Dynamic Consolidation of Virtual Machines for Energy Efficient Management of Cloud Data Centers

Md Anit Khan

Master of Information Technology

A thesis submitted for the degree of *Doctor of Philosophy* at IT

Monash University in 2019

Faculty of Information Technology

Copyright notice

© Md Anit Khan (*2019*).

I certify that I have made all reasonable efforts to secure copyright permissions for third-party content included in this thesis and have not knowingly added copyright content to my work without the owner's permission.

Declaration

This thesis is an original work of my research and contains no material which has been accepted for the award of any other degree or diploma at any university or equivalent institution and that, to the best of my knowledge and belief, this thesis contains no material previously published or written by another person, except where due reference is made in the text of the thesis.

Signature:

Print Name: Md Anit Khan

Date: ………………………….

Publications during enrolment

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

- *Khan MA, Paplinski A, Khan AM, Murshed M, Buyya R. Dynamic Virtual Machine Consolidation Algorithms for Energy-Efficient Cloud Resource Management: A Review. Sustainable Cloud and Energy Services: Springer; 2018. p. 135-65.*

Thesis including published works declaration

I hereby declare that this thesis contains no material which has been accepted for the award of any other degree or diploma at any university or equivalent institution and that, to the best of my knowledge and belief, this thesis contains no material previously published or written by another person, except where due reference is made in the text of the thesis.

This thesis includes (2) original papers published in peer reviewed conference and (1) published book chapter publications. The core theme of the thesis is (insert theme). The ideas, development and writing up of all the papers in the thesis were the principal responsibility of myself, the student, working within the Faculty of Information and Technology under the supervision of (insert name of supervisor). (The inclusion of co-authors reflects the fact that the work came from active collaboration between researchers and acknowledges input into team-based research.)

In the case of (2 & 3) my contribution to the work involved the following:

Thesis Chapter	**Publication Title**	**Status** *(published, in press, accepted or returned for revision, submitted)*	**Nature and % of student contribution**	**Co-author name(s) Nature and % of Co-author's contribution***	**Co-author(s), Monash student Y/N***
2	*Dynamic Virtual Machine Consolidation Algorithms for Energy-Efficient Cloud Resource Management: A Review.*	*Accepted*	*60%. Concept and collecting data and writing first to final draft*	*1) Andrew P Paplinski, guidence with contents and review the entire writing, 10%* *2)Abdul Malik Khan, update contents and review the entire writing, 10%* *3) Manzur Murshed, Update Abstract and teaching Basic Concepts Related to VM Consolidation, 10%* *4)Rajkumar Buyya, Guidance with Writing, Selecting where to Publish 10%.*	*Yes* *No* *No* *No* *No*
3	*Exploiting user provided information in dynamic consolidation of virtual machines to minimize energy consumption of cloud data centers*	*Accepted*	*60%. Concept, Algorithm and novel thinking and collecting data and writing first to final draft*	*1) Andrew P Paplinski, guidence with contents and review the entire writing, 10%* *2) Abdul Malik Khan, update contents and review the entire writing, 10%* *3) Manzur Murshed, Teaching Basic Concepts Related to VM Consolidation, Help with Core Idea of the Paper*	*Yes* *No* *No* *No* *No*

				and Guidence with Simulation, 10% 4) *Rajkumar Buyya, Guidance with Writing and Experiments, 10%.*	

I have renumbered sections of submitted or published papers in order to generate a consistent presentation within the thesis.

Student name: Md Anit Khan

Student signature: Date:

I hereby certify that the above declaration correctly reflects the nature and extent of the student’s and co-authors’ contributions to this work. In instances where I am not the responsible author I have consulted with the responsible author to agree on the respective contributions of the authors.

Main Supervisor name: Abdul Malik Khan

Main Supervisor signature: Date:

Acknowledgements

I am ever-indebted and ever-thankful to almighty Allah, as he has fulfilled my wish by giving me the opportunity to undertake this research in such a world-renowned university as Monash University and assisted me all the way to accomplish it. The person I am mostly indebted in my life is my late mom who had a dream to see his son complete his PhD. Only thing she ever wanted in her life is her son's success, as she had sacrificed almost everything she possibly could in her life for that. Next, I would like to express my love to my dad. Almighty has showered me with his love and support through my parents.

I can never give my supervisors Dr. Abdul Malik Khan, Assoc. Professor Andrew P Paplinski, Professor Manzur Murshed and Professor Rajkumar Buyya enough thanks for being such strong supporters of mine throughout this journey. Without their love, kindness, care, effort, hard work, research directions, professional guidance, insightful advice, motivation, constant assistance and endless patience from the very beginning till the completion of this thesis, I would not be able to complete this thesis. In my eyes, my supervisors are the best supervisors.

My heartfelt thanks to my wife for her enormous sacrifice and profound affection. Without her hard work and support, this dissertation could not have reached this stage.

Last but not the least, I feel grateful from my heart to Monash University for providing me this research and scholarship opportunities, an excellent environment and all sorts of support to accomplish this research.

Acronyms

VM	Virtual Machine
PM	Physical Machine
CSU	Cloud Service User
CSP	Cloud Service Provider
CDC	Cloud Data Center
IAAS	Infrastructure as a Service
PAAS	Platform as a Service
SAAS	Software as a Service
SLA	Service Level Agreement
QoS	Quality of Service
DPM	Dynamic Power Management
CRMS	Cloud Resource Management System
O-UPM	Over-Utilized PM
U-UPM	Under-Utilized PM
SVMC	Static VM Consolidation
DVMC	Dynamic VM Consolidation
VMRT	VM Release Time
PMRT	PM Release Time
CDVMC	Centralized Dynamic VM Consolidation
DDVMC	Distributed Dynamic VM Consolidation
STDVMC	Static Threshold based Dynamic VM Consolidation
ATDVMC	Adaptive Threshold based Dynamic VM Consolidation
NPDVMC	Non-Predictive Dynamic VM Consolidation
PDVMC	Predictive Dynamic VM Consolidation
RPS	Random PM Selection
RS	Random Selection
RC	Random Choice
MMT	Minimum Time to Migration
MVM	Minimization of VM Migration
HPG	Highest Potential Growth
MC	Maximum Correlation
MU	Minimum Utilization
ACO	Ant Colony Optimization
RTDVMC	Release Time based Dynamic VM Consolidation
RC	Resource Constraint
MUTC	Maximum Utilization Threshold Constraint
CS	*candidateSources*
CD	*candidateDestinations*
SVMRT	Stochastic VM Release Time
SPMRT	Stochastic PM Release Time
SRTDVMC	Stochastic Release Time based Dynamic VM Consolidation

PVMRT	Predicted VM Release Time
PPMRT	Predicted PM Release Time
PRTDVMC	Predicted Release Time based Dynamic VM Consolidation

Nomenclature

Notations	Meaning
$\|\|$	Cardinality of a Set
$P = \{P_i\}_{\|P\|}$	The set of $\|P\|$ PMs
$V = \{V_j\}_{\|V\|}$	The set of $\|V\|$ VMs
$V^i = \{V_j^i\}_{\|V^i\|}$	Set of VMs hosted in PM, P_i
R_{P_i}	Resource utilization ratio of a PM P_i
$\overline{R_{CDC}}$	Mean resource utilization ratio of CDC
x_a	the index of the a^{th} VMRT record
y_a	a^{th} original VMRT value
w_a	weight at (x_a, y_b)
$T(u)$	*tricube weight function*
x_1	the index of the b^{th} observation/*VMRT* record from the right boundary
$\Delta_i(x_b)$	the distance between x_i and x_b
$\hat{y}_a$	*PVMRT* value for the original a^{th} *VMRT* value, y_a
$\hat{\varepsilon}_a$	Residuals or Distance between y_a and $\hat{y}_a$
$r_a(x)$	Robustness weight for an observation/*VMRT* record (x_a, y_a)
$B(u)$	*bisquare weight function*
Δ_a^{RLR}	Distance between a user's a^{th} original *VMRT* value, y_a and the corresponding predicted *VMRT* value, $\hat{y}_a^{RLR}$ obtained through RLR model
δ_a^{RLR}	The safety margin for a^{th} *PVMRT* value
$\hat{y}_a^{RLR\_SM}$	*PVMRT* value including the safety margin for an observation/*VMRT* record (x_a, y_a)
$\hat{T}_{V_j}$	*PVMRT* of VM *Vj* and let denotes the set of *PVMRT* of VMs hosted in PM, P_i
$\hat{T}_{V^i} = \{\hat{T}_j^i\}$	The set of *PVMRT* of VMs hosted in PM, P_i
$\hat{T}_{P_i}$	Predicted PMRT (*PPMRT*) of PM, P_i.
T_{V_j}	*VMRT* of VM, V_j
$T_{V_j^i}$	*VMRT* of VM, V_j^i
$T_{V^i} = \{T_{V_j^i}\}$	The set of *VMRT* of VMs hosted in PM, P_i
T_{P_i}	*PMRT* of PM, P_i. $T_{P_i} = \max(T_{V^i})$
Y	Random variable ranging $[-1, +1]$
α	Maximum deviation of *VMRT*

Notations	Meaning
S_{V_j}	S*VMRT* of VM, V_j
$S_{V_j^i}$	S*VMRT* of VM, V_j hosted in PM, P_i
$S_{V^i} = \left\{S_{V_j^i}\right\}$	The set of S*VMRT* of VMs hosted in PM, P_i
S_{P_i}	Stochastic *PMRT* of PM, P_i. $S_{P_i} = \max(S_{V^i})$
$x = \{x_{i,j}\}_{\lvert P \rvert \times \lvert V \rvert}$	Placement Matrix
$R = \{R_k\}_{\lvert R \rvert}$	The set of Different Types of Resources
D_j^k	Demand of Resource, R_k by VM, V_j
U_i^k	Utilization of Resource, R_k of PM, P_i
C_i^k	Capacity of PM, P_i in terms of Resource, R_k
θ_{MAX}	Maximum Threshold
$OP = \{P_o\}_{\lvert OP \rvert}$	The set of *O-UPMs*
$V^o = \{V_q^o\}_{\lvert V^o \rvert}$	The set of VMs hosted in an *O-UPM*, P_o
D_q^k	Demand of Resource, R_k by VM, V_q^o
U_o^k	Utilization of Resource, R_k of PM, P_o
C_o^k	Capacity of PM, P_o in terms of Resource, R_k
$NOP = \{P_x\}_{\lvert NOP \rvert}$	The set of Non-*Over-utilized PMs*, which are neither *O-UPMs* nor in Sleep mode or switched off
$SP = \{P_s\}_{\lvert SP \rvert}$	The set of PMs that are in sleep mode or switched off
P_d	Destination PM
$vmsToMigrate = \{V_l\}_{\lvert vmsToMigrate \rvert}$	The set VMs to migrate out into new PMs
$destinationPMs = \{P_d\}_{\lvert destinationPMs \rvert}$	The set of new destination PMs for migrating VMs from source *O-UPMs*
$candidateSources = \{P_c\}_{\lvert candidateSources \rvert}$	The set of PMs from which a PM would be selected as *U-UPM*
$V^c = \{V_n^c\}_{\lvert V^C \rvert}$	The set of VMs hosted in PM, P_c
$candidateDestinations = \{P_m\}_{\lvert candidateDestinations \rvert}$	The set of PMs from which a PM would be selected to host migrating VM of a *U-UPM*
$destinations$	The set of new destination PMs for migrating VMs of source *U-UPMs*
E_i	Energy Consumption by PM, P_i
E_{CDC}	Energy Consumption by the CDC
$\bar{E}_{CDC}$	Mean Energy Consumption by the CDC
f_R	Objective Function of *RTDVMC*
f_1	First Objective Function of *SRTDVMC* and *PRTDVMC*

Notations	Meaning
f_2	Second Objective Function of *SRTDVMC* and *PRTDVMC*
$Nectar = \{Nectar_{NT}\}_{\lvert 3\rvert} = \{\text{Nov 2013, Dec 2013, Jan 2014}\}$	Set of different months of Nectar Cloud
$PLab = \{PLab_{PL}\}_{\lvert 4\rvert} = \{\text{6 March, 9 March, 9 April, 20 April}\}$	Set of different days of PlanetLab Cloud
$\bar{E}^S = \{\bar{E}^S_{NT,PL}\}_{\lvert Nectar\rvert \cdot \lvert PLab\rvert}$	Set of mean energy consumption values for *SRTDVMC* algorithm under different workload combinations of Nectar and PlanetLab
$\bar{E}^R = \{\bar{E}^R_{NT,PL}\}_{\lvert Nectar\rvert \cdot \lvert PLab\rvert}$	Set of Mean energy consumption values for *RTDVMC* algorithm under different workload combinations of Nectar and PlanetLab
$np = \{1,2,\ldots,\lvert Nectar\rvert \cdot \lvert PLab\rvert\}$	Index to denote np th smallest element of $\bar{E}^S$, $\bar{E}^R$, $\bar{\psi}^S$and $\bar{\psi}^R$
a_{np}	Weights related to Shapiro-Wilk Normality Test
$\bar{E}^S_{(np)}$	np th smallest element of $\bar{E}^S$
$\bar{E}^R_{(np)}$	np th smallest element of $\bar{E}^R$
$\bar{\bar{E}}^S$	Mean of $\bar{E}^S$
$\bar{\bar{E}}^R$	Mean of $\bar{E}^R$
ψ	Total Number of VM migration
$\bar{\psi}$	Mean Total Number of VM migration
$\bar{\psi}^S = \{\bar{\psi}^S_{NT,PL}\}_{\lvert Nectar\rvert \cdot \lvert PLab\rvert}$	Set of Mean Number of VM migration for *SRTDVMC* algorithm under different workload of Nectar and PlanetLab
$\bar{\psi}^R = \{\bar{\psi}^R_{NT,PL}\}_{\lvert Nectar\rvert \cdot \lvert PLab\rvert}$	Set of Mean Number of VM migration for *RTDVMC* algorithm under different workload of Nectar and PlanetLab
$X^{\bar{E}} = \{X^{\bar{E}}_{NT,PL}\}_{\lvert Nectar\rvert \times \lvert PLab\rvert}$	Minimization of $\bar{E}_{CDC}$ by *SRTDVMC* compared to *RTDVMC* under different workload combination of Nectar and PlanetLab
$X^{\bar{\psi}} = \{X^{\bar{\psi}}_{NT,PL}\}_{\lvert Nectar\rvert \times \lvert PLab\rvert}$	Minimization of $\bar{\psi}$ by *SRTDVMC* compared to *RTDVMC* under different workload combination of Nectar and PlanetLab
$SW^S_{\bar{E}}$	Test statistics for the S-W normality test with $\bar{E}^S$
$SW^R_{\bar{E}}$	Test statistics for the S-W normality test with $\bar{E}^R$
$SW^S_{\bar{\psi}}$	Test statistics for the S-W normality test with $\bar{\psi}^S$
$SW^R_{\bar{\psi}}$	Test statistics for the S-W normality test with $\bar{\psi}^R$

Notations	**Meaning**
$t_{\bar{X}^{\bar{E}}}$	Test statistic for the *t-test* with data samples of $\bar{X}^{\bar{E}}$
$t_{\bar{X}^{\bar{\psi}}}$	Test statistic for the *t-test* with data samples of $\bar{X}^{\bar{\psi}}$
$\hat{\sigma}_{\bar{X}^{\bar{E}}}$	Standard Deviation of $\bar{X}^{\bar{E}}$
$\hat{\sigma}_{\bar{X}^{\bar{\psi}}}$	Standard Deviation of $\bar{X}^{\bar{\psi}}$
$\bar{X}^{\bar{E}}$	Mean of $X^{\bar{E}}$
$\bar{X}^{\bar{\psi}}$	Mean of $X^{\bar{\psi}}$
$SVMC = \{SVMC_{SV}\}_{\lvert 4\rvert} = \{\mathrm{FF}, \mathrm{BF}, \mathrm{NF}, \mathrm{RS}\}$	Set of different SVMC approaches
$PlanetLab = \{PlanetLab_{PL}\}_{\lvert 10\rvert} = \left\{\begin{matrix} \text{3 March, 6 March, 9 March,} \\ \text{22 March, 25 March, 3 April,} \\ \text{9 April, 11 April, 12 April,} \\ \text{20 April} \end{matrix}\right\}$	Set of different days of PlanetLab Cloud
$\bar{E}_{SV}^{P} = \{\bar{E}_{SV,PL}^{P}\}_{\lvert PlanetLab\rvert}$	Set of mean energy consumption values for *PRTDVMC* algorithm with different days of PlanetLab workload for a particular SVMC algorithm denoted by index, *SV*
$\bar{E}_{SV}^{TMMT} = \{\bar{E}_{SV,PL}^{TMMT}\}_{\lvert PlanetLab\rvert}$	Set of mean energy consumption values for THR-MMT algorithm with different days of PlanetLab workload for a particular SVMC algorithm denoted by index, *SV*
$w = \{1, 2, \ldots, \lvert PlanetLab\rvert\}$	Index to denote wth smallest element of $\bar{E}_{SV}^{P}$, $\bar{E}_{SV}^{TMMT}$, $\bar{\psi}_{SV}^{P}$ *and* $\bar{\psi}_{SV}^{TMMT}$
$\bar{E}_{(w)}^{P,SV}$	wth smallest element of $\bar{E}_{SV}^{P}$
$\bar{E}_{(w)}^{TMMT,SV}$	wth smallest element of $\bar{E}_{SV}^{TMMT}$
$\bar{\bar{E}}_{SV}^{P}$	Mean of $\bar{E}_{SV}^{P}$
$\bar{\bar{E}}_{SV}^{TMMT}$	Mean of $\bar{E}_{SV}^{TMMT}$
$\bar{\psi}_{SV}^{P} = \{\bar{\psi}_{SV,PL}^{P}\}_{\lvert PlanetLab\rvert}$	Set of Mean Number of VM migration for *PRTDVMC* algorithm with different days of PlanetLab workload for a particular SVMC algorithm denoted by index, *SV*
$\bar{\psi}_{SV}^{TMMT} = \{\bar{\psi}_{SV,PL}^{TMMT}\}_{\lvert PlanetLab\rvert}$	Set of Mean Number of VM migration for THR-MMT algorithm with different days of PlanetLab workload for a particular SVMC algorithm denoted by index, *SV*
$\bar{\psi}_{(w)}^{P,SV}$	wth smallest element of $\bar{\psi}_{SV}^{P}$
$\bar{\psi}_{(w)}^{TMMT,SV}$	wth smallest element of $\bar{\psi}_{SV}^{TMMT}$
$\bar{\bar{\psi}}_{SV}^{P}$	Mean of $\bar{\psi}_{SV}^{P}$
$\bar{\bar{\psi}}_{SV}^{TMMT}$	Mean of $\bar{\psi}_{SV}^{TMMT}$

Notations	**Meaning**
$X_{SV}^{\bar{E}} = \left\{X_{SV,PL}^{\bar{E}}\right\}_{\vert PlanetLab \vert}$	Minimization of $\bar{E}_{CDC}$ by *PRTDVMC* compared to THR-MMT under a particular SVMC algorithm denoted by SV for different days of PlanetLab workload
$X_{SV}^{\bar{\psi}} = \left\{X_{SV,PL}^{\bar{\psi}}\right\}_{\vert\vert PlanetLab \vert}$	Minimization of $\bar{\psi}$ by *PRTDVMC* compared to THR-MMT under a particular SVMC algorithm denoted by SV for different days of PlanetLab workload
se_{SV}	Standard Error for distribution
a_w	Weights related to Shapiro-Wilk Normality Test
$SW_{\bar{E}}^{P,SV}$	Test statistics for the S-W normality test with $\bar{E}_{SV}^{P}$
$SW_{\bar{E}}^{TMMT,SV}$	Test statistics for the S-W normality test with $\bar{E}_{SV}^{TMMT}$
$SW_{\bar{\psi}}^{P,SV}$	Test statistics for the S-W normality test with $\bar{\psi}_{SV}^{P}$
$SW_{\bar{\psi}}^{TMMT,SV}$	Test statistics for the S-W normality test with $\bar{\psi}_{SV}^{TMMT}$
$\bar{X}_{SV}^{\bar{E}}$	Mean of sampling distribution of sample mean $X_{SV}^{\bar{E}}$
$\bar{X}_{SV}^{\bar{\psi}}$	Mean of sampling distribution of sample mean $X_{SV}^{\bar{\psi}}$
$\hat{\sigma}_{X_{SV}^{\bar{E}}}$	Standard Deviation of sampling distribution of sample mean $X_{SV}^{\bar{E}}$
$\hat{\sigma}_{X_{SV}^{\bar{\psi}}}$	Standard Deviation of sampling distribution of sample mean $X_{SV}^{\bar{\psi}}$
$t_{\bar{X}_{SV}^{\bar{E}}}$	Test statistic of the two tail paired *t-test* with $\bar{X}_{SV}^{\bar{E}}$
$t_{\bar{X}_{SV}^{\bar{\psi}}}$	Test statistic of the two tail paired *t-test* with $\bar{X}_{SV}^{\bar{\psi}}$
SE	Standard Error
$SE_{X_{FF}^{\bar{E}}}$	Standard Error for distribution of $X_{SV}^{\bar{E}}$
$SE_{X_{SV}^{\bar{\psi}}}$	Standard Error for distribution of $X_{SV}^{\bar{\psi}}$
CI	Confidence Interval
$CI_{\bar{X}_{SV}^{\bar{E}}}$	Confidence Interval for $\bar{X}_{SV}^{\bar{E}}$
$CI_{\bar{X}_{SV}^{\bar{\psi}}}$	Confidence Interval for $\bar{X}_{SV}^{\bar{\psi}}$
t_{CI}	Critical value for t interval

1 TABLE OF CONTENTS

List of Figures

List of Tables

List of Algorithms

Abstract

The thesis describes our effort to minimize energy consumption in the Cloud Data Centers (CDCs). Our general approach is to do so by smartly using user provided information. The research challenges herein are twofold. Firstly, finding the right benign information to be received from a Cloud Service User (CSU), which can complement the energy-efficiency of CDC. Secondly, smart application of such information to significantly reduce the energy consumption of CDC.

To address those research challenges, in this dissertation, we have proposed a novel heuristic Dynamic VM Consolidation (DVMC) algorithm, namely *Release Time Based DVMC* (*RTDVMC*) algorithm, which minimizes the energy consumption of CDC through utilizing CSU provided information. Our research exemplifies the fact that if VMs are dynamically consolidated based on the time when a VM can be removed from CDC – a useful information to be received from respective CSU, then more physical machines can be turned into sleep state, yielding lower energy consumption.

Our initial novel approach, *RTDVMC* and other traditional DVMC algorithms assume that optimal energy-efficiency can be achieved via maximum load on Physical Machines (PMs). Such assumption has become invalid with the advent of the highly energy proportional PMs. Consequently, these algorithms may fail to proffer optimal performance under real Cloud scenarios. Although minimization of VM migration brings massive benefit for CDC, it is complete opposite of what is needed to minimize energy consumption through DVMC. The energy-efficiency through *RTDVMC* comes at a cost of excessive VM migration. Hence, we have proposed a new multi-objective DVMC algorithm, namely *Stochastic Release Time based DVMC* (*SRTDVMC*) algorithm, which is unique in addressing concomitant minimization of energy consumption and VM migration in the presence of new generation state-of-the-art PMs.

Both *RTDVMC* and *SRTDVMC* algorithms pose unrealistic conditions that Infrastructure as a Service (IAAS) users must be able to estimate VMRT and must be willing to proffer that VMRT information to IAAS users. Pitfalls of such conditions are massive. Estimation of VMRT by IAAS users compels additional expenses after developing skills and facilities to manage and operate large volume of previous usage records. To mitigate the investment burden of IAAS users, IAAS providers would need to provide proper incentive back to IAAS users, yielding into elevated business operational cost for IAAS users. We have hence proposed our third novel DVMC algorithm, namely *Predicted Release Time Based DVMC* (*PRTDVMC*) algorithm, which is modeled to completely eradicate such caveats.

We have simulated the performance of our proposed algorithms with real Cloud data centers workload traces. The empirical figures and result analysis affirm the superiority of proposed algorithms over existing prominent Static and Adaptive Threshold based DVMC algorithms.

1 Introduction

Cloud computing, visioned by Leonard Kleinrock [1], has transformed the dream of ‘computing as a utility’ into reality. It has turned out as the latest computing paradigm [2]. Ever since the remarkable financial benefits of using Clouds to the use of own data centers have been realized, growing number of business institutions and entrepreneurs have been moving their databases as well as other applications into the Cloud environment that are being sold as products and services to their own customers. Furthermore, cloud providers have been offering to rent high performance computing resources as well as software development platforms including but not limited to operating systems and a wide range of application development tools anytime and for any duration in accordance to their clients’ requirements. Beyond that, Cloud users may opt for expansion or shrinking their cloud services usage that perfectly fits to their own business and clients’ demands at any point of time through a payment of no more than their aggregate quantity and duration of usage. Consequently, the provisions of on-demand ordering along with flexibility in service usage with paying in accordance to the meticulous quantity and length of consumption have eroded institutions’ initial extreme financial burden of purchasing software platforms as well as establishing own computing hardware infrastructures including data centers. In today’s modern era, no institution and business may operate without IT support, as leveraging of IT services has revolved into an absolute necessity. Besides, we are living in an era of ever-increasing competitive marketplaces. Hence, Cloud computing has turned out as a highly lucrative means to ensure long term financial sustainability of multitude of businesses through drastic reduction of fixed term and operating costs. Thus, by far it has become impossible for entrepreneurs to outperform other competitors in terms of cost without diving towards electing Cloud services for IT support.

1.1 Background

Research conducted by [3] unlocks that Cloud data centers (CDCs) consume a lot of electricity, which on average is twenty-five thousand times more than a household’s energy demand. Their research has revealed that energy consumption of Cloud origins more carbon emission to that of two countries, Netherlands and Argentina [3]. Beloglazov [4] has highlighted that worldwide energy usage of cloud data centers climbed up to 56% from 2005 to 2010, and was projected to be between 1.1% and 1.5% of the total electricity use in 2010 [5]. Moreover, approximate carbon emissions by IT industry is 2% of the global emissions, which is equivalent to the emissions of the aviation industry [6]. Koomey [7] projected that CDC energy consumption would remain to rise rapidly without energy- efficient resource management solutions being applied. As such, many developed countries are getting more concerned these days to reduce carbon emission [8].

Furthermore, in order to have a competitive edge in the business, big Cloud Service Providers' (CSPs') need to have their own DCs across major cities in the world. Consequently, DC construction has increased 47% in 2017 [9], resulting in consumption of 3% of world's energy [10] - equivalent to the energy consumption by airline industry [11]. Research on DCs of the United States has revealed that in 2014, the sum of energy usage by all DCs in the US was 70 billion kWh - accounting for 1.8% of the country's total energy usage [12]. More importantly, the trend of energy consumption of DCs highlights that energy consumption is rising every year and is expected to increase by 4% from 2014 to 2020. The energy consumption by the DCs of the US data centres is estimated to reach up to 73 billion kWh in 2020. Further studies, such as [13, 14] has highlighted that Google consumed as much energy as the city of San Francisco in 2015. Subsequently, countries across the world have come forward to address the challenge of increasing energy consumption by CDC [15]. For instance, Joint Research Centre (JRC) of European Commission has formed the Code of Conduct for Energy Efficiency in DCs with an aim to inform and encourage DC owners and operators to effectively level off the energy consumption [16]. Most recently in 2018, JRC has proposed a detailed guideline in relation to the best practices to limit the energy consumption of Data Centres [17]. Standing on such recent evidences of increasing energy consumption despite the usage of modern energy-efficient servers and considering the intrinsic unpredictable behaviour of nature in renewable energy sources, we argue that regardless of the types of used energy sources in DCs, researchers' contributions to cut down the huge energy consumption of CDC remains highly significant to present day.

1.2 CDC Energy Regulation Techniques

By far, we have discussed the necessity of cloud energy consumption reduction. In this section, we will introduce the energy reservation techniques applied in CDC. Researchers have broadly classified Cloud energy consumption minimization techniques into three groups [18]:

- Workload Prediction
- Resource Overcommitment

- Virtual Machine (VM) Consolidation

In the following three sections these three techniques have been briefly discussed.

1.2.1 Workload Prediction

A physical server or PM may have any of following four states:

- **Active state:** One or more VMs are hosted in the PM. The PM keeps consuming energy, which primarily depends upon the amount of CPU usages as demanded by the hosted VMs.
- **Idle state:** Although no VMs are currently residing in the server, yet the server is kept as turned on. In this state a PM consumes as much as 50% energy as it consumes in the active state [19].
- **Sleep state:** An intermediate state between Idle state and complete switched off state. In this state the PM consumes very low amount of energy.
- **Inactive/Switched Off state:** The physical server is switched off and consumes no energy at all.

To save energy, it is vital to keep those PMs into lower energy consuming states, for instance, sleep state when no VMs are hosted in them. However, if a PM is kept in sleep state for a short period and then turned back into the active state, then the amount of energy saved by keeping it in sleep state would be lower than the amount of transient energy spent after switching it back to the active state. Besides, the service disruption would be experienced by Cloud Service Users (CSUs) in that intermediate period required to switch the PM back from sleep state to the active state. Therefore, it is crucial to know the expected workload beforehand, so that required amount PMs can be kept in the active state, while the rest of the PMs can be kept in either sleep or switched off state and thus can be saved.

The workload prediction method proposed by [18] first categorizes VMs into different groups or clusters based on the VMs' resource demand. Then for each group, number of expected VMs is predicted. Thus, the total expected workload is calculated and respective number of PMs are kept on or in the active state to handle the total predicted workload, while the rest of PMs are kept in either sleep or switched Off state [18]. From the literature [20], we have observed that workload prediction techniques for Cloud is vast. Researchers have proposed diverse workload prediction techniques that utilize including but not limited to deep learning [21] and neural network [22].

The issue with workload prediction is that the prediction may go off target, since prediction is based on past and current usage, whereas there is always a possibility of mismatches between present and future or past and future. Consequently, over-estimation or under-estimation of workload may take place, which causes less energy saving than expected and Service Level Agreement (SLA) violation, respectively. Furthermore,

1.2.2 Resource Overcommitment

It has been observed that CSUs tend to reserve more resources than the actual required amount for their respective VMs, which causes poor resource utilization. Therefore, VMs resource demand are projected to a new amount, which is lower than the amount as reserved by respective CSUs. Thus, more VMs can now be placed in less number of PMs with a lower projected resource demand than the number of PMs, which would have been required to accommodate those VMs as per the amount mentioned by respective CSUs at the point of reservation. The issue with resource overcommitment is that one might argue that it is not moral to provide less amount of resources than respective CSUs paid for. Besides, the efficiency of resource overcommitment technique greatly depends upon the projected or expected resource amount for a VM, while the projection based on past usage may become off target leading towards performance degradation. From studying literature, we have observed that researchers [23, 24] have utilized diverse prediction techniques to implement resource overcommitment.

1.2.3 VM Consolidation

More energy is consumed by a physical server if it is in turned on state (i.e., the active state) compared to a low energy consumption state, for instance, sleep state, whereas no energy is consumed if it is in turned off state. In order to reduce the energy consumption, Pinheiro et al. proposed that power consumption of a set of PMs can be reduced rather through load concentration, or unbalancing technique, whilst switching the idle machines off [25]. Furthermore, virtualization, which is the core of Cloud computing has advanced to a level that a VM is now possible to be moved from one PM to another without interrupting it from running on the source PM, also called as live VM migration [26].

Dynamic Power Management (DPM) technique uses VM migration for evacuating and turning off an underutilized host after migrating its VMs to another host, which is currently being more utilized, also entitled as VM consolidation. As delineated in Fig. 1.1, several physical servers may contain a small number of VMs, for instances, one or two. If all these

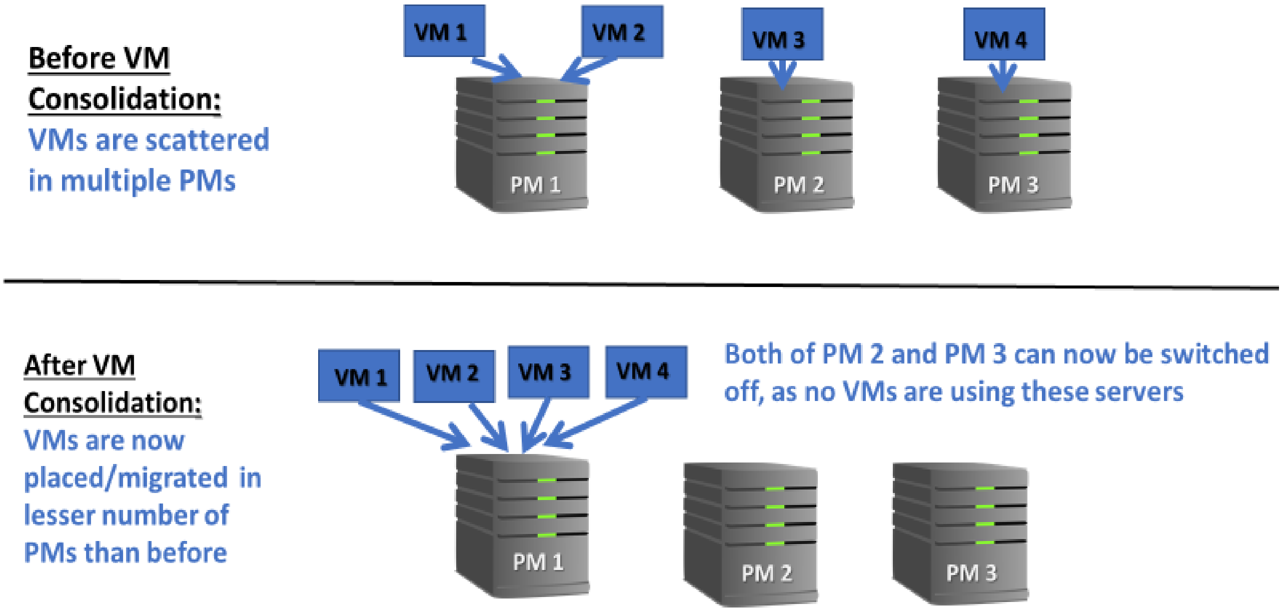
Before VM Consolidation:
VMs are scattered in multiple PMs
VM 1
VM 2
VM 3
VM 4
PM 1
PM 2
PM 3
After VM Consolidation:
VMs are now placed/migrated in lesser number of PMs than before
VM 1
VM 2
VM 3
VM 4
Both of PM 2 and PM 3 can now be switched off, as no VMs are using these servers
PM 1
PM 2
PM 3

Fig. 1.1 VM Consolidation

VMs, which are scattered in multiple physical servers are moved away from those servers and placed into one single physical server, then those physical servers, which would now contain no VM; can be put into sleep state or turned off state and hence energy can be saved. This technique is called VM consolidation. It is one of the fundamental and popular DPM techniques applied to achieve energy-efficiency in CDC. The algorithm, which is used to accomplish VM consolidation is called VM consolidation algorithm.

1.3 Benefits of VM Consolidation

VM consolidation provides many benefits including the following:

1. Workload Prediction calculates the total expected workload from which the number of required PMs are estimated. However, efficient VM placement algorithm is still essential to choose the right PM for a particular VM at the point of initial VM placement and VM migration to further minimize the energy consumption. Therefore, without VMC algorithm Workload Prediction method would not be as effective in terms of reducing energy consumption.
2. Resource Overcommitment assumes that users reserve more resources for their VMs than the actual requirement and therefore projects the estimated VM resource demand

to a lower amount. Consequently, more VMs can be consolidated in lesser number of PMs than the number of PMs, which would be required as per User perceived VM resource demand. Therefore, resource overcommitment is inherently using VM consolidation as the means of energy conservation.

3. Cloud computing is called as elastic computing, since it is claimed to be scalable with any amount of increase of user demand at any moment. VM Consolidation is the natural requirement to maintain the scalability and availability of Cloud with the perpetual increase of users' demand, since it utilizes resources conservatively to make the utilization as high as possible.
4. One primary motivation for innovation of Cloud computing is to increase the utilization of computing resources. VM Consolidation minimizes resource wastage and enhances Cloud resource utilization.
5. VM Consolidation minimizes the number of active physical servers, by consolidating more VMs in lesser number of PMs. Thus, VM consolidation increases server utilization and promotes green Cloud by reducing the power consumption of CDC. In the following section we have presented the classification of VM consolidation.

1.4 Classification of VM Consolidation

VM Consolidation can be classified into two groups:

- **Static VM Consolidation (SVMC):** A CDC may consist of thousands of PMs and hundreds of thousands of VMs. Prior creation of a VM, a PM is needed to be chosen to host that VM. SVMC algorithm is used to select a PM among a number of PMs to host a VM. Diverse SVMC algorithm is developed to uphold diverse objectives, including but not limited to minimization of number of active PMs, minimization of network related energy consumption, better throughput. If the initial VM placement is not done without considering energy usage optimization, then the overall energy-efficiency of CDC would be greatly affected.
- **Dynamic VM Consolidation (DVMC):** Cloud is a multi-tenant environment. Multiple VMs hosted in a single PM shares the underlying resources of that PM. Resource demand by a VM varies over time. With the change in resource demand of hosted VMs, changes in resource availability of a PM takes place. To elucidate further, when resource demand by hosted VMs drops, the host PM's unutilized amount of

resources increases, unfolding the opportunity to host additional VMs, aka VM consolidation. Resource demand by VMs hosted in a PM may also rise over time, while the PM might not have adequate resources to meet the increased resource demand resulting into delayed service and decreased throughput. In such case, one or more VM(s) are needed to migrate out in different PMs. Furthermore, hardware failure may occur, resulting into addition or deletion of new PMs, which changes the amount of available resources in CDC. As such, with the progression of time, the VM placement solution provided by SVMC algorithm loses efficiency. DVMC algorithms provides the solution of reallocation of existing VMs in lesser number of PMs such that the number of active PMs is minimized.

