A chromosome ‘CH’	Exchange the elements of chromosome, at randomly generated two points r1 and r2 (1<=r1, r2<=segment size).
3	Overall Roulette Wheel Algorithm	VMList, PMList, CloudletList	Allocation of VMs

3.5 EXPERIMENTAL RESULT

3.5.1 CODE

3.5.1.1 Initialization

The first generation of the random population is generated from a string which is generated as the result of integer encoding. Figure 3.4 shows the code regarding the initialization of first generation of the random population.

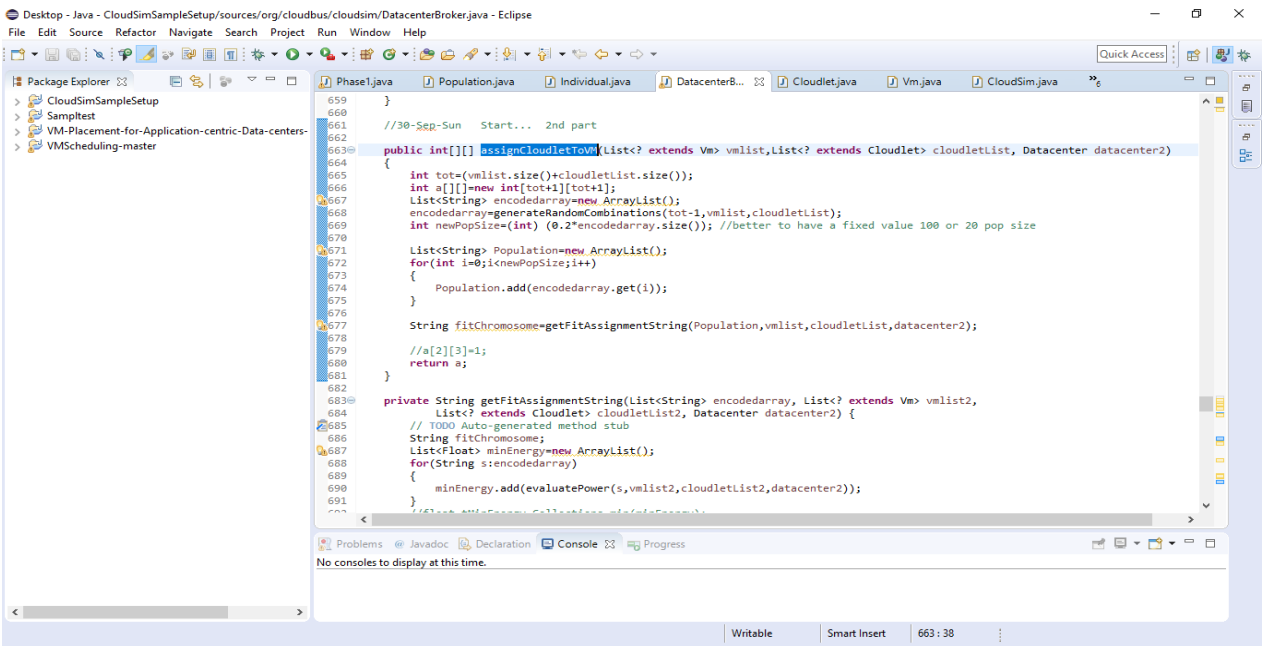


Figure 3.4 Initialization

3.5.1.2 Integer Encoding

An encoded string is generated based on the number of VM and number of instance type. Figure 3.5 shows the code regarding the generation of integer encoded string.