1.5 Benefits of DVMC Compared to SVMC

Application of SVMC is limited to initial VM placement. It is unable to stop Quality of Service (QoS) degradation, yielding into SLA violation. CDC energy consumption can be further minimized through recurrent VM migrations considering the fluctuations of resource demand and resource availability. DVMC algorithm considers the change in workload of PMs and dynamically migrates VMs according to the change in workload of PMs. DVMC algorithm uplifts the energy-efficiency of CDC and inhibits SLA violation. These were the reasons behind our motivation to research with DVMC algorithm.

1.6 Research Motivations

In Section 1.3, we have articulated the benefits of VM consolidation. There are a number of research issues that need to be addressed to advance the existing concept of the DVMC algorithm. They are introduced in the following section.

1. In this dawn of big data and billions of IoT devices, usage of Cloud based services is growing tremendously. As a result, the energy consumption of CDC is rapidly increasing, and many cities hence do not allow CSPs to build any new CDC. Although existing energy conservation techniques are being applied in CDC, the intensity of the problem is still swelling. Therefore, there is a need to introduce a new approach in order to resolve the issue. While CSUs receive immense benefits of Cloud based services, thus far, the responsibility to minimize CDC energy consumption is solely shouldered by CSP. To the best of our knowledge, any potential collaboration between CSU and

CSP to reduce CDC energy consumption is unheard of. We propose that instead of CSP being alone carrying the burden, both CSU and CSP should agree to work together to reduce CDC energy consumption. Our research motivation is to present a model in which CSUs and CSP would work hand in hand towards energy-efficient management of CDC.

2. With the rapid advancement of modern technology around the world, most countries have become aware of the intense global demand for more and more energy. Energy can be produced in many ways including but not limited to burning non-renewable and environmentally polluting resources such as coal and gas. As climate change and global warming has become a global threat, it is, therefore, necessary to regulate CDC energy consumption and thus stop damaging our environment. For this reason, we have focused on developing more energy-efficient DVMC algorithms compared to existing literature.
3. Through VM migrations, VMs are attempted to be consolidated in lesser number of PMs than before in order to reduce energy consumption. However, excessive VM migrations increase network overhead, decreases network throughput causing QoS degradation. The importance of minimizing energy consumption and regulating VM migrations are alike. However, since, VM migration is inherent in DVMC, therefore, minimizing both energy consumption and VM migration at the same time are confronting objectives. Our research motivation is to address the challenge of designing a DVMC algorithm, which satisfies both objectives.
4. Academic research, thus far, has seen very limited numbers of studies focused on critical review on existing DVMC algorithms. DVMC algorithm is a popular research topic. Plethora of research and advancement is ongoing in this area. Hence, undertaking a research to present a critical review on cutting-edge DVMC algorithms is extremely important. Such review of contemporary algorithms would present crucial future research ideas.
5. In Section 1.3, we have articulated the benefits of VM consolidation. Nevertheless, existing DVMC algorithms have a number of limitations, such as:
    - Cloud being a multi-tenant environment and distributed computing in nature, VMs of different CSUs can be hosted in a single PM. While, these VMs hosted in the same PM share underlying resources of the host VM, assigned workload in these

VMs are heterogeneous in terms of workload finishing time. To the best of our knowledge, academic research, thus far, has not seen studies focused on developing DVMC algorithm for heterogeneous workload in terms of workload finishing time. This necessitates to design a DVMC algorithm considering heterogeneous workload finishing time at the bedrock.

- For legacy PMs, the higher is the CPU utilization, the lower is the ratio of energy consumption to CPU utilization. Existing DVMC algorithms hence attempt to consolidate maximum possible number of VMs in minimum possible number of PMs based on the assumption that maximum energy-efficiency is achievable at maximum load level on PMs. However, such assumption has become invalid for state-of-the-art highly energy proportional PMs. For highly energy proportional PMs, energy-efficiency rather drops beyond 70% load level due to the drastic rise of energy consumption at that level. Performance of existing DVMC algorithms loses optimality on account of not being developed considering the changed energy-efficiency characteristics of state-of-the-art highly energy proportional PMs.

1.7 Research Objectives

To address those research issues raised in the previous Section, we have aimed to work on following objectives:

1. At present, the extent of effort to minimize CDC energy consumption is limited to the effort of CSP. Our objective is to construct a bridge between CSUs and CSP, so that CSUs and CSP can work hand in hand towards building more energy-efficient of CDC together. To achieve that we aim to create a platform for CSUs, so that alongside CSP, CSUs can contribute in effective minimization CDC energy consumption.
2. To address the lack of performance of existing DVMC algorithms in the presence of heterogeneous workload, we aim to develop innovative DVMC algorithms considering heterogeneous workload finishing time at its bedrock.
3. To develop robust DVMC algorithm that will have inherent self-adjusting capability with the change in energy-efficiency characteristics of underlying PMs; consequently, it will be suitable for a heterogeneous CDC consisting both state-of-the-art highly energy proportional PMs.
4. To develop a DVMC algorithm that can minimize both CDC energy consumption and VM migration simultaneously.

5. To classify and critically review contemporary DVMC algorithms, which is currently missing in existing literature. Deeper understanding of pros and cons of different types of DVMC algorithms would assist in making the conscious choice of a DVMC algorithm for a CDC.
6. To achieve more accurate performance estimation of proposed DVMC algorithms in real Cloud scenarios. Hence, we aim to perform evaluations and analysis of proposed DVMC algorithms with real Cloud based heterogeneous workload.
7. To provide future researchers valuable research ideas to extend the concept of DVMC algorithms further.

In the following section, we have articulated our main research contributions.

1.8 Research Contributions

To fulfil the objectives mentioned in Section 1.7, following contributions has been made through our research:

1. In this dissertation, we have classified and critically reviewed VMC algorithms from multitude of viewpoints, so that the readers can be truly benefitted (**Objective** 1).
2. One of our research accolades is the innovation of a model to effectively minimize CDC energy consumption utilizing CSU provided information. Our study presents the pathway for CSP to incorporate CSU provided information in DVMC algorithm in order to minimize CDC energy consumption further (**Objective** 2). The information received from CSUs is release time of their respective VMs (i.e., workload finishing time). We have brought forth a novel DVMC algorithm, which takes CSU provided heterogeneous workload finishing time into VM consolidation decision process. Hence, our proposed DVMC algorithm is suitable for heterogeneous workload in terms of workload finishing time (**Objective** 3).
3. We have provided novel heterogeneous workload finishing time aware DVMC algorithms for CDC comprised of state-of-the-art highly energy proportional PMs (**Objective** 3 and 4).
4. Prior any VM migration, estimation of the difference between benefit and cost for a VM migration is embodied in our proposed techniques. A VM is only migrated, if the net energy gain is found positive. Consequently, our proposed DVMC algorithms are

more optimized compared to existing techniques in terms of minimizing both CDC energy consumption and VM migration (**Objective** 5).

5. We have developed Cloud simulation models for our proposed algorithms and other existing notable DVMC algorithms, using a Cloud based discrete event simulation tool, namely CloudSim [27]. The performance of our approaches has been analysed and compared with existing techniques in terms of different performance metrics such as mean CDC energy consumption and mean total number of VM migration. For performance comparison real Cloud based workload has been drawn from Nectar Cloud [28] and PlanetLab [29] (**Objective** 6).
6. Last but not the least, we have elucidated valuable future research directions so that it would pave the way for fellow researchers to further contribute in this area (**Objective** 7).

In the following section, we have presented the structure of the thesis.

1.9 Thesis Structure

The thesis structure is organized as follows:

In Chapter 2, we have focused on previous studies on DVMC algorithms. Based on our study, we have classified existing DVMC techniques. We have elucidated pros and cons of each of those techniques. Studying the literature has aided us finding useful research gaps as articulated in the chapter. We have set the direction of our research to address those gaps. Most of the content of this chapter was formulated and published as:

1. *Khan MA, Paplinski A, Khan AM, Murshed M, Buyya R. Dynamic Virtual Machine Consolidation Algorithms for Energy-Efficient Cloud Resource Management: A Review. Sustainable Cloud and Energy Services: Springer; 2018. p. 135-65.*

Our proposed heterogeneous VMRT aware DVMC algorithm, namely *Release Time based DVMC* (*RTDVMC*) algorithm including mathematical modelling of resource demand, resource utilization and energy consumption, diverse constraints, objective functions, the algorithm and its unique characteristics under different scenarios are detailed in Chapter 3. In this chapter, we have also presented the empirical evaluation of our proposed algorithm and compared with other notable existing algorithms under real Cloud based diverse CPU utilization distributions. Most of the content of this chapter was formulated and published as:

1. *Khan MA, Paplinski A, Khan AM, Murshed M, Buyya R. Exploiting User Provided Information In Dynamic Consolidation of Virtual Machines to Minimize Energy Consumption of Cloud Data Centers. Third International Conference on Fog and Mobile Edge Computing (FMEC); April 23-26, 2018; Barcelona, Spain. 2018.*

RTDVMC presents a novel method to utilize CSU provided information in minimizing CDC energy consumption. Nonetheless, the improved energy-efficiency through *RTDVMC* comes at a cost of excessive VM migration. In Section 1.6.3, we have explained necessities of regulation of VM migration in CDC. Furthermore, in Section 1.6.5.2, we have elucidated the issues with existing DVMC algorithms in the presence of state-of-the-art highly energy proportional PMs. To address these issues, in Chapter 4, a novel DVMC algorithm, namely *Stochastic Release Time based DVMC* (*SRTDVMC*) algorithm has been proposed. *SRTDVMC* is suitable for state-of-the-art highly proportional PMs, as it is robust regardless of the energy-efficiency characteristics of underlying PMs. In addition, *SRTDVMC* is a multi-objective DVMC algorithm, which aims to minimize both CDC energy consumption and VM migration at the same time. Similar to *RTDVMC*, *SRTDVMC* is developed considering heterogeneous VMRT. The characteristics of *SRTDVMC* and performance measure under diverse heterogeneous VMRT distributions and heterogeneous resource utilization distributions drawn from real Cloud, namely Nectar Cloud and PlanetLab, respectively are highlighted in detail in Chapter 4.

RTDVMC and *SRTDVMC* algorithms come with the constraint that CSUs must provide VMRT information to CSP prior VM consolidation. Such constraint conflicts with the concept of pay-as-you-go service, since freedom is provided to CSUs to first use the service as long as they want and pay later. Pay-as-you-go consumers might not always be able to inform VMRT information prior service usage. Furthermore, estimation of VMRT by PAAS provider/IAAS user demands PAAS provider/IAAS user to store, maintain and analyse large volume of past resource usage records. Many PAAS provider/IAAS user are unable to afford the cost and facilities needed to do so. Consequently, IAAS provider/CDC owner would need to provide incentives to PAAS provider/IAAS user, which would increase the business operational cost and decrease the profit margin. We have addressed these issues in Chapter 5. The chapter presents two regression based mathematical models, which CDC owner can use to generate predicted VMRT utilizing past VMRT records. Since, as part of billing process, CDC owner must record VMRT information, hence unlike *RTDVMC* and *SRTDVMC*, generating VMRT utilizing past VMRT records does not incur any additional cost on CDC owner. Next, we have introduced a novel DVMC algorithm, namely *Predicted Release Time based DVMC* (*PRTDVMC*) algorithm, which utilizes predicted VMRT. Similar to *SRTDVMC*, *PRTDVMC* is suitable for state-of-the-art highly energy proportional PMs. The characteristics and advantages of the proposed algorithm compared to *RTDVMC* and *SRTDVMC* have been explained in detail and included in the chapter. The chapter highlights performance analysis of *PRTDVMC* algorithm in combination with diverse SVMC algorithms under real cloud based heterogeneous workload.

Finally, the thesis is concluded with Chapter 6 summarising our research and critical findings. The chapter also presents potential research scopes emerged from the findings of our research project.

2 Review on Dynamic Virtual Machine Consolidation Algorithms

2.1 Introduction

Previously, in Chapter 1, we have elucidated the importance of VM consolidation to regulate energy consumption of CDC. We have also presented our motivations to develop DVMC algorithms with an aim of concomitant minimization of CDC energy consumption and VM migration. In doing so, we have first undertaken a research on available literature to obtain knowledge on existing VM consolidation algorithms. In this chapter, we have discussed on diverse SVMC and DVMC algorithms available in the literature. Most of the content of this chapter was formulated and published as.

- *Khan MA, Paplinski A, Khan AM, Murshed M, Buyya R. Dynamic Virtual Machine Consolidation Algorithms for Energy-Efficient Cloud Resource Management: A Review. Sustainable Cloud and Energy Services: Springer; 2018. p. 135-65.*

While, VM consolidation effectively minimizes CDC energy consumption, it may cause resource contention leading towards QoS degradation, translated as SLA violation. To elucidate further, as portrayed in Fig. 1.1, before VM Consolidation is applied, VMs are scattered in multiple PMs. VM consolidation migrates the VMs from lower utilized PMs to higher utilized PMs and thus consolidate VMs in lesser number of PMs than before. Meanwhile, the state of those PMs having no VMs, can be changed from the active state (i.e., turned on state) into a lower energy consuming state, such as sleep state and consequently, energy consumption can be minimized. Moreover, on account of compacting more number of VMs into fewer number of PMs, *resource utilization ratio* of a PM P_i, R_{P_i} (2.1) would become higher, which in turn would increase the *mean resource utilization ratio of CDC*, $\overline{R_{CDC}}$ (2.2), where N is the total number of active hosts in CDC.

$$R_{P_i} = \frac{Utilized\ Amount\ of\ Resource\ of\ P_i}{Total\ Amount\ Resource\ Of\ P_i} \tag{2.1}$$

$$\overline{R_{CDC}} = \frac{\sum_{i=1}^{N} R_{P_i}}{N} \tag{2.2}$$

However, as delineated in Fig. 2.1, VMs hosted in a PM share the underlying physical resources of that PM. Therefore, with the increased number of VMs sharing underlying resources of a single PM, waiting time for a VM prior receiving its required resources becomes higher. Thus, if more VMs are placed in a single PM, resource contention may arise, which would lead towards poor QoS. Consequently, possibility of SLA violation arises.

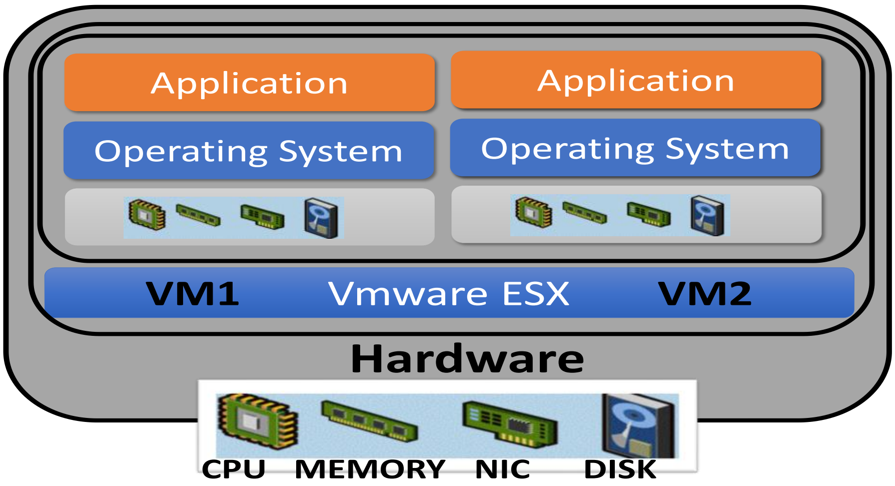

Fig. 2.1 A PM Hosting Multiple VMs [30]

To balance the trade-off between QoS and energy efficiency, it is extremely challenging to design such VM consolidation algorithm, which increases both resource utilization and energy efficiency without compromising the QoS of running applications as agreed with respective CSUs during SLA. Recently, VM consolidation has attracted interest of Cloud researchers, while designing efficient VM consolidation algorithms is extremely challenging as it needs to be *scalable*, to the millions of VMs and PMs, as well as *robust*, such that the performance does not degrade with the fluctuation in resource demand of VMs. In this chapter, we have presented detail discussion on a wide range of VM consolidation algorithms. Our classification of VM Consolidation algorithms has been delineated through Fig. 2.2. In brief, our contributions are as follows:

- A group of surveys on VM consolidation algorithms [31-41] have greatly assisted our research with VM consolidation algorithms. However, VM consolidation is an extremely popular research area and none of the available survey papers have presented

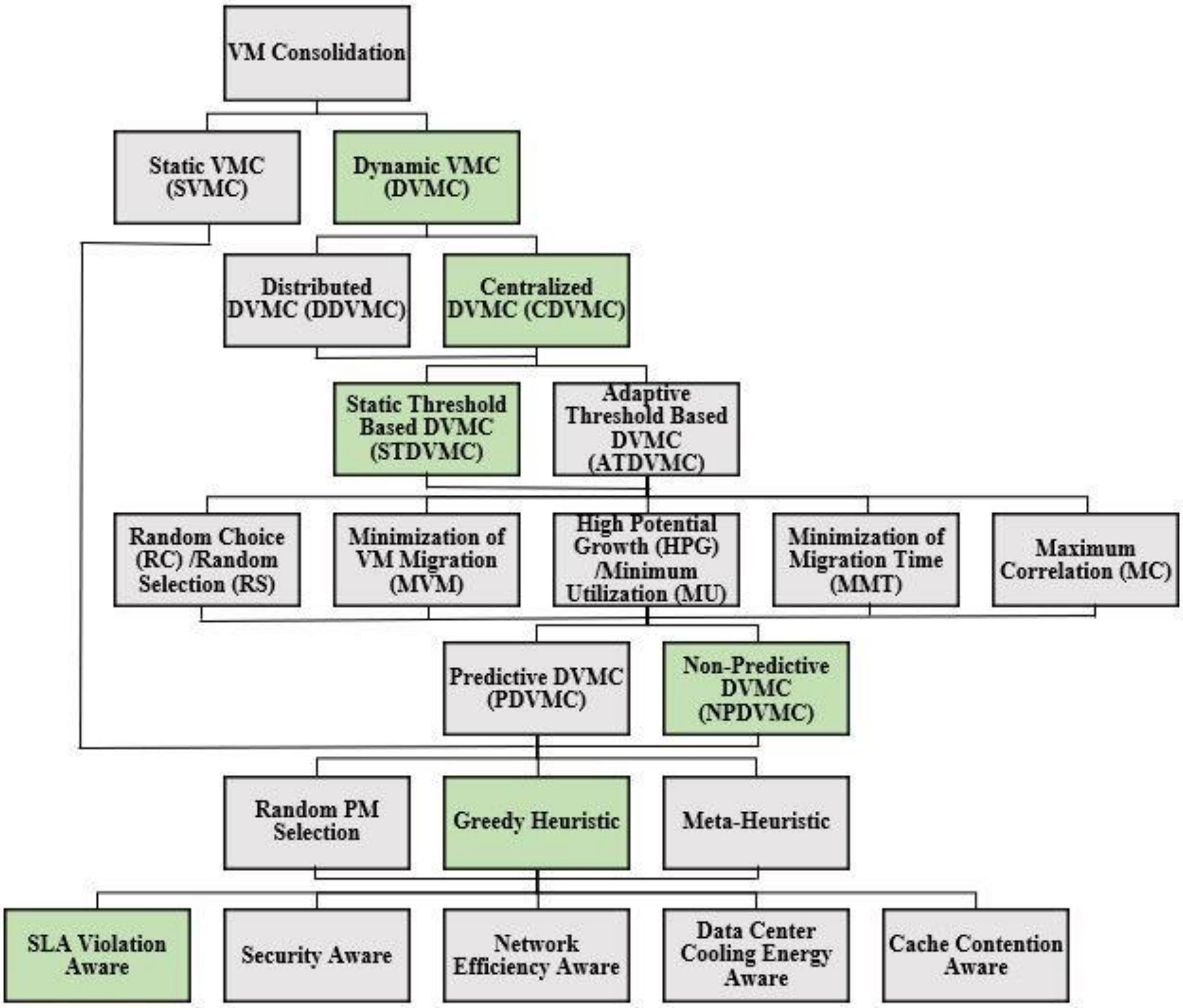
VM Consolidation
Static VMC (SVMC)
Dynamic VMC (DVMC)
Distributed DVMC (DDVMC)
Centralized DVMC (CDVMC)
Static Threshold Based DVMC (STDVMC)
Adaptive Threshold Based DVMC (ATDVMC)
Random Choice (RC) /Random Selection (RS)
Minimization of VM Migration (MVM)
High Potential Growth (HPG) /Minimum Utilization (MU)
Minimization of Migration Time (MMT)
Maximum Correlation (MC)
Predictive DVMC (PDVMC)
Non-Predictive DVMC (NPDVMC)
Random PM Selection
Greedy Heuristic
Meta-Heuristic
SLA Violation Aware
Security Aware
Network Efficiency Aware
Data Center Cooling Energy Aware
Cache Contention Aware

Fig. 2.2 Classification of VM Consolidation Algorithms

discussion on VM consolidation algorithms published since 2016. To address that, in this chapter, we have primarily focused on VM consolidation algorithms, which are not available in any of these survey papers.

- Related literature of VM consolidation algorithm is extremely broad. Taking the time constraints for our dissertation into account and based on our level of understanding, we have presented our own critical review on different types of VM consolidation algorithms, which is missing in the available literature.
- There are several existing VM consolidation algorithms proposed before the year of 2016, which strongly influenced subsequent researchers through introducing unique research directions. Those researches presented their own distinct techniques, which are strong enough to be used as classification criterions. Therefore, to the best of our understanding, we have also included elaborate discussion on such prominent VM

consolidation algorithms proposed before the year of 2016 to clarify the important concepts based on which we have reached our own classification of VM consolidation algorithms.

- Authors of [37] have mentioned that presenting a survey and classification of VM consolidation algorithms with an equal justice to all viewpoints is hardly possible. In the light of such admitted belief, based on our level of understanding of existing literature within the given time for this dissertation, we have proposed our own analysis and classification of existing VM consolidation algorithms with an emphasis towards incorporation of future resource demand of Cloud resources, since considering the future resource demand is essential to prevent the SLA violation, which is one of the major drawbacks of VM consolidation algorithms.
- Based on our level of understanding of existing literature, we have pointed out potential important research scopes, which have not been explored so far.

In the following section we have elaborately discussed our classification of VM consolidation algorithms.

2.2 Classification of VM Consolidation Algorithms

As delineated in Fig. 2.2, we have broadly classified VM consolidation algorithms into two groups:

- Dynamic VM Consolidation (DVMC) Algorithm
- Static VM Consolidation (SVMC) Algorithm

2.2.1 Dynamic VM Consolidation (DVMC) Algorithm

In Cloud, received workloads are run in VMs, while these VMs accomplish the assigned workload through consuming the resources of the respective hosting PMs. With the advancement of time, progression of previously accepted workloads continues, while at the same time, new workloads keep being accepted by CSP. Furthermore, removal of some PMs due to hardware failure and addition of new PM also takes place. Thus, the overall workload with corresponding resource requirement and resource availability in the CDC keeps evolving over time. In DVMC Algorithm, current VM-to-Server assignment is taken into consideration in the VM consolidation process. Note that, the workload or resource requirement of any VM and its location (i.e., its hosting PM) can be dynamic, as it changes with time. If the VM consolidation algorithm consolidates VMs considering the dynamic (i.e., changing) workload and location of the VM (i.e., current VM-to-Server assignment), then it is called DVMC algorithm. In simple words, DVMC algorithms provides the solution of reallocation of existing VMs in lesser number of PMs such that the number of active PMs is minimized.

2.2.2 Static VM Consolidation (SVMC) Algorithm

In contrast to DVMC algorithm, Static VMC (SVMC) algorithms, also referred as consolidated VM placement algorithms do not consider the current VM-to-Server assignment while choosing a new destination PM for any VM. In [42], the authors have mentioned that SVMC algorithms work with a set of fully empty PMs and a set of VMs with specific resource requirement. In simple words, SVMC algorithm provides the solution of initial VM placement in minimum number of active PMs so that energy-efficiency and resource utilization of CDC increases. However, it does not provide the solution for reallocation of VMs in new PMs considering the current VM-to-Server assignment. Since, the dynamic (i.e., changing) load and placement of VMs are not considered, therefore, it is called as SVMC algorithm. [43, 44] are examples of SVMC algorithm is which do not consider the current VM-to-Server assignment while choosing a new destination PM for a VM.

Energy-efficiency of CDC would be hampered without the initial consolidated VM placement, as provided by SVMC algorithms. Besides, the energy-efficient initial placement keeps VMs consolidated in fewer PMs from the very beginning and consequently, the intermediate period before the awakening of the necessity to reallocate VMs can be prolonged. VMC has network overhead and it hampers QoS due to inherent service downtime. The prolonged period between initial VM placement and VM consolidation or between two consecutive VM consolidation reduces the overhead of VM consolidation. However, the dynamic VM-to-Server assignment is not taken into consideration in SVMC algorithm. Therefore, the migration cost of a VM from its current hosting PM to its new destination PM is ignored. Consequently, SVMC algorithms are only applicable for initial placement of VMs or migrating VMs of one CDC into another CDC. As the time progresses, both of workload and resource availability changes in CDC. Therefore, apart from the initial consolidated VM placement, DVMC is one of the key techniques that uphold the energy-efficiency, resource usage optimization and profit maximization of CSPs.

The breakdown of different components of SVMC and DVMC algorithms has been highlighted in the following sections.

2.3 Fundamental Components of DVMC Algorithms

DVMC consolidation algorithm is comprised of three core components [4, 45], which are as follows:

2.3.1 Source Host Selection

First, among all the PMs, a set of PMs are selected from where VMs are migrated out. The *Source Host Selection* component takes all the PMs and VMs as input and selects one or more PMs as source PM(s) from where VMs would be migrated out.

2.3.2 VM Selection

Secondly, one or more VM(s) are selected for migration from a source PM. The *VM Selection* component takes a PM as input, which has been selected by *Source Host Selection* component and selects one or more VMs from that source PM for migration into a different PM.

2.3.3 Destination Host Selection/VM Placement

Finally, the *Destination Host Selection/VM Placement* component selects a PM for each of the migrating VM, which was selected by *VM Selection* component.

However, after a VM being created for the first time (i.e., for new VMs), the initial placement of that newly created VM can also be considered as VM consolidation, if the corresponding destination PM is selected with an aim to minimize the total number of active PMs and increase the resource utilization, given that the hosting PM has adequate resources to fulfil the resource demand of the new VM. Hence, for new VMs, only *Destination Host Selection Algorithm/VM Placement Algorithm* does the VM consolidation, as no source host selection and VM selection are needed for new VMs. A number of VM placement algorithms are available in the literature. While VM consolidation algorithm is our primary focus, we have also included VM placement algorithms in the discussion of this chapter, as we have denoted VM placement algorithms as VM consolidation algorithms in case of new VMs. In the following section, we have presented the classification of VM consolidation algorithms.

2.4 Classification of DVMC Algorithms based on the Controlling Architecture

From our literature study, we have observed that based on controlling architecture DVMC algorithms can be categorized into two groups:

- Centralized DVMC (CDVMC) Algorithm
- Distributed DVMC (DDVMC) Algorithm

2.4.1 Centralized DVMC (CDVMC) Algorithm

As proposed in [46-48], in centralized architecture, there is a single controller, which has the information about present resource availability of all the PMs. The controller runs the CDVMC Algorithm, which selects a destination PM for a migrating VM considering the resource availability of all the PMs.

2.4.2 Distributed DVMC (DDVMC) Algorithm

Instead of having a single controller, which poses the information of present available resource availability of all the PMs and selects a destination PM for any migrating VM considering that information; in distributed architecture, PMs exchange information of their present resource availability with their own neighbour PMs and thus, each PM has the resource availability information of its neighbourhood PMs. If a PM wants to migrate out one of its VMs, it executes Distributed VMC Algorithm to select one of the neighbour PMs as the destination PM where the migrating VM would be hosted. Example of distributed DVMC algorithms are [49, 50].

Major DVMC algorithms found in the literature are centralized DVMC and only a few Distributed DVMC [49, 50] has been proposed. In [49], authors have presented their distributed DVMC algorithm for a P2P network oriented CDC. According to [49, 50], the growing number of PMs becomes a bottleneck for CDVMC at the time of selecting a destination for any migrating VM, since the asymptotic time complexity of the centralized DVMC algorithm is proportional to the number of PMs in the CDC, whereas the number of potential PMs to choose from for a migrating VM is relatively small in distributed DVMC. Thus, distributed DVMC is more scalable for CDC with huge number of PMs. However, distributed DVMC has message passing overhead, as every PM must update its present resource availability to all of its neighbours. Every time a VM is migrated, both the sender PM and the destination PM must update their present resource availability status to all of their neighbours. Besides, a central monitoring system is indispensable in Cloud, which monitors the accepted workload progression status and allocate/deallocate resources accordingly to accomplish the workload in time. Furthermore, at the time of accepting new workload, the overall resource availability status of CDC must be known, so that accurate decision can be made on whether the new workload would be possible to serve within deadline. Therefore, central DVMC can be implemented without adding any additional resources. Moreover, message passing as required by distributed DVMC algorithms, increases network overhead, decreases network throughput and increases network related energy consumption.

As discussed earlier in Section 2.1, first component of DVMC algorithm is to select the source PM. A DVMC algorithm can either randomly select a source PM from where one or more VM(s) are migrated out or VM(s) can be selected from over-utilized and under-utilized PMs. In the following section, we have presented our classification of DVMC algorithms based on different source PM selection techniques.

2.5 Classification of DVMC Algorithms Based on Different Source PM Selection Techniques

From examining literature on DVMC algorithms, we have observed that in majority cases, upper and lower threshold values are used to identify a PM as overloaded or underloaded respectively, from the perspective of resource utilization ratio of a PM P_i, R_{P_i} (2.1). The key point is that R_{P_i} is compared against the values of some thresholds, which can be either static or adaptive (explained in detail in Section 2.4.1 and 2.4.2). As presented in [51], if R_{P_i} goes past the upper utilization threshold value, then P_i is identified as overloaded or over-utilized PM and VMs are migrated out from P_i, until R_{P_i} becomes lower than the upper-threshold, since high R_{P_i} is a strong indicator of potential QoS degradation or SLA violation, which is arisen because of the higher resource demand of the hosted VMs. Again, if R_{P_i} is smaller than the lower utilization threshold value, then P_i is identified as underloaded or under-utilized and potential destination PMs are looked for, where VMs of P_i can be migrated out so that P_i can be put in sleep state. Based on the type of used thresholds to identify a PM as overloaded or underloaded, threshold-based DVMC algorithms can be classified into two groups. The classification of threshold-based DVMC algorithms has been presented in the following.

2.5.1 Static Threshold-Based DVMC (STDVMC) Algorithm

In STDVMC algorithms, fixed values or predefined values are used as upper and lower thresholds to identify a PM as overloaded or underloaded. As the values of the thresholds do not change over time, therefore it is called as static threshold. Examples of STDVMC algorithms are [49, 51, 52]. In [52], the authors has used 100% CPU utilization as upper utilization threshold and 50% CPU utilization as lower utilization threshold. In other words, if the CPU utilization of a PM is found as 50%, then that PM is considered as lower utilized PM and VMs are migrated out from that PM into other PMs. Similarly, if the total resource demand of all the VMs hosted in a particular PM is found as higher than the CPU capacity of that PM, then that PM is considered as overloaded PM and VMs are migrated out from that PM into other PMs.

2.5.2 Adaptive Threshold-Based DVMC (ATDVMC) Algorithm

On the contrary, in ATDVMC algorithms, the values of the thresholds based on which a PM is selected as overloaded or underloaded, changes dynamically as the resource utilization ratio a PM P_i, R_{P_i} (2.1) changes with time. In other words, the threshold value adapts with the change of resource utilization. Examples of ATDVMC algorithms are [45, 47, 53, 54].

To the best of our knowledge, authors of [45] are pioneers in proposing adaptive threshold-based DVMC algorithm, as they proposed a number of adaptive thresholds-based on which a PM is detected as overloaded. One such adaptive threshold is referred to as Median Absolute Deviation (MAD). To illustrate more, let, $T = \{ \mathrm{t_j} \mid \mathrm{t_j} \text{ denotes time j and j} \in \mathbb{N}\}$ where ℕ is the set of positive integers and $R_{P_i}^{t_j}$ is the resource utilization ratio a PM P_i at time t_j. For each PM P_i, R_{P_i} (2.1) across different time (i.e., $R_{P_i}^{t_1}$, $R_{P_i}^{t_2}$, $R_{P_i}^{t_3}$ and so forth) would be recorded and then MAD is calculated using (2.3), while the upper utilization threshold, T_u is calculated using (2.4), where $s \in \mathbb{R}^+$ a parameter, which defines how strongly the system tolerates host overloads.

$$MAD = Median(\left|R_{P_i}^{t_j} - Median(R_{P_i}^{t_j})\right|) \tag{2.3}$$

$$T_u = 1 - s.MAD \tag{2.4}$$

The lower is the value of s, the system is more tolerant to variation in resource utilization. If the current $R_{P_i}^{t_j}$ is found as greater than T_u, then P_i is considered as overloaded [4]. Note that, value of MAD (2.3) is not fixed or static, as $R_{P_i}^{t_j}$ changes with time and hence, T_u (2.4) also changes with the change in resource utilization. The lower is the value of s, the system is more tolerant to variation in resource utilization.

One of the aims of VM consolidation algorithm is to increase the resource utilization, which helps to minimize the energy consumption. However, very high resource utilization or higher R_{P_i} (2.1) causes QoS degradation or potential SLA violation. In other words, the higher is the upper and lower utilization threshold, the higher is the energy saving. However, the higher is the upper utilization threshold, the higher is the SLA violation. Hence, there is a trade-off between energy efficiency and SLA violation, while the values of upper and lower utilization thresholds have great impact in controlling such trade-off. Hence, a balance is needed to be maintained between energy efficiency and SLA violation through controlling or changing the threshold values, which is not possible with static threshold policy.

The key idea of using upper and lower static thresholds is to keep the resource utilization restricted into a certain range (i.e., in between upper and lower utilization threshold), so that a balance exists between energy efficiency and SLA violation. However, the workload pattern as experienced by an application running inside of a VM changes over time. Besides, multiple VMs, which are all hosted in a single PM may exhibit different workload pattern. Since, the threshold is static and it cannot be changed with the change of workload, therefore, SLA violation increases with the increase of workload. ATDVMC algorithms partially mitigate this problem by changing the utilization threshold values with the variation in workload. For instance, as the workload grows in VMs, MAD (2.3) increases and the upper utilization threshold (2.4) becomes lower accordingly. Consequently, VMs are migrated out from P_i before R_{P_i} (2.1) reaches to a very high level and as a result, VMs of P_i do not suffer from degraded QoS due to high R_{P_i} (2.1). In general, compare to static threshold-based approach, adaptive threshold-based approach decreases SLA violation rate more from the perspective of high resource utilization. However, it provides less energy-efficiency than static threshold-based approach, since the lower is the upper and lower utilization threshold, the lower is the energy consumption minimization. Apart from that, adaptive based approach causes more number of VM migration than static based approach, which increases both energy consumption and SLA violation.

Thus far, we have reviewed different types of threshold based DVMC algorithms. As highlighted previously in section 2.1, one of the core components of VM consolidation algorithm is VM selection. In the following section, we have presented our discussion on different DVMC algorithms with different VM selection policies.

2.6 Classification of DVMC Algorithms Based on VM Selection Policy

Once source PMs are selected, the following step of VM consolidation is to select one or more VM(s) from source PM to migrate out. Different prominent VM selection strategies as found in the literature are mentioned in the following:

2.6.1 Random Choice (RC)/Random Selection (RS)

Among all the VMs residing in the source PMs, a VM is randomly selected [51, 55]. This is also named as Random Selection (RS) [4]. Random VM Selection can select a VM in $O(1)$ time.

2.6.2 Minimization of VM Migration (MVM)

Minimum number of VMs are migrated to make the current resource utilization of a PM lower than the upper utilization threshold. MVM algorithm as proposed by [51], first sorts the VMs in descending order with respect to CPU demand and then selects the VM that satisfy the two criterions: First, the VM’s CPU utilization should be higher than the difference between the host’s present overall CPU utilization and the upper utilization threshold; Second, that VM is selected for which the difference between the upper threshold and the new utilization is the minimum compare to the values provided by all the VMs. If no such VM is found, then the VM with the highest utilization is selected and the process is repeated until the new utilization becomes lower than the upper utilization threshold.

2.6.3 High Potential Growth (HPG)/Minimum Utilization (MU)

The VM with lowest ratio of actual resource usage to its initial claimed resource demand is selected [51] . A number of authors [46, 47] have referred this technique as Minimum Utilization (MU) while considering only resource utilization and ignoring resource demand part. Asymptotic running time of the algorithm is $O(n)$.

2.6.4 Minimization of Migration Time (MMT)

The VM, which requires minimum time to complete the migration is selected for migration, while the migration time is estimated as the amount of RAM utilized by a VM divided by the spare network bandwidth available for the hosting PM [45]. Asymptotic running time of the algorithm is $O(n)$.

2.6.5 Maximum Correlation (MC)

VM that has the highest correlation of the resource utilization with other VMs are selected [4]. *Multiple Correlation Coefficient* as proposed by [56], is used to determine the correlation between the resource utilization of VMs.

The RC may help to find the globally optimal solution. However, if the solution space is confided prior, such as source PM is selected using the heuristic that the PM with highest or lowest resource utilization would be the source PM and after then that, a VM is randomly chosen from that PM, then RC might not provide the global optimal solution. Rest of the greedy heuristics such as MVM, HPG, MMT and MC provide the local best solution. RC probabilistically may choose a solution, which will not be locally optimal. To illustrate more, VMs experience degraded QoS during the period of migration. Therefore, selecting the VM, which would take the least migration time (i.e., MMT) would certainly assist in keeping the SLA violation lower [57]. In contrast, RC may choose a VM with higher migration time. Consequently, SLA violation rate may become higher for RC compared to rest of the techniques.

MVM minimizes the number of VM migrations with minimal decreasing of resource utilization ratio of a PM P_i, R_{P_i}, (2.1). As a result, *mean resource utilization ratio of CDC,* $\overline{R_{CDC}}$ (2.2) would remain as higher compare to rest of the above-mentioned algorithms and thus it turns out to be more energy-efficient. However, higher R_{P_i}, (2.1) may cause degraded QoS and more SLA violation. Hence, MVM shows higher SLA violation than MMT.

Another critical issue with MVM is that VMs need to be sorted first with respect to resource utilization, as otherwise the asymptotic running time is exponential. However, it is not possible to sort the VMs with respect to all types of resource demand, since a VM has three different types of resource demand, such as CPU demand, memory demand and network bandwidth demand, which are not related. For instance, a VM may have high CPU demand and low network bandwidth demand, whereas another VM may have low CPU demand and high network bandwidth demand. Therefore, it is not possible to sort VMs based on VM resource demand, since one distinctive feature of CDC is location transparency [58], which arises from the fact that a VM can be placed in any of the PMs. Hence, a PM may have VMs with varied resource demand with respect to different resource types [59].

Because of such diverse resource utilization value of a VM across various types of resources, it is not possible to select a VM with highest potential growth ratio (i.e., ratio of actual resource usage to a VM's initial claimed resource demand) among all the VMs across all resource types. Consequently, HPG is only possible to be implemented considering one resource type, such as CPU or memory or network bandwidth and thus, it does not ensure the minimization of energy-efficiency or SLA violation. In contrast, since MMT primarily selects VM based on memory size and hence, it is free from such issue.

Thus far, we have analyzed different DVMC algorithms with different VM selection policies. Another critical distinguishing aspect among DVMC algorithms is that whether estimated future resource demand has been considered in the VM consolidation process or not, as we have presented our discussion about it in the following section.

2.7 Classification of DVMC Algorithms based on Consideration of Estimated Future Resource

From the literature, it can be viewed that consideration of estimated future workload in a PM is commonplace in a wide range of DVMC algorithms. However, there are still numerous DVMC algorithms, which make the consolidation decision based on the current resource utilization of PMs instead of the estimated future resource utilization. Consideration of future can create a significant difference on the performance of VM consolidation algorithm, compare to those VM consolidation algorithms, which takes the decision based on the current resource utilization. Hence, in this section, we have reviewed both types of VM consolidation algorithms from that perspective.

2.7.1 Non-Predictive DVMC (NPDVMC) Algorithm

Instead of considering the estimated future resource utilization of a PM, NPDVMC algorithms consider the current aggregated resource demand of VMs. Note that the aggregated resource demand of hosted VMs in a PM is equal to the resource utilization of that PM. VM migration decisions are taken when the current resource utilization of a PM P_i, R_{P_i} (2.1) becomes very high or very low so that SLA violation can be avoided or energy consumption can be minimized.

One such example of NPDVMC algorithm is [46], where source and destination PMs for consolidation of VMs are selected based on the current resource utilization status of the PM. If the current R_{P_i} is found as equal or greater than 90%, then P_i is considered as overloaded or over-utilized and VMs are migrated out from P_i. Again, if current R_{P_i} is found as equal or lower than 10%, then P_i is considered as overloaded or over-utilized and VMs are migrated out from P_i to place in new PMs. Other prominent non-predictive VMC algorithms are [46-48, 60-63].

2.7.2 Predictive DVMC (PDVMC) Algorithm

On the contrary, PDVMC algorithms take the decision to migrate VMs from one PM to another PM considering the estimated future resource demand of VMs instead of current resource demand. Examples of PDVMC algorithms are [52, 64].

In [52], both of current and future resource utilization of a PM is considered while making the consolidation decision. Linear regression [54] is used to generate an estimated future resource utilization of a PM from analysing its past resource utilization statistics. If the current resource utilization of a PM is found as higher than the upper-utilization threshold, then that PM is identified as *overloaded*. Furthermore, although the current resource utilization of a PM is found as lower than the upper-utilization threshold, yet its estimated future resource utilization is found as higher than the upper-utilization threshold, then that PM is identified as *predicted overloaded*. Both of *overloaded* and *predicted overloaded* PMs are elected as source PMs from where one or more VM(s) are selected to migrate out into new PMs. Again, both of *overloaded* and *predicted overloaded* PMs are excluded from the list of potential destination PMs where migrating VMs would be placed.

Consolidation of VMs considering the estimated workload is a more proactive approach than NPDVMC, as VMs are migrated out from those PMs, which are predicted to be overloaded in future. Aim of such proactive approach is to move VMs out prior QoS degradation or SLA violation takes place. Consequently, compare to NPDVMC algorithms, PDVMC algorithms will display lower SLA violations due to less occurrences of resource contention. However, because of migrating more VMs out of the higher utilized hosts than NPDVMC, the *mean resource utilization ratio of CDC*, $\overline{R_{CDC}}$ (2.2) would become lower and thus, total number of inactive PMs may become less for PDVMC. Hence, PDVMC would display lower energy consumption minimization than that of NPDVMC.

Another challenging aspect of PDVMC is that PDVMC relies on prediction techniques to estimate the future resource utilization of PMs. Predictive techniques are based on the correlation between the past history of the system behavior and its near future [4]. The efficiency of prediction-based techniques greatly depends on the actual correlation between past and future events and the quality of adjustment with a specific workload type. In Cloud environment, different VMs are hosted in a single PM, while these VMs are expected to exhibit different behavioural pattern from each other in terms resource demand. Consequently, no single prediction technique would be a perfect fit for all PMs. A non-ideal prediction causes over or under prediction, which lead towards either inefficient resource usage or more SLA violation.

Thus far, we have reviewed diverse approaches to select source PMs and VMs, as we have also analysed both prediction-based and non-prediction-based DVMC algorithms. One of the core components of DVMC algorithms is destination PM selection where migrating VMs are placed. This is also referred as **VM placement problem, aka SVMC**. In the following section, we have presented our discussion on diverse approaches to select destination PMs for migrating VMs as incorporated in different VM consolidation algorithms.

2.8 Classification of VM Consolidation Algorithms based on Destination PM Selection Strategies

Destination PM selection strategy plays an important role in increasing the energy-efficiency of CDC. Aim of destination PM selection is to select such new PMs for migrating VMs so that the total number of active PMs becomes minimum without violating the resource constraint of any PM. However, destination PM selection /SVMC is a NP-Hard problem and hence a number of heuristic as well as meta-heuristic algorithms have been proposed in the literature. Based on different destination PM selection strategies, we have broadly classified VM consolidation algorithms into three groups:

2.8.1 Random PM Selection (RPS)

Authors of [60] and [18] have compared their proposed methods with a VM consolidation algorithm, which randomly selects a destination PM from the list of suitable PMs. The asymptotic running time of FF is $O(n)$, where n is the total number of VMs.

2.8.2 Greedy Heuristic

Greedy Heuristic algorithms are most widespread in the literature to select the destination PM for migrating VMs. Several popular heuristic based algorithms are as follows:

2.8.2.1 First Fit (FF)

In FF, PMs are ordered in a sequence and for each VM, the first available PM from the ordered list of PMs is selected. In other words, for every single VM, the searching of destination PM always starts from the first PM. If the first PM cannot accommodate a VM, then the second PM is checked and if the second PM cannot accommodate it, then the third PM is checked, as the searching continues to the next PM while always following the initial order of PMs until a suitable destination PM with adequate resource capacity is found. Since, a VM may have larger resource demand than the available remaining resource of a PM, therefore, the asymptotic running time of FF is $O(nm)$, where n is the total number of VMs and m denotes the total number of PMs.

2.8.2.2 First Fit Decreasing (FFD)

FFD is same as FF, except the VMs are first sorted in the decreasing order of their resource demand. Then the destination PM for the first VM with highest resource demand is first searched using FF algorithm, as the searching continues for the VM with second highest resource demand and so on. The asymptotic running time of FF is $O(nlogn + nm)$, where n is the total number of VMs and m denotes the total number of PMs. Note that, $O(nlogn)$ is running time of sorting algorithm.

2.8.2.3 Next Fit (NF)/ Round Robin (RR)

Like FF, NF also performs a sequential search, except it starts from the last server selected in the previous placement. To explain more, if the last VM was placed in the second PM, then checking will start from the second PM for the following VM placement and so on, whereas in FF and FFD the checking would have always started from the first PM for any VM. NF is also referred to as **Round Robin (RR)** [65]. The asymptotic running time of NF is same as FF.

2.8.2.4 Best Fit (BF)

In BF, the PM with the minimum residual resource is selected as its destination PM [66]. The residual resource of a PM is the difference between the total resource capacity of the PM and the aggregated resource demand of the hosted VMs in it along with the resource demand of the target VM for which destination PM is under search. If PMs are first sorted based on resource utilization ratio, then running time of BF would be identical to that of FFD. However, if no sorting is applied, then the running time of BF would be $O(nm^2)$.

2.8.2.5 Best Fit Decreasing (BFD)

VMs are first sorted in the decreasing order based on their resource demand. Then the destination PM for the first VM with highest resource demand is first searched using BF algorithm, as the searching continues for the VM with second highest resource demand and so on. The asymptotic running time of BFD is same as FFD.

2.8.2.6 Power Aware Best Fit Decreasing (PABFD)

PABFD proposed by [51], is a modified version of BFD, as the VMs are first sorted in decreasing order based on their CPU demand and then the destination PM is selected with the least power increase compare to all the suitable PMs, which could host the target VM. The asymptotic running time of PABFD is same as FFD.

RPS is most time efficient; but least optimal compare to rest of the above mentioned VM selection strategies from the perspective of energy-efficiency, since it does not ensure to first choose a suitable PM from the set of currently turned on PMs, so that unnecessary waking up of PMs, which are currently in sleep state can be avoided.

Because of random PM selection nature of RPS, it may cause sparse VM placement or VMs may be found as more scattered compare to those of FF, FFD, BF and BFD. The rationale of BF heuristic is that placing VMs on the PM with the least remaining available resource would provide other turned on PMs with large remaining available resources, which can be used to support future larger VMs, while this strategy would concomitantly increase the *mean resource utilization ratio of CDC*, $\overline{R_{CDC}}$ (2.2). Experimental results of [18], suggests that energy consumption is lowest for BF and BFD, as BF and BFD packs the VMs more tightly compare to FF and FFD.

Difference between BFD and PABFD is that BFD will select the smallest PM among all the suitable PMs for the first VM in terms of total resource capacity of PMs and then consolidating more VMs into it, whereas PABFD will initially select the most power-efficient PM among all the suitable PMs for the first VM and then consolidating more VMs into it. PABFD focuses on utilizing power-efficient PMs more, which certainly has an impact on increasing the energy-efficiency of CDC. On the contrary, BFD leads towards utilizing the smallest PMs first and leaving larger PMs for future, while ignoring to ensure the usage of power-efficient PMs. With the increased number of VMs, larger PMs have to be turned on eventually. Hence, as opposed to BFD, since, PABFD ensures the more usage of power-efficient PMs, therefore PABFD is more energy-efficient compare to BFD.

2.8.3 Meta-heuristic

Greedy Heuristic algorithms may become stuck with local minima or local maxima. Therefore, several meta-heuristic based destination PM selection/SVMC algorithms have been proposed in the literature. In the following, we have discussed on several meta-heuristics embodied in VM consolidation techniques.

2.8.3.1 Evolutionary Algorithm

The general steps of evolutionary algorithm are as follows:

- At each step (generation), the algorithm starts working with a population comprised of a range of solutions (members), while different evolutionary based VM consolidation algorithms use different heuristics to generate the initial solution.

- Next, a few members are first selected, also called parents from the generation to produce new solutions (children). Different evolutionary algorithms propose different methods to select parents from the population. One common method to select parents is to check value(s) of objective function(s) for each of the member and then select the members with higher values. The optimization functions are called as Pareto set and the values of the Pareto set achieved by the members are called as Pareto front [65]. For VM consolidation, one prevalent object function is to maximize the number of physical servers with no VMs running in it or minimize the number of active physical servers.
- In order to produce a new solution (child), different parts collected from different parents are combined together. This technique is called mutation, which takes place with a certain probability. The objective of mutation is the faster production of more optimized solutions.
- Furthermore, after mutation, swapping or interchanging (crossover) among different parts of a child takes place with a certain probability. The goal of crossover is to complement the faster creation of new children that are more optimized. In the context of VM placement or VM consolidation, one widespread crossover technique is to interchange the hosts between two VMs [65].
- Through mutation and crossover, children are generated from parents, which are added in the population.
- The entire process is repeated or more generations are run, until no improvements are found from consecutive repetitions of the algorithm.
- Finally, a solution is chosen from the Pareto front based on an objective function.

2.8.3.2 Ant Colony Optimization (ACO)

In ACO, a virtual ant selects a PM for a VM through considering two factors: Heuristic, and Pheromone. Ants can either work sequentially or in parallel while constructing their own solutions. Each ant can either follow its own heuristic or all the ants can follow a common heuristic. The heuristic is the key, which guides to construct an optimal solution in aligned with the optimization function. Based on the diverse objective functions as presented in different ACO based VM placement or VM consolidation algorithms, the proposed heuristic varies from one ACO based VM placement or VM consolidation to another. We have discussed about different heuristics previously. One common heuristic found in a number of ACO based VM placement or VM consolidation is BFD [52]. Apart from heuristic, the *Pheromone* plays a critical role in constructing an ant's solution, which guides ants to find diverse solutions through exploring the search space. One key distinguishing aspect, which makes ACO meta-heuristic different from heuristic based algorithms is that some probability exists for an ant to choose the PM, which is not optimal from the perspective of heuristic and thus stagnation into local minima or local maxima is avoided.

2.8.3.3 Simulated Annealing

Authors of [67] have proposed a Simulated Annealing meta-heuristic based VM consolidation algorithm. In perturbation phase, instead of randomly choosing source or destination PMs, solutions are generated by selecting source PMs with lower utilization ratio and destination PMs with neither very high utilization ratio nor very low utilization ratio. Thus, VMs are consolidated in lesser number of active PMs. However, in order to avoid stagnation in local minima or local maxima, exploration is adopted through accepting solution, which is even less optimal than the optimal solution found so far.

Aim of VM consolidation is to minimize the energy-efficiency, as VM consolidation minimizes energy consumption by placing more VMs in a single PM. However, the higher number of VMs are placed in a single PM, the higher is the probability of QoS degradation or SLA violation. Hence, minimization of energy consumption and minimization of SLA violation are two confronting goals. While most researchers have focused to maintain a balance between minimization of PMs' energy consumption and SLA violation, some researchers have considered other aspects too, such as security, energy consumption by network, network throughput and so forth. In the following section, classification of VM consolidation algorithms based on their objectives has been presented.

2.9 Classification of VM Consolidation Algorithms based on Different Objectives

Different DVMC algorithms with diverse objectives have been observed in the literature, which we have mentioned in the following:

2.9.1 SLA Violation Aware

VM migration is intrinsic in VM consolidation. However, the services, which the VM is providing to its users' needs to be suspended temporarily at the time of migrating that VM from one PM to another PM. Hence, VM consolidation causes SLA violation. Furthermore, since VMs share the underlying physical resources of their hosting PM such as CPU, RAM, network bandwidth and so forth, therefore, the waiting time to receive the required resources for each VM increases with the increase of number of VMs in a single PM. Many VM consolidation algorithms focus to minimize such SLA violation by limiting the number of VM migrations as well as minimize resource oversubscription and thus decrease SLA violation. [60] is an example of SLA Violation aware VM consolidation algorithms.

2.9.2 Security Aware

Cloud is a multi-tenant environment, where VMs of different clients are hosted in same PM, while these VMs also share the underlying physical resources. Hence, security is one of the major challenging aspects in Cloud. In [46], authors have proposed a security-based DVMC algorithm.

2.9.3 Network Efficiency Aware

Network efficiency Aware VM consolidation algorithms consolidate VMs with an aim to uphold the network efficiency through considering diverse network related aspects, such as traffic among VMs, bandwidth and so forth. The aim is to reduce of network congestion, improve QoS and so forth. [68] is an example of network efficiency aware VM consolidation algorithm.

2.9.4 Data Center Cooling Aware

CDC is the physical backbone of any Cloud based services. One big challenging aspect of any data center is that an appropriate temperature must be always maintained, as the challenge escalates with the increase of the volume of data center along with the growth of number of PMs, network devices and so forth. Cooling of CDC is extremely crucial to ensure the smooth and continuous functioning of PMs, routers, switches and so forth. However, energy spending after cooling of CDC is very high and such energy requirement rises with the increase of quantity of VMs as well as with the increase of VMs' resource demand. Therefore, researchers have presented VM consolidation algorithms, which consolidate VMs in such a way that energy spending after cooling the data center can be minimized. [62] is an example of such VM consolidation approach, which minimizes the energy related to data center cooling.

2.9.5 Cache Contention Aware

Cache contention refers to the situation that a VM may experience extra cache misses, as other VMs co-located on the same CPU fetch their own data into the Lowest Level Cache (LLC), which forces to evict the VM's data from the LLC and later fetching back that VM's data into the LLC again causes cache misses to other co-located VMs. In order to minimize cache misses due to VM consolidation, [43] has proposed a cache contention aware VM consolidation algorithm that considers the expected cache misses at the time of destination PM selection for a migrating VM.

Until now, we have reviewed different types of VM consolidation algorithms. We have delineated the classification of VM consolidation algorithms in Fig. 2.2. In order to present further details of contemporary VM consolidation algorithms, we have discussed about different aspects of recent VM consolidation algorithms in the following section.

2.10 Detailed Analysis of Contemporary VMC Algorithms

Table 2.1 illustrates several aspects of the notable recent research works on VMC as found in the published materials. The description of the attributes, which are being considered to review the existing VM consolidation algorithms are as follows:

- **Research Project**: Name of the research project.
- **Type of VMC:** Whether the VM consolidation is SVMC or DVMC.
- **VM consolidation Decision Process:** If destination PM selection decision is taken centrally (i.e., centralized) or a source PM itself chooses another destination PM where the migrating VM will be placed (i.e., distributed).
- **Source PM Selection Strategy:** Type of threshold, which has been applied to select source PMs.
- **VM Selection Criteria:** What VM selection algorithm has been used.
- **Application of Prediction Technique**: Whether any prediction technique has been incorporated in the proposed system to predict the future resource utilization of PMs.

- **Destination PM Selection Strategy:** What algorithm has been used to select the destination PM for the migrating VMs.
- **Performance Evaluation Technique**: What technique is used to evaluate the performance of the proposed system.

Table 2.1 Different Aspects of the Notable VM Consolidation Algorithms

Research Project	**VM consolidation Type**	**VM consolidation Decision Process**	**Source PM Selection Strategy**	**VM Selection Criteria**	**Application of Prediction Technique**	**Destination PM Selection Strategy**	**Performance Evaluation Technique**
Security Aware and Energy-Efficient Virtual Machine Consolidation in Cloud Computing Systems [46]	DVMC	Centralized	Static and Adaptive Threshold-	RS, MMT, MC and MU.	Non-Predictive	Greedy Heuristic	Simulation using CloudSim [27]
Dynamic virtual machine consolidation for improving energy efficiency in cloud data centers [47]	DVMC	Centralized	Adaptive Threshold	RS, MMT, MC and MU	Predictive	Greedy Heuristic	Simulation using CloudSim
Thermal aware workload consolidation in cloud data centers[62]	DVMC	Centralized				Meta-heuristic	Simulation
Optimizing Virtual Machine Consolidation in Virtualized Data centers Using Resource Sensitivity [63]	SVMC	Centralized					Simulation in Matlab
Virtual Machine Consolidation with Multiple Usage Prediction for Energy-Efficient Cloud Data Centers [69]	DVMC	Centralized	Static Threshold to detect *O-UPM* and Adaptive Threshold to detect *U-UPM*	The VM with highest resource demand is selected	Predictive	Greedy Heuristic	Simulation with real Cloud workload traces
An efficient resource utilization technique for consolidation of virtual machines in cloud computing environments [55]	DVMC	Centralized	Adaptive Threshold	RC, MMT, HPG, MC	Non-Predictive	Greedy Heuristic	Simulation in CloudSim
Energy and migration cost-aware dynamic virtual machine consolidation in heterogeneous cloud data centers [70]	DVMC	Centralized				Experimented with both Heuristic and Meta-heuristic destination PM Selection algorithms	Simulation with real Cloud workload traces

Research Project	**VM consolidation Type**	**VM consolidation Decision Process**	**Source PM Selection Strategy**	**VM Selection Criteria**	**Application of Prediction Technique**	**Destination PM Selection Strategy**	**Performance Evaluation Technique**
Energy and migration cost-aware dynamic virtual machine consolidation in heterogeneous cloud data centers [70]	DVMC	Centralized				Experimented with both Heuristic and Meta-heuristic based destination PM Selection algorithms	Simulation with real Cloud workload traces
Cache contention aware Virtual Machine placement and migration in cloud data centers [43]	SVMC	Centralized				Greedy Heuristic that selects the PM with least cache contention	Simulation with real Cloud workload traces
Improved Virtual Machine Migration Approaches in Cloud Environment [71]	DVMC	Centralized		MMT	Predictive	Greedy Heuristics	Simulation in CloudSim with real Cloud workload traces
Self-Adaptive Resource Management System in IaaS Clouds [53]	DVMC	Centralized	Adaptive Threshold		Predictive	Greedy Heuristics	Simulation in CloudSim with real Cloud workload traces
Energy-aware vm consolidation in cloud data centers using utilization prediction model [66]	DVMC	Centralized	Static Threshold	VM with minimum resource demand	Predictive	Greedy Heuristics	Simulation in CloudSim with real Cloud workload traces

Research Project	VM consolidation Type	VM consolidation Decision Process	Source PM Selection Strategy	VM Selection Criteria	Application of Prediction Technique	Destination PM Selection Strategy	Performance Evaluation Technique
Robust Server Consolidation: Coping with Peak Demand Underestimation [72]	SVMC	Centralized			Predictive	Greedy Heuristics	Simulation
Achieving Intelligent Traffic-Aware Consolidation of Virtual Machines in a Data Center Using Learning Automata [68]	SVMC	Centralized		Graph Partitioning Algorithm used to create VM cluster		Simulated Annealing based Meta-Heuristic	Simulation
Energy optimized VM placement in cloud environment [73]	SVMC	Centralized			Non-Predictive	Greedy Heuristics	Simulation in CloudSim
GLAP: Distributed Dynamic Workload Consolidation through Gossip-Based Learning [50]	DVMC	Distributed			Predictive		Simulation in PeerSim [74] with real Cloud workload traces
A Consolidation Strategy Supporting Resources Oversubscription in Cloud Computing [75]	SVMC	Centralized				Greedy Heuristic FF	Testbed
Bayesian networks-based selection algorithm for virtual machine to be migrated [57]	DVMC	Centralized	Adaptive Threshold	Bayesian Network Based Model to select Migrating VMs		Greedy Heuristic	Simulation in CloudSim
Performance-aware server consolidation with adjustable interference levels [44]	DVMC	Centralized	Static Threshold			Greedy Heuristics	Simulation
A Gossip-Based Dynamic Virtual Machine Consolidation Strategy for Large-Scale Cloud Data Centers [49]	DVMC	Distributed	Adaptive Threshold		Non-Predictive	Greedy Heuristic	Simulation in PeerSim [74]

Research Project	**VM consolidation Type**	**VM consolidation Decision Process**	**Source PM Selection Strategy**	**VM Selection Criteria**	**Application of Prediction Technique**	**Destination PM Selection Strategy**	**Performance Evaluation Technique**
Energy-efficient migration and consolidation algorithm of virtual machines in data centers for cloud computing [76]	DVMC	Centralized	Static Threshold	The VM with the highest resource demand is selected from overloaded PMs, all VMs are selected from underloaded PMs	Non-Predictive	Meta-Heuristic	Simulation in CloudSim
SOC: Satisfaction-Oriented Virtual Machine Consolidation in Enterprise Data Centers [77]	SVMC	Centralized				Greedy Heuristic	Simulation
Virtual machine placement optimizing to improve network performance in cloud data centers [78]	SVMC	Centralized				Metaheuristic	Simulation using C++
Virtual machine consolidated placement based on multi-objective biogeography-based optimization [79]	SVMC	Centralized				Metaheuristic	Simulation using MATLAB
An energy-aware heuristic framework for virtual machine consolidation in Cloud computing [80]	DVMC	Centralized	Adaptive Threshold	RS, MMT, MC, MU		Greedy Heuristic	Simulation in CloudSim

Research Project	**VM consolidation Type**	**VM consolidation Decision Process**	**Source PM Selection Strategy**	**VM Selection Criteria**	**Application of Prediction Technique**	**Destination PM Selection Strategy**	**Performance Evaluation Technique**
Dynamic Virtual Machine Consolidation for Energy Efficient Cloud Data Centers [81]	DVMC	Centralized	Static Threshold	VMs from underutilized hosts	Predictive	Meta-Heuristic	Testbed
Correlation-based virtual machine migration in dynamic cloud environments [82]	DVMC	Centralized	Static Threshold	Maximum Correlation	Predictive	Greedy Heuristic	Simulation
VM consolidation approach based on heuristics, fuzzy logic, and migration control [83]	DVMC	Centralized	Static Threshold	Fuzzy VM selection	Predictive	Greedy Heuristic	Simulation in CloudSim
Server Consolidation with Minimal SLA Violations [84]	DVMC	Centralized	Static Threshold	MMT	Non-Predictive	Heuristic	Simulation in CloudSim

2.11 Research Review Findings and Research Challenges

The summary of our research review findings are as follows:

- Both of SVMC and DVMC algorithms are crucial for energy-efficiency of CDC. SVMC is the first step towards limiting the energy consumption, while DVMC further minimizes the energy consumption while increasing the resource utilization of CDC.
- CDVMC algorithms have been found as more popular than DDVMC algorithms. For P2P networks, DDVMC algorithms are useful. Although, DDVMC algorithms are more robust and reliable in case of hardware failure; however, it poses more network overhead compare to CDVMC algorithms.
- Threshold-based approach is extremely popular. However, since Threshold-based approach limits the search space by selecting PMs based on the threshold value, therefore stagnation in local minima or local maxima may arise.
- One of the drawbacks of VM consolidation is that SLA violation may arise because of aggressive consolidation. STDVMC algorithms cannot control the SLA violation. In contrast, ATDVMC algorithms limits SLA violation by prior migration VMs from potential overloaded PMs. However, in terms of minimization of energy consumption, ATDVMC algorithms is less efficient and it causes more number of VM migrations.
- MMT is the most widely used VM selection strategy, as with the decrease of migration time, service disruption time decreases, which certainly lowers the SLA violation.
- A wide range of prediction techniques have been proposed in the literature to estimate the future resource demand of VMs a well as future resource utilization of PMs, as PDVMC algorithms minimizes SLA violation more than NPDVMC Algorithms. However, PDVMC algorithms may exert the overhead of more VM migrations.
- Unlike meta-heuristic algorithms, greedy based heuristic algorithms may become stagnant in local minima or local maxima, yet these heuristic algorithms provide acceptably sub optimal solution in quick time. Among all the meta-heuristics, Evolutionary algorithms has appeared to be most popular. Considering both heuristics and meta-heuristics, Modified version of Best Fit Decreasing is found as most prevalent destination PM selection algorithm.

From our extensive study, we have found following potential research openings, which are yet to be explored:

- User provided information is used in reservation and pricing scheme. Based on our literature study, researchers are yet to contemplate on how CSUs can contribute in minimizing CDC energy consumption. Hence, utilization of CSU provided information in effective minimization of CDC energy consumption is a challenging research direction.
- From examining such contemporary DVMC algorithms as [45, 49, 51-53, 64, 85-96], we have found that these algorithms use legacy PMs, such as HP ProLiant ML110 G4 [97] and HP ProLiant ML110 G5 [98] for performance validation. For legacy PMs, energy-efficiency increases as the load increases. However, for modern highly energy proportional PMs, such as Dell PowerEdgeR940 (Intel Xeon Platinum 8180, 112 cores⇥25000 MHz, 384 GB) [99], HP ProLiant DL560 Gen10 (Intel Xeon Platinum 8180, 112 cores⇥25000 MHz, 384 GB) [100] and HP ProLiant ML350 Gen10 (Intel Xeon Platinum 8180, 28 cores⇥25000 MHz, 192 GB) [101], energy-efficiency rather drops beyond 70% utilization level. As such, several researchers [102, 103] argue that consolidation towards maximum increase of PM resource utilization does not feature the optimal minimization of CDC energy consumption with new generation of highly energy proportional PMs. To the best of our knowledge, no DVMC algorithm has addressed this issue.
- The effectiveness of PDVMC algorithms hinges on the actual correlation between past and future resource demand and the quality of adjustment with a specific workload type. However, different VMs co-hosted in in a single PM have varied behavioral pattern in terms resource demand. Hence, no single prediction technique would fit for all PMs, whereas existing PDVMC algorithms apply a common prediction technique for all PMs. Furthermore, there is always a possibility of inaccurate prediction because of potential mismatches between past and present. Hence, there exists a research gap, which is yet to be addressed.
- Apart from MMT, rest of the VM selection algorithms only consider CPU demand of VMs and ignore the memory, network bandwidth and disk I/O requirement of VMs. However, as argued by authors of [59], selecting a VM only based on CPU will cause saturation in terms of CPU and can lead towards no further improvement in utilization while leaving other types of resource underutilized. It is highly challenging to determine a single converging point representing the equivalent total resource demand of multitude of resource types, while different types of resources represent different dimensions.

2.12 Summary

In this chapter, we have critically reviewed multitude of VM consolidation algorithms with varied viewpoints, as we have also highlighted different aspects of most contemporary VM consolidation algorithms. Furthermore, analyzation of VM consolidation algorithms has provided prominent research directions. With the spirit of addressing those critical research challenges found from the literature, in Chapter 3 of this dissertation, we have proposed a novel DVMC algorithm, namely *RTDVMC*, which minimizes CDC energy consumption through exploiting CSU provided information. Next, in Chapter 4, we have brought forth a novel DVMC algorithm, namely *SRTDVMC*, which address the issue of changed energy-efficiency characteristics of modern highly energy proportional PMs. Finally, in Chapter 5, we have proposed a predicted release time based DVMC algorithm, namely *PRTDVMC*, which is designed to address issues with *RTDVMC* and *SRTDVMC*.

3. User Information Based Dynamic Consolidation of Virtual Machines for Energy-Efficient Cloud Data Centers

3.1 Introduction

In the previous chapter, we have presented the limitations of existing DVMC algorithms. We have discussed that CSU provided information play a crucial role in many aspects of Cloud resource management system, such as resource scheduling, resource reservation and reservation-based pricing scheme. However, from our extensive literature review on VM consolidation algorithms [104], we have discovered that incorporation of CSU provided information in DVMC algorithm and its impact on energy-efficiency of CDC has not been investigated yet. Since, VM consolidation is a NP-Hard problem, no VM consolidation algorithm can guarantee or provide optimal solution in polynomial time. With this fact being underlined, we have presented a novel heuristic DVMC algorithm, referred to as *Release Time based DVMC* (*RTDVMC*) that utilizes CSU provided information to make more efficient VM consolidation decision in terms of reducing CDC energy consumption to promote green cloud. This chapter is derived from the following book chapter.

- *Khan MA, Paplinski AP, Khan AM, Murshed M, Buyya R. Exploiting user provided information in dynamic consolidation of virtual machines to minimize energy consumption of cloud data centers. 2018 Third International Conference on Fog and Mobile Edge Computing (FMEC); 2018 23-26 April 2018.*

Before presenting our proposed algorithm, in section 3.1, we have explained various important concepts and terms used in the algorithm, followed by mathematical modelling of different components of VM consolidation in section 3.2. In section 3.3, we have articulated and elaborately explained our proposed heuristic DVMC algorithm, *RTDVMC*. In section 3.4, empirical evaluation technique of the proposed algorithm has been presented, followed by analysis of experimental outcome has been highlighted in Section 3.5. Finally, in Section 3.6, we have summarized our research.

3.2 CSU provided information and Important Concepts

In our proposed DVMC algorithm, the CSU provided information is one of key distinguishing features used in VM consolidation decision process. To ensure easy understanding of our proposed algorithm in the following section, we have first explained the assumptions, key terms, CSU provided information as well as notations that have been used in the proposed algorithm.

3.2.1 VM Release Time, PM Release Time and Assumptions

The information that would be required to receive from a CSU is the time when the CSU would release the resources that are in his acquisition. Since, DVMC algorithm attempts to dynamically select VMs to migrate into fewer number of active or turned on PMs, hence, for our DVMC algorithm, we consider VM as the resource of a CSU and the received information from a CSU is the time when the respective VM(s) can be removed from CDC. We refer to such time as *VM Release Time* (*VMRT*). We assume that every VM would have *VMRT*, which would be provided prior by respective CSU.

Apart from *VMRT*, another crucial term noteworthy explaining is *PM Release Time* (*PMRT*). *PMRT* refers to the time when a PM can be either shut down or put into a sleep state that would consume no energy, or lower amount of energy compared to its active state. A PM can be shut down or put into sleep state, if it has either no VM hosted on it, or none of its hosted VMs is in the active state. Since *VMRT* refers to the maximum time until which the VM would be in the active state, hence *PMRT* denotes the maximum *VMRT* value among all the *VMRT* values of VMs that are hosted in that PM.

3.3 Modelling Resource Utilization, Constraints, Energy Consumption and Objective Function

Using the notations presented previously in Nomenclature section at the outset of this thesis, we have first explained the modelling of the resource utilization and the constraints used in proposed *RTDVMC* algorithm followed by the explanation of power consumption by CDC and objective function.

3.3.1 Modelling Resource Utilization

DVMC algorithm migrates VMs from one PM to another PM, so that VMs would be placed in minimum number of PMs and thus increase resource utilization and energy-efficiency. VMs hosted in a PM utilizes the resources of that PM. Therefore, resource utilization of a PM corresponds to the total resource demand by all the VMs hosted in that PM. Let, U_i^k denotes utilization of Resource type, R_k of PM, P_i. Hence, the equation for calculating U_i^k [59] is as follows:

$$U_i^k = \sum_{j=1}^{|V|} D_j^k \cdot x_{i,j} \tag{3.1}$$

where D_j^k denotes Demand of Resource type, R_k by VM, V_j, $x_{i,j}$ denotes the element of placement matrix, x and value of $x_{i,j}$ is determined as follows:

$$x_{i,j} = \begin{cases} 1, & if\ V_j\ is\ placed\ in\ P_i \\ 0, & otherwise \end{cases} \tag{3.2}$$

3.3.2 Modelling Resource Constraint

Since, Resource Capacity of a PM, P_i is fixed and it cannot provide additional resources to its hosted VMs than its capacity, Therefore, a VM, V_j can only be placed in a PM, P_i if the amount of available resources in P_i is adequate to meet the resource demand of V_j. Hence, V_j can only be placed in P_i if the following equation is satisfied [59]:

$$C_i^k - U_i^k \geq D_j^k \tag{3.3}$$

In other words, a PM, P_i cannot host a VM, V_j if the available resource of P_i is lesser than the resource demand of V_j. We denote such constraint presented in (3.3) as *Resource Constraint* (*RC*).

3.3.3 Modelling *O-UPM* and Maximum Utilization Threshold Constraint

At times, workload of VMs could rise very high resulting in steep resource utilization of the hosting PM. We denote such PM with heavy resource utilization as *O-UPM* and use a threshold, referred to as *Maximum Threshold*, θ_{max} to distinguish whether a PM is *Over-utilized* or not. Let us denote *OP* as a set of *O-UPMs* then,

$$OP = \{P_i \mid U_i^k \geq \theta_{max} \; for \; any \; R_k \in R \; and \; 1 \leq i \leq |P|\} \quad (3.4)$$

If VM(s) were not migrated out of an *O-UPM, then* SLA violation would unfold. In order to avoid causing SLA violation, during destination PM selection for a migrating VM, it is essential to ensure that hosting the migrating VM would not turn the destination into an *O-UPM,* which we have modelled through the following equation:

$$D_j^k + U_i^k < \theta_{max} \quad (3.5)$$

We refer such constraint presented in (3.5) as *Maximum Utilization Threshold Constraint* (*MUTC*).

3.3.4 Modelling Energy Consumption

Most of the existing VM consolidation algorithms have mentioned that energy consumption of a PM is primarily dominated by its CPU utilization [52, 59]. Hence, our energy consumption model is a function of CPU utilization (3.6), where, E_i denotes the energy consumption by PM, P_i.

$$E_i = f\left(U_i^{CPU}\right) \quad (3.6)$$

Based on (3.6), we can determine the total energy consumption of the CDC through (*3.7*), where E_{CDC} denotes the total energy consumption of the CDC.

$$E_{CDC} = \sum_{i=1}^{|P|} E_i \quad (3.7)$$

In order to relate closely to the real world energy consumption by PMs, for our energy consumption model, we have opted to draw energy consumption benchmark results of two different types of PMs: Hewlett-Packard Company ProLiant ML110 G4 [97] and Hewlett-Packard Company ProLiant ML110 G5 [98]. In Table 3.1, we have articulated respective energy consumption of these two types of PMs at varying load level [97, 98]. Based on U_i^{CPU} of P_i, we can determine the respective E_i from Table 3.1. After measuring E_i for each P_i, we can determine the total energy consumption of the CDC by following (3.7).