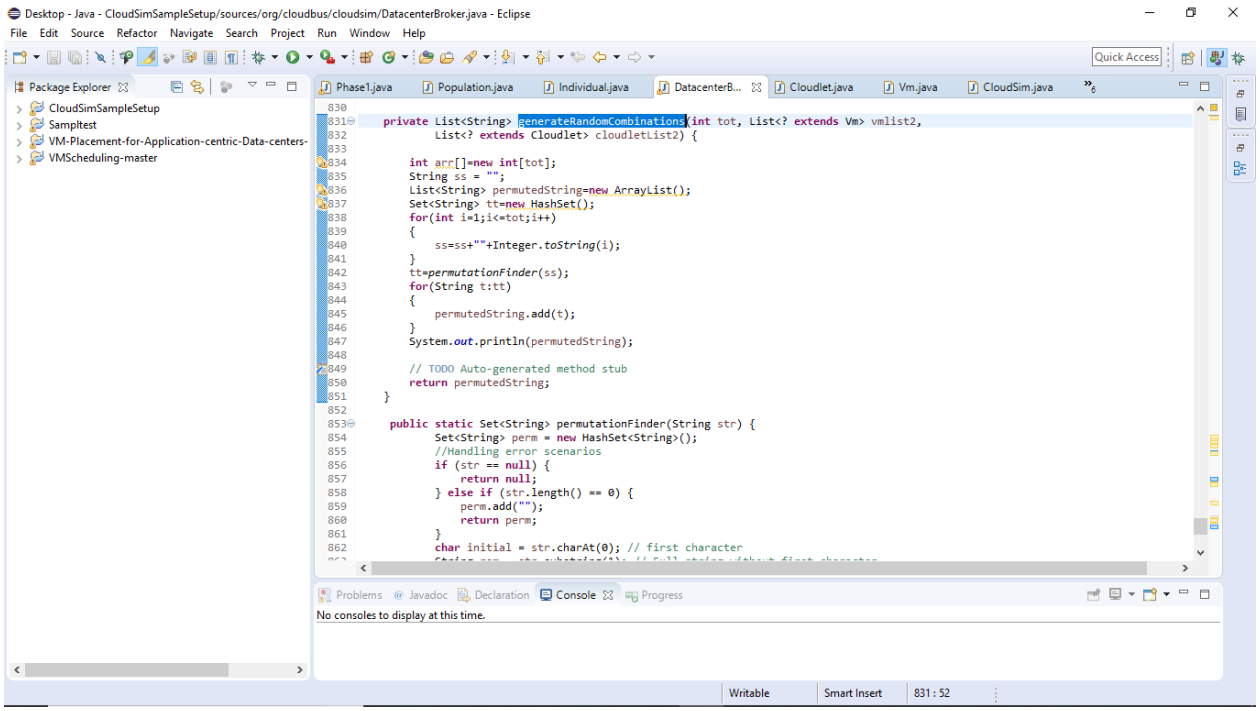


Figure 3.5 Integer encoding

3.5.1.3 Power Evaluation

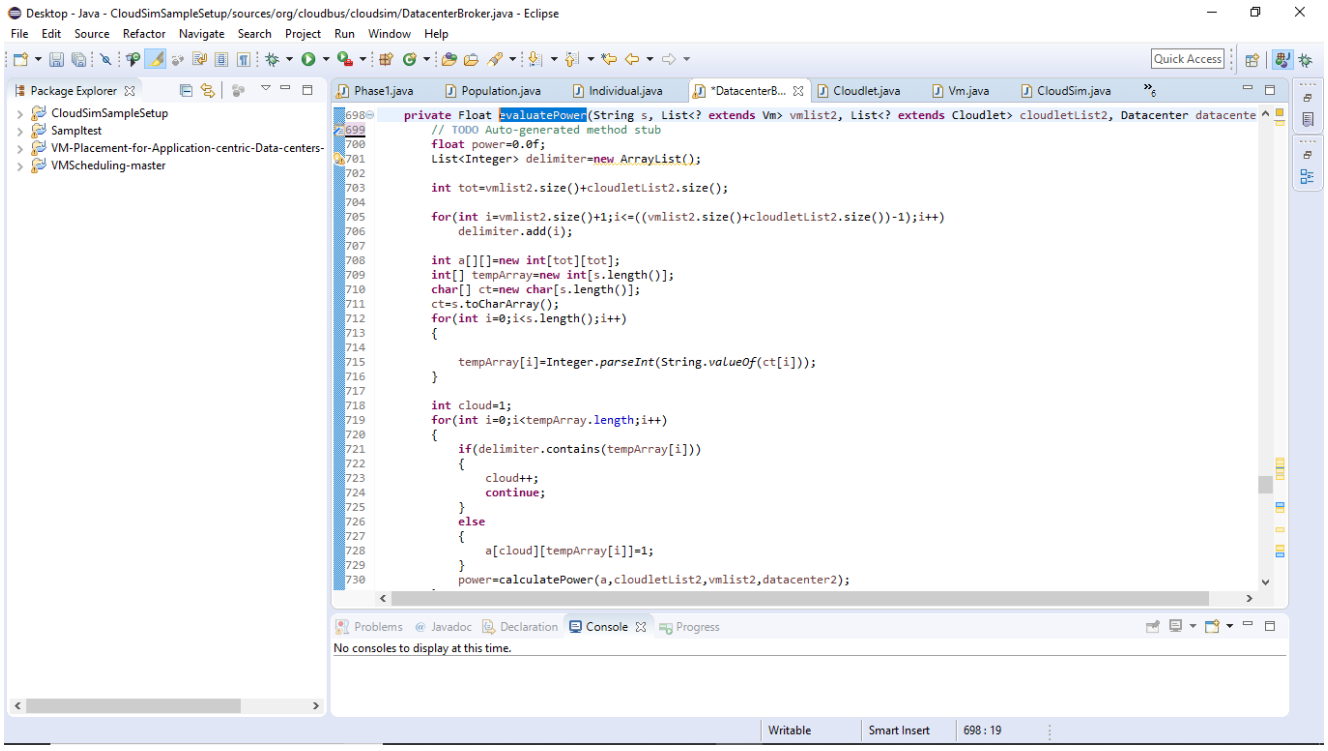


Figure 3.6 Power Evaluation

Figure 3.6 deals with Power Evaluation on how power is calculated for each string of chromosomes, it splits the chromosome based on delimiter and sends to power calculation along with VM, cloudlet and datacentre.