3.3.5 Objective Functions

Due to decreased resource demand by hosted VMs, a PM's resource utilization may drop. We denote such PM with low resource utilization as *Under-utilized PM* (*U-UPM*). If all VMs from an *Under-utilized PM* can migrated out into new PM, which is not *Over-utilized,* then that *Under-utilized* PM, which would now have no VM, can be put either into a lower energy consuming state, such as sleep state or shut down state and hence, energy consumption can be minimized. Since, aim of DVMC is to minimize energy consumption, hence, the objective function of *RTDVMC*, f_R (3.8) is as follows, where $\bar{E}_{CDC}$ denotes mean CDC energy consumption:

$$f_R = \min \bar{E}_{CDC} \tag{3.8}$$

Thus far, we have modelled various components of our proposed DVMC algorithm. In the following section, we have articulated our proposed solution.

Table 3.1 Energy Consumption Values of Contemporary Servers at Different Load Level [97, 98]

	Energy Consumption (kW) at Different Percentage of Load Level										
	Sleep	10%	20%	30%	40%	50%	60%	70%	80%	90%	100%
Hewlett-Packard Company ProLiant ML110 G4 [97]	86	89.4	92.6	96	99.5	102	106	108	112	114	117
Hewlett-Packard Company ProLiant ML110 G5 [98]	93.7	97	101	105	110	116	121	125	129	133	135

3.4 Proposed Solution

VM consolidation attempts to consolidate VMs in minimum number of PMs without violating any of the PMs' resource capacity, so that the total number of active PMs and subsequent energy consumption can be minimized. Multi-dimensional Vector Packing Problem (MDVPP) refers to an NP-hard combinatorial optimization problem in which a set of items is required to be packed into minimum number of bins without violating any of the bins' capacities. Note that, if we refer to bins as PMs and items as VMs that are needed to be packed into minimum possible bins or PMs, given that none of bins or PMs' resource capacity is violated, then VM consolidation turns into a NP-hard problem. As such, our proposed heuristic-based DVMC algorithm, *RTDVMC*, which aims to minimize the number of active PMs is presented in the following.

3.4.1 Two Phases of *RTDVMC* Algorithm

In this section, we have explained our proposed *RTDVMC* algorithm. The objectives of

Algorithm 3.1 The *RTDVMC* Algorithm

```
Input: P, V, R, SP
Output: VM Placement
The first phase: O-UPMs
1:  for each $P_i$ in P do
2:      if (3.4) is satisfied then
3:          $OP \leftarrow \{P_i \cup OP\}$
4:      end if
5:  end for
6:  for each $P_o$ in OP do
7:      migratingVMs ← Invoke the VSO algorithm with $P_o$
8:      vmsToMigrate ← {migratingVMs ∪ vmsToMigrate}
9:  end for
10: Sort vmsToMigrate in the order of decreasing VMRT
11: $NOP \leftarrow \{P - OP - SP\}$
12: for each $V_l$ in vmsToMigrate do
13:     $P_d$ ← Invoke the DPSVO algorithm with $V_l$ and NOP
14:     destinationPMs ←{$P_d$ ∪ destinationPMs}
15:     if $P_d$ is in SP then
16:         $SP \leftarrow \{SP - P_d\}$
17:         $NOP \leftarrow \{P_d \cup NOP\}$
18:     end if
19: end for
The second phase: U-UPMs
20: candidateSources ← {$P - OP - SP$ − destinationPMs}
21: candidateDestinations ← {$P - OP - SP$}
22: Sort candidateSources in the order of increasing PMRT
23: for each $P_c$ in candidateSources do
24:     candidateDestinations ← {candidateDestinations − $P_c$}
25:     destinations ← Invoke the DPSVU algorithm with $P_c$ and candidateDestinations
26:     candidateSources ←{candidateSources − destinations}
27: end for
```

migrating VMs out of a PM and then placing those migrating VMs into new destination PMs are twofold: *Firstly*, limiting SLA violations, which are taking place in *O-UPMs* and *Secondly*, limiting the total number of active PMs by migrating VMs of a *U-UPM* into new appropriate *U-UPMs*, so that the *U-UPM* having no VM can be put into sleep state. Hence, *RTDVMC* algorithm works in two phases. In **the first phase**, SLA violations of *O-UPMs* are limited through migrating out required number of VMs from those *O-UPMs* (Line 1 to 19 of Algorithm

3.1); while in **the second phase**, VMs from a range of *U-UPMs* are migrated out and then consolidated in lesser number of active PMs (Line 20 to 27 of Algorithm 3.1). In the following section, we have comprehensibly discussed each of these phases and its components:

3.4.1.1 The first phase *O-UPMs*

In order to identify *O-UPM*s, we compare a PM's resource (i.e., CPU, RAM and Bandwidth) utilization (*3.1*) to a fixed threshold value, θ_{max}. As expressed in (*3.4*), if for any resource type, the resource utilization of a PM is found equal or greater than θ_{max}, then the PM is referred to as *O-UPM* (Line 1 to Line 5 of Algorithm 3.1). Next, VM(s) are selected from *O-UPM*s to migrate out into new destination PMs, which we have discussed in the following section.

- **VMs Selection From *O-UPMs***

The *VSO* algorithm, articulated as Algorithm 2, provides the solution to select VM(s), which would be migrated out from an *O-UPM*. In order to select VM(s) from an *O-UPM*, a list of VMs sorted in the order of decreasing *VMRT* is first prepared (Line 1 of Algorithm 3.2) (i.e., the first VM of the sorted list has the largest *VMRT* value, while the second VM of the sorted list has the second largest *VMRT* value and so forth) and then, the first VM of the sorted list is selected to migrate out, followed by the second VM and so forth, until the resource utilization of the *O-UPM* drops below θ_{max} (Line 2 to 11 of Algorithm 3.2).

Algorithm 3.2 The VMs Selection From *O-UPM* (*VSO*) algorithm

```
Input: The O-UPM, $P_o$
Output: List of VMs to be migrated out from the given O-UPM
 1:  Sort $V^o$, the set of VMs of $P_o$ in order of decreasing VMRT
 2:  $q \leftarrow 1$
 3:  while $q \leq |V^o|$ and $\theta_{max} < U_o^k$ for any $R_k$ in $R$ do
 4:      $migratingVMs \leftarrow \{V_q^o \cup migratingVMs\}$
 5:      for each $R_k$ in $R$ do
 6:          $U_o^k \leftarrow U_o^k - D_q^k$
 7:          $C_o^k \leftarrow C_o^k + D_q^k$
 8:      end for
 9:      $q \leftarrow q + 1$
10:  end while
11:  return migratingVMs
```

For every *O-UPM*, such sorted list of migrating VMs is created and then, combining all those lists a single list of VMs to be migrated out of all *O-UPM*s is created, which is further sorted in descending order of *VMRT* (Line 6 to 10 of Algorithm 3.1). We denote the list as *vmsToMigrate*. The reason to migrate out VMs with greater *VMRT* is to minimize the duration of active state of *O-UPM*s, so that at a particular point in time, more PMs can be found that are in sleep state, which would lower the energy consumption.

- **Destination PMs Selection For VMs of *O-UPMs***

After creating the sorted list of migrating VMs from *O-UPM*s (i.e., *vmsToMigrate*), new destination PMs for those migrating VMs are selected through executing the *DPSVO* algorithm (referred to as Algorithm 3.3).

Algorithm 3.3 The Destination PM Selection for VM of *O-UPM* (*DPSVO*) algorithm

```
Input: The VM $V_j$ to be migrated out from an O-UPM
Input: NOP
Output: The new destination PM for the given migrating VM
1:  Sort NOP in the order of increasing PMRT
2:  for each $P_x$ in NOP do
3:      suitable ← Invoke the PST algorithm with $P_x$ and $V_j$
4:      if suitable is true then
5:          return $P_x$
6:      end if
7:  end for
8:  return the most energy-efficient $P_s$ from SP
```

The new destination PMs are chosen from the set of PMs, which are neither Over-utilized PMs nor inactive PMs (i.e., the PMs that are currently in the sleep state or in the switched off state). We denote such a set of PMs as *NOP* (Line 11 of Algorithm 3.1). The PMs belong to NOP are first sorted in the order of increasing *PMRT* (Line 1 of Algorithm 3.3). Next, for the first VM of *vmsToMigrate*, it is checked whether the first PM of sorted *NOP* is suitable to host that PM or not. The *PST* algorithm, listed as Algorithm 3.4 is invoked to decide whether the PM is suitable to host the given VM.

Algorithm 3.4 The PM Suitability Test (*PST*) algorithm

```
Input: PM, VM
Output: A decision whether the given PM can host the given VM
if (3.3) and (3.5) are satisfied for all $R_k$ in $R$ then
    return true
end if
return false
```

As per *PST* algorithm, if the PM satisfies both *RC* (3.3) and *MUTC* (3.5) constraints, then the PM is considered as suitable (Line 1 to 4 of Algorithm 3.4). If the first PM of sorted *NOP* is found as unsuitable, then the second PM of sorted *NOP* is checked, followed by the third PM and so forth, until a suitable PM is found (Line 2 to 7 of Algorithm 3.3). However, if none of the PMs from sorted *NOP* is found as suitable, then the most energy-efficient PM is chosen from the set of inactive PMs (i.e., the PMs, which are currently in the sleep state or in the switched off state) to host the migrating VM (Line 8 of Algorithm 3.3). The same process is repeated to select a destination PM for the second VM of *vmsToMigrate*, followed by the third VM and so forth (Line 12 to 19 of Algorithm 3.1). Thus, new destination PMs are selected for all migrating VMs of *O-UPM*s.

Although, selecting the VM with highest *VMRT* to migrate out from *O-UPM*s minimizes the active duration of *O-UPM*s; one might argue that in worst-case scenario, it can pose the increased active duration on destination PMs. To alleviate such worst-case scenario, we have followed the approach to sort all the migrating VMs of *O-UPM*s in descending order of *VMRT* and sort all the candidate destination PMs in ascending order of *PMRT* and then compared the highest *VMRT* against the lowest *PMRT* first, followed by the second lowest *PMRT* and so forth, which ensures that the hosting of a new VM would incur the least amount of increased active duration of destination PMs.

However, given *RC* (3.3) and *MUTC* (3.5) constraints, in best-case scenario, without any increase of active duration of the destination PM, which is going into sleep state first compared to all other candidate PMs, will finish the most time-consuming task before it moves into sleep state, while leaving the shorter tasks for rest of candidate PMs. In other words, considering *RC* (3.3) and *MUTC* (3.5) constraints, the destination PM selection process would provide the best-fit solution in terms of maximizing utilization of a PM before it moves into sleep state or switched-off state, since the most time-consuming job is finished by the PM, which is going to be into sleep state faster than the rest of the candidate PMs. Hence, by ensuring the increased utilization of active PMs that are going into sleep state quicker, the probability of increased active duration of rest of the PMs due to upcoming workload is minimized, which limits the total number of active PMs at a particular point in time. Furthermore, if no suitable PM is found in the list of active PMs of *NOP*, then that switched off PM or the PM, which is in sleep state is selected, which would exert the least amount of increase in energy consumption due to hosting the migrating VM. Thus, energy-efficiency is ensured.

3.4.1.2 The Second phase *U-UPMs*

After completion of VM migrations from *O-UPM*s, the algorithm enters into **the second phase** where VMs from *U-UPM*s are attempted to consolidate in fewer number of PMs (Line 20 to 27 of Algorithm 3.1). The PMs with resource utilization lower than θ_{max} are denoted as *U-UPM*s. As previously discussed, workload of *U-UPM*s might drop over time, allowing to pack more VMs in it. Hence, if VMs of an *U-UPM* can be migrated out into rest of the *U-UPM*s, then that *U-UPM*, which would not be having any VM, can be either shut down or put into sleep state, resulting in reduced number of active PMs and lower energy consumption. We aim to maximize the utilization of an *U-UPM* without increasing its active duration, which eventually would minimize the load of rest of the PMs. Consequently, with reduced workload remaining, it would be possible to consolidate more VMs in fewer number of *U-UPM*s than before. Hence, more *U-UPM*s can be put in the sleep state, resulting reduced energy consumption.

To implement this strategy in our proposed *RTDVMC* algorithm, we first prepare the set of PMs, which are potential candidates as *U-UPM*s, denoted by *candidateSources* (CS) and the set of candidate destination PMs, denoted by *candidateDestinations* (CD), which can potentially host those VM(s) that would be migrated out from *U-UPM*s. Next, we assign all the PMs in both CS and CD except *O-UPM*s and the set of PMs that are in sleep state or in switched off state (denoted by *SP*), so that *O-UPM*s do not turn into *O-UPM*s again due to hosting more VMs, while not waking up the PMs, which are in sleep state or switched off state would complement the energy consumption minimization. In addition, from CS, we remove the set of PMs, which hosted those VMs that were migrated out of *O-UPM*. These set of PMs is denoted by *destinationPMs* (Line 20 to 21 of Algorithm 3.1). Since, VMs may migrate out from an *U-UPM*, therefore, if PMs of *destinationPMs* were selected as *U-UPM*s, then a number of such VMs might migrate out from PMs of *destinationPMs*, which had already been migrated out once in the first phase of the algorithm. Hence, re-migration would have taken place, which would incur the increased SLA violation overhead.

- **Source *U-UPMs* Selection**

After preparing the set of candidate source *U-UPM*s (i.e., CS) and set of potential destination PMs to host migrating VMs of *U-UPM*s (i.e., CD), we reach to the state where one or more source *U-UPM*s belong to CS are selected from which VM(s) would be migrated out into suitable destination PMs of CD. In order to do so, we first sort the PMs of CS in increasing order of *PMRT* (Line 22 of Algorithm 3.1) and then we start to select destination PMs for the VM(s) of the first PM of sorted CS (i.e., the PM with lowest *PMRT* or the PM, which is going into sleep state first), followed by the second PM and so forth (Line 23 to 25 of Algorithm 3.1). The migrating VMs selection and the corresponding destination PMs selection process are accomplished through the *DPSVU* algorithm, referred to as Algorithm 3.5 and is articulated in the following section.

- **Selection of Migrating VMs and Corresponding Destination PMs**

In the destination PMs selection part for migrating VMs, we start with the first PM of sorted CS, followed by the second PM and so forth. The PM, which is selected to be checked whether its VMs can be migrate out into suitable destination PMs or not, we first remove that PM from CD, since a source PM cannot be the destination PM of VMs that were migrated out from itself in the first place (Line 24 of Algorithm 3.1). Next, we sort the VMs of that selected PM in the order of decreasing *VMRT* (Line 1 of Algorithm 3.5) and start searching for a suitable PM for the first VM with highest *VMRT* (Line 2 of Algorithm 3.5). The rationale to select the VM with highest *VMRT* is that it would shorten the active duration of the source PM.

Algorithm 3.5 The Destination PMs Selection for VMs of *U-UPMs* (*DPSVU*) algorithm

```
Input: A U-UPM, $P_c$ from the set of candidateSources
Input: candidateDestinations
Output: List of new destination PMs to host migrating VMs
1:   Sort $V^c$, the set of VMs of $P_c$ in order of decreasing VMRT
2:   for each $V_n^c$ in $V^c$ do
3:       $P_d$ ← null
4:       Sort candidateDestinations in order of increasing PMRT
5:       for each $P_m$ in candidateDestinations do
6:           suitable ← Invoke the PST algorithm with $P_m$ and $V_n^c$
7:           if suitable is true
8:               $P_d$ ← $P_m$
9:               hostList ← {$P_d$ ∪ hostList }
10:              break loop
11:          end if
12:      end for
13:      if $P_d$ is null then
```

```
14:         break loop
15:       end if
16:   end for
17:   return hostList
```

In order to find the respective destination PM for a VM, the PMs of CD are first sorted in the order of increasing *PMRT* (Line 4 of Algorithm 3.5) and checked if the first PM of sorted CD is suitable to host the VM or not. The *PST* algorithm, listed as Algorithm 3.4 is utilized to check if a PM is suitable for hosting the given VM. However, if the first PM of sorted CD is found as unsuitable, then the suitability of the second PM of sorted CD is checked, followed by the third PM and so forth, unless a suitable PM is found (Line 5 to 12 of Algorithm 3.5). If no such suitable PM can be found in sorted CD, the VM is chosen not to be migrated out and the destination PM selection process for VMs of an *U-UPM* terminates (Line 13 to 16 of Algorithm 3.5). To explain more, if a VM with higher *VMRT* cannot be migrated out, then the rest of the VMs with lower *VMRT* are not migrated out, since a PM's active duration can only be lowered if the VM with highest *VMRT* among all of its hosted VMs can be migrated out.

One noteworthy point to underline is that after finding the set of new destination PMs, denoted by destinations for VMs of a source *U-UPM* through the *DPSVU* algorithm, those new destination PMs of destinations are removed from CS (Line 26 of Algorithm 3.1), which restricts unnecessary VM re-migration and consequent SLA violation as well as improves the running time of the *RTDVMC* algorithm.

3.4.2 Characteristics of *RTDVMC* Algorithm

In summary, the VM selection process from an *U-UPM* only selects VMs with higher *VMRT*, which minimizes the active duration of the source PM, while the destination PM selection process selects suitable destination PMs with higher *PMRT* than the source PM (Line 22 to 27 of Algorithm 3.1) ensuring that destination PMs' active duration is not extended because of hosting any migrating VM of a *U-UPM*. Consequently, active duration of source PM is shortened without exerting prolonged active duration of destination PM as it is illustrated in Fig. 3.1. To elucidate Fig. 3.1 further, there are two PMs, P_1 and P_2. Prior VM consolidation, P_1 had two VMs, $V_1^1$and V_2^1, while P_2 had two VMs, $V_1^2$and V_2^2. Release time of V_1^1, V_2^1, V_1^2 and V_2^2 are denoted by $T_{V_1^1}$, $T_{V_2^1}$, $T_{V_1^2}$ and $T_{V_2^2}$. Since, $T_{V_1^2} > T_{V_1^1} > T_{V_2^2} > T_{V_2^1}$, therefore, Release time of P_1 and P_2, denoted by T_{P_1} and T_{P_2} are equal to $T_{V_1^1}$ and $T_{V_1^2}$, respectively. Since, $T_{V_1^1} < T_{V_1^2}$, therefore, $T_{P_1} < T_{P_2}$. Consequently, *RTDVMC* selects the largest VM, V_1^1 in terms of release time from P_1 to migrate out to P_2. Since, $T_{V_1^1}$ is lower than T_{P_2}, therefore, hosting V_1^1 does not increase T_{P_2}, while T_{P_1} becomes smaller to $T_{V_2^1}$. Hence, at a particular point in time, $T_{V_2^1}$ the total number of PMs in sleep state would increase, yielding lower energy consumption. In addition, given *RC* (3.3) and *MUTC* (3.5) constraints, the destination PM selection process places a migrating VM in the PM, which is going to be in sleep state sooner than rest of the suitable PMs, resulting in increased resource utilization of a PM while it is in the active state. In other words, considering *RC* (3.3) and *MUTC* (3.5) constraints, *RTDVMC* provides the best fit solution in terms of maximizing utilization of an *U-UPM* before it moves into sleep state or switched off state; since, the longest job across time dimension is assigned to the PM, which is going to be switched into idle state soonest compared to all the available suitable PMs.

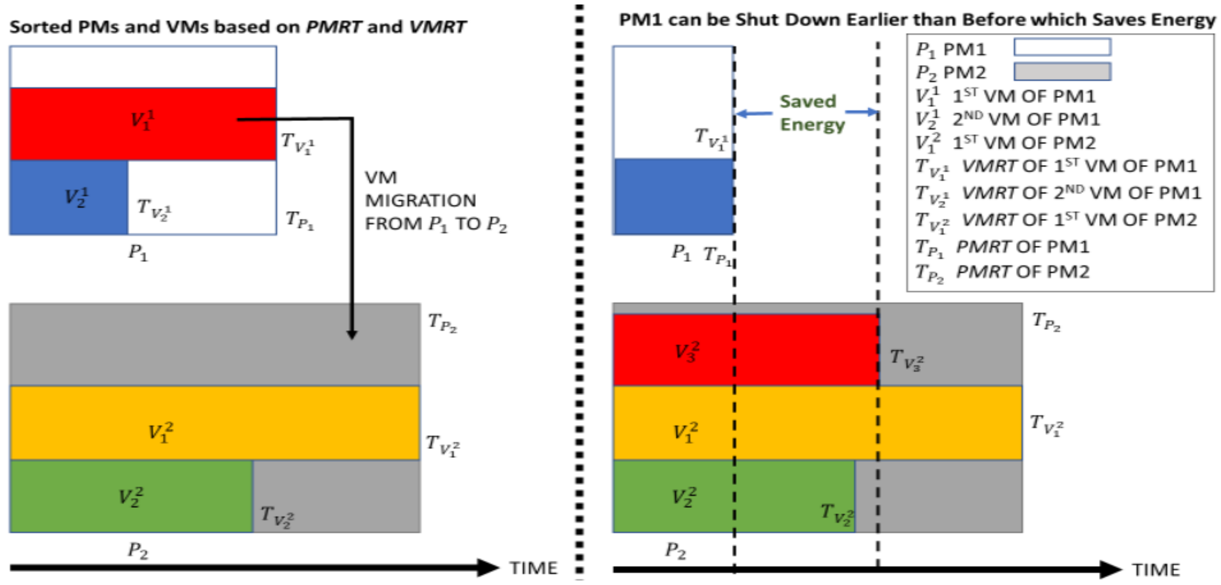

Fig. 3.1 How VM Consolidation Based on VM Release Time Minimizes Saves Energy

3.5 Performance Evaluation

To evaluate the performance of *RTDVMC*, we have modelled and simulated a cloud environment and implemented our proposed algorithm, *RTDVMC* in CloudSim [27]. Then, we have simulated *RTDVMC* algorithm in different workload scenarios. The other notable DVMC algorithms, namely, THR-MMT [51], MAD-MMT [45], IQR-MMT [45], LR-MMT [45], LRR-MMT [45] with which we have compared the performance of *RTDVMC*, have also been evaluated using CloudSim. The rationale of comparing *RTDVMC* with THR-MMT [51], MAD-MMT [45], IQR-MMT [45], LR-MMT [45], LRR-MMT [45] is explained in the following:

RC, MVM, HPG, MMT and MC are the most widely used VM selection algorithms incorporated in existing VMC algorithms. The RC algorithm has found to be the least energy-efficient. The issue with MVM and HPG is that, VMs are sorted with respect to CPU demand avoiding the consideration of other types of resources, such as RAM and Network bandwidth. However, selection of migrating VM(s) based on only one specific type of resource, negatively affects the resource utilization maximization in terms of other types of resources. Similar issue exists with the MC algorithm. Unlike MVM, HPG and MC algorithms, the MMT algorithm selects the VM having least migration time; resulting lower SLA violation. However, it does not consider energy consumption minimization aspect. Since, both the MMT algorithm and our proposed *RTDVMC* algorithm, consider time aspect of VMs to select migrating VMs, therefore, for performance comparison, we have selected those DVMC algorithms which uses the MMT algorithm as VM selection process. Based on our literature review, we have found THR-MMT [51], MAD-MMT [45], IQR-MMT [45], LR-MMT [45], LRR-MMT [45] as the pioneer and most popular MMT based DVMC algorithms.

3.5.1 Experimental Setup

To ensure the fairness in performance comparison of *RTDVMC* with THR-MMT, MAD-MMT, IQR-MMT, LR-MMT and LRR-MMT we have carried out our simulation using same environment in CloudSim in relation to CDC, VM, PM and energy module, as used by respective authors in their research. Hence, our simulated CDC is comprised of 800 heterogeneous PMs. Two different server configurations HP ProLiant ML110 G4 (Intel Xeon 3040, 2 cores⇥1860 MHz, 4 GB) [97], and HP ProLiant ML110 G5 (Intel Xeon 3075, 2 cores⇥2660 MHz, 4 GB) [98] have been used. Each server is provided with 1 GB/s network bandwidth. The energy consumption characteristics of these servers with varying workload is articulated in Table 3.1**.**

The characteristics of the different VM types match with the VMs used by THR-MMT, MAD-MMT, IQR-MMT, LR-MMT, LRR-MMT and correspond to Amazon EC2 instance types [105]. However, the difference between the simulated VMs and Amazon EC2 instance types is that the simulated VMs are single-core, which is explained by the fact that the workload data used for the simulations come from single-core VMs. Since, the single-core is used, the amount of RAM is divided by the number of cores for each VM type: High-CPU Medium Instance (2500 MIPS, 0.85 GB); Extra Large Instance (2000 MIPS, 3.75 GB); Small Instance (1000 MIPS, 1.7 GB); and Micro Instance (500 MIPS, 613 MB). At the outset, VMs are provided with the resources defined by the VM types. However, during the lifetime, VMs utilize less resources according to the workload data, widening opportunities for dynamic consolidation. With every single day of PlanetLab workload, each DVMC algorithm has been run twice to generate mean CDC energy consumption by that DVMC algorithm under such workload scenario. For each time, the simulation has been run until one-hour CloudSim simulation clock time. As chosen by authors of THR-MMT, the upper utilization threshold for such STDVMC algorithms as *RTDVMC* and THR-MMT is set to 80%. The parameters for such ATDVMC algorithms as MAD-MMT, IQR-MMT, LR-MMT and LRR-MMT are set to as mentioned by respective authors.

3.5.2 Performance Metric and Workload Data

The objective of DVMC is to minimize the energy consumption of CDC. Hence, we have evaluated the performance of *RTDVMC* in terms of CDC energy consumption and compared with that of THR-MMT, MAD-MMT, IQR-MMT, LR-MMT and LRR-MMT. As expressed in (3.7), CDC energy consumption is the sum of energy consumption of all the PMs, while each PM's energy consumption is derived from Table 3.1 according to its current CPU utilization.

In order to make a simulation-based evaluation applicable in real world, it is crucial to use workload traces from a real system in experiments [45]. Therefore, the performance of *RTDVMC* and other DVMC algorithms have been measured with real Cloud workload traffic traces representing time varying resource utilization. Real workload data is provided as part of the CoMon project, a monitoring infrastructure for PlanetLab [29]. Data of CPU usage of thousands of VMs has been collected every five minutes, while these VMs had been hosted in PMs spread globally across 500 locations. Four different days of PlanetLab workload from two different months: 3 March, 6 March, 12 April and 20 April of have been applied for performance testing. The characteristics of diverse days' workload data has been articulated in Table 3.2. The distribution of VMRT has been selected from the range of $[1\ day, 365\ days]$ through a uniformly distributed random variable.

3.6 Simulation Results and Result Analysis

Alongside performance evaluation and comparison of *RTDVMC* with other DVMC algorithms, we have embodied diverse inferential statistical techniques to prove the statistical

Table 3.2 Characteristics of PlanetLab Data (CPU Utilization)

Day	Number of VMs	Mean (%)	St. dev. (%)	Quartile 1	Quartile 2	Quartile 3
3 March	1052	12.31	17.09	2%	6%	15%
6 March	898	11.44	16.83	2%	5%	13%
12 April	1054	11.54	15.15	2%	6%	16%
20 April	1033	10.43	15.21	2%	4%	12%

significance of experimental outcome. In this section, we have first highlighted experimental results under different workload scenarios. Next, statistical significance of such experimental results has been examined through significance test followed by determination of population mean of performance improvement achieved through *RTDVMC* compared to existing DVMC algorithms. Finally, we have analysed the results.

3.6.1 Simulation Results

In Table 3.3, mean CDC energy consumption of diverse DVMC algorithms for four different days of PlanetLab workload: 3 March, 6 March, 12 April and 20 April has been articulated.

Table 3.3 Mean CDC Energy Consumption (kW) for Different DVMC Algorithms under Diverse Workload Scenarios

	Mean CDC Energy Consumption (kW)			
	3 March	**6 March**	**12 April**	**20 April**
RTDVMC	3.12	2.8	3.58	3.55
THR-MMT	3.21	3.04	3.86	3.48
MAD-MMT	3.63	3.13	4.23	4.3
IQR-MMT	3.49	2.7	3.98	3.71
LR-MMT	3.6	3.52	4.22	3.67
LRR-MMT	3.25	3.37	3.76	4.48

Fig. 3.2 – Fig. 3.5 represents performance comparison of *RTDVMC* with existing DVMC algorithms for different days of PlanetLab workload. The mean regulation of energy consumption achieved through *RTDVMC* compared to existing DVMC algorithms considering all of the different days is delineated in Fig. 3.6 and Fig. 3.7.

From experimental results, as portrayed in Fig. 3.6 and in Fig. 3.7, we can observe that *RTDVMC* significantly reduces CDC energy consumption compared to existing DVMC

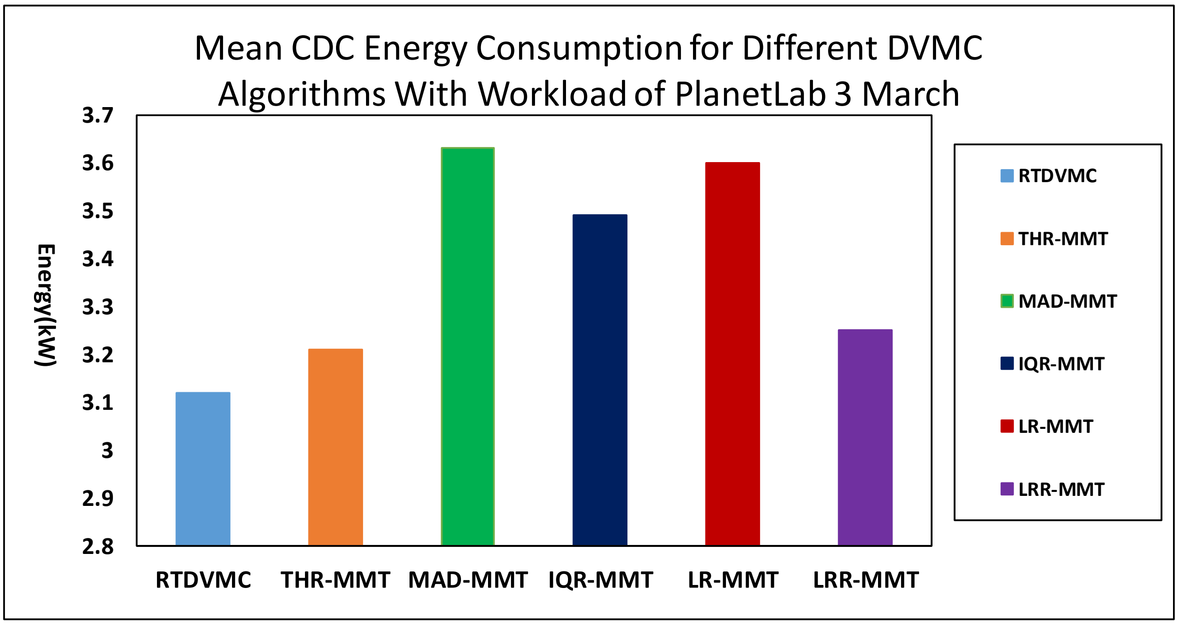

Fig. 3.2 Mean CDC Energy Consumption by *RTDVMC* and Existing DVMC Algorithms with PlanetLab Workload of 3 March

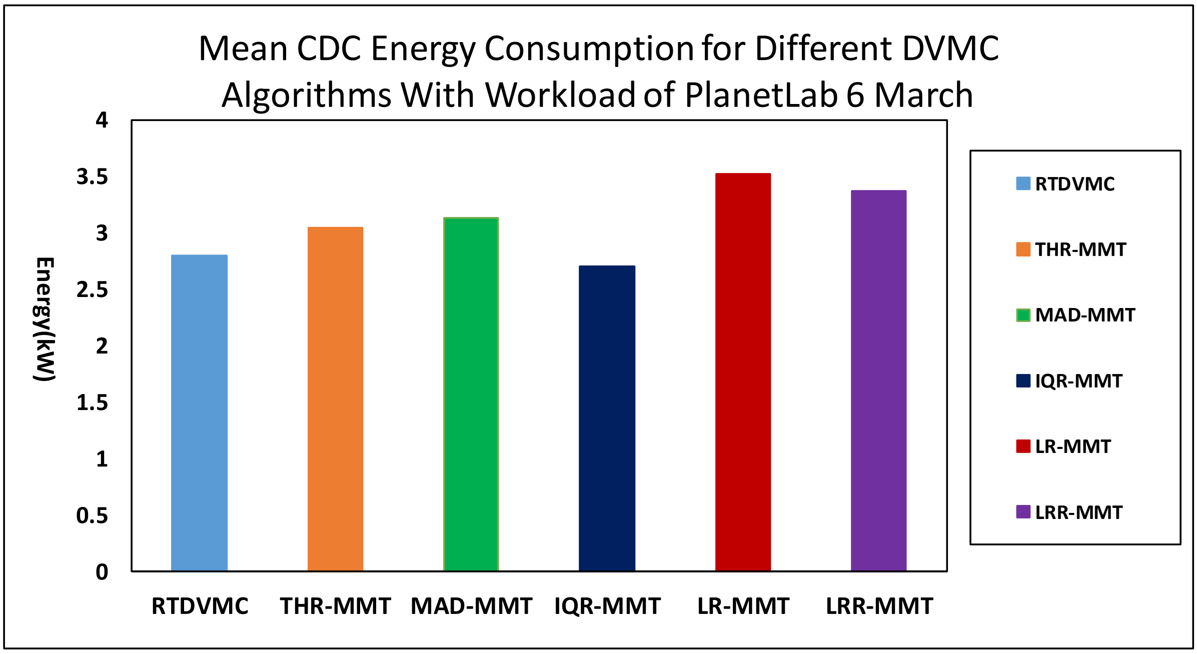

Fig. 3.3 Mean CDC Energy Consumption by *RTDVMC* and Existing DVMC Algorithms with PlanetLab Workload of 6 March

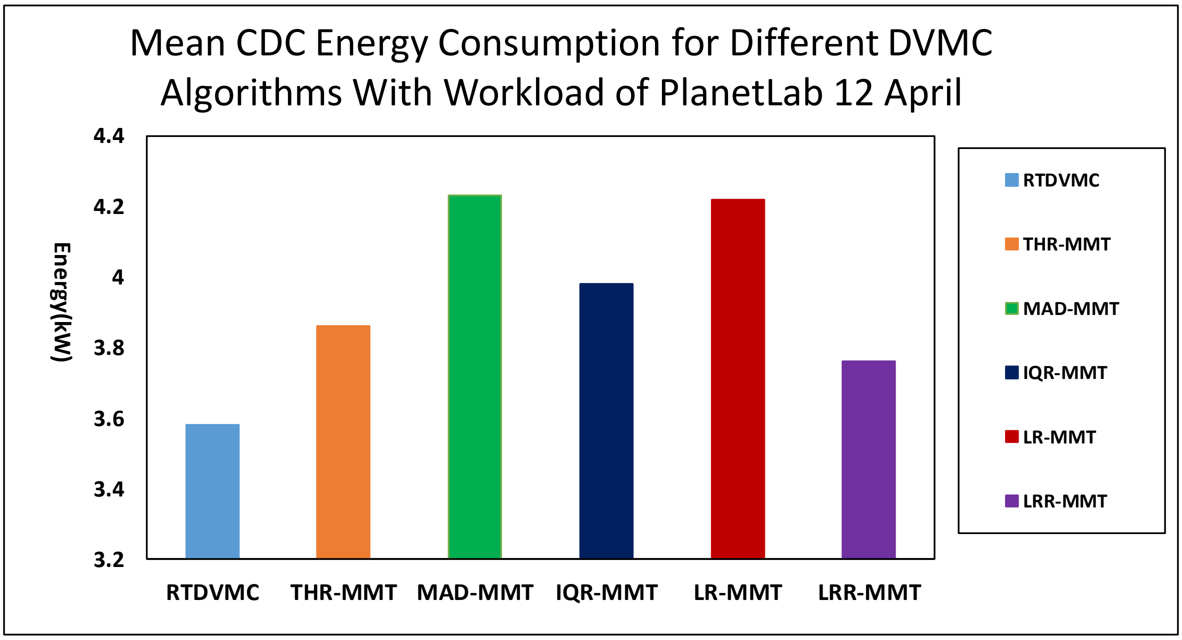

Fig. 3.4 Mean CDC Energy Consumption by *RTDVMC* and Existing DVMC Algorithms with PlanetLab Workload of 12 April

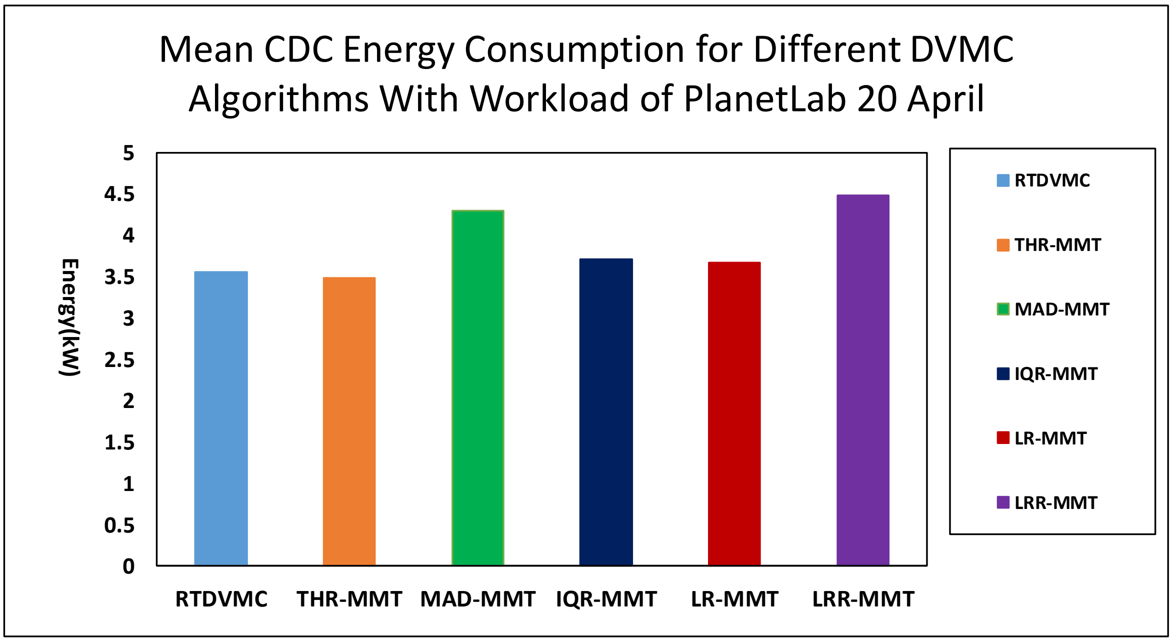

Fig. 3.5 Mean CDC Energy Consumption by *RTDVMC* and Existing DVMC Algorithms with PlanetLab Workload of 20 April

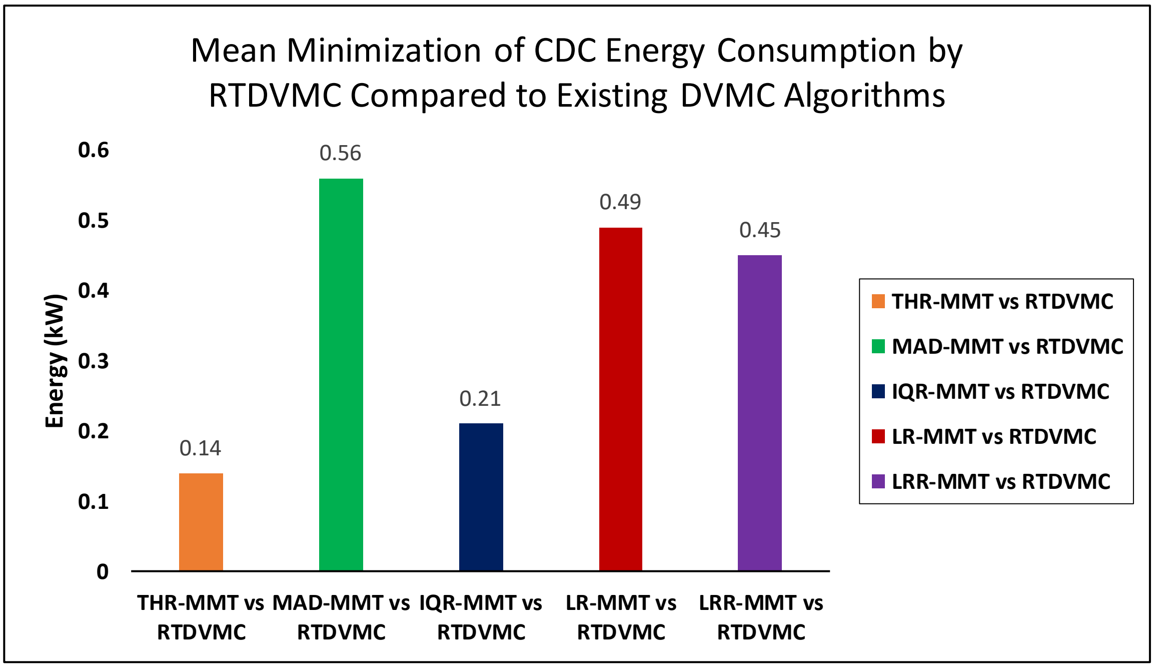

Fig. 3.6 Minimization of Mean CDC Energy Consumption (kw) by *RTDVMC* Compared to Existing DVMC Algorithms

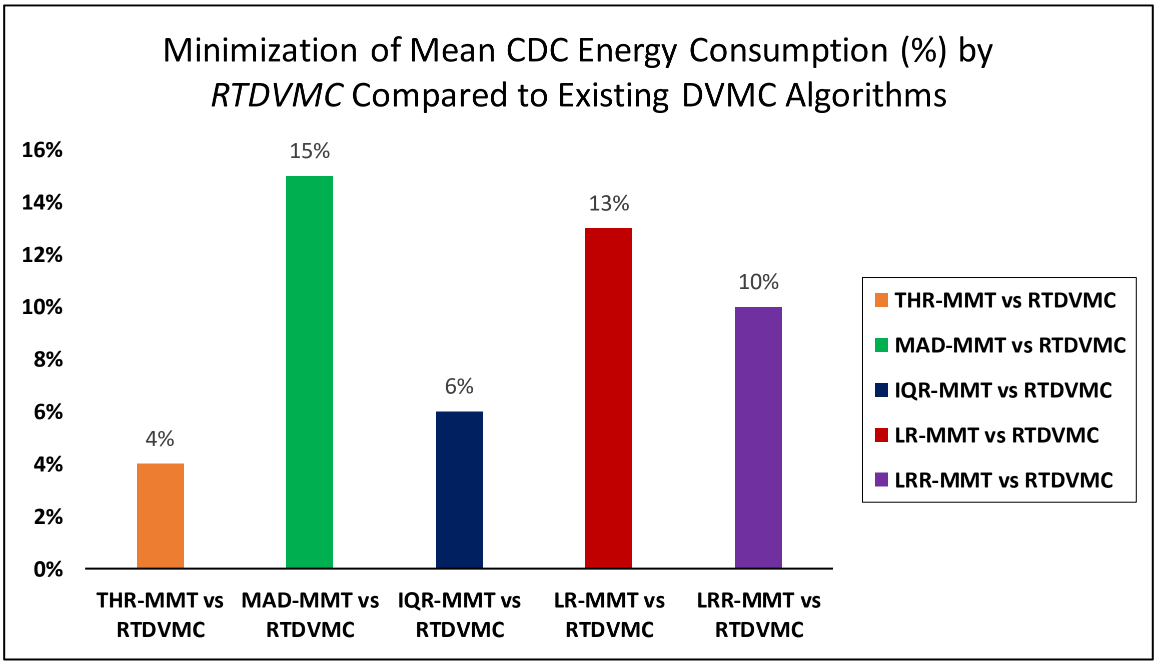

Fig. 3.7 Minimization of Mean CDC Energy Consumption (%) by *RTDVMC* Compared to Existing DVMC Algorithms

algorithms. However, such mean performance improvement is sample mean, whereas population mean is also important. Furthermore, one might reject the superiority of *RTDVMC* over existing DVMC algorithms based on the argument that no proof of statistical significance has been provided. To address such arguments, in the following section, we have presented diverse statistical testing.

3.6.2 Statistical Tests

Without any proof being present that the experimental result is statistically significant, readers may refuse to accept any claim made on the basis of such experimental outcome. We have hence performed the *t-test*, which is universally accepted to offer the evidence that the experimental outcome is statistically significant. Our experimental results suggest that *RTDVMC* is more effective in terms of regulating CDC energy consumption to existing DVMC algorithms. Hence, we aim to prove that the mean minimization of CDC energy consumption obtained through *RTDVMC* compared to existing literature as portrayed in Fig. 3.6 is statistically significant.

3.6.2.1 Normality and Significance Test

One critical point to note that the *t-test* cannot prove statistical significance of target data, if data is not normally distributed. To address that issue, normality testing is required to prove that the data representing mean minimization of CDC energy consumption obtained through *RTDVMC* compared to existing literature as portrayed in Fig. 3.6 is normally distributed. To elucidate the steps clearly, we have first presented mean CDC energy consumption with different days of PlanetLab workload for THR-MMT and *RTDVMC* along with corresponding minimization of CDC energy consumption in Table 3.4. Next, in the following section, we have elaborately discussed normality tests performed on the data presented in Table 3.4 to meet the prior condition of the *t-test*.

- **Normality Test**

Usage of the Shapiro-Wilk (S-W) normality test is prevalent in literature to test normality. In our research, we have utilized the software tool collected from [106], to perform the S-W normality test. The null hypothesis with the S-W normality test is that data is normally distributed. For the S-W normality test on set of Mean CDC energy consumption by THR-MMT, $\{3.21, 3.04, 3.86, 3.48\}$ as shown in Table 3.4 and set of Mean CDC energy consumption by *RTDVMC*, $\{3.12, 2.8, 3.58, 3.55\}$ as shown in Table 3.4, the corresponding *p* values are found as 0.74 and 0.36. Now, both 0.74 and 0.34 are greater than critical value 0.1. Therefore, there is no strong evidence to reject the null hypothesis that sets of Mean CDC energy consumption by THR-MMT and *RTDVMC* are normally distributed. It is important to note that the resulting distribution from subtracting corresponding elements of two normal distributions, is a normal distribution. As such, the elements of set of minimizations of CDC energy consumption by *RTDVMC* compared to THR-MMT, $\{0.09, 0.24, 0.28, -0.07\}$ as shown in Table 3.4, are also normally distributed. Consequently, if we had repeated the experiments more and more, and created a distribution of mean of such sampling distribution of sample mean as $\{0.09, 0.24, 0.28, -0.07\}$, then the resulting distribution representing set of mean of minimization of mean CDC energy consumption by *RTDVMC* compared to THR-MMT would also had been a normal distribution.

Table 3.4 Mean Minimization of CDC Energy Consumption (kW) by *RTDVMC* compared to THR-MMT

PlanetLab Workload	**Mean CDC Energy Consumption (kW)**		**Minimization of CDC Energy Consumption by *RTDVMC* Compared to THR-MMT**
	THR-MMT	***RTDVMC***	
3 March	3.21	3.12	0.09
6 March	3.04	2.8	0.24
12 April	3.86	3.58	0.28
20 April	3.48	3.55	-0.07
Mean of Minimization of Mean CDC Energy Consumption by *RTDVMC* Compared to THR-MMT			0.14

- **Significance Test**

The *t-test* is used to prove statistical significance. According to the two sample paired one tail *t-test*, the null hypothesis is that there is no improvement of mean of minimization of mean CDC energy consumption by *RTDVMC* compared to THR-MMT. In other words, the mean of minimization of mean CDC energy consumption by *RTDVMC* compared to THR-MMT is 0. Hence, given the null hypothesis is true, the mean of the normal distribution representing mean of minimization of mean CDC energy consumption by *RTDVMC* compared to THR-MMT is 0. From our experiment, we have found that the mean of minimization of mean CDC energy consumption by *RTDVMC* compared to THR-MMT is 0.14 (i.e., as shown in Table

3.4). We now measure that how likely it is to obtain such as extreme value of 0.14, given that the null hypothesis is true. Using two sample paired one tail *t -test*, we have found that the respective probability is 0.09, which is lower than the critical value 0.1. In other words, if the null hypothesis is true, then there is less than 10% chance to obtain 0.14 as the mean of minimization of mean CDC energy consumption by *RTDVMC* compared to THR-MMT. However, we have obtained 0.14 as the mean through our rigorous experiment. Therefore, we can claim that the null hypothesis is not true, as we reject the null hypothesis. As such, we have proved that the mean minimization of CDC energy consumption by *RTDVMC* compared to THR-MMT is statistically significant, as otherwise, the *p* value would had been greater than or equal to 0.1. Hence, we can accept the alternative hypothesis that mean CDC energy consumption by *RTDVMC* is lower compared to that of THR-MMT.

Similar to THR-MMT and *RTDVMC*, we have performed the S-W normality test and the *t-test*s to prove that mean minimization of CDC energy consumption through *RTDVMC* compared to MAD-MMT, IQR-MMT, LR-MMT and LRR-MMT is statistically significant. In Table 3.5, Table 3.6, Table 3.7 and Table 3.8, we have presented mean minimization of CDC energy consumption (kW) through *RTDVMC* compared to MAD-MMT, IQR-MMT, LR-MMT and LRR-MMT, respectively. Respective *p* values for the S-W normality tests with set of Mean CDC energy consumption by MAD-MMT, IQR-MMT, LR-MMT and LRR-MMT have been articulated in Table 3.9.

Table 3.5 Mean Minimization of CDC Energy Consumption (kW) by *RTDVMC* compared to MAD-MMT

PlanetLab Workload	**Mean CDC Energy Consumption (kW)**		**Minimization of CDC Energy Consumption by *RTDVMC* Compared to MAD-MMT (kW)**
	MAD-MMT	***RTDVMC***	
3 March	3.63	3.12	0.51
6 March	3.13	2.8	0.33
12 April	4.23	3.58	0.65
20 April	4.3	3.55	0.75
Mean of Minimization of Mean CDC Energy Consumption by *RTDVMC* Compared to MAD-MMT			0.56

Table 3.6 Mean Minimization of CDC Energy Consumption (kW) by *RTDVMC* compared to IQR-MMT

PlanetLab Workload	**Mean CDC Energy Consumption (kW)**		**Minimization of CDC Energy Consumption by *RTDVMC* Compared to IQR-MMT (kW)**
	IQR-MMT	***RTDVMC***	
3 March	3.49	3.12	0.37
6 March	2.7	2.8	-0.1
12 April	3.98	3.58	0.04
20 April	3.71	3.55	0.16
Mean of Minimization of Mean CDC Energy Consumption by *RTDVMC* Compared to IQR-MMT			0.21

Table 3.7 Mean Minimization of CDC Energy Consumption (kW) by *RTDVMC* compared to LR-MMT

PlanetLab Workload	**Mean CDC Energy Consumption (kW)**		**Minimization of CDC Energy Consumption by *RTDVMC* Compared to LR-MMT (kW)**
	LR-MMT	***RTDVMC***	
3 March	3.21	3.12	0.48
6 March	3.04	2.8	0.72
12 April	3.86	3.58	0.64
20 April	3.48	3.55	0.12
Mean of Minimization of Mean CDC Energy Consumption by *RTDVMC* Compared to LR-MMT			0.49

Table 3.8 Mean Minimization of CDC Energy Consumption (kW) by *RTDVMC* compared to LRR-MMT

PlanetLab Workload	**Mean CDC Energy Consumption (kW)**		**Minimization of CDC Energy Consumption by *RTDVMC* Compared to LRR-MMT (kW)**
	LRR-MMT	***RTDVMC***	
3 March	3.25	3.12	0.13
6 March	3.37	2.8	0.57
12 April	3.76	3.58	0.18
20 April	4.48	3.55	0.93
Mean of Minimization of Mean CDC Energy Consumption by *RTDVMC* Compared to LRR-MMT			0.45

Table 3.9 The *p* Values Generated through the S-W Normality Tests

Distribution of mean CDC Energy Consumption	**The *p* Value with the S-W normality Test**
Distribution of Mean CDC Energy Consumption by MAD-MMT, $\{3.63, 3.13, 4.23, 4.3\}$ as shown in Table 3.5	0.4
Distribution of Mean CDC Energy Consumption by IQR-MMT, $\{3.49, 2.7, 3.98, 3.71\}$ as shown in Table ***3.6***	0.46
Distribution of Mean CDC Energy Consumption by LR-MMT, $\{3.21, 3.04, 3.86, 3.48\}$ as shown in Table 3.7	0.13
Distribution of Mean CDC Energy Consumption by LRR-MMT, $\{3.25, 3.37, 3.76, 4.48\}$ as shown in Table 3.8	0.4

From Table 3.9, we can observe that the *p* values for the S-W normality tests with set of Mean CDC energy consumption by MAD-MMT, IQR-MMT, LR-MMT and LRR-MMT, are 0.4, 0.46, 0.13 and 0.4, respectively. Now, 0.4, 0.46, 0.13 and 0.4 are greater than critical value 0.1, therefore, there is no strong evidence to reject the null hypothesis that sets of mean CDC energy consumption by MAD-MMT, IQR-MMT, LR-MMT and LRR-MMT are normally distributed. Previously, in section 0, we showed that set of Mean CDC energy consumption by *RTDVMC* is normally distributed. As such, the prior condition of the *t-test* that the target data must be normally distributed is met. Given the null hypothesis is true, *RTDVMC* cannot improve energy efficiency any further than that of MAD-MMT, IQR-MMT, LR-MMT and LRR-MMT. In other words, the mean of minimization of mean CDC energy consumption by *RTDVMC* compared to MAD-MMT, IQR-MMT, LR-MMT and LRR-MMT is 0. Articulated in Table 3.5, Table 3.6, Table 3.7 and Table 3.8, our experimental results highlight that mean values of minimization of mean CDC energy consumption by *RTDVMC* compared to MAD-MMT, IQR-MMT, LR-MMT and LRR-MMT are 0.56, 0.21, 0.49 and 0.45. We have performed the two sample paired one tail *t-test* to measure the likelihood (i.e., corresponding *p* values) to obtain such extreme values, given the null hypothesis is true. The results of the *t-test*s have been highlighted in Table 3.10.

From Table 3.10, we can observe that the *p* values are lower than critical value, 0.1. Therefore, respective null hypotheses are not true, as we can accept alternative hypotheses that mean CDC energy consumption by *RTDVMC* is lower than that of MAD-MMT, IQR-MMT, LR-MMT and LRR-MMT. As such, we have proved that the minimization of mean energy consumption by *RTDVMC* compared to MAD-MMT, IQR-MMT, LR-MMT and LRR-MMT is statistically significant. One critical aspect is to highlight test error related to significance test and to determine population mean using Confidence Interval (*CI*), which we have discussed in the following section.

Table 3.10 The *p* Values Generated through the Two sample paired One tail *t-test*

Experimental Mean of Minimization of Mean CDC Energy Consumption by *RTDVMC* Compared to Diverse ATDVMC Algorithms	**The *p* Value with the Two Sample Paired *t-test***
Mean of Minimization of Mean CDC Energy Consumption by *RTDVMC* Compared to MAD-MMT, 0.56	0.004
Mean of Minimization of Mean CDC Energy Consumption by *RTDVMC* Compared to IQR-MMT, 0.21	0.085
Mean of Minimization of Mean CDC Energy Consumption by *RTDVMC* Compared to LR-MMT, 0.49	0.017
Mean of Minimization of Mean CDC Energy Consumption by *RTDVMC* Compared to LRR-MMT, 0.45	0.047

3.6.2.2 Test Error and Confidence Interval (CI)

Errors related to the *t-test* can be classified into two groups: Type I error and Type II error. Type I error refers to the total probability of falsely rejecting the null hypothesis while it was true, and Type II error refers to the total probability of falsely rejecting alternative hypothesis while it was true. The *p* value refers to the probability to obtain the test statistic assuming that the null hypothesis is true. Based on the *p* values of 0.09, 0.004, 0.085, 0.017 and 0.047, we have rejected null hypotheses. Hence, the probability to falsely reject null hypothesis while those were true, aka Type I errors are 0.09%, 0.004%, 0.085%, 0.017% and 0.047%, respectively.

Thus far, minimization of mean CDC energy consumption by *RTDVMC* compared to existing DVMC algorithms as obtained through experiments are set of samples mean and not population mean. CI is used to generate a range within which population mean can be located. We have used 80% CI. A common way of stating 80% CI of a mean is we are 80% confident that the population mean would be within that range. Fig. 3.8 illustrates values of 80% *CI* of Mean of Minimization of Mean CDC Energy Consumption by *RTDVMC* compared to existing DVMC algorithms.

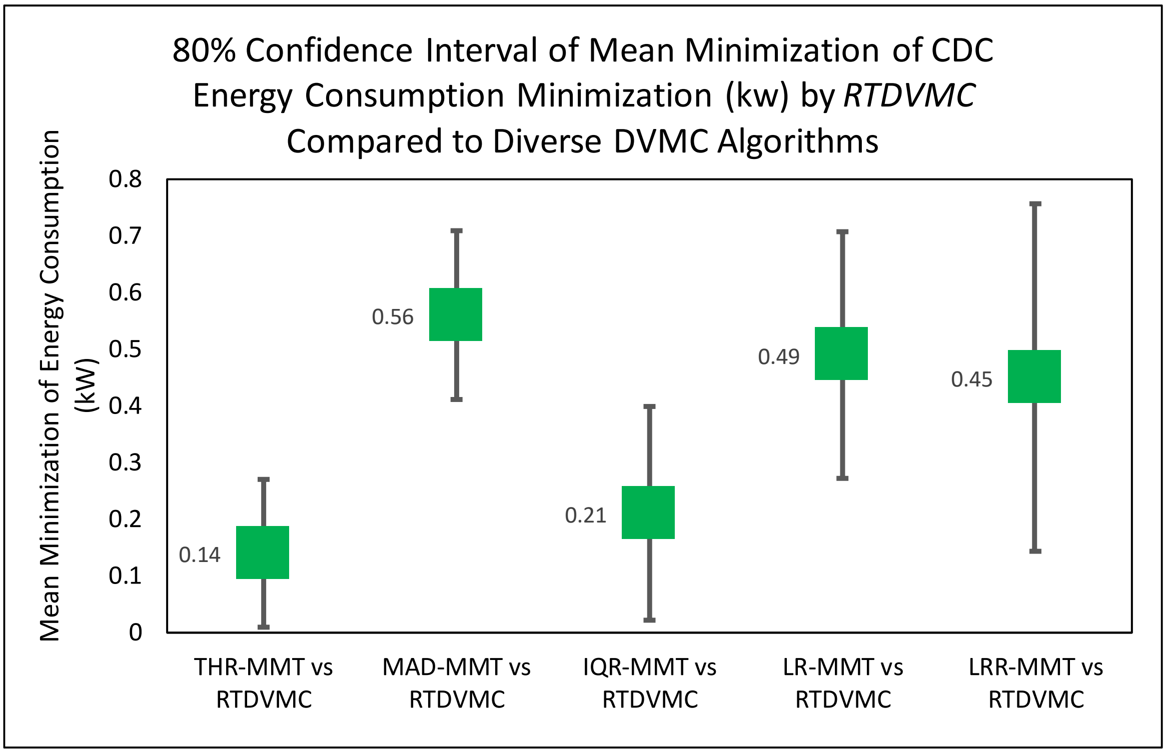

Fig. 3.8 80% *CI* of Mean of Minimization of Mean CDC Energy Consumption by *RTDVMC* Compared to Existing DVMC Algorithms

3.6.3 Result Analysis

Thus far, we have illustrated our experimental results and performed diverse statistical testing. In the following we have presented result analysis.

Observation 1: Empirical evaluation as portrayed through Fig. 3.2 - Fig. 3.7, highlight that for majority days of workload, *RTDVMC* edges out existing DVMC algorithms in terms of minimizing of mean CDC energy consumption. The key difference between *RTDVMC* and existing DVMC algorithms is that the former one takes such user provided information as VMRT into account, whereas later ones do not. Elucidated though Fig. 3.1, *RTDVMC* concomitantly attempts to optimize from two aspects: first, minimize number of active PMs and second, minimize release time of source PMs without causing increased release time of destination PMs. In contrast, as illustrated in Fig. 3.1, existing algorithms only attempt to optimize from one aspect: minimize number of active PMs, as minimization of release time of source PM and inhibition of increase of release time of destination PMs are not considered at all. Consequently, in the presence of heterogeneous VMRT, existing algorithms are outperformed by *RTDVMC*.

Observation 2: Fig. 3.2 – Fig. 3.5 show that 18 out of 20 times of performance comparisons between *RTDVMC* and existing DVMC algorithms, the former one has come out as superior to rest of the algorithms. In one such cases as PlanetLab 6 March, IQR-MMT outperformed *RTDVMC* and in one such case as PlanetLab 20 April, THR-MMT outperformed *RTDVMC*. This observation highlights that with such NP-hard problem as DVMC, no algorithm, including *RTDVMC* can guarantee that it would always find optimal solution for any input set in every possible scenario compared to rest of the algorithms. As such, in practise, a target algorithm is run multiple times with different days of data and then performance is compared with that of rest of the algorithms. From our experimental results delineated in Fig. 3.6 and Fig. 3.7, considering performance under different days of workload, *RTDVMC* minimizes mean CDC energy consumption by a minimum of 4% and a maximum of 15% compared to rest of the algorithms.

Observation 3: The improvement of performance in terms of minimizing mean energy consumption as exhibited by *RTDVMC* compared to rest of the algorithms is found as statistically significant. Such proof of statistical significance further supports the claim that consideration of such user provided information as VMRT in DVMC algorithm can benefit in regulating mean CDC energy consumption.

Observation 4: Experiment results suggest that incorporation of such user provided information as VMRT in VM consolidation decision process can lower mean CDC energy consumption further. However, set of mean obtained through experiments are sample mean, while sample mean carries less weight than population mean. Therefore, set of population mean is presented in Fig. 3.8. Fig. 3.8 show that, minimization of mean CDC energy consumption by *RTDVMC* compared to existing algorithms is positive in terms of population mean. Such result further strengthens the evidence that *RTDVMC* is superior to existing algorithms in terms of minimizing mean CDC energy consumption.

Observation 5: Fig. 3.6 and Fig. 3.7 show that such STDVMC algorithms as *RTDVMC* and THR-MMT are more energy-efficient than ATDVMC algorithms. The underlying reason is that in comparison with STDVMC, ATDVMC algorithm regulates SLA violation further through migrating out VMs prior excessive resource utilization takes place. However, increased VM migration increases the probability of requirement of more PMs to host migrating VMs, resulting into higher energy consumption.

In the following section we have summarised my research presented in this chapter.

3.7 Summary

VMC is one of the state-of-the-art energy-efficient Cloud resource management technique, which effectively lowers CDC energy consumption through increasing Cloud resource utilization. Through our extensive literature review on VM consolidation algorithms, we have learned that incorporation of CSU provided information in DVMC algorithm and its impact on energy-efficiency of CDC has not been investigated. In this research work, we have hence articulated a novel heuristic DVMC algorithm, *RTDVMC*, which exploits such CSU provided information as *VMRT* in its substratum.

RTDVMC takes *Release Time* of VMs and PMs as well as energy-efficiency of PMs into account in three components of DVMC algorithm: Source PM selection, VM selection and Destination PM Selection. The *PMRT* based source PM selection (i.e., *U-UPM* selection) and *VMRT* based VM selection strategy combined with *PMRT* based destination PM selection technique increase the resource utilization of PMs. Furthermore, *RTDVMC* endeavours to lower the turned-on durations of source PMs through migrating out VM(s) with higher *VMRT* without undesirably affecting (i.e., increasing) the turned-on durations of destination PMs. Hence, at a given point in time, the *RTDVMC* can shut down more PMs compared to existing DVMC algorithms, which do not consider *VMRT* or any of the CSU feedback in VM consolidation decision process. Consequently, significantly improved energy-efficiency is achieved as reflected in empirical evaluations. For performance testing real cloud-based workload has been incorporated. Two key points has been noted through result analysis. First, based on our simulation results, we have found that opportunity of making more optimal VM migration decision might arise through considering such user provided information as *VMRT* leading towards improved energy-efficiency. Second, from our experiments we have found STDVMC algorithms as more energy-efficient to ATDVMC algorithms. All findings of this chapter are incorporated in our subsequent research to develop a further optimized DVMC algorithm as presented in next chapter.

4. Multi-Objective Dynamic Virtual Machine Consolidation Algorithm for Cloud Data Centers with Highly Energy Proportional Servers and Heterogeneous Workload

4.1 Introduction

Existing DVMC algorithms, such as [45, 51-53, 64, 85-96] either mentioned that homogeneous VMRT has been used for experiments or have not articulated any assumption made on VMRT. Through experimenting with homogenous VMRT (i.e., all VMs are assigned with tasks of equal length in terms of task finishing time) using CloudSim [27, 107] as used by respective researchers, we have obtained similar results as presented in respective literature. However, in real scenario, CDC consists of VMs with heterogeneous VMRT. In other words, VMs are assigned with tasks of unequal lengths in terms of task finishing time.

The data representing VM Release Time (VMRT) (i.e., the lifespan of a VM) of real VMs resided in Nectar Cloud [28] highlighted through Fig. 4.1 - Fig. 4.3 exhibit that VMRT varies from one VM to another. It is critical to note that at the end of this Chapter (Observation 3 and Observation 4 of Section 4.5), we have exhibited through experiments that larger VMRT increases CDC energy consumption, since the total resource utilization by a VM grows as the lifespan of that VM's grows. Hence, without considering heterogenous VMRT, the overall estimation about the impact of any DVMC algorithm on CDC energy consumption and subsequent QoS would be less accurate.

As such, in contrast with traditional DVMC algorithms, which only consider homogeneous VMRT, in Chapter 3, we proposed a Release time based DVMC algorithm, namely *RTDVMC* [108], which takes the heterogeneous VMRT into consideration. Nevertheless, several limitations exist with *RTDVMC* as elucidated in the following:

- To the best of our level of understanding of such existing DVMC algorithms as [45, 49, 51-53, 64, 85-96], including *RTDVMC* [108] are developed based on the fundamental underlying assumption that optimal energy efficiency is achievable at maximum PM resource utilization. The basis of such claim against these existing DVMC algorithms is that these algorithms use legacy PMs, such as HP ProLiant ML110 G4 [97] and HP ProLiant ML110 G5 [98] for performance validation. From Fig. 4.6, we can see that for those legacy PMs the energy-efficiency (i.e., Ratio of Power to Throughput) is as high as its incurred utilization. In stark contrast, for modern highly energy proportional PMs, such as Dell PowerEdgeR940 (Intel Xeon Platinum 8180, 112 cores→25000 MHz, 384 GB) [99], HP ProLiant DL560 Gen10 (Intel Xeon Platinum 8180, 112 cores→25000 MHz, 384 GB) [100] and HP ProLiant ML350 Gen10 (Intel Xeon Platinum 8180, 28 cores→25000 MHz, 192 GB) [101], energy-efficiency rather drops beyond 70% utilization intervals, as delineated in Fig. 4.6. The underlying reason is that, while the throughput increases uniformly with the increase of load or utilization (Fig. 4.4), the power consumption of modern PMs beyond 70% utilization level rises such drastically (Fig. 4.5) that it exceeds the respective linear increase of throughput. Consequently, the ratio of throughput to power consumption, translated as energy-efficiency, drops [109]. As such, several researchers [102, 103] argue that consolidation towards maximum increase of PM resource utilization does not feature the optimal minimization of CDC energy consumption with new generation of highly energy proportional PMs. Based on our literature study [45, 49, 51-53, 64, 85-96], no DVMC algorithm is found to address this issue.

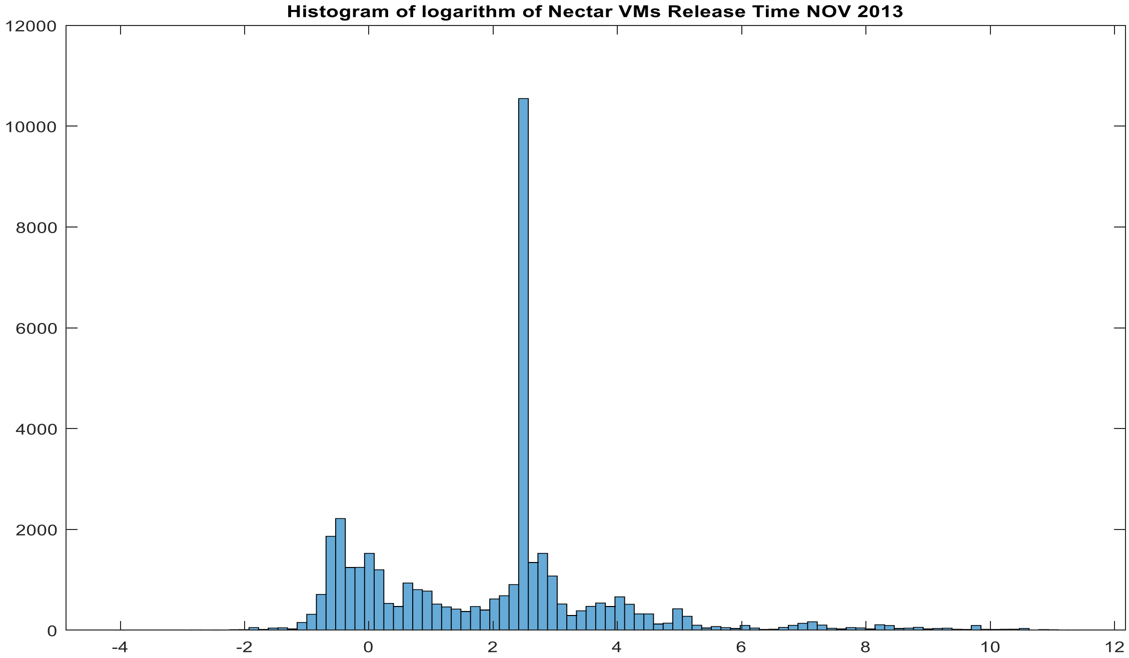

Fig. 4.1 Histogram of logarithm of Release Time (in Second) of VMs Created in Nectar Cloud in November 2013

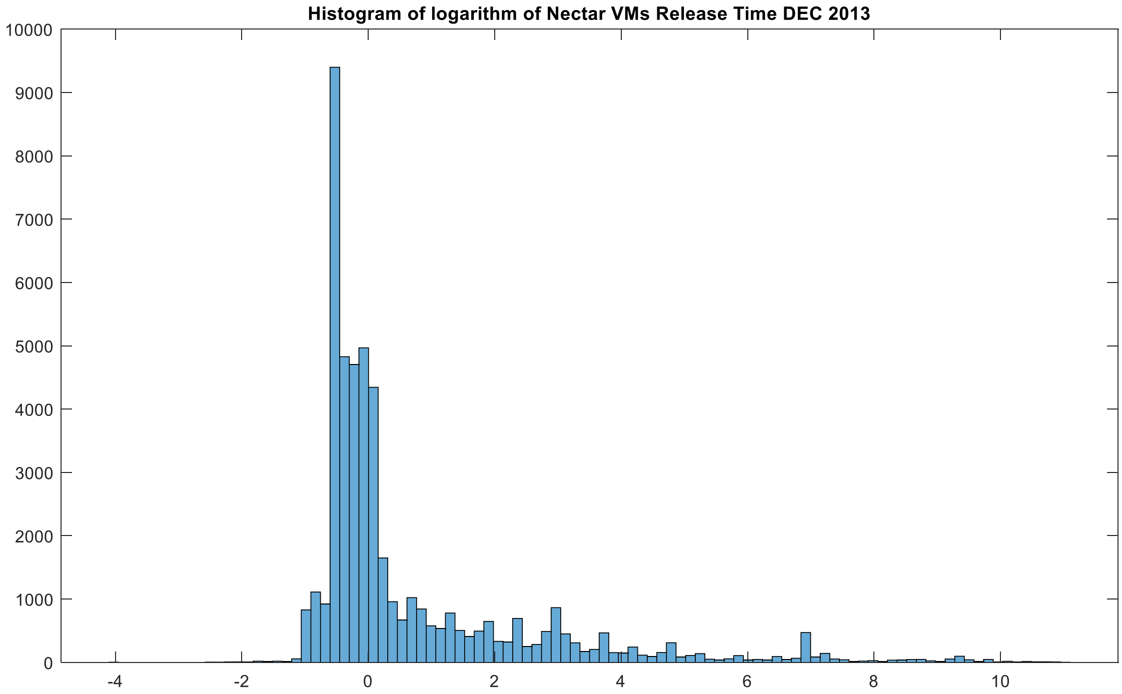

Fig. 4.2 Histogram of logarithm of Release Time (in Second) of VMs Created in Nectar Cloud in December 2013

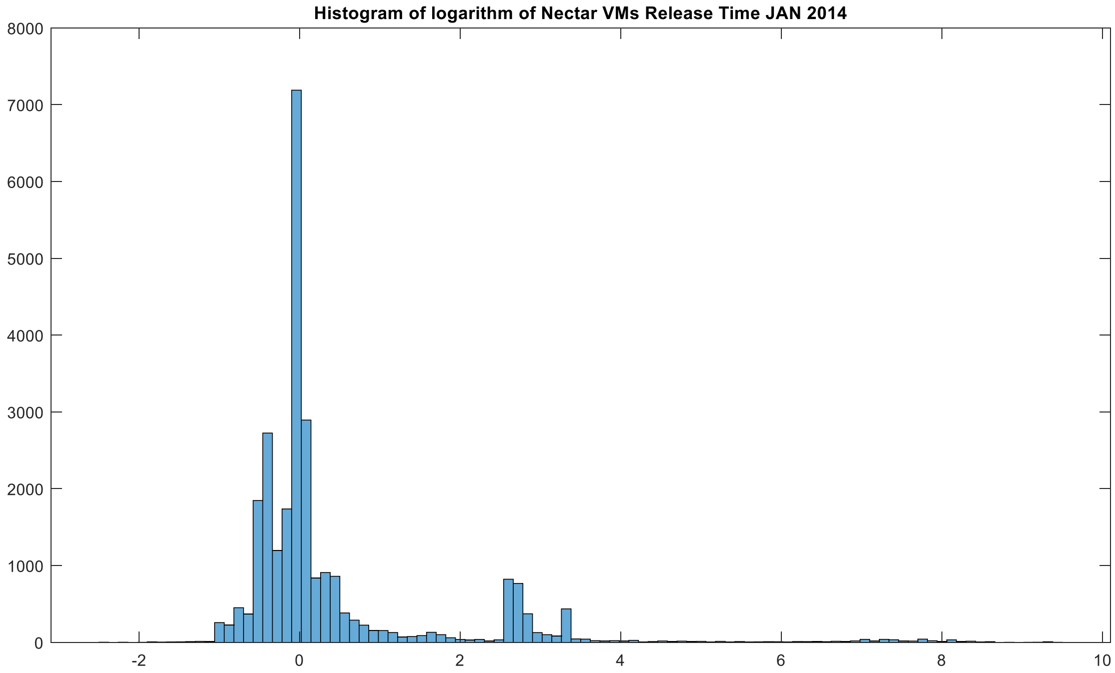

Fig. 4.3 Histogram of logarithm of Release Time (in Second) of VMs Created in Nectar Cloud in January 2014

- Preventing Quality of Service (QoS) degradation in CDC due to excessive VM migration is as much important as it is to minimize CDC energy consumption. However, higher energy consumption minimization by *RTDVMC* comes at a cost of excessive increase of VM migration.