3.5.1.4 Power Calculation

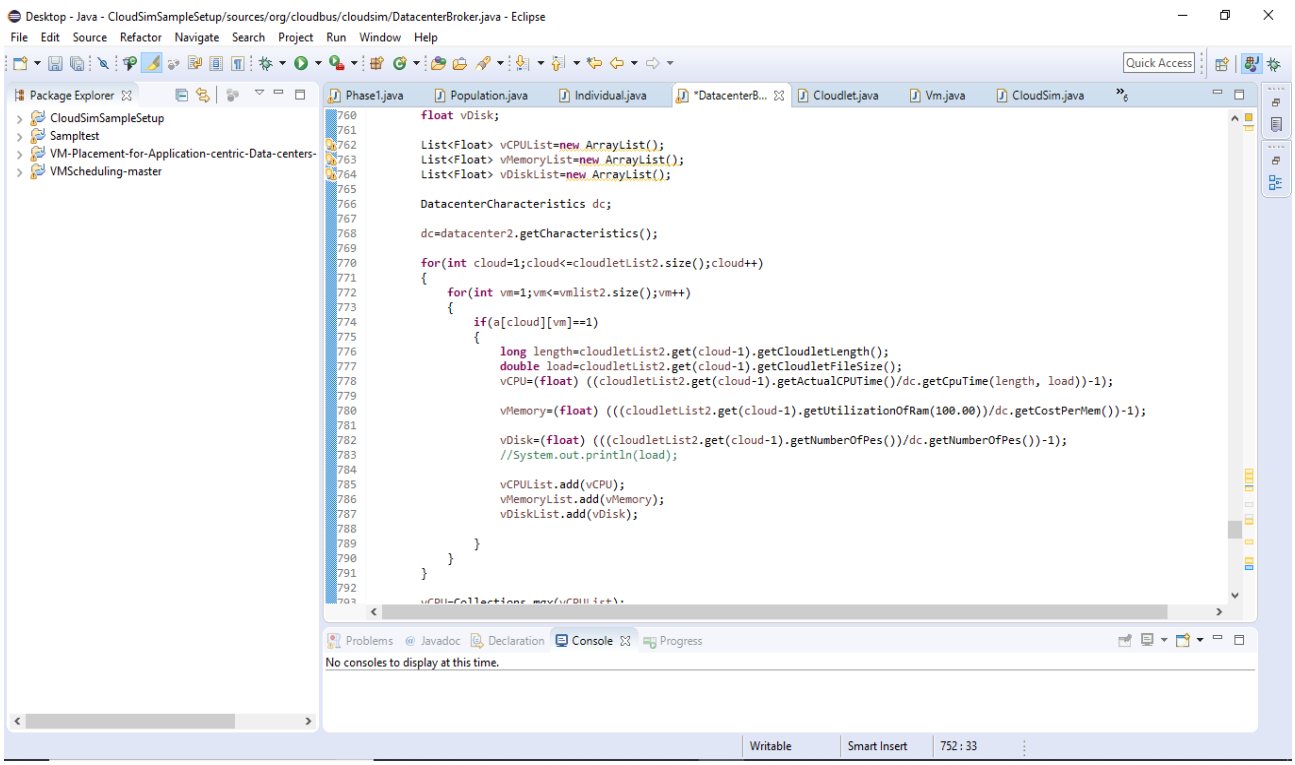


Figure 3.7 Power Calculation

Figure 3.7 shows the code regarding how Power Calculation is used to calculate the power of a VM running on a cloudlet as discussed on Evaluation Function 3.3.3.