- While, performance of *RTDVMC* has strong correlation with the value of VMRT, instead of VMRT values of real Cloud workload, only simulated VMRT values have been used. The complex behaviour and interaction of VMs have impact on VMRT values, which cannot be reflected in simulated VMRT values.
- *RTDVMC* consolidates VMs primarily on the value of VMRT with an assumption that VMRT would be precisely known in advance, whereas, in reality VMRT would not be strictly accurate in many scenarios.

With respect to above-mentioned challenges, our contributions in this chapter is briefly outlined in the following.

- A novel heuristic DVMC algorithm, namely *Stochastic Release Time Based DVMC* (*SRTDVMC*) algorithm has been proposed, which is robust to uphold optimal performance in terms of minimizing energy consumption regardless of the potential change in underlying PMs' energy efficiency characteristics.
- One crucial goal of experiment is to predict behaviour of an algorithm in the real world [110]. For performance evaluation, VMRT values extracted from real Cloud, namely Nectar Cloud has been used, which assists to obtain a stronger performance prediction under real Cloud scenarios.
- While minimization of VM migration reduces network overhead and subsequent energy consumption by networking equipment, it is intrinsic in VM consolidation. *SRTDVMC* is multi-objective, which concomitantly addresses two confronting goals : minimizing energy consumption and minimizing VM migration.
- *SRTDVMC* eliminates the restriction of strictly accurate VMRT through the conversion of VMRT into respective Stochastic VMRT (SVMRT).

Prior proffering our proposed algorithm, Stochastic VM Release Time and various terms used in the algorithm have been elucidated in Section 4.1. Next, in Section 4.2, we have brought forth the proposed heuristic DVMC algorithm, *SRTDVMC* followed by discussion of its various characteristics in Section 4.3. In Section 4.4, the experimental setup and performance evaluation of the proposed algorithm have been articulated. In Section 4.5, we have elucidated our critical observations extracted from empirical evaluations. Finally, in

Section 4.6, we have summarized our research with future research directions and motivation.

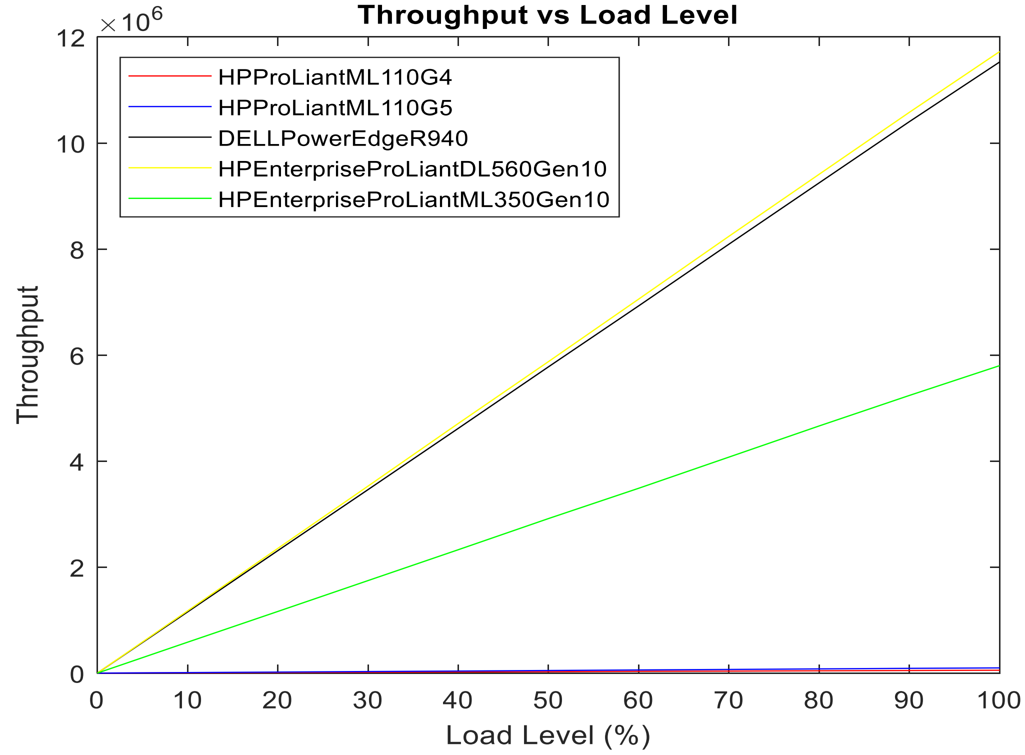

Fig. 4.4 Change of Throughput with Varying Load Level for Various PMs

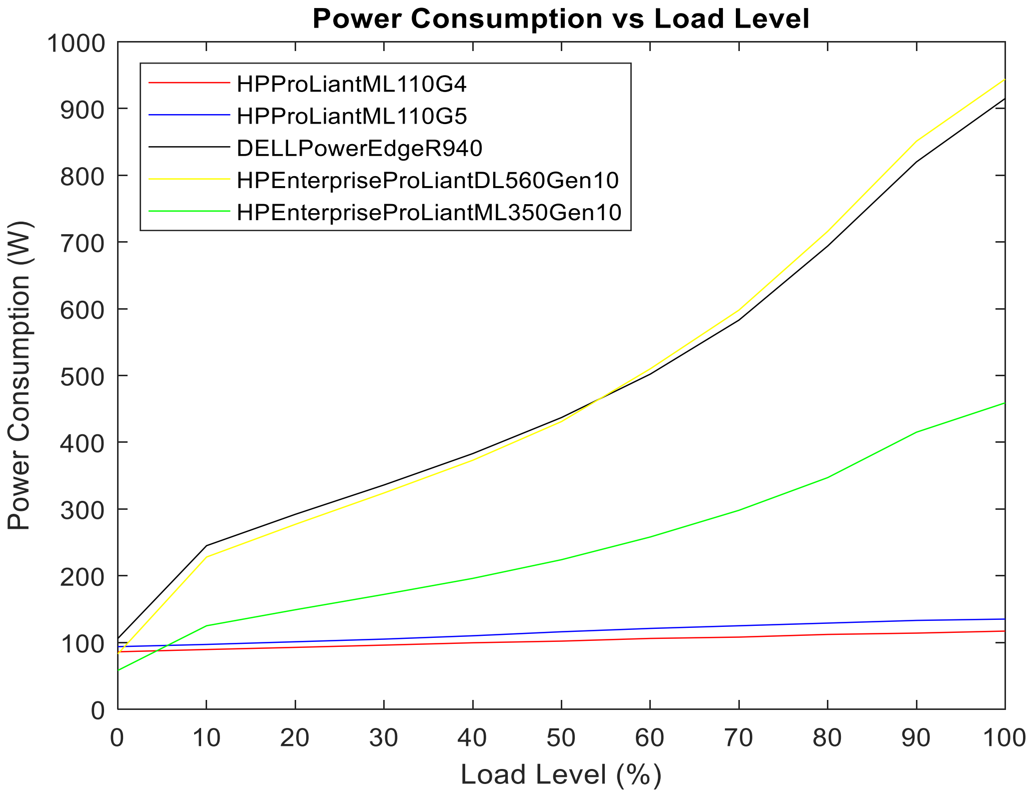

Fig. 4.5 Change of Power Consumption with Varying Load Level for Various PMs

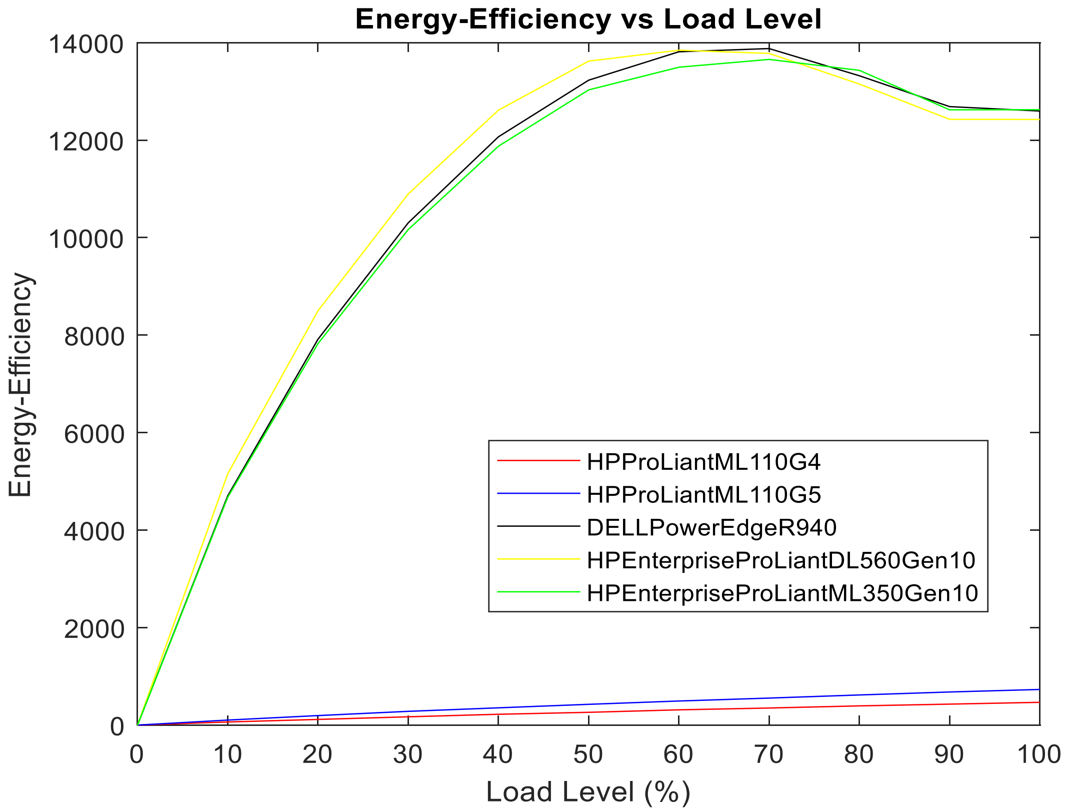

Fig. 4.6 Change of Energy-Efficiency with Varying Load Level for Various PMs

4.2 Modelling of Stochastic VM Release Time and Important Concepts

In our proposed *SRTDVMC* algorithm, stochastic release time is one of the key distinctive features used in VMC decision process. To ensure easy understanding of our proposed algorithm in the following, we have first explained the key terms used in the proposed algorithm.

4.2.1 Modelling Stochastic VM Release Time and PM Release Time

Workload finishing time or lifetime of a VM is referred to as VMRT. Prior receiving any service, negotiation of service related conditions including service expiry date followed by an acceptance of the contract takes place between Cloud Service Providers (CSPs) and CSUs, interpreted as Service Level Agreement (SLA). For many VMs, VMRT is equal to the contract of service period between the respective CSU (i.e., VM owner) and the CSP as agreed during

SLA. Before the contract is expired, both CSUs and CSPs might agree/disagree to renew, extend or early termination of the contract and respective VMRT would be updated accordingly. Some web applications hosted in CDC remains unremoved for a very long period. Estimated VMRT of such VMs would be large values referring to the time when the respective contract of service between CSU and CSP would expire as agreed prior the service during SLA. If renewal or early termination of contract of service takes place, VMRT would be readjusted accordingly.

For some applications, VMRT corresponds to QoS and potential resource demand. Resource demand may change with the variation in number of users, causing creation of additional VMs and later deletion of such VMs. At the time of SLA, a SAAS provider/PAAS user must consider the potential number of application users and mention it to PAAS provider so that by taking the potential number of end users and corresponding resource demand into account a certain standard of QoS can be uphold. Pattern of changing resource demand over time derived from past data can also be utilized to recognize the change of resource demand in future [111]. Considering the change in resource demand over time and demanded QoS, PAAS provider/IAAS user can estimate resource/VM release time, which would be proffered to IAAS provider.

In many cases, the prior estimated VMRT might not turn out as strictly accurate in future. Hence, to reflect the reality further closely, we propose to embody a stochastic version of VMRT, referred to as Stochastic VM Release Time (SVMRT) in *SRTDVMC*. SVMRT can be calculated from (4.1), At the outset of the thesis, in the Nomenclature Table, we have articulated the meaning of the notations used in this chapter.

$$S_{V_j} = (1 + \alpha \cdot Y) \cdot T_{V_j} \tag{4.1}$$

Apart from *VMRT*, another crucial term noteworthy explaining is *PM Release Time* (*PMRT*). *PMRT* refers to the time when a PM can be either shut down or put into a sleep state that would consume no energy, or lower amount of energy compared to its active state. A PM can be shut down or put into sleep state, if it has either no VM hosted on it, or none of its hosted VMs is in the active state. Since *SVMRT* refers to the maximum time until which the VM would be in the active state, hence *PMRT* denotes the maximum *SVMRT* value among all the *VMRT* values of VMs that are hosted in that PM, as articulated in (4.2).

$$T_{P_i} = \max\left(S_{V^i}\right) \tag{4.2}$$

4.2.2 Modelling Resource Utilization, Constraints and Energy Consumption

Prior presenting the *SRTDVMC* algorithm, resource utilization model, respective constraints, energy consumption model by a PM and CDC have been discussed in this section followed by multiple objective functions, which *SRTDVMC* aim to optimize. For *SRTDVMC*, we have utilized the resource utilization model presented in (3.1). The constraints (3.3) and (3.5) used in the algorithm have been previously explained in Section 3.2. The Energy consumption model of PM and CDC have been outlined in (3.6) and (3.7). In order to relate closely to the real energy consumption by PMs, for our energy consumption model, we have opted to draw energy consumption benchmark results of three different types of state-of-the-art PMs: Dell PowerEdgeR940 (Intel Xeon Platinum 8180, 112 cores⇥25000 MHz, 384 GB) [99], HP ProLiant DL560 Gen10 (Intel Xeon Platinum 8180, 112 cores⇥25000 MHz, 384 GB) [100] and HP ProLiant ML350 Gen10 (Intel Xeon Platinum 8180, 28 cores⇥25000 MHz, 192 GB) [101]. In Table 4.1, we have articulated respective energy consumption of these three types of PMs at varying load level. From (3.7), we can observe that the total energy consumption of the CDC is the sum of energy consumption of all the PMs in the CDC. Using Table 4.1, we can determine the energy consumption of a PM based on the CPU utilization of that PM.

Table 4.1 Energy Consumption Values of Contemporary Servers at Different Load Level

	Energy Consumption (kW) at Different Percentage of Load Level										
	Sleep	10%	20%	30%	40%	50%	60%	70%	80%	90%	100%
Dell PowerEdge R940	106	245	292	336	383	437	502	583	694	820	915
HP ProLiant DL560 Gen10	82.9	228	277	324	373	431	510	598	716	851	944
HP ProLiant ML350 Gen10	58.1	125	149	172	196	224	258	298	347	415	459

4.2.3 Objective Functions

DVMC algorithms aim to minimize CDC energy consumption through migrating VM(s) out of lower utilized PMs and placing those VM(s) in relatively higher utilized PM(s). As such, the first objective function, f_1 of *SRTDVMC* has been characterized through (4.3).

$$f_1 = \min(\bar{E}_{CDC}) \tag{4.3}$$

One downfall of DVMC is that energy consumption minimization through VM consolidation cannot be achieved without VM migration, which itself deteriorates QoS as well as raises network overhead leading towards increased SLA violation and increased energy consumption by networking equipment of CDC. Therefore, restricting VM migration in CDC is no less important than lowering CDC energy consumption. As such, (4.4) characterizes the second objective function, f_2 of *SRTDVMC*.

$$f_2 = \min(\bar{\psi}) \tag{4.4}$$

It is worthwhile to note that f_1 and f_2 are two confronting objective functions. Value of f_1 can only be minimized through migrating VM(s) from lower utilized PM(s) to relatively higher utilized PMs, which increases ψ, whereas, increased ψ negatively affects f_2. Hence, it is more challenging to design a DVMC algorithm, which can show better performance in terms of both f_1 and f_2. In the following section, our proposed multi-objective heuristic DVMC algorithm, namely *SRTDVMC* has been presented, which aims to optimize f_1 and f_2.

4.3 Proposed Solution

At the outset of this chapter, we have elucidated the limitations of *RTDVMC*. To address those limitations, we have proposed our *SRTDVMC* algorithm presented in Algorithm 4.1. The meaning of notations used in *SRTDVMC* algorithm has been presented earlier in Nomenclature Table.

Algorithm 4.1 The *SRTDVMC* algorithm

```
Input: $P$, $V$, $R$, $SP$
Output: VM Placement
The first phase: O-UPMs
 1:  for each $P_i$ in $P$ do
 2:      if (3.4) is satisfied then
 3:          $OP \leftarrow \{P_i \cup OP\}$
 4:      end if
 5:  end for          // end of for loop from Line 1 to 5
 6:  for each $P_o$ in $OP$ do
 7:      $migratingVMs \leftarrow$ Invoke the $VSO$ algorithm with $P_o$
 8:      $vmsToMigrate \leftarrow \{migratingVMs \cup vmsToMigrate\}$
 9:  end for          // end of for loop from Line 6 to 9
10:  Sort $vmsToMigrate$ in the order of decreasing $SVMRT$
11:  $NOP \leftarrow \{P - OP - SP\}$
12:  for each $V_l$ in sorted $vmsToMigrate$ do
```

```
13:     $P_d$ ← Invoke the DPSVO algorithm with $V_l$ and NOP
14:     destinationPMs ← {$P_d$ ∪ destinationPMs}
15:     if $P_d$ is in SP then
16:         SP ← {SP − $P_d$}
17:         for each $R_k$ in R do
18:             $U_d^k \leftarrow U_d^k + D_q^k$
19:             $C_d^k \leftarrow C_d^k - D_q^k$
20:         end for     // end of for loop from Line 17 to 20
21:         NOP ← {$P_d$ ∪ NOP}
22:     end if          // end of if at Line 15
23: end for             // end of for loop from Line 12 to 23
        The second phase: U-UPMs
24: candidateSources ← {P − OP − SP − destinationPMs}
25: candidateDestinations ← {P − OP − SP}
26: Sort candidateSources in the order of increasing PMRT
27: for each $P_c$ in sorted candidateSources do
28:     candidateDestinations ← {candidateDestinations − $P_c$}
29:     destinations ← Invoke the DPSVU algorithm with $P_c$
                       and candidateDestinations
30:     candidateSources ← {candidateSources − destinations}
31: end for             // end of for loop from Line 27 to 31
```

4.3.1 Two Phases of *SRTDVMC* Algorithm

In this section, we have explained our proposed *SRTDVMC* algorithm. Resource demand of VMs may change over time, resulting variation of resource utilization in host PMs. When resource demand of hosted VM(s) grow higher over time, the hosting PM might fall short of adequate resources to prevent potential performance degradation, translated as SLA violation. Again, hosted VMs resource demand may decline over time, bringing forth a gap in the hosting PM to accommodate additional VMs from other low utilized PMs, leading towards potential opportunity to reduce the number of active PMs and subsequent reduction of CDC energy consumption. Based on this principal, *SRTDVMC* algorithm works in two phases. In **the first phase**, VMs from *O-UPMs* are migrate out to control SLA violations (Line 1 to 23 of Algorithm 4.1) and in **the second phase**, VMs from *U-UPMs* are migrated out to consolidate in lesser number of active PMs (Line 24 to 31 of Algorithm 4.1). In the following section, we have comprehensibly discussed each of these phases and its components:

4.3.1.1 The first phase *O-UPMs*

As expressed in (4.4), for any resource type (i.e., CPU, RAM and Bandwidth), if a PM's resource utilization is found as equal or greater than θ_{max}, then the PM is denoted as *O-UPM* (Line 1 to Line 5 of Algorithm 4.1). Next, VM(s) are selected from all *O-UPMs* to migrate out into new destination PMs (Line 7 of Algorithm 4.1).

- **VMs Selection From *O-UPMs***

The *VSO* algorithm, articulated as Algorithm 4.2, proffers to the VM(s), which are attempted to migrate out from an *O-UPM*. VMs of an *O-UPM* are first sorted in decreasing order of VMRT (Line 1 of Algorithm 4.2). The first VM from the sorted list of VMs is first selected and checked whether the respective PM's utilization drops lower than θ_{max}. If yes, then the respective PM is no more *O-UPM* and VM selection process stops (Line 2 to Line 3 of Algorithm 4.2). Otherwise, the second VM from the sorted list of VMs is selected and then the third VM and so forth, until the PM's utilization is decreased below θ_{max} (Line 3 to Line 10 of Algorithm 4.2). The rationale of selecting VM(s) with largest VMRT is to minimize the active duration of source *O-UPM*, which might aid in minimization of energy consumption.

Algorithm 4.2 The VMs Selection From *O-UPM* (*VSO*) algorithm

Input: The *O-UPM*, P_o

Output: List of VMs to be migrated out from the given *O-UPM*

```
 1: Sort $V^o$, the set of VMs of $P_o$ in order of decreasing $VMRT$
 2: $q \leftarrow 1$
 3: while $q \le |V^o|$ and $\theta_{MAX} < U_o^k$ for any $R_k$ in $R$ do
 4:     $migratingVMs \leftarrow \{V_q^o \cup migratingVMs\}$
 5:     for each $R_k$ in $R$ do
 6:         $U_o^k \leftarrow U_o^k - D_q^k$
 7:         $C_o^k \leftarrow C_o^k + D_q^k$
 8:     end for
 9:     $q \leftarrow q + 1$
10: end while
11: return $migratingVMs$
```

- **Destination PMs Selection for Migrating VMs from *O-UPMs***

SRTDVMC algorithm develops a set, denoted as *vmsToMigrate*, comprised of all migrating VMs from *O-UPMs* (Line 6 to Line 9 of Algorithm 4.1), as these VMs of *vmsToMigrate* are sorted in decreasing order of VMRT (Line 10 of Algorithm 4.1). The migrating VMs are attempted to host in PMs, which are neither *O-UPMs*, nor in sleep state or turned off state. Such set of PMs, which are not *O-UPMs* and not in either turned off or sleep state is referred to as *NOP* (Line 11 of Algorithm 4.1). *SRTDVMC* algorithm then keeps invoking *DPSVO* algorithm, presented as Algorithm 4.3 to determine the destination PM for each of the migrating VMs of *vmsToMigrate* starting from the largest VM to the smallest VM in terms of VMRT (Line 12 to Line 23 of Algorithm 4.1).

Algorithm 4.3 The Destination PM Selection for VM of *O-UPM* (*DPSVO*) algorithm

```
Input: The VM $V_j$ to be migrated out from a O-UPM
Input: NOP
Output: The new destination PM for the given migrating VM
 1: Sort NOP in the order of increasing PMRT
 2: for each $P_x$ in sorted NOP do
 3:     suitable ← Invoke the PST algorithm with $P_x$ and $V_j$
 4:     if suitable is true then
 5:         return $P_x$
 6:     end if
 7: end for
 8: return most energy-efficient $P_s$, which satisfies (4) and (6)
```

Algorithm 4.4 The PM Suitability Test (*PST*) algorithm

```
Input: PM, VM
Output: A decision whether the given PM can host the given VM
 1: if (3.3) and (3.5) are satisfied for all $R_k$ in $R$ then
 2:     return true
 3: end if
 4: return false
```

In order to determine a PM from *NOP* as destination host for a migrating VM of vmsToMigrate, the *DPSVO* algorithm first sorts the PMs of *NOP* is increasing order of PMRT (Line 1 of Algorithm 4.3). The smallest PM in terms of PMRT from the sorted *NOP* is first checked whether it is suitable to accommodate the migrating VM or not (Line 2 of Algorithm 4.3). The *PST* algorithm presented in Algorithm 4.4 is invoked to check the suitability of a PM as a potential destination PM (Line 3 of Algorithm 4.3). A PM is considered suitable, if RC (3.3) and MUTC (3.5) constraints are not violated (Line 1 to Line 4 of Algorithm 4.4). If that PM is found as suitable as per *PST* algorithm, then it is selected as the new destination PM for the migrating VM and the destination PM selection process ends (Line 4 to Line 6 of Algorithm 4.3). In case the PM is found as unsuitable, suitability of the second smallest PM in terms of

PMRT from the sorted *NOP* is checked and then the third smallest PM and so forth until a suitable PM is found (Line 2 to Line 7 of Algorithm 4.3). If no PM from *NOP* can accommodate that particular migrating VM, then the most energy-efficient and suitable PM from the set of PMs, which are in either sleep or turned-off state, referred to as *SP* is awoke and selected as destination PM (Line 8 of Algorithm 4.3). If the destination PM is selected from *SP*, then that PM is removed from the set *SP*, since it is no more in sleep or turned-off state and its utilization and capacity values across all resource types are adjusted (Line 15 to Line 20 of Algorithm 4.1). Furthermore, the PM is added in the set *NOP*, so that it can be considered as a potential destination PM for following migrating VMs of *vmsToMigrate* (Line 21 to Line 22 of Algorithm 4.1).

4.3.1.2 The Second Phase *U-UPMs*

U-UPMs refers to the set of those PMs, which had not been determined as *O-UPMs* in the first phase of *SRTDVMC* and which do not belong to *SP*. After determining the destination PMs for VMs of *O-UPMs*, the second phase of *SRTDVMC* is commenced when VMs from *U-UPMs* are migrated out, followed by strategic destination PMs selection for those migrating VMs with an aim to lower the number of active PMs so that CDC energy consumption can be minimized (Line 24 to Line 31 of Algorithm 4.1).

- **Source PMs Selection From *U-UPMs***

In the second phase, *SRTDVMC* first rounds up a set of *U-UPMs*, namely *candidateSources* – the set representing potential source *U-UPM*(s) from which VM(s) would be attempted to migrate out. The set of PMs, which were identified as *O-UPMs* in the first phase, referred to as *OP* along with the set of PMs, which hosted migrating VMs of *O-UPMs* are excluded from *candidateSources* to avoid repeated handling of PMs from *OP* and to inhibit re-migration of those VMs, which had been migrated out once in the first phase (Line 24 of Algorithm 4.1). The PMs of *candidateSources* are then sorted in the increasing order of PMRT (Line 26 of Algorithm 4.1). All PMs of sorted *candidateSources* starting from the smallest PM to the largest PM in terms of PMRT is sequentially selected as source *U-UPM*. However, the set of *U-UPMs* denoted by *destinations*, which are determined as the new destination PM(s) for migrating VM(s) from a source *U-UPM* is excluded from *candidateSources* and hence, those new destination *U-UPMs* cannot become source *U-UPM*, which prevents repeated migration of same VMs (Line 30 of Algorithm 4.1).

- **Migrating VMs and Destination PMs Selection**

SRTDVMC algorithm invokes the *DPSVU* algorithm (Algorithm 4.5) to select migrating VMs from *U-UPMs* and corresponding new destination PMs. Once a *U-UPM* from sorted *candidateSources* is selected as source *U-UPM*, the hosted VMs in that source *U-UPM* is sorted in decreasing order of VMRT (Line 1 of Algorithm 4.5). The VMs starting from the largest to the smallest in terms of VMRT are attempted to migrate out (Line 2 of Algorithm 4.5). The reason of selecting VMs in descending order of VMRT is that migrating out the largest VM can reduce the PMRT of the source PM leading towards energy consumption minimization. If for any VM, a suitable new destination *U-UPM* cannot be found, the migrating VM(s) selection from a source *U-UPM* terminates (Line 18 to 20 inside of Line 2 to 21 from Algorithm 4.5). In the following we have discussed the process of determining the new destination PM for such migrating VM.

Algorithm 4.5 The Destination PMs Selection for VMs of *U-UPMs* (*DPSVU*) algorithm

```
Input: A UPM, $P_c$ from the set of candidateSources
Input: candidateDestinations
Output: List of new destination PMs to host migrating VMs
 1: Sort $V^c$, the set of VMs of $P_c$ in order of decreasing VMRT
 2: for each $V_n^c$ in sorted $V^c$ do
 3:     $P_d \leftarrow$ null
 4:     Sort candidateDestinations in order of increasing PMRT
 5:     for each $P_m$ in sorted candidateDestinations do
 6:         suitable $\leftarrow$ Invoke the PST algorithm with $P_m$ and $V_n^c$
 7:         if suitable is true
 8:             energyDrop $\leftarrow$ Energy drop in $P_c$ without $V_n^c$
 9:             energyRise $\leftarrow$ Energy rise of $P_m$ for hosting $V_n^c$
10:             netEnergyGain $\leftarrow$ EnergyDrop – EnergyRise
11:             if NetEnergyGain > 0
12:                 $P_d \leftarrow P_m$
13:                 hostList $\leftarrow \{P_d \cup$ hostList$\}$
14:                 break loop
15:             end if
16:         end if
17:     end for
18:     if $P_d$ is null then
19:       break loop
20:     end if
21: end for
22: return hostList
```

In order to select the destination PM for a migrating VM of *U-UPM*, *SRTDVMC* algorithm first creates a set of potential destination PMs, referred to as *candidateDestinations*. The PMs of *SP, OP* and the source *U-UPMs* hosting the migrating VMs are excluded from *candidateDestinations,* since a source PM cannot be the new destination PM of its own VMs and to avoid increasing the likelihood of turning the PMs from *OP* into *O-UPMs* again (Line 25, 27 and 28 of Algorithm 4.1). The *DPSVU* algorithm (Algorithm 4.5) is then invoked to select the destination PM from *candidateDestinations* (Line 29 of Algorithm 4.1). The PMs of *candidateDestinations* are first sorted in increasing order of PMRT (Line 4 of Algorithm 4.5) and then the suitability of these PMs from sorted *candidateDestinations* are sequentially checked starting from the smallest to the largest PM in terms of PMRT (Line 5 to Line 6 of Algorithm 4.5). If a PM is found as suitable satisfying both RC (3.3) and MUTC (3.5) constraints as per *PST* Algorithm (Algorithm 4.4), then net energy gain for the potential VM migration is estimated from the difference between reduced energy consumption of source *U-UPM* and increased energy consumption of new destination *U-UPM*. If net energy gain is found as positive, then that PM is selected as the new destination PM (Line 7 to Line 17 of Algorithm 4.5). In the following section, we have discussed about the characteristics of *SRTDVMC* algorithm.

4.3.2 Characteristics of Proposed Algorithm

SRTDVMC attempts to fit the largest VM in terms of VMRT from the smallest PM in terms of PMRT into the next smallest possible PM. Such consolidation approach shortens the PMRT of source PM without raising the PMRT of destination PM, resulting into decreased energy consumption. Additionally, selecting the next smallest possible PM as destination PM ensures that the PM is accomplishing largest possible jobs before moving into sleep state or turned off state. Consequently, remaining workload for the existing active PMs becomes lower, which aids in energy consumption minimization. Furthermore, lesser remaining workload for existing active PMs increases the likelihood that upcoming workload can be served by these active PMs without turning on PMs, which are in lower energy consumption state, for instance, sleep or turned off state. Hence, energy consumption minimization is complemented.

One critical aspect of *SRTDVMC* is that both rise of energy in potential destination PM (i.e., cost) and drop of energy in potential source PM (i.e., benefit) is checked prior any potential VM migration. VMs from *U-UPMs* are migrated only if the net energy gain (i.e., energy drop – energy rise) is positive, which limits the number of VM migrations and improves QoS without compromising energy-efficiency. Hence, *SRTDVMC* can concurrently satisfy both objective functions (4.3) and (4.4). Furthermore, *SRTDVMC* smartly selects destination PMs ensuring that the increased energy consumption of potential destination *U-UPM* does not outweigh the reduced energy consumption of potential source *U-UPM*. It aids to uphold the energy-efficiency of the solution regardless of the drastic rise of state-of-the-art PMs' energy consumption causing declined energy-efficiency at utilization level beyond 70%. Thus, *SRTDVMC* encounters the lack of energy-efficiency issue in the presence of state-of-the-art PMs as experienced with existing DVMC algorithms. As a result, *SRTDVMC* is robust against underlying PMs' change of energy-efficiency characteristics with varying load.

4.4 Performance Evaluation

Fig. 4.1 - Fig. 4.3 representing the diverse range of VMRT of Nectar Cloud, reveal the heterogeneous nature of real Cloud workloads in terms of finishing time. To the best of our knowledge, none of the existing DVMC algorithms except *RTDVMC* is designed considering heterogeneous VMRT in their bedrock assuming all jobs finish at the same time, which is unrealistic. Consequently, these traditional DVMC algorithms cannot provide optimal solution for heterogeneous VMRT. As such, we have compared the performance of *SRTDVMC* with *RTDVMC*, since both are developed considering heterogeneous VMRT in their bedrocks.

4.4.1 Experimental Setup

Elucidated earlier in Chapter 3, performance of *RTDVMC* has been evaluated through CloudSim [27]. Since, performance of *SRTDVMC* has been compared with *RTDVMC*, therefore, we have modelled and simulated a cloud environment in CloudSim [27], which we have used to simulate *SRTDVMC* algorithm under different workload scenarios. For fair comparison, both algorithms have been simulated using same environment with respect to the characteristics of CDC, VM, PM and energy module. The simulated CDC consists of 800 heterogeneous PMs. Three different modern generation of PMs, such as Dell PowerEdgeR940 (Intel Xeon Platinum 8180, 112 cores⇥25000 MHz, 384 GB) [99], HP ProLiant DL560 Gen10 (Intel Xeon Platinum 8180, 112 cores⇥25000 MHz, 384 GB) [100] and HP ProLiant ML350 Gen10 (Intel Xeon Platinum 8180, 28 cores⇥25000 MHz, 192 GB) [101] have been used. Each server is provided with 1 GB/s network bandwidth. The energy consumption characteristics of these servers with varying workload is articulated in Table 4.1.

The characteristics of different VM types match with the VMs used by *RTDVMC* and correspond to Amazon EC2 instance types [105]. However, the difference between the simulated VMs and Amazon EC2 instance types is that the simulated VMs are single-core, which is explained by the fact that the workload data used for the simulations come from single-core VMs. Since, the single-core is used, the amount of RAM is divided by the number of cores for each VM type: High-CPU Medium Instance (2500 MIPS, 0.85 GB); Extra Large Instance (2000 MIPS, 3.75 GB); Small Instance (1000 MIPS, 1.7 GB); and Micro Instance (500 MIPS, 613 MB). Lifetime of a VM V_j, aka VMRT of a VM V_j, denoted by T_{V_j} can be different from one VM to another (i.e., heterogeneous). For further accurate estimation of the performance of both *SRTDVMC* and *RTDVMC* algorithms under real Cloud scenario, T_{V_j} values are drawn from VMRT traces of a real Cloud, namely Nectar Cloud. Nectar Cloud consists of over thousands of VMs across multiple data centers located in eight different cities of Australia [28]. For *SRTDVMC* algorithm, T_{V_j} is converted into *SVMRT*, S_{V_j} as per (3.1), using 0.05 as the value of α and a uniformly distributed random variable ranging $[-1, +1]$ as X. For further clarity, maximum deviation of T_{V_j} from S_{V_j} is ± 5%. At the outset, VMs are provided with the resources defined by the VM types. However, during the lifetime, VMs utilize less resources according to the workload data, widening opportunities for dynamic consolidation. The workload data also reflects traces of real Cloud workload traffic, originated as part of the CoMon project, a monitoring infrastructure for PlanetLab [29]. For both *RTDVMC* and *SRTDVMC*, upper utilization threshold, θ_{max} is considered as 80%. With every workload scenario, a DVMC algorithm has been run twice to generate mean CDC energy consumption and mean total number of VM migration by that DVMC algorithm under such workload scenario. Each time, the simulation has been run until 24 hours CloudSim simulation clock time.

4.4.2 Workload Data and Performance Metrics

4.4.2.1 Workload Data

Workload data reflects traces of real Cloud workload, namely PlanetLab [29]. Earlier in section 3.5.2, we have elucidated the rationale of using workload traces of a real system in simulation-based evaluation. Data of CPU usage of thousands of VMs has been collected every five minutes constituting workload. These VMs had been hosted in PMs spread globally across 500 locations. Both algorithms have been tested with the PlanetLab workload data of four different days: 6 March, 9 March, 9 April and 20 April, as these days are randomly selected from a set of available daily PlanetLab data featuring different sets of varying resource demand over time. The characteristics of different PlanetLab workload data is articulated in Table 4.2.

Table 4.2 Characteristics of PlanetLab Data (CPU Utilization)

Day	Number of VMs	Mean	St. dev.	Quartile 1	Quartile 2	Quartile 3
6 March	898	11.44	16.83	2%	5%	13%
9 March	1061	10.70	15.57	2%	4%	13%
9 April	1358	11.12	15.09	2%	6%	15%
20 April	1033	10.43	15.21	2%	4%	12%

For each workload, the associated VMs' release time or workload finishing time have been drawn from monthly VMRT traces of real Cloud, namely Nectar Cloud. Traces of VMs created in Nectar Cloud over a month along with respective release time of those VMs constitutes the monthly VMRT data. The latest available VMRT data of three different months: November-2013, December-2013 and January-2014 have been used for experiments. To explain more, a single day's PlanetLab workload data is tested with Nectar VMRT data of three different months offering diverse VMRT distributions, so that the impact of heterogeneous workload finishing time or release time can be analysed. Histogram of different months of Nectar VMRT data has been articulated through Fig. 4.1 – Fig. 4.3. The number of VMs in Nectar VMRT data of a month is greater than the number of VMs in the PlanetLab workload data of a day. Therefore, a uniformly distributed random variable has been used to select a smaller set of VMs from monthly Nectar data to match the number of VMs of the daily PlanetLab data. Uniformly distributed random variable proffers the smaller set of VMs with similar VMRT distribution present in the monthly Nectar data.

4.4.2.2 Performance Metrics

SRTDVMC is a multi-objective DVMC algorithm, which aims to minimize the CDC energy consumption and VM migrations. Hence, performance of *SRTDVMC* and *RTDVMC* algorithms have been measured and compared in terms E_{CDC} and ψ. Expressed in (3.7), E_{CDC} is the sum of energy consumption of all the PMs, while each PM's energy consumption is derived from Table 4.1 according to its current CPU utilization. The second metric, ψ is the number of VM migrations initiated during the VM placement adaptation.

4.4.3 Simulation Results and Analysis

SRTDVMC and *RTDVMC* have been simulated under different workload scenarios described earlier in Section 4.4.2.1. Four different days of PlanetLab workload data has been randomly selected (i.e., 6 March, 9 March, 9 April and 20 April). PlanetLab workload data of every single day featuring varying resource demand over time has then been blended with three diverse sets of VMRT data originated from three different months of Nectar Cloud Data (i.e., Nectar Nov, Nectar Dec and Nectar Jan) featuring heterogeneous VMRT. Thus, from a single set of daily PlanetLab workload data, three diverse sets of workload data are produced featuring time variant resource demand and diverse workload finishing time, which matches with real Cloud. Both algorithms are reiterated over multiple times for each set of time variant workload representing a unique combination of PlanetLab and Nectar Cloud data, to produce corresponding $\bar{E}_{CDC}$ and $\bar{\psi}$.

4.4.3.1 Energy Consumption

Values of $\bar{E}^{R}$ and $\bar{E}^{S}$ representing mean CDC energy consumption by *RTDVMC* and *SRTDVMC*, respectively are highlighted in Fig. 4.8. Let, $X^{\bar{E}}$ (4.5) denotes the set representing difference between mean energy consumption by *RTDVMC* and mean energy consumption by *SRTDVMC* for different workload scenarios. In other words, $X^{\bar{E}}_{NT,PL}$ represents the minimization of mean energy consumption proffered by *SRTDVMC* compare to *RTDMC* for diverse workloads, as articulated in Table 4.3. A range of statistical testing are performed with achieved simulation results, which are discussed in following sections.

$$X^{\bar{E}} = \{X^{\bar{E}}_{NT,PL}\}_{|Nectar|\cdot|PLab|} = \{\bar{E}^{R}_{NT,PL} - \bar{E}^{S}_{NT,PL}\}_{|Nectar|\cdot|PLab|} \tag{4.5}$$

Table 4.3 Minimization of Mean CDC Energy Consumption (kW) by *SRTDVMC* compared to *RTDVMC*

		PlanetLab (CPU utilization distribution)			
		6 March	9 March	9 April	20 April
Nectar VMRT	NOV	3.01	2.71	2.75	2.85
	DEC	2.09	2.77	2.3	2.59
	JAN	1.16	0.52	1.3	0.32

- **Normality Testing**

Parametric tests are reported as more powerful than non-parametric tests. Assumption of parametric tests is that data samples are normally distributed. Therefore, prior parametric tests, normality testing is executed. The capability to accurately figure out if a data sample has come from a non-normal distribution, referred to as power, is the most widespread measure of the strength of a normality test [112]. Chi-square test for normality is not as powerful and unsuitable for small data samples. Kolmogorov-Smirnov (K-S) test is reported to have low power to test normality [113] and has high sensitivity issue with extreme values, which is handled by Lilliefors correction [114]. The S-W normality test is regarded as more powerful than the K-S test even after the Lilliefors correction [115] and recommend as the best option for testing the normality of data [112].

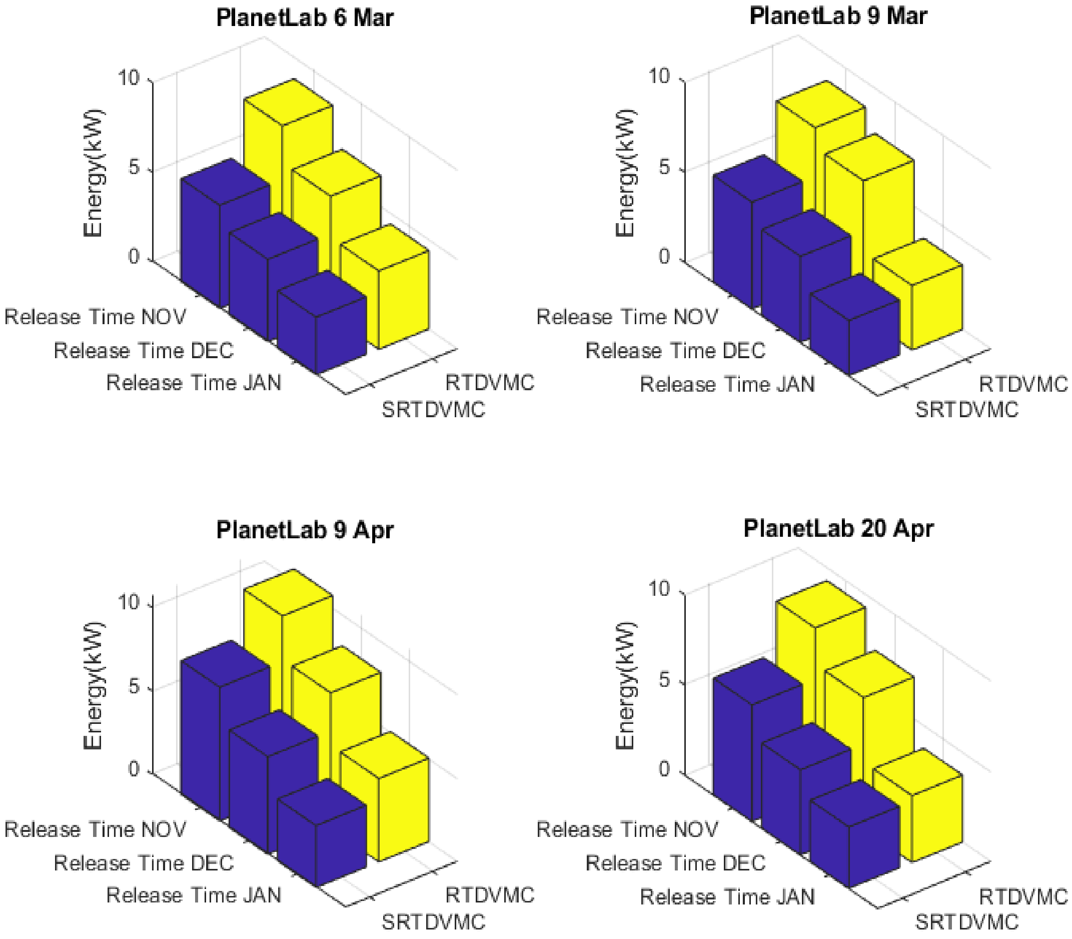

Fig. 4.7 Mean Energy Consumption of *SRTDVMC* vs *RTDVMC*

Test statistics for the S-W normality test with $\bar{E}^R$ and $\bar{E}^S$, referred to as $SW_{\bar{E}}^{R}$ and $SW_{\bar{E}}^{S}$, respectively can be calculated from (4.6) and (4.7) [116, 117]. a_{np} weights are available in Shapiro-Wilk Table [118]. Different $\bar{E}^R_{(np)}$ and $\bar{E}^S_{(np)}$ values are presented in Table 4.4 and Table 4.5.

$$SW_{\bar{E}}^{R} = \left(\sum_{np=1}^{|np|} a_{np} \cdot \left(\bar{E}^{R}_{(|np|+1-np)} - \bar{E}^{R}_{(np)}\right)\right)^2 / \sum_{w=1}^{|w|}\left(\bar{E}^{R}_{(np)} - \bar{\bar{E}}^{R}\right)^2 \tag{4.6}$$

$$SW_{\bar{E}}^{S} = \left(\sum_{np=1}^{|np|} a_{np} \cdot \left(\bar{E}^{S}_{(|np|+1-np)} - \bar{E}^{S}_{(np)}\right)\right)^2 / \sum_{w=1}^{|w|}\left(\bar{E}^{S}_{(np)} - \bar{\bar{E}}^{S}\right)^2 \tag{4.7}$$

$$\bar{\bar{E}}^{R} = \sum_{np=1}^{|np|} \bar{E}^{R}_{(np)} / (|np|) \tag{4.8}$$

$$\bar{\bar{E}}^{S} = \sum_{np=1}^{|np|} \bar{E}^{S}_{(np)} / (|np|) \tag{4.9}$$

Table 4.4 Mean CDC Energy Consumption for *RTDVMC*

np	1	2	3	4	5	6	7	8	9	10	11	12
$\bar{E}^R_{(np)}$	3.58	3.72	4.37	5	6.68	7.3	7.53	8.14	8.62	8.75	9.31	10.76

Table 4.5 Mean CDC Energy Consumption for *SRTDVMC*

np	1	2	3	4	5	6	7	8	9	10	11	12
$\bar{E}^S_{(np)}$	3.06	3.21	3.4	3.7	4.59	4.71	4.76	5.74	5.84	5.92	6.46	8.01

For the S-W normality tests with data samples of $\bar{E}^R$ and $\bar{E}^S$, the Null Hypothesis is that data is normally distributed. We have utilized the software collected from [106] to perform the S-W normality test. For distribution of $\bar{E}^R$ and $\bar{E}^S$, corresponding p values are found as 0.54 and 0.65 respectively, which are greater than critical value, α as 0.05. Hence, no strong evidence could be found to reject Null Hypothesis that elements of $\bar{E}^R$ and $\bar{E}^S$ have come from normal distribution.

- **Parametric Hypothesis Testing and Test Error**

Results for normality testing through the S-W normality test have suggested that elements of $\bar{E}^R$ and $\bar{E}^S$ follow normal distribution, which meets the prior condition of parametric tests. Positive numeric value of every element of $X^{\bar{E}}$ (4.5) articulated in Table 4.3 refer to the fact that energy consumption by *SRTDVMC* is numerically lower compare to *RTDVMC* for different workload scenarios as presented in experiments. We hence aim to perform parametric hypothesis test to find whether simulation output samples, $X^{\bar{E}}_{NT,PL}$ (4.5) featuring difference between two corresponding means of *RTDVMC* and *SRTDVMC* associated to a unique combination of Nectar and PlanetLab workload is statistically significant. Our sample size is less than 30 and means of no more than two DVMC algorithms (i.e., *SRTDVMC* and *RTDVMC*) would be compared. Therefore, among different parametric tests we opt to use the *t-test*, instead of *Z-test*, *F-test* and *ANOVA*. Based on the data samples, the *t-tests* can be classified into three groups: One sample, Two Independent Samples and Paired Samples *t-test*. For a specific combination of *NL* and *PL*, corresponding $\bar{E}^{R}_{NT,PL}$ and $\bar{E}^{S}_{NT,PL}$ has a relationship, as $\bar{E}^{R}_{NT,PL}$ and $\bar{E}^{S}_{NT,PL}$ represent $\bar{E}_{CDC}$ for *RTDVMC* and *SRTDVMC* respectively, under a particular workload distribution scenario. Therefore, the paired two tail *t-test* is performed.

The null hypothesis with the *t-test* is that mean CDC energy consumption with *RTDVMC*, $\bar{E}^{R}$ and mean CDC energy consumption with *SRTDVMC*, $\bar{E}^{S}$ are same, while the alternative hypothesis is that $\bar{E}^{S} < \bar{E}^{R}$. Utilizing (4.10) – (4.12), the test statistic for the *t-test*, denoted by $t_{\bar{X}^{\bar{E}}}$ is found as 2.13 and the corresponding *p* value is found as 7.10693×10^{-6}, which is lower than critical value, α as 0.05. For clear understanding of the interpretation of the *t-test* result, we have first explained the performed the *t-test* in more details in the following.

$$t_{\bar{X}^{\bar{E}}} = \left(\left(\bar{X}^{\bar{E}} - 0\right) / \left(\hat{\sigma}_{\bar{X}^{\bar{E}}}\right)\right) \tag{4.10}$$

$$\bar{X}^{\bar{E}} = \left(\left(\sum_{np=1}^{|Nectar|\cdot|PLab|} \left(X^{\bar{E}}_{np}\right)\right) / \left(|Nectar|\cdot|PLab|\right)\right) \tag{4.11}$$

$$\hat{\sigma}_{\bar{X}^{\bar{E}}} = \sqrt{\frac{\left(\sum\left(X^{\bar{E}}_{(np)} - \bar{X}^{\bar{E}}\right)^{2} / \left(\left(|Nectar|\cdot|PLab|\right)-1\right)\right)}{\left(|Nectar|\cdot|PLab|\right)}} \tag{4.12}$$

Previously, in Section 4.4.3.1, we have shown that distributions of $\bar{E}^{R}$ and $\bar{E}^{S}$ are normal distributions. It is critical to note that, if we subtract two corresponding elements of two different normal distributions, then the resulting distribution is a normal distribution. Hence, elements of $X^{\bar{E}}$ (4.5) featuring difference between two corresponding means of *RTDVMC* and *SRTDVMC* are normally distributed. Since, elements of $X^{\bar{E}}$ are normally distributed, therefore, the distribution of their means, denoted by $\bar{X}^{\bar{E}}$ (4.11) is also a normally distribution. Now, assuming the null hypothesis true that $\bar{E}^{R}$ and $\bar{E}^{S}$ are same, the mean of the distribution of $\bar{X}^{\bar{E}}$ is 0. Nonetheless, utilizing (4.11) with our experimental results articulated in Table 4.3, value of $\bar{X}^{\bar{E}}$ has been found as 2.03. As a result, we have estimated the probability of $\bar{X}^{\bar{E}}$ to be 2.03, given the null hypothesis is true. In order to determine such probability, utilizing (4.10), we have calculated that how many standard deviations far away is our experimental mean (i.e., 2.03) from the distribution mean (i.e., 0), aka the test statistic for the *t-test*.

The test statistic for the *t-test*, denoted by $t_{\bar{X}^{\bar{E}}}$ is found as 2.13 and the corresponding probability is found as 7.10693×10^{-6}. Now, 7.10693×10^{-6} is less than 0.05. In other words, the outcome of the *t-test* shows that if the null hypothesis is true that mean of distribution of $\bar{X}^{\bar{E}}$ is 0, there remains less than 5% chance for $\bar{X}^{\bar{E}}$ to be 2.03. According to the rule of the *t-test*, we can then argue that despite such low probability, since we still have received $\bar{X}^{\bar{E}}$ as 2.03, therefore, the null hypothesis itself cannot be true. So, we retain the alternative hypothesis as true. If $\bar{X}^{\bar{E}}$ (i.e., the mean of minimization of mean CDC energy consumption by *SRTDVMC* compared to *RTDVMC*) was not significant from the perspective of such inferential statistics as the *t-test*, then respective *p* value would not have been found as lower than 0.05, which gives strong evidence to reject the null hypothesis itself. Hence, we have provided evidence through utilizing inferential statistics that the performance improvement through *SRTDVMC* compared to *RTDVMC* in terms of mean CDC energy consumption is statistically significant.

Earlier in Section 3.5.2.2, we have explained two types of error related to the *t-test* - Type I and Type II error. The null hypothesis has been rejected based on the corresponding probability value of 7.10693×10^{-6}. Hence, the probability is 7.10693×10^{-6} that we have rejected the null hypothesis while it was true, aka Type I error. In the following section, the simulation results in relation to VM migration has been presented.

4.4.3.2 VM Migration

VM consolidation is applied to regulate CDC energy consumption. However, one major downside of VM consolidation is that VM consolidation is impossible without VM migration, while, increased VM migration increases network overhead. *SRTDVMC* being a multi-objective DVMC algorithm, is designed to minimize CDC energy consumption without incurring increased VM migration. In Fig. 4.9, we have illustrated mean total number of VM migrations with *RTDVMC* and *SRTDVMC*, denoted by $\bar{\psi}^R$ and $\bar{\psi}^S$, respectively. Let, $X^{\bar{\psi}}$ (4.13) denotes the set representing difference of mean total number of VM migrations between *RTDVMC* and *SRTDVMC* under different workload scenario, as articulated in Table 4.6. In the following section, we have discussed diverse statistical tests performed on our experimental results.

$$X^{\bar{\psi}} = \{X^{\bar{\psi}}_{NT,PL}\}_{|Nectar|\cdot|PLab|} = \{\bar{\psi}^R_{NT,PL} - \bar{\psi}^S_{NT,PL}\}_{|Nectar|\cdot|PLab|} \tag{4.13}$$

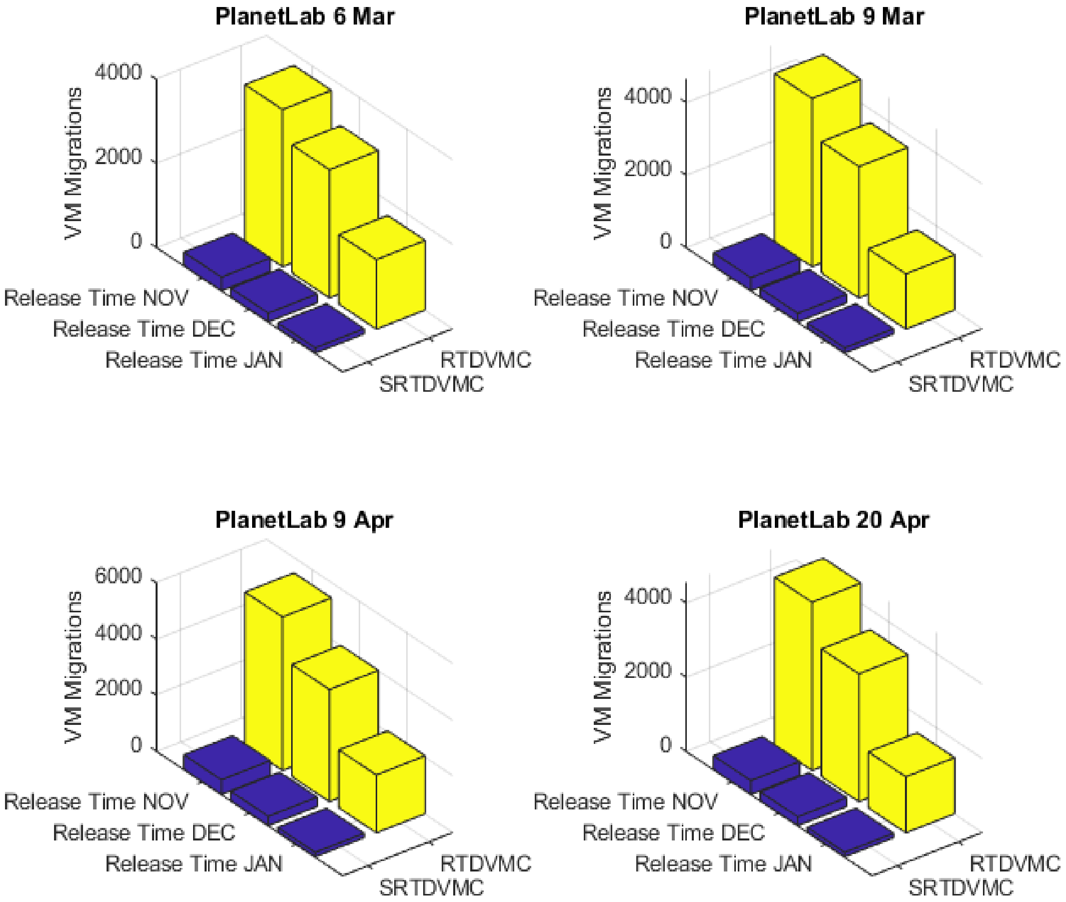

Fig. 4.8 Mean Total Number of VM Migration of *SRTDVMC* vs *RTDVMC*

Table 4.6 Minimization of mean Number of VM Migration by *SRTDVMC* compared to *RTDVMC*

		PlanetLab (CPU utilization distribution)			
		6 March	9 March	9 April	20 April
Nectar VMRT	NOV	3416	4279	4951	4162
	DEC	2838	3385	3667	3206
	JAN	1531	1381	1908	1390

- **Normality Testing**

From Table 4.6, we can observe that *SRTDVMC* outperforms *RTDVMC* in terms of mean of total number of VM migration. One might argue that such improvement is merely a random event and the results are not statistically significant. To rebut such argument, the *t-test* can be performed on experimental results to check if results are statistically significant or not. The condition to be met prior the *t-test* is that data must be normally distributed. We have hence applied the S-W normality test with set of mean of total number of VM migration by *RTDVMC* and *SRTDVMC*, denoted by $\bar{\psi}^R$ and $\bar{\psi}^S$, respectively. Test statistics for the S-W normality test with $\bar{\psi}^R$ and $\bar{\psi}^S$, referred to as $SW_{\bar{\psi}}^R$ and $SW_{\bar{\psi}}^S$, respectively can be calculated from (4.14) and (4.15) [116, 117]. a_{np} weights are available in Shapiro-Wilk Table [118]. Different $\bar{\psi}^R_{(np)}$ and $\bar{\psi}^S_{(np)}$ values are presented in Table 4.7 and Table 4.8.

$$SW_{\bar{\psi}}^R = \left(\sum_{np=1}^{|np|} a_{np} \cdot \left(\bar{\psi}^R_{(|np|+1-np)} - \bar{\psi}^R_{(np)}\right)\right)^2 / \sum_{np=1}^{|np|}\left(\bar{\psi}^R_{(np)} - \bar{\bar{\psi}}^R\right)^2 \tag{4.14}$$

$$SW_{\bar{\psi}}^S = \left(\sum_{np=1}^{|np|} a_{np} \cdot \left(\bar{\psi}^S_{(|np|+1-np)} - \bar{\psi}^S_{(np)}\right)\right)^2 / \sum_{np=1}^{|np|}\left(\bar{\psi}^S_{(np)} - \bar{\bar{\psi}}^S\right)^2 \tag{4.15}$$

$$\bar{\bar{\psi}}^R = \sum_{np=1}^{|np|} \bar{\psi}^R_{(np)} / (|np|) \tag{4.16}$$

$$\bar{\bar{\psi}}^S = \sum_{np=1}^{|np|} \bar{\psi}^S_{(np)} / (|np|) \tag{4.17}$$

Table 4.7 Mean of Total Number of VM Migration for *RTDVMC*

np	1	2	3	4	5	6	7	8	9	10	11	12
$\bar{\psi}^R_{(np)}$	1511	1524	1654	2066	3046	3444	3619	3738	3999	4535	4639	5472

Table 4.8 Mean of Total Number of VM Migration for *SRTDVMC*

np	1	2	3	4	5	6	7	8	9	10	11	12
$\bar{\psi}^S_{(np)}$	120	123	143	157	208	234	238	322	332	360	373	521

The null hypothesis for the S-W normality tests with data samples of $\bar{\psi}^R$ and $\bar{\psi}^S$ is that data is normally distributed. For distribution of $\bar{\psi}^R$ and $\bar{\psi}^S$, corresponding *p* values are found as 0.42 and 0.37 respectively, which are greater than critical value, α as 0.05. Hence, no strong evidence could be found to reject Null Hypothesis that elements of $\bar{\psi}^R$ and $\bar{\psi}^S$ have come from normal distribution. As such, in the following section we have proceeded with the *t-test* to verify if the reduction of VM migration by *SRTDVMC* compared to *RTDVMC* as obtained through experiments is statistically significant.

- **Parametric Hypothesis Testing and Test Error**

Previously in section 4.4.3.1, we have explained the reason of choosing the two sample paired *t-test*. We aim to prove that minimization of mean of total number of VM migration by *SRTDVMC* compared to *RTDVMC* is statistically significant. As null hypothesis we assume that the opposite is true. Hence, the null hypothesis is that mean of total number of VM migration, $\bar{\psi}^R$ and mean of total number of VM migration with *SRTDVMC*, $\bar{\psi}^S$ are same, while the alternative hypothesis is that $\bar{\psi}^S < \bar{\psi}^R$. Utilizing (4.18) – (4.20), the test statistic for the *t-test*, denoted by $t_{\bar{X}^{\bar{\psi}}}$ is found as 2.48 and the corresponding *p* value is found as 1.64×10^{-6}, which is lower than critical value, α as 0.05. In the following we have elaborately discussed the interpretation of the *t-test* result.

$$t_{\bar{X}^{\bar{\psi}}} = \left((\bar{X}^{\bar{\psi}} - 0)/(\hat{\sigma}_{\bar{X}^{\bar{\psi}}})\right) \tag{4.18}$$

$$\bar{X}^{\bar{\psi}} = \left(\left(\sum_{np=1}^{|Nectar|\cdot|PLab|} \left(X_{np}^{\bar{\psi}}\right)\right)/(|Nectar|\cdot|PLab|)\right) \tag{4.19}$$

$$\hat{\sigma}_{\bar{X}^{\bar{\psi}}} = \sqrt{\frac{\left(\sum\left(X_{(np)}^{\bar{\psi}} - \bar{X}^{\bar{\psi}}\right)^2/((|Nectar|\cdot|PLab|)-1)\right)}{(|Nectar|\cdot|PLab|)}} \tag{4.20}$$

In section 4.4.3.2, we have shown that such distributions as $\bar{\psi}^R$ and $\bar{\psi}^S$ are normal distributions. As a result, elements of $X^{\bar{\psi}}$ (4.13) are normally distributed. Furthermore, since elements of $X^{\bar{\psi}}$ are normally distributed, therefore, distribution of their means, denoted by $\bar{X}^{\bar{\psi}}$ (4.19) is also a normally distribution. Now, under the given null hypothesis that $\bar{\psi}^R$ and $\bar{\psi}^S$ are same, the mean of the distribution $\bar{X}^{\bar{\psi}}$ is 0. Applying results articulated in Table 4.6, into (4.19), $\bar{X}^{\bar{\psi}}$ is found as 3010. We have then determined the probability of $\bar{X}^{\bar{\psi}}$ to be 3010, under the scenario that null hypothesis is true (i.e., $\bar{X}^{\bar{\psi}}$ is 0). To determine that probability, we have estimated the distance between our experimental mean (i.e., 3010) and the distribution mean (i.e., 0) in terms of standard deviation, aka the test statistic for the *t-test*, $t_{\bar{X}^{\bar{\psi}}}$ (4.18).

$t_{\bar{X}^{\bar{\psi}}}$ is found as 2.48 and the corresponding probability is found as 1.64×10^{-6}. Now, 1.64×10^{-6} is less than 0.05. In other words, the outcome of the *t-test* shows that given the null hypothesis is true, there is less than 5% chance of $\bar{X}^{\bar{\psi}}$ to be 3010. Nevertheless, despite such low probability, since we still have received $\bar{X}^{\bar{\psi}}$ as 3010, therefore, we can rebut that the null hypothesis itself is not true. Hence, we retain the alternative hypothesis as true. If the mean of minimization of mean CDC energy consumption by *SRTDVMC* compared to *RTDVMC* was insignificant, then respective *p* value would not have been found as lower than 0.05. Thus, through the *t-test* results we have provided evidence that the reduction of VM migration by *SRTDVMC* compared to *RTDVMC* is statically significant. Since, the null hypothesis has been rejected based on the estimated probability of 1.64×10^{-6}, therefore, the probability of false rejection of null hypothesis while it was true, aka Type I error is 1.64×10^{-6}, which is very low.

4.5 Observations

Experimental results also reveal several critical aspects as discussed in the following.

Observation 1: From experiments results portrayed in Fig. 4.8 – Fig. 4.9, we can observe that such traditional DVMC algorithm as *RTDVMC* lacks in performance compared to *SRTDVMC* with the presence of state-of-the-art highly energy proportional PMs. Performance lacking by traditional DVMC algorithm including *RTDVMC* is attributed to its flawed working principal that maximum energy-efficiency is attainable through maximum load on PMs.

Observation 2: DVMC algorithm reduces energy consumption through VM migration, which detrimentally affects QoS. As such, concomitant minimization of energy consumption and VM migration are confronting objectives. Hence, developing a DVMC algorithm, which optimizes energy-efficiency without increasing the number of VM migration is a much greater challenge than designing an algorithm that merely focuses on the former aspect and ignores the later aspect. *SRTDVMC* being designed to optimize both aspects, only migrates a VM if the respective benefit is greater than the corresponding cost. Our research outcome, as illustrated in Fig. 4.8 and Fig. 4.9, substantiates the success of such strategic VM consolidation technique of *SRTDVMC* in both aspects.

Observation 3: Experimental results reveal that energy consumption and VM migration correspond to VMRT. The number of PMs has not been altered. PlanetLab workload data of every single day has been combined with three different VMRT distributions of Nectar Cloud. Hence, from one single set of PlanetLab data of a day, three different sets of workload data have been created representing same number of VMs and same varying resource demand, but different VMRT distributions and performances of both algorithms change with the change in VMRT distributions. The underlying reason is that considering the entire lifetime, aggregated resource consumption by a VM with a relatively large VMRT is likely to be greater than the aggregated resource consumption by a VM with smaller VMRT. In addition, longer VMRT provides more time resulting into more likelihood of VM migrations.

Observation 4: Our research has highlighted that a correlation exists between energy consumption and VMRT. Existence of such correlation has also been found as true between number of VM migrations and VMRT. To elucidate further, one common pattern with both *SRTDVMC* and *RTDVMC* is unfolded that for any day's PlanetLab data, VMRT distribution of

Nov 2013 displays the highest energy consumption and highest number of VM migration, while VMRT distribution of Jan 2014 proffers to the lowest energy consumption and lowest number of VM migration (Fig. 4.8 and Fig. 4.9). The answer lies within VMRT distributions of these months (Fig. 4.1 - Fig. 4.3). Considering the sum of VMRT of all VMs, VMRT distribution of Nov 2013 consists of total 8343 days, which is the highest among all months, while VMRT distribution of Jan 2014 consists of total 727 days, which is the lowest among all months. Since, Nov 2013 represents the highest total workload duration resulting into highest total resource utilization; therefore, maximum energy consumption is observed for Nov 2013. Due to maximum duration of total workload existence, number of VM migrations is also found as maximum with Nov 2013. Similarly, since Jan 2014 features the lowest total workload duration resulting into lowest total resource utilization, hence, minimum energy consumption is observed for Jan 2014. Minimum duration of workload existence results into lowest number of VM migration for Jan 2014.

Observation 5: Experimental results in Fig. 4.8 and Fig. 4.9 also present the fact that energy consumption and VM migration are affected by the change in resource demand. Nectar VMRT data of one single month is blended with four different days of PlanetLab data resulting into four different sets of workloads, representing same VMRT, but different sets of resource demand and the performance of both algorithms change due to the variation in resource demand. The reason can be explained through equations (3.6) and (3.7), showing that energy consumption is a function of CPU utilization, while the utilization refers to the sum the resource demand (3.1). Hence, energy consumption is affected by the change in resource demand. The reason of observing the change in total VM migrations with the change in resource demand is that further opportunities of VM consolidation arises as the resource demand changes. DVMC algorithm keeps capitalizing such consolidation opportunities through further VM migration and hence, number of VM migration changes with the change in resource demand.

Observation 6: Furthermore, energy consumption and VM migration are associated with number of VMs. PlanetLab 9 April features the highest number of VMs, while PlanetLab 6 March holds the lowest number of VMs, which reflects on energy consumption (Fig. 4.8) and VM migration (Fig. 4.9). Higher the number of VMs, higher is the energy consumption and VM migration. The reason is that more VMs consume more resources, resulting into higher energy consumption and more VMs would normally contribute to a greater number of VM migrations.

4.6 Summary

While correlation exists between VMRT and energy consumption, traditional DVMC algorithms except *RTDVMC* do not consider heterogenous VMRT in VM consolidation decision process. Furthermore, existing algorithms including *RTDVMC* consolidate VMs in as few PMs as possible based on the premise that optimal energy efficiency can be achieved with maximum load on PM. However, for state-of-the-art PMs, energy-efficiency rather drops above 70% load level. Combining lack of consideration of heterogeneous VMRT and ignoring changed energy-efficiency characteristics of underlying PMs, existing DVMC algorithms lack in performance in the context of real Cloud scenario with heterogeneous VMRT and state-of-the-art PMs.

Issue with *RTDVMC* are twofold – first, it does not take the changed energy-efficiency characteristics of modern PMs into account and second, it only aims to minimize energy consumption without considering VM migration minimization. VM migration, nonetheless, increases network overload causing degraded QoS and increased energy consumption by networking equipment. VM migration being an unavoidable part of VM consolidation, minimizing both energy consumption and VM migrations at the same time are confronting objectives. As such, in this chapter, we have brought forth a novel multi-objective DVMC algorithm, namely *SRTDVMC*, which aims to reduce VM migrations without compromising energy-efficiency. Consideration of heterogeneous VMRT values in VM consolidation decision process enables *SRTDVMC* to be more energy-efficient. On top of that, contrast to *RTDVMC*, *SRTDVMC* incorporates consideration both benefit and cost prior any VM migration. As a result, it is robust against the changed energy-efficiency characteristics of underlying PMs and can reduce VM migration without compromising energy-efficiency compare to *RTDVMC*.

Performance of *SRTDVMC* has been tested through most popular Cloud based simulation tool, namely CloudSim in the context of hundreds of different cutting-edge PMs and thousands of VMs representing heterogenous VMRT of real Nectar Cloud, as the assigned workload reflects real Cloud workload obtained from PlanetLab. The empirical outcome exhibits the superiority of *SRTDVMC* over *RTDVMC* in both metrics - CDC energy consumption and VM migration. Three key elements are extracted from our research. First, based on our experiments, VMRT impacted on both aspects - energy consumption and VM migration, and hence, we have suggested to develop DVMC algorithm considering the presence of heterogeneous VMRT. Second, such working principal of existing algorithms that maximum energy-efficiency is achievable at maximum load on PM is found as false for state-of-the-art PMs, resulting into performance inefficiencies. Third, simulation results show that if corresponding cost and benefit are considered prior VM migration, then concomitant optimization of both aspects - reduction of energy consumption and VM migration can be achieved. Findings of this chapter have been carried in developing a further optimized DVMC algorithm, which we have presented in the next chapter.

5. Predictive Release Time based Multi-Objective Dynamic VM Consolidation for Cloud Data Centers with Highly Energy-Proportional PMs

5.1 Introduction

In last two chapters we have presented two DVMC algorithms: *RTDVMC* and *SRTDVMC*, which include heterogeneous VM Release Time (*VMRT*) into every aspect of VM consolidation decision process. In addition, existing DVMC algorithms are constructed on the basis that maximum energy-efficiency is achievable at maximum load level of a PM, whereas Fig. 4.6 pointed out that energy-efficiency rather decreases beyond 70% load level for state-of-the-art highly energy proportional PMs. As such, *SRTDVMC* algorithm has been presented previously in Chapter 4, which is robust regardless of the energy-efficiency characteristics of underlying PMs. Nevertheless, several critical caveats are needed to be addressed in relation to *RTDVMC* and *SRTDVMC* alike, as highlighted in the following:

- For both *RTDVMC* and *SRTDVMC* algorithms, *VMRT* is one key input parameter required to be known prior initiation of consolidation of VMs. Both algorithms assume that CDC owner/IAAS provider would receive *VMRT* information in advance from respective users, which is used as input to the DVMC algorithm. However, in real Cloud environment, hundreds of thousands of users are often provisioned with pay-as-you-go privilege. In other words, such users are provided with the freedom to first use VMs as long as they wish and pay later just for that duration of usage. Consequently, users might not always be able to provide *VMRT* information in advance to IAAS providers. Hence, in the context of real Cloud scenario, neither *RTDVMC* nor *SRTDVC* is applicable.
- With respect to some applications, QoS and potential resource demand regulates *VMRT*. To explain more, variation in number of users causes fluctuation in resource demand, resulting into addition of new VMs and later deletion of such VMs. During SLA, a SAAS provider/PAAS user hence must consider the potential number of application users and mention it to PAAS provider so that by taking the potential number of end users and corresponding potential resource demand into account a certain standard of

QoS can be uphold. Considering the promised standard of QoS to be uphold and potential time-variant resource demand, PAAS provider/IAAS user can estimate resource/VM release time [111] and must provide such information to IAAS provider.

As such, CDC owner/IAAS provider cannot apply any of *RTDVMC* and *SRTDVMC* algorithms in CDC unless on two conditions: firstly, PAAS providers/IAAS users must be capable of calculating *VMRT* and secondly, PAAS providers/IAAS users must be willing to share such information of *VMRT* with CDC owner/IAAS provider. It is important to realise that in order to estimate *VMRT*, PAAS providers/IAAS users must store large volume of records related to previous time-variant resource demand and then analyse such large amount of data to obtain potential resource demand. Storing, maintaining and analysing large volume of data exerts an additional inconvenient commitment to be undertaken by PAAS providers/IAAS users, which is not ideal in the context present Cloud business model. As such, in the context of existing relationship dynamics between CSUs and CSPs, scope of embodiment of *RTDVMC* and *SRTDVMC* algorithms in CDC is limited.

- In relation to existing Cloud service-oriented business models, CSUs pay to CSPs for the usage of Cloud based services through which CSPs make profit [119] and CSPs do not commonly share profit back with CSUs to any extent. For PAAS providers/IAAS users (i.e., CSUs), determination of *VMRT* begets further investment in development of required skills and facilities to manage and operate large volume of data. However, DVMC algorithm being exercised by IAAS providers in CDC, respective profit achieved through reduced CDC energy consumption is only accumulated in the pockets of CDC owners/IAAS providers. In current business model, no financial stimulation from CDC owners/IAAS providers is provided towards PAAS providers/IAAS users. As such, *RTDVMC* and *SRTDVMC* algorithms would stretch the current operational cost of CDC, since additional cost of providing incentives to PAAS providers/IAAS to obtain *VMRT* would be incurred on IAAS providers.
- To regulate energy consumption both SVMC and DVMC algorithms are applied together in CDC. SVMC algorithm being the first energy consumption minimization measure strategically selects the initial PM for a VM when it is first created in CDC, as DVMC algorithm is the following step to consolidate VMs in lesser number of PMs if opportunity arises [104]. Different SVMC algorithms work in different ways because of being developed to fulfil diverse objectives. Different DVMC algorithms also work

different ways. Same as DVMC problem, SVMC itself is another NP-hard problem. Consequently, to the best of our level of understanding of existing literature [120-140], many researchers focus either only with SVMC problem or cannot evaluate performance of their proposed DVMC algorithm under diverse SVMC algorithms. However, performance of a DVMC algorithm may change with the choice of SVMC algorithm. Hence, it is critical to examine the performance of a DVMC algorithm with diverse SVMC algorithms.