3.5.1.5 Roulette Wheel selection:

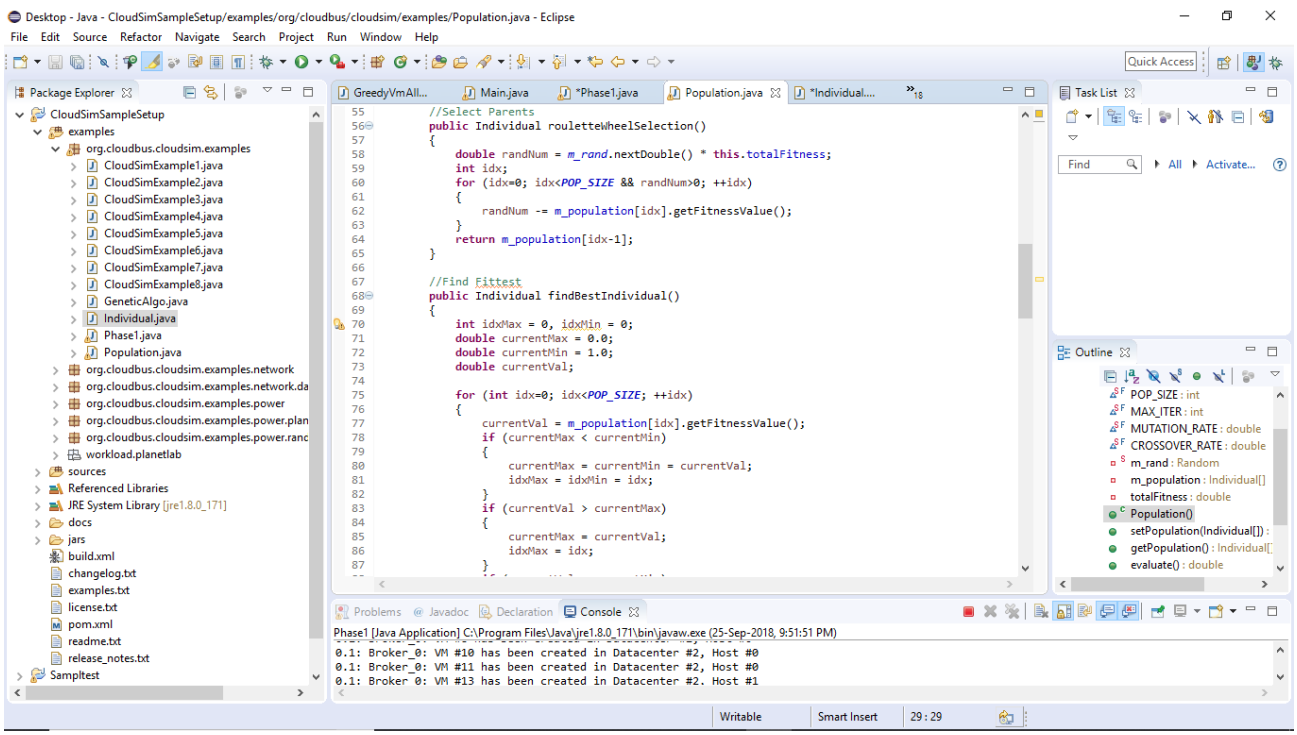


Figure 3.8 Roulette Wheel Selection

Figure 3.8 shows how to perform Roulette wheel selection from entire population set on which crossover and mutation is applied and carried over to next new population set.

3.5.1.6 Crossover Operator:

Figure 3.9 shows how to do crossover on two chromosomes, here we choose a random point based on which crossover is performed.

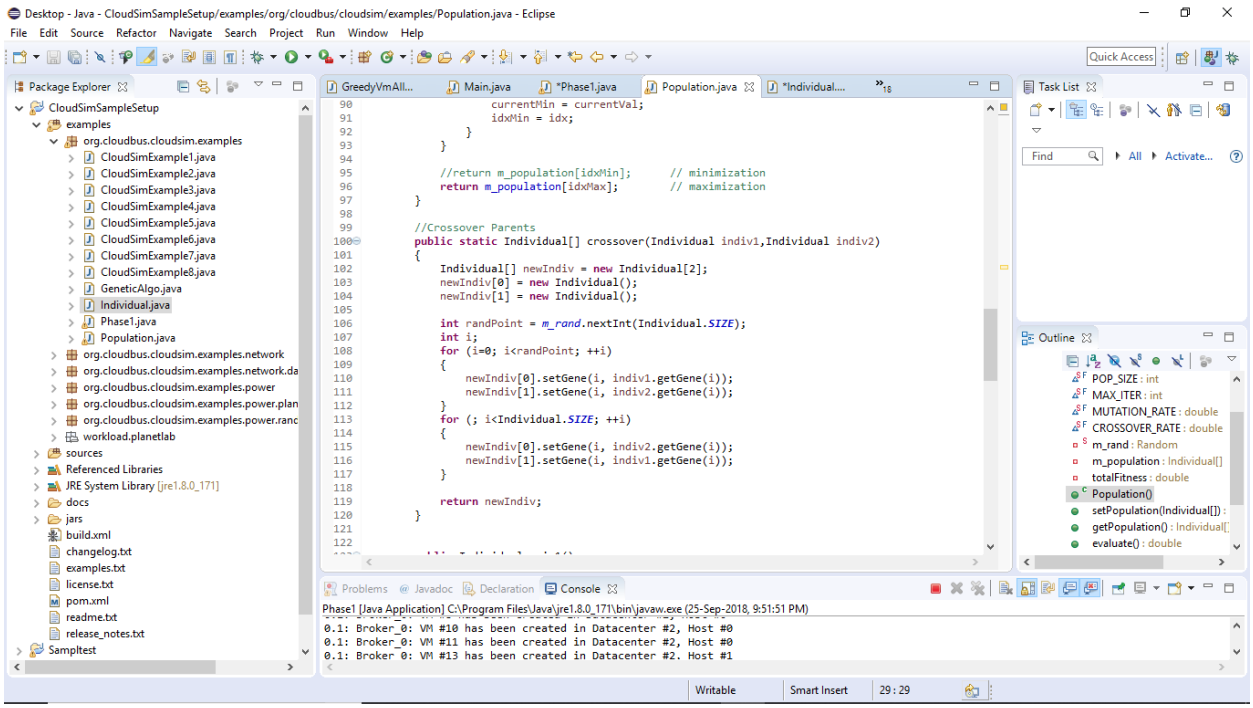


Figure 3.9 Crossover operator

3.5.1.7 Mutation Operator:

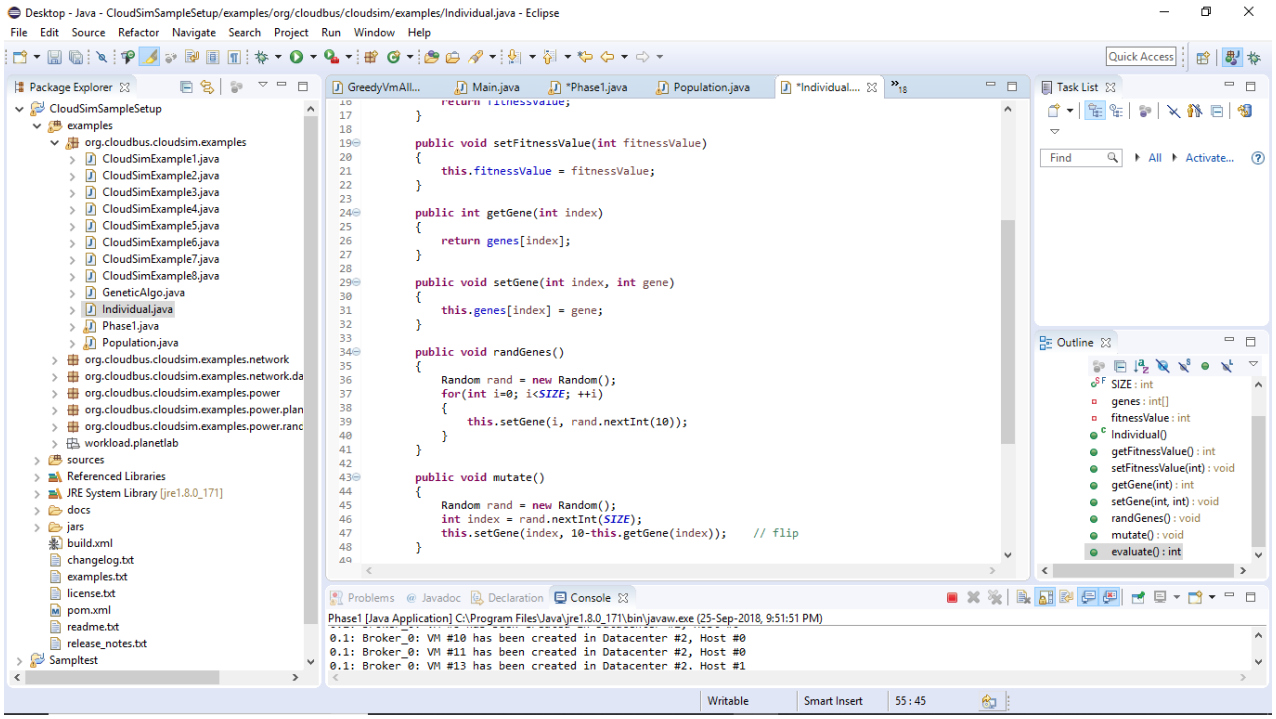


Figure 3.10 Mutation Operator

Figure 3.10 shows the mutation of single random gene for a given chromosome.

3.5.2 OUTPUT

Simulation configuration

The characteristics of the VM, Host and cloudlet used in the simulation are detailed below:-

VM requirements(Each host has a VM)

MIPS rating: 250

Image Size on Disk: 10000 MB

RAM: 512 MB

Band width: 1000

Number of Requirement Pes: 1

Hypervisor: Xen

Characteristics of Host (each datacentre has a host)

RAM: 200000

Storage: 1000000

Bandwidth: 100000

MIPS rating of processing entity: 1000

Characteristics of the cloudlets

Length (in terms of instruction):40000

Input file size: 300

Output file size: 300

The cloudsim 3.0.3 toolkit is used for simulation and measuring the performance of large scale virtualized datacentre. The simulation is done with 3 datacentre, 4 host and 4 VM.

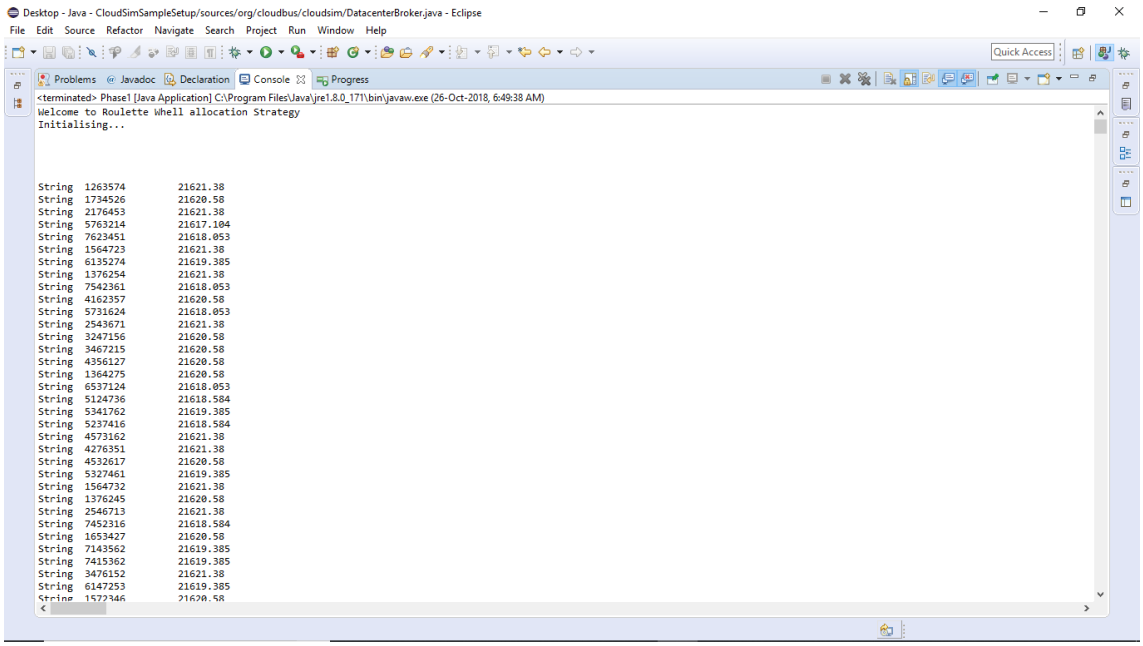


Figure 3.11 Initial Population with its Power value

Figure 3.11 shows how initial population set is generated with different chromosome values and its corresponding power values which is calculated from power calculation function.

Figure 3.12 shows how the best chromosome returned by the system and how the allocation of VM to cloudlet is done.

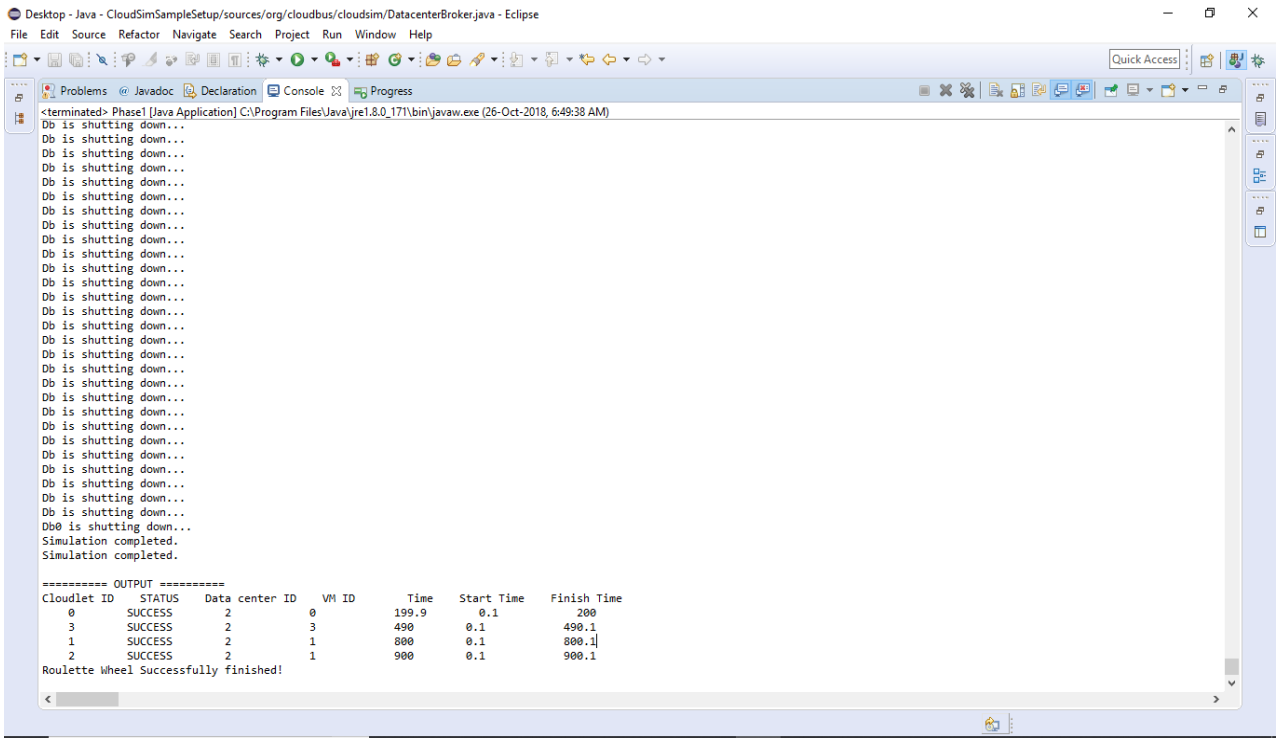


Figure 3.12 Final allocation of VM to Cloudlets

CHAPTER 4

RESULT ANALYSIS AND DISCUSSION

This Chapter deals with analysis of results. First the metrics to be considered for evaluation of the modules are discussed and then analyse the performance of the system based on the values obtained.

4.1 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

$$\text{Fitness value} = \frac{\text{Total fitness of the population}}{\text{number of generation (population)}}$$

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

$$v_i^{cpu} = \max\left(\frac{\sum_{j=1}^{L}\sum_{k=1}^{N} R_j^{CPU} Xijk}{Yi * C_i^{CPU}} - 1.0\right), \qquad \forall i \in \{1,2,\dots,M\}$$

$$v_i^{Memory} = \max\left(\frac{\sum_{j=1}^{L}\sum_{k=1}^{N} R_j^{Memory} Xijk}{Yi * C_i^{Memory}} - 1.0\right), \qquad \forall i \in \{1,2,\dots,M\}$$

$$v_i^{Disk} = \max\left(\frac{\sum_{j=1}^{L}\sum_{k=1}^{N} R_j^{Disk} Xijk}{Yi * C_i^{Disk}} - 1.0\right), \qquad \forall i \in \{1,2,\dots,M\}$$

$$v_T = \frac{\sum_{i=1}^{M}\left(v_i^{CPU} + v_i^{Memory} + v_i^{Disk}\right)}{M}, \qquad \forall i \in \{1,2,\dots,M\}$$

F= (0.5 * F2) + (1 + β * v_T)

Refer Table 3.1 for notations, here F is the fitness Value

4.2 GRAPH REPRESENTING GENERATIONS VS FITNESS VALUE

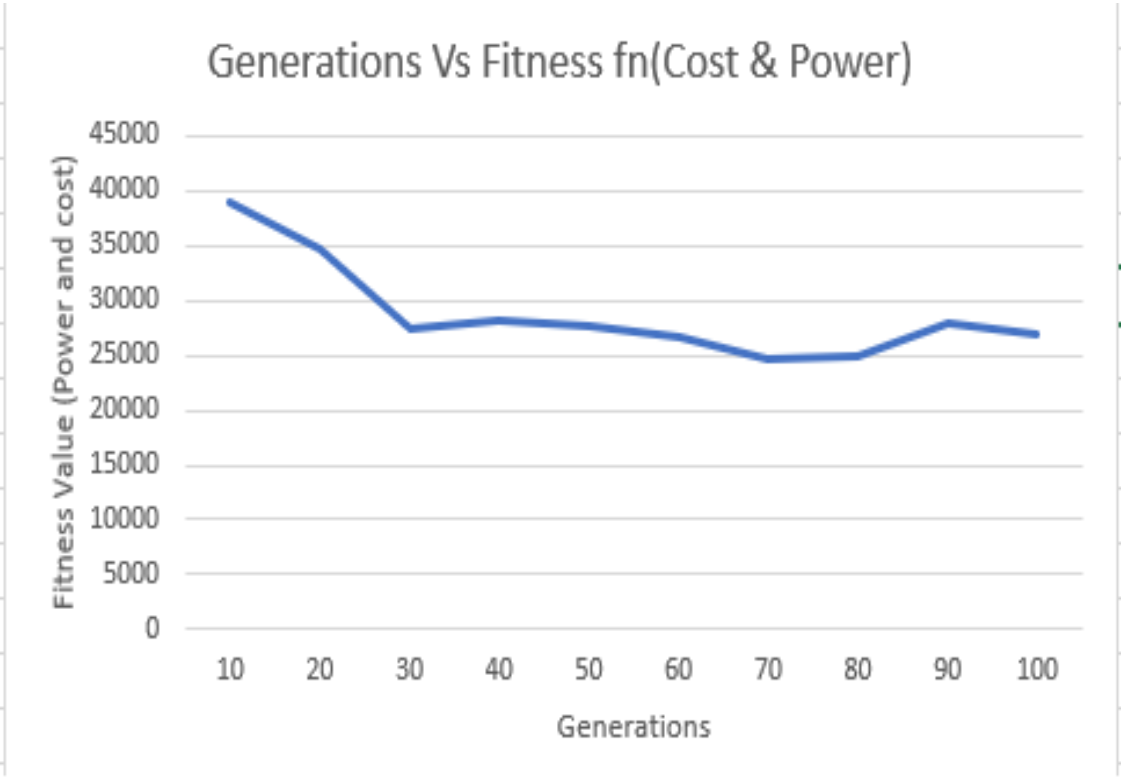


Figure 4.1 GENERATIONS VS FITNESS VALUE

The graph represents the number of generations (population) and its impact on fitness value. X-axis in the graph (figure 4.1) 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.

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

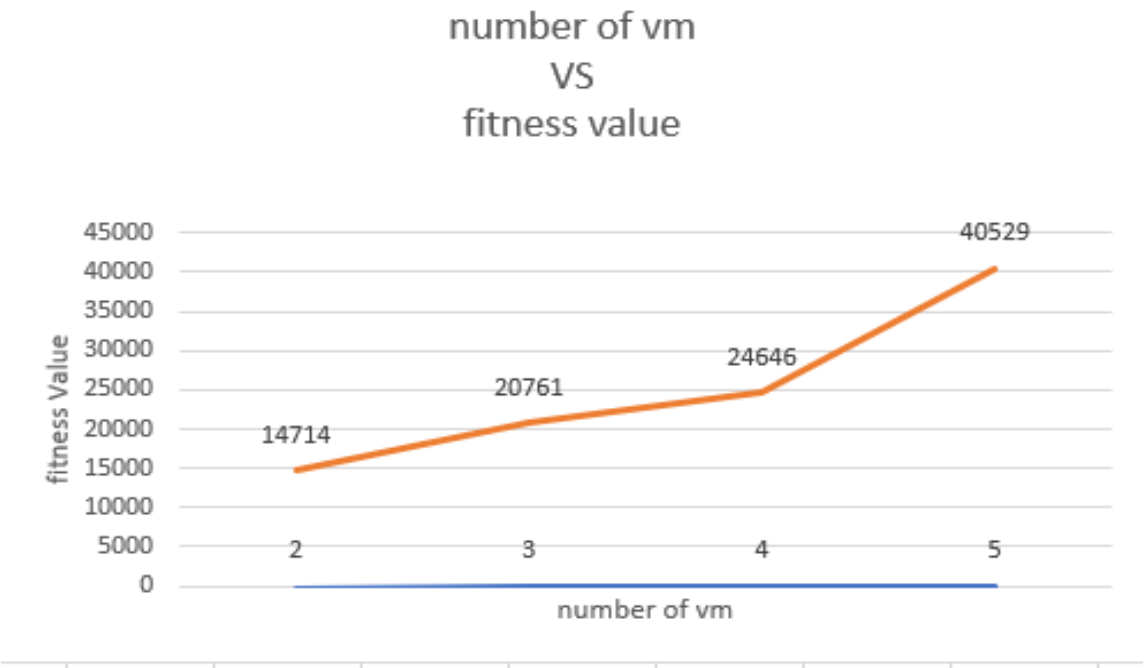


FIGURE 4.2 NUMBER OF VM VS FITNESS VALUE

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. Figure 4.2 shows the increasing fitness value with respect to increasing VM count.

CHAPTER 5

CONCLUSION AND WORK SCHEDULE FOR PHASE II

This chapter discusses the results of the project, the limitations and the future work to be carried out.

5.1 CONCLUSION

In cloud computing, power management has been studied extensively with the objective of minimizing the total power consumption in data centers. To this aim, researchers have proposed a wide range of approaches ranging from software-based techniques (such as server consolidation) to hardware-based solutions (such as DVFS). In this paper, we focused on software-based optimizations using genetic-algorithm based technique and, in particular, on effective VM placement techniques for VM consolidation. 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.

5.2 WORK SCHEDULE FOR PHASE II

Future work is to carry out binding of Data Centres and VM using Genetic algorithm for the given data centres, cloudlet and VM. Binding of VM and cloudlets & VM and Data Centres will provide much efficient results in terms of power consumption when compared to only binding of VM and cloudlets, as Data Centres also play a major role in the total power consumption, moreover different selection strategies other than roulette wheel selection strategy can also be implemented to checkout which would produce good results in terms of power consumption.

REFERENCES

1. 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.

2. 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.

3. 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.

4. 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.

5. 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.

6. 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.

7. 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.