In this chapter, we aim to address those critical research challenges discussed above. Our research contributions are articulated in the following:

- Instead of *VMRT* being calculated by IAAS users, which necessitates payment of proper premiums to IAAS users attributing increased operational cost for IAAS providers, we have introduced a mathematical model to be used by IAAS providers to estimate *VMRT*. The model predicts *VMRT* of a user based on the past *VMRT* records of that user. As part of existing billing process, IAAS providers need to collect the resource usage time/*VMRT* records of respective IAAS users. Therefore, as opposed to *RTDVMC* and *SRTDVMC*, management of users' past *VMRT* information in order to obtain predicted *VMRT* (*PVMRT*) through our proposed mathematical model does not pose any unprecedented additional cost on IAAS users. Hence, *PVMRT* pulls down the business operational cost, which is a huge advantage.
- We have next presented a novel DVMC algorithm, namely *Predicted Release Time based DVMC* (*PRTDVMC*) algorithm, which uses *PVMRT* and hence, excludes such constraints as IAAS users' obligations to afford the calculation of *VMRT* and providing that information to IAAS providers. Eradication of such inconvenient constraints on IAAS providers transforms *PRTDVMC* as non-conflicting with existing relationship dynamics between CSUs and CSPs and perfectly aligns with the present Cloud business model. Even for pay-as-you-go scenario, IAAS user can utilize previous pay-as-you-go type users' *VMRT* records to obtain respective *PVMRT*. Hence, *PRTDVMC* precludes such biggest drawback as limited scope of applicability of *RTDVMC* and *SRTDVMC* in real CDC.
- In previous chapters we have highlighted that existing DVMC algorithms loses efficiency on account of not being developed considering energy-efficiency characteristics of highly energy proportional PMs and heterogeneous workload in their bedrocks. *PRTDVMC* is immune against changed energy-efficiency characteristics of

state-of-the-art highly energy proportional PMs and suitable for workload featuring heterogeneous *VMRT*.

- In contrast with existing literature [120-140], scope of which is confined to either SVMC problem or their proposed DVMC algorithms are not tested with diverse SVMC algorithms, we have stretched our research to analyse the impact of four diverse SVMC algorithms on *PRTDVMC* algorithm and compared with an existing DVMC algorithm, namely THR-MMT [51].

In Section 5.1, mathematical models to generate *PVMRT* and *Predicted PM Release Time* (*PPMRT*) have been presented, followed by objective functions and various constraints related to DVMC algorithm in Section 5.2. In Section 5.3, we have brought forth our proposed heuristic DVMC algorithm, *PRTDVMC*, as the characteristics of the algorithm has been discussed in Section 5.4. To examine the effect SVMC algorithm for *PRTDVMC*, we have presented four diverse notable SVMC algorithms in Section 5.5. The performance evaluation of the proposed algorithm in conjunction with diverse SVMC algorithms under real cloud based workload has been elucidated in Section 5.6, followed by analysing results and rigorous statistical testing in Section 5.7. Finally, in Section 5.8, we have summarized our research.

All notations along with respective meaning, which are used in this chapter, have been articulated previously in the Nomenclature Table, at the outset of the thesis. If necessary, reader can easily figure out the meaning of a notation through looking up the table.

5.2 VM Release Time Predictor Model and PM Release Time

One key feature of our proposed *PRTDVMC* algorithm is that *PVMRT* is invested in VMC decision process. There is plenty of choices available in the literature that can be utilised to generate *PVMRT*. Different prediction techniques work in different ways, as are designed to be more suitable in different scenarios. It is not always possible to specify a function that fits all past *VMRT* data of a CSU. The biggest advantage of Loess Regression (LR) proposed in the year 1979 [141] is that it is ideal for modelling complex processes for which no theoretical model exists [142]. LR works on the basis that any function can be well approximated in a small neighbourhood by a low-order polynomial and simple models can fit to data. Based on that LR does not require the specification of a function to fit a model to the entire sample data (i.e., all past *VMRT* data of a user). In the following we have presented elaborate discussion on applying LR to obtain *PVMRT*.

5.2.1 Loess Regression Based VM Release Time Predictor Model

The application method of LR model is that with every single *VMRT* point/record, the $slope$ and $intercept$ of a low degree polynomial would be first determined, which fits to a subset of *VMRT* records extracted from the entire data set of past *VMRT* records of a user. The subset would consist of some *VMRT* records located near that particular single *VMRT* point whose response is being measured. In other words, *VMRT* records belong to the subset localized around the target *VMRT* point. Utilising the $slope$ and the $intercept$, the corresponding predicted *VMRT* value would be determined for every single original *VMRT* record. A curve would eventually hence be built up to approximate the entire original data.

The size of the subset is defined by a parameter of LR model, namely bandwidth. To avoid higher biases at boundaries [143] and to avoid over-fitting [142], the degree of the polynomial as used to be fitted by our LR model is taken as 1. The equation of the parametric function with degree 1 is presented in (5.1).

$$y = (slope \cdot x) + inercept \tag{5.1}$$

The parametric function (5.1) is fitted to the subset of *VMRT* records using the *weighted least-squares* method with weight $w_a(x)$ at $(x_a\, , y_a)$, where x_a denotes the index of the a^{th} *VMRT* record and y_a denotes the a^{th} original *VMRT* value. In other words, using *weighted least-squares* method values of $slope$ and $intercept$ are determined through minimizing (5.2).

$$\sum\left(\left(w_a(x)\right) \cdot \left(y_a - \; intercept - \; \left((slope) \cdot (x_a)^2\right)\right)^2\right) \tag{5.2}$$

As per *weighted least-square method*, in order to minimize (5.2) respective weight $w_a(x)$ of a *VMRT* record $(x_a\, , y_a)$ is required. According to LR model, weight $w_a(x)$ is calculated using (5.3) on the basis of *tricube weight function*, $T(u)$ presented in (5.4), where b denotes the number of elements in the subset, x_1 refers to the index of the b^{th} observation/*VMRT* record from the right boundary and $\Delta_i(x_b)$ denotes the distance between x_i and x_b (5.5).

$$w_a(x) = \; T\left(\left(\Delta_a(x_b)\right)/\left(\Delta_1(x_b)\right)\right) \tag{5.3}$$

$$T(u) = \begin{cases} (1 - |u|^3)^3, & if\ |u| < 1 \\ 0, & otherwise \end{cases} \tag{5.4}$$

$$\Delta_a(x_b) = |x_b - x_a| \tag{5.5}$$

Since, $1 < a < b$, therefore $x_1 < x_a < x_b$. Hence, $\left((\Delta_a(x_b))/(\Delta_1(x_b))\right) < 1$. Consequently, based on (5.3), weight $w_a(x)$ can now be simplified through (5.6).

$$w_a(x) = T\left((\Delta_a(x_b))/(\Delta_1(x_b))\right) = \left(1 - |((\Delta_a(x_b))/(\Delta_1(x_b)))|^3\right)^3 \tag{5.6}$$

After calculating the $slope$ and $intercept$ through minimizing (5.2), the corresponding *PVMRT* value, $\hat{y}_a$ for the original a^{th} *VMRT* value, y_a of a user can be determined from (5.7).

$$\hat{y}_a = (slope \cdot x_a) + intercept \tag{5.7}$$

Every prediction technique comes with its own set of advantages and disadvantages. The LR model [141] we have discussed thus far is vulnerable to outliers. However, outlier is highly likely to be present in data. To resolve that issue, an updated version of LR has been proposed in the year 1988 [144]. We refer to this updated LR model as Robust Loess Regression (RLR) model. For our proposed *PRTDVMC* algorithm, we have utilised RLR model to predict *PVMRT*.

5.2.2 Robust Loess Regression Based VM Release Time Predictor Model

One big advantage of RLR is that, contrast to LR, RLR incorporates residuals, $\hat{\varepsilon}_a$ into the prediction model. Residuals, $\hat{\varepsilon}_a$ can be calculated from (5.8), where $\hat{y}_a$ is determined from (5.7).

$$\hat{\varepsilon}_a = (y_a - \hat{y}_a) \tag{5.8}$$

RLR model includes residuals, $\hat{\varepsilon}_a$ into weight, as the weight is updated based on residuals, $\hat{\varepsilon}_a$. We name such weight as robustness weight, $r_a(x)$. Each observation/*VMRT* record (x_a, y_a) is assigned with the robustness weight, $r_a(x)$. Robustness weight, $r_a(x)$ is calculated from (5.9) using *bisquare weight function*, $B(u)$ articulated in (5.10).

$$r_a(x) = B\left((\hat{\varepsilon}_a)/\left(6 \cdot median(\{\hat{\varepsilon}_a\}_{|b|})\right)\right) \tag{5.9}$$

$$B(u) = \begin{cases} (1 - |u|^2)^2, & if\ |u| < 1 \\ 0, & otherwise \end{cases} \tag{5.10}$$

For RLR model, as per *weighted least-squares* method values of $slope$ and $intercept$ are determined through minimizing (5.11).

$$\sum\left(\left(r_a(x)\right)\cdot\left(y_a - intercept - \left((slope)\cdot(x_a)^2\right)\right)^2\right) \tag{5.11}$$

Values of $slope$ and $intercept$ generated through minimizing (5.11) are then utilised to obtain the corresponding *PVMRT* value, $\hat{y}_a$ for the original a^{th} *VMRT* value, y_a of a user. The equation to determine $\hat{y}_a$ is articulated in (5.7).

5.2.3 Safety Margin

Even though RLR incorporates residuals, prediction errors, such as over-estimation or under-estimation of *VMRT* may still take place resulting into degraded performance. Being inspired from the research presented by [145], we have hence opted to a further conservative approach through investing a safety margin in our *VMRT* predictor model. Opposed to static approach, adaptive approach has been embodied to set the safety margin, since it is adjusted based on the accuracy of the predictor. The safety margin value essentially depends upon the distance of *PVMRT* values obtained through RLR model from the corresponding original ones. Let, Δ_a^{RLR} denotes the distance between a user's a^{th} original *VMRT* value, y_a and the corresponding predicted *VMRT* value, $\hat{y}_a^{RLR}$ obtained through RLR model. Δ_a^{RLR} can be estimated from (5.12).

$$\Delta_a^{RLR} = |y_a - \hat{y}_a^{RLR}| \tag{5.12}$$

For RLR based *VMRT* predictor model, we have invested *Exponential Moving Average* (EMA) of Δ_a^{RLR} as safety margin, since EMA adapts more quickly than simple moving average [146, 147]. The safety margin for a^{th} *PVMRT* value, δ_a^{RLR} can be obtained from (5.13), where $0 < \gamma < 1$.

$$\delta_a^{RLR} = \left((1-\gamma)\cdot\delta_{a-1}^{RLR}\right) + \left(\gamma\cdot\Delta_{a-1}^{RLR}\right) \tag{5.13}$$

Including the safety margin, the *PVMRT* value, $\hat{y}_a^{RLR\_SM}$ for corresponding $\hat{y}_a^{RLR}$ can be obtained from (5.14).

$$\hat{y}_a^{RLR\_SM} = \hat{y}_a^{RLR} + \delta_a^{RLR} \tag{5.14}$$

5.2.4 Predicted PM Release Time

Another important term used in our proposed *PRTDVMC* algorithm is *Predicted PM Release Time* (*PPMRT*). *PPMRT* refers to the time when a PM can be placed into lower amount of energy state such as sleep state or shut down state. A PM can be placed into lower energy consumption state, if it has no VM hosted in it. Since *PVMRT* refers to the maximum time until which the VM would exist, *PPMRT* denotes the maximum among all the *PVMRT* values of VMs hosted in that PM. Let, $\hat{T}_{V_j}$ denotes *PVMRT* of VM V_j and let $\hat{T}_{V^i}$ denotes the set of *PVMRT* of VMs hosted in PM, P_i, as shown in (5.15).

$$\hat{T}_{V^i} = \left\{\hat{T}^i_{V_j}\right\}_{|V^i|} \tag{5.15}$$

Hence, *PPMRT* of PM, P_i denoted by $\hat{T}_{P_i}$ can be calculated from (5.16).

$$\hat{T}_{P_i} = \max(\hat{T}_{V^i}) \tag{5.16}$$

5.3 Modelling Resource Utilization, Constraints, Energy Consumption and Objective Functions

Before bringing forth the *PRTDVMC* algorithm, resource utilization model, respective constraints, energy consumption model and objective functions are needed to be discussed. For *PRTDVMC*, we have utilized the resource utilization model presented in (3.1). The constraints (3.3) and (3.5) used in the algorithm have been previously explained in Section 3.4. The energy consumption model of PM and CDC have been outlined in (3.6) and (3.7).

In order to relate closely to the real energy consumption by PMs, for our energy consumption model, we have opted to draw energy consumption benchmark results of three different types of state-of-the-art PMs: Dell PowerEdgeR940 (Intel Xeon Platinum 8180, 112 cores→25000 MHz, 384 GB) [99], HP ProLiant DL560 Gen10 (Intel Xeon Platinum 8180, 112 cores→25000 MHz, 384 GB) [100] and HP ProLiant ML350 Gen10 (Intel Xeon Platinum 8180, 28 cores→25000 MHz, 192 GB) [101]. In Table 4.1, we have articulated respective energy consumption of these three types of PMs at varying load level. As articulated in (3.7), the total energy consumption of the CDC is the sum of energy consumption of all the PMs in the CDC. From Table 4.1, we can determine the respective energy consumption of a PM based on its CPU utilization.

Similar to *SRTDVMC*, *PRTDVMC* maintains equity between energy consumption and VM migration. Hence, *PRTDVMC* aims to minimize energy consumption and VM migration alike. The corresponding objective function has been outlines in (4.3) and (4.4).

5.4 Proposed Solution - *PRTDVMC* Algorithm

Based on such models of resource utilization, CDC energy consumption and constraints, in the following we have presented the *PRTDVMC* algorithm (Algorithm 5.1), which invests *PPMRT*, *PVMRT* calculated through RLR model in conjunction with the safety margin module.

5.4.1 Two Phases of *PRTDVMC* Algorithm

PRTDVMC algorithm (Algorithm 5.1) has two phases. In *the first phase* (Line 1 – 18 of Algorithm 5.1), the VMs from *O-UPMs* are migrated out to encounter resource contention issue and *the second phase* (Line 19 – 26 of Algorithm 5.1), VMs are consolidated in lesser number of PMs so that energy consumption can be regulated. First, we have elaborately discussed on *the first phase* and later, we have elucidated *the second phase*.

Algorithm 5.1 The *PRTDVMC* algorithm

```
Input: $P$, $V$, $R$, $SP$
Output: VM Placement
The first phase: O-UPMs
1:    for each $P_i$ in $P$ do
2:        if (3.4) is satisfied then
3:            $OP \leftarrow \{P_i \cup OP\}$
4:        end if
5:    end for        // end of for loop from Line 1 to 5
6:    for each $P_o$ in $OP$ do
7:        $migratingVMs \leftarrow$ Invoke the $VSO$ algorithm with $P_o$
8:        $vmsToMigrate \leftarrow \{migratingVMs \cup vmsToMigrate\}$
9:    end for        // end of for loop from Line 6 to 9
10:   $NOP \leftarrow \{P - OP\}$
11:   for each $V_l$ in sorted $vmsToMigrate$ do
12:       $P_d \leftarrow$ Invoke the $DPSVO$ algorithm with $V_l$ and $NOP$
13:       $destinationPMs \leftarrow \{P_d \cup destinationPMs\}$
14:       if $P_d$ is in $SP$ then
15:           $SP \leftarrow \{SP - P_d\}$
16:       end if          // end of if from Line 14 to 16
18:   end for         // end of for loop from Line 11 to 18
      The second phase: U-UPMs
19:   $candidateSources \leftarrow \{P - OP - SP - destinationPMs\}$
20.   Sort $candidateSources$ in order of increasing $PPMRT$
21:   $candidateDestinations \leftarrow \{P - OP - SP\}$
22:   for each $P_c$ in sorted $candidateSources$ do
23:       $candidateDestinations \leftarrow \{candidateDestinations - P_c\}$
```

```
24:         destinations ← Invoke the DPSVU algorithm with $P_c$
                    and candidateDestinations
25:         candidateSources ← {candidateSources – destinations}
26:     end for          // end of for loop from Line 22 to 26
```

5.4.1.1 The first phase *O-UPMs*

Resource demand of VMs, which constitutes hosting PMs' resource utilization (3.1), changes over time. When a PM's Resource utilization level exceeds a certain threshold value, θ_{max} then that PM is considered as *O-UPM* (3.4). VMs of an *O-UPM* are afflicted with resource contention, yielding into degraded QoS. To control such degraded QoS, VMs are migrated out from *O-UPM*s. As such, for every PM, utilization of different types of resources, such as CPU, Memory and Bandwidth are compared against θ_{max} and if utilization for any type of resource is found as greater than θ_{max}, then that PM is considered as *O-UPM* (Line 1 - 5 of Algorithm 5.1). With every *O-UPM*, one or more VM(s) are needed to be selected strategically, which will be migrated out so that QoS degradation can be halted and then these VMs are required to be intelligently placed in new PMs to regulate the potential increase of energy consumption. *PRTDVMC* algorithm (Algorithm 5.1) first invokes The VMs Selection From *O-UPM* (*VSO*) algorithm (Algorithm 5.2) to determine which VMs to migrate out from the *O-UPM* (Line 6 – 9 of Algorithm 5.1). As part of the new destination PMs selection process to host VMs of *O-UPMs* (Line 10 – 18 of Algorithm 5.1) the Destination PM Selection for VM of *O-UPM* (*DPSVO*) algorithm (Algorithm 5.3) is utilized. In the following section, we have presented the *VSO* algorithm (Algorithm 5.2), which is used to determine VM(s) to be migrate out from an *O-UPM*.

- **VMs Selection From *O-UPMs***

VSO algorithm (Algorithm 5.2) is used to determine the set of VMs, which would be migrated out from an *O-UPM*. The name of such set of VMs is *migratingVMs*. Among all VMs hosted in an *O-UPM*, the largest ones in terms of *PVMRT* are selected to migrate out until the resource utilization of that PM drops below θ_{max}. In order to do that VMs are first sorted in descending order to *PVMRT* (Line 1 of Algorithm 5.2). The first VM from the sorted list of VMs is added in *migratingVMs* in order to migrate out and then checked if the resource utilization of the PM becomes lower than θ_{max}. If yes, then the algorithm returns *migratingVMs* and terminates. If not, the second VM from the sorted list of VMs is included in *migratingVMs* and checked whether the PM's utilization now crosses below θ_{max}. Such addition of VMs in *migratingVMs* is repeated until the PM's utilization plummets lower then θ_{max} (Line 2 – 6 of Algorithm 5.2). Once the utilization drops lower than θ_{max}, the PM is no more an *O-UPM* and formation of *migratingVMs* is completed. The VSO algorithm then returns the *migratingVMs* and terminates (Line 7 of Algorithm 5.2).

Algorithm 5.2 The VMs Selection From O-UPM (*VSO*) algorithm

Input: The *O-UPM*, P_o

Output: List of VMs to be migrated out from the given *O-UPM*

```
1:  Sort $V^o$, the set of VMs of $P_o$ in order of decreasing $PVMRT$
2:  $q \leftarrow 1$
3:  while $q \leq |V^o|$ and $\theta_{max} < U_o^k$ for any $R_k$ in $R$ do
4:      $migratingVMs \leftarrow \{V_q^o \cup migratingVMs\}$
5:      $q \leftarrow q + 1$
6:  end while
7:  return migratingVMs
```

PRTDVMC (Algorithm 5.1) identifies all *O-UPMs* and forms a set of such PMs, namely *OP*. For every PM of *OP*, the VMs Selection From *O-UPM* (*VSO*) algorithm (Algorithm 5.2) is invoked to determine which VMs to migrate out from an *O-UPM*. *VSO* algorithm returns the set of migrating VMs, namely *migratingVMs* from an *O-UPM*. Combining sets of *migratingVMs* extracted from all *O-UPMs*, the set of all migrating VMs from *O-UPMs*, namely *vmsToMigrate* is generated (Line 6 – 9 of Algorithm 5.1). Next, the destination PMs for VMs of *vmsToMigrate* is strategically selected, which is elucidated in the following section.

• Destination PMs Selection for Migrating VMs from *O-UPMs*

The Destination PM Selection for VM of *O-UPM* (*DPSVO*) algorithm (Algorithm 5.3) is used to determine destination PMs for VMs, which are migrated out of *O-UPMs*. Migrating VMs out of *O-UPMs* and place in new PMs is unavoidable, as it is necessary to cease degradation of QoS. However, it is contrary to consolidated VM placement in as few PMs as possible and hence, it negatively affects the minimization of energy consumption. As such, *DPSVO* aims to ensure least increase of energy consumption on account of placing migrating VMs of *O-UPMs* (i.e., VMs of *vmsToMigrate*) in new PMs. *DPSVO* algorithm calculates the potential rise of energy consumption of a *Non O-UPM* (i.e., A PM, which is not Over-utilized), if the target VM would had been hosted in that PM and compare with that of other PMs. The PM with the least increase of energy consumption is selected as destination PM for that VM (Line 1 – 11 of Algorithm 5.3).

Algorithm 5.3 The Destination PM Selection for VM of O-UPM (*DPSVO*) algorithm

Input: The VM V_j V_j to be migrated out from an *O-UPM*

Input: *NOP*

Output: The new destination PM for the given migrating VM

```
leastIncreasedEnergy ← $maximum\ value$
for each $P_z$ in $NOP$ do
    energyRise ← Energy rise of $P_z$ for hosting $V_j$
    if energyRise < leastIncreasedEnergy then
        if $P_z$ satisfies both (3.3) and (3.5) then
            leastIncreasedEnergy ← energyRise
            hostWithLeastIncreasedEnergy ← $P_z$
        end if
    end if
end for
return $P_z$
```

5.4.1.2 The Second Phase *U-UPMs*

After migrating VMs out of all *O-UPMs*, the *PRTDVMC* algorithm initiates its second phase (Line 19 - 26 of Algorithm 5.1). Aim of this phase is to capitalize consolidation opportunities arisen through change of workload over the period. VMs are attempted to be placed in lesser number of PMs so that overall energy consumption can be lowered. First, a number of PMs are selected from which VMs would be migrated out. After selecting the source PM, strategic selection of VM(s) from that PM followed by smart selection of respective new PMs as destination PMs to host those migrating VMs are carried out. In the following section we have presented the source PM selection strategy.

- **Source PMs Selection From *U-UPMs***

The PM, which is not in sleep state or not in turned off state (i.e., not belong to the set of *SP*) and was not found as *O-UPM* in *the first phase* of *PRTDVMC* algorithm is denoted as *Under-Utilized PM* (*U-UPM*). *candidateSources* refers to the set of potential source *U-UPM*s from which VMs would be migrated out and consolidated in lesser number of PMs. *destinationPMs* refers to the set of PMs, which were selected as destination PMs to host migrating VMs of *O-UPMs*. Since, PMs of *destinationPMs* contain VMs, which had been migrated out from *O-UPMs* in the first phase, PMs of *destinationPMs* are excluded from *candidateSources* to prevent re-migration of VMs (Line 19 of Algorithm 5.1).

PMs of *candidateSources* are sorted in ascending order of *PPMRT* (Line 20 of Algorithm 5.1). The first PM from sorted *candidateSources* (i.e., the PM with the lowest *PPMRT*) is first selected as source *U-UPM* from which VMs are attempted to migrate out with the aim of consolidated VM placement. After a source *U-UPM* is selected, the Destination PMs Selection for VMs of *U-UPMs* (*DPSVU*) algorithm (Algorithm 5.4) is invoked to strategically determine which VMs will be migrated out from that PM and to select the set of respective new destination PMs, namely *destinations*. Since, the selected destination PMs for migrating VMs of source *U-UPM* (i.e., PMs belong to *destinations*) host VMs that had just been migrated out, therefore, to cease VM re-migration, PMs belong to *destinations* are removed from *candidateSources* (Line 25 of Algorithm 5.1).

After removing PMs of *destinations* from *candidateSources,* the next PM (i.e., with the second lowest *PPMRT*) from the updated sorted *candidateSources* is chosen as the next source *U-UPM*. Once again, the *DPSVU* algorithm is re-invoked, followed by subsequent update of sorted *candidateSources* (i.e., PMs of *destinations* are excluded from *candidateSources*) and the next PM from the updated sorted *candidateSources* (i.e, the PM with the third lowest *PPMRT*) is then selected as the source *U-UPM*. Such process is repeated until *candidateSources* has no PM left (Line 22 – 26 of Algorithm 5.1). After selecting source *U-UPMs*, migrating VM(s) from that source *U-UPM* and respective destination PMs are strategically chosen, which we have elucidated in the following section.

- **Migrating VMs and Destination PMs Selection**

The Destination PMs Selection for VMs of *U-UPMs* (*DPSVU*) algorithm (Algorithm 5.4) is used to select migrating VM(s) from a source *U-UPM* and respective destination PMs. *candidateDestinations* represent the set of potential candidates *U-UPMs* from which destination PMs are selected. It consists all *U-UPMs* (Line 21 of Algorithm 5.1) (i.e., PMs, which are neither in sleep or turned-off state nor had been found as *O-UPM* in the first phase) except the source *U-UPM* of the migrating VM. The reason to exclude source *U-UPM* of a migrating VM from *candidateDestinations* is to avoid selecting a source PM of a VM as its destination PM (Line 23 of Algorithm 5.1). In order to be able to host a migrating VM, the destination PM must also not violate RC (3.3) and MUTC (3.5) constraints, which is ensured by invoking The PM Suitability Test (*PST*) algorithm (Line 1 – 4 of Algorithm 5.5). To select which VM to migrate out from the source *U-UPM*, VMs of that source *U-UPM* is sorted in descending order of *PVMRT* (Line 1 of Algorithm 5.4). The first VM from the sorted list of

VMs (i.e., the VM with the largest *PVMRT*) is attempted to migrate out, as a new destination PM is sought from *candidateDestinations* to host that VM.

Algorithm 5.4 The Destination PMs Selection for VMs of U-UPMs (*DPSVU*) algorithm

Input: A *U-UPM*, P_c from the set of *candidateSources*

Input: *candidateDestinations*

Output: List of new destination PMs to host migrating VMs

```
1:  Sort $V^c$, the set of VMs of $P_c$ in order of decreasing PVMRT
2:  for each $V_n^c$ in sorted $V^c$ do
3:      $P_d \leftarrow$ null
4:      Sort candidateDestinations in ascending order of PPMRT
5:      for each $P_m$ in sorted candidateDestinations do
6:          suitable ← Invoke the PST algorithm with $P_m$ and $V_n^c$
7:          if suitable is true
8:              energyDrop ← Energy drop in $P_c$ without $V_n^c$
9:              energyRise ← Energy rise of $P_m$ for hosting $V_n^c$
10:             netEnergyGain ← EnergyDrop – EnergyRise
11:             if NetEnergyGain > 0
12:                 $P_d \leftarrow P_m$
13:                 hostlist ← {$P_d \cup$ hostlist }
14:                 break loop  // Go to Line 18
15:             end if        // End of if from Line 11 to 15
16:         end if            // End of if from Line 7 to 16
17:     end for               // End of for Loop from Line 5 to 17
18:     if $P_d$ is null then
19:         break loop        // Go to Line 22
20:     end if                // End of if from Line 18 to 20
21: end for                   // End of for Loop from Line 2 to 21
22: return hostlist
```

Algorithm 5.5 The PM Suitability Test (*PST*) algorithm

Input: *PM*, *VM*

Output: A decision whether the given *PM* can host the given *VM*

```
if (3.3) and (3.5) are satisfied for all $R_k$ in $R$ then
    return true
end if
return false
```

In order to select the destination PM, PMs of *candidateDestinations* are sorted in increasing order to *PPMRT* and checked if the first PM from the sorted *candidateDestinations* (i.e., the PM with the smallest *PPMRT*) is suitable to host the VM. The *PST* algorithm (Algorithm 5.5) is invoked to test the suitability of a PM for hosting a particular VM. If found suitable, then the net energy gain related to the VM migration, which is the difference between the potential drop of energy of the source PM due to migrating a VM out and potential rise of energy of the destination PM on account of hosting an additional VM is measured. If the net energy gain is positive, then the first PM of the sorted *candidateDestinations* is selected as the new destination PM for that migrating VM. If the net energy gain is negative, then the suitability and respective net energy gain for the next PM from the sorted *candidateDestinations* is checked. Such suitability and net energy gain testing continue until both are found positive from a PM from sorted *candidateDestinations*. The PM for which both suitability and net energy gain are found positive is selected as the destination PM (Line 4 – 17 of Algorithm 5.4).

After selecting the destination PM for the first VM, the same process is followed to find the destination PM for the next VM from the sorted list of VMs. If for any VM, no PM can be found from *candidateDestinations* for which both suitability and net energy gain are positive, then the VM selection and destination PM selection process stops (Line 2 – 22 of Algorithm 5.4). The reason is that migrating out the largest VM in terms of *PVMRT* of a source *U-UPM* shortens the *PMRT* or expedites the moving of the PM into sleep state or turned-off state, which saves energy. If the largest VM in terms of *PVMRT* among all the VMs of a source PM is not migrated out, then then *PMRT* of that PM cannot be reduced. For easier understanding of the readers, we have elucidated the characteristics of *PRTDVMC* algorithm in the following section.

5.5 Characteristics of Proposed Algorithm

With contrast to existing DVMC algorithms, *PRTDVMC* algorithm is developed considering heterogeneous nature of *VMRT* in its bedrock. In order to regulate CDC energy consumption, traditional DVMC algorithms only focus on minimizing the total number of active PMs, whereas *PRTDVMC* algorithm not only minimizes the total number of active PMs, but also expedites the move of an active PM into lower energy consumption state, such as sleep state or turned off state. Largest VMs in terms of *VMRT* are migrated out from the smallest PM in terms of *PPMRT* and placed in the next smallest possible PM in terms of *PPMRT*. Such source PM, migrating VM and destination PM selection technique helps to achieve two objectives – consolidation of VMs in lesser number of PMs and reduction of the active period of the source PM without increasing the active period of the destination PM. Hence, *PRTDVMC* is more energy-efficient compare to traditional DVMC algorithms.

Previously in Chapter 4, we have highlighted that lack of efficiency occurs if DVMC algorithm is not developed by taking changed energy-efficiency characteristics of state-of-the-art highly energy proportional PMs. With highly energy proportional state-of-the-art PMs, energy consumption rises drastically beyond 70% load level and energy-efficiency consequently drops. Prior any VM migration for the purpose of consolidation, *PRTDVMC* checks whether the rise of energy in the destination PM exceeds the drop of energy of the source PM or not. If the rise of energy in the destination PM does not outweigh the energy drop in destination PM, only then VM is migrated. Hence, *PRTDVMC* is robust against underlying PMs' energy-efficiency characteristics.

Apart from the difference in obtaining *VMRT* and *PMRT*, there is a stark contrast in terms of destination PM selection technique for migrating VMs of *O-UPMs* between *PRTDVMC* and our previously proposed algorithms (i.e., *RTDVMC* and *SRTDVMC*). Previously, we mentioned that placing migrating VMs of *O-UPMs* into *U-UPMs* is contrary to VM consolidation and increases energy consumption. Both *RTDVMC* and *SRTDVMC*, attempts to place the largest VM of *O-UPM* in terms of *VMRT* into the smallest possible *U-UPM* in terms of *PMRT*. However, *VMRT* of the largest VM of *O-UPM* can be greater than the smallest *PMRT* of *U-UPM*, or in other words, the largest *VMRT* of the source *O-UPM* can be greater than the largest *VMRT* of the destination *U-UPM*. Hence, least increase of energy can be guaranteed with such destination PM selection technique. To address this issue, in order to select the destination PM for a migrating VM of *O-UPM*, *PRTDVMC* checks the potential increase of energy consumption for all *U-UPMs* and selects the *U-UPM* with least increase of energy consumption as destination PM. As a result, *PRTDVMC* is designed as more energy-efficient compared to *RTDVMC* and *SRTDVMC*.

At the outset of this chapter, we have highlighted that performance of a DVMC algorithm varies with the change in SVMC algorithm, it is coupled with. Like DVMC, SVMC is a NP-hard problem and hence, heuristics and meta-heuristics are only possible approaches to address the problem. In the following section, we have presented different heuristics based SVMC approaches.

5.6 SVMC algorithms

Initial VM Placement, also called as SVMC plays a crucial role in energy-efficient placement of VMs compelling reduced energy consumption. Performance of *PRTDVMC* has hen been tested with four diverse and well-known heuristics based SVMC algorithms as elucidated in the following.

5.6.1 First Fit (FF)

A VM is always attempted to be assigned in the first PM. If not possible, then the next PM is chosen to host the VM. For instance, the first VM would be placed in the first PM. The second VM would also be attempted to be placed in the first PM. Every single VM would be first attempted to be placed in the first PM. If the first PM is unable to accommodate a VM because of the resource restriction, then the second PM would be checked. Assume that the first and second VM are hosted in the first PM, while the third PM is hosted in the second PM. For the fourth VM, the first PM would be first check whether it can host the fourth VM.

5.6.2 Next Fit (NF)

A VM is attempted to be placed in the PM, which is the immediate next to that PM hosted the last VM. To explain more, the first VM would be attempted to be placed in the first PM. If the first VM can be placed in the first PM, then the second VM would be attempted to be hosted in the second PM. Then the next VM would be attempted to be hosted in the third PM and so on. If a PM, for instance the second PM is unable to accommodate a VM, then the next PM (i.e., the third PM) is checked if it has adequate resource to host the VM. The immediate next PM to that last PM (i.e., the fourth PM) becomes the candidate for the next VM.

5.6.3 Best Fit (BF)

The Best FIT (BF) can be defined in different ways. We have defined BF that it would choose the PM to be experienced the least increase of energy consumption on account of hosting a target VM. This is a greedy heuristic approach. To explain more, let a VM is needed to be initially hosted in any of the three PMs. If the VM is hosted, then the first PM's energy consumption would rise by 5kW, while the second PM's energy consumption would grow by 3kW and the third PM's energy consumption would arise by 4kW. If the second PM has adequate resources to host the VM, then BF would choose the second PM. If the second PM cannot host it, then the third PM would be checked.

5.6.4 Random Selection (RS)

As opposed to previous greedy heuristics, RS randomly choses a PM for a VM. For instance, suppose there are ten PMs. Probability of any host being chosen by RC is 0.1. In other words, every host is equiprobable to be chosen as destination host for the initial VM placement. RC is proven to be effective in avoiding local minima, which may help in obtaining global maxima.

In the following section, we have articulated the empirical evaluation of *PRTDVMC* and compared with an existing notable DVMC algorithm under diverse heuristics-based approaches.

5.7 Performance Evaluation

CloudSim has been used to simulate *PRTDVMC*. Working principle of both *RTDVMC* and *SRTDVMC* is that the *VMRT* is known in advance, or in other words, the future information, which is assumed to be known by the users is received in prior and used as input parameter. In contrast, the approach of *PRTDVMC* is that predicted *VMRT* is first generated from past usage records, which is used as input, since *PRTDVMC* does not receive any future information from users. If future is known in advance, then the performance is beyond compare. It would not be truly an ideal comparison between two algorithms, if one knows the future in advance and uses it, while the other does not know the future. As such, we have avoided comparing *PRTDVMC* with *RTDVMC* and *SRTDVMC*.

We explained earlier that Threshold based DVMC algorithms are of two types- Static Threshold based DVMC (STDVMC) and Adaptive Threshold based DVMC (ATDVMC). ATDVMC exalts number of VM migration compare to STDVMC and is less energy-efficient than STDVMC. *PRTDVMC* is hence designed as a STDVMC algorithm. Based on our extensive literature review on DVMC algorithms, we have found THR-RS [51], THR-MC [51], THR-MMT [51] are pioneers and one of the most popular STDVMC algorithms. In comparison with THR-RS [51] and THR-MC [51], THR-MMT [51] is found as more energy-efficient [51]. Hence, we have compared *PRTDVMC* with THR-MMT. As per objective functions (4.10) and

(4.11), We have measured performance of *PRTDVMC* and THR-MMT CDC in the context of energy consumption and VM migration.

5.7.1 Experimental Setup

We have modelled and simulated a cloud environment in CloudSim [27] to simulate *PRTDVMC* algorithm in combination with diverse SVMC algorithms under different workload scenarios. Equity is given for both *PRTDVMC* and THR-MMT algorithms in terms of the simulated cloud environment comprised of CDC, VM, PM, energy module and workload. The CDC is built with 800 heterogeneous PMs. Three different modern generation of highly energy proportional PMs have been used. The characteristics of those PMs have been articulated in Table 5.1. The energy consumption characteristics of these PMs with varying workload is articulated in Table 4.1.

Table 5.1 Characteristics of the PMs Used in the Simulation

Name of the PM	Name of CPU	Number of CPU Cores	CPU Clock Frequency	RAM Size	Network Bandwidth
Dell PowerEdgeR940 [99]	Intel Xeon Platinum 8180	112 cores	25000 MHz	384 GB	1 GB/s
HP ProLiant DL560 Gen10 [100]	Intel Xeon Platinum 8180	112 cores	25000 MHz	384 GB	1 GB/s
HP ProLiant ML350 Gen10 [101]	Intel Xeon Platinum 8180	28 cores	25000 MHz	192 GB	1 GB/s

The characteristics of different VM types match with the VMs used by the authors of THR-MMT and correspond to Amazon EC2 instance types [105]. However, the difference between the simulated VMs and Amazon EC2 instance types is that the simulated VMs are single-core, which is explained by the fact that the workload data used for the simulations come from single-core VMs. Since, the single-core is used, the amount of RAM is divided by the number of cores for each VM type. In Table 5.2, we have presented the diverse VMs used in the simulation.

Table 5.2 Characteristics of the VMs Used in the Simulation

Name of the VM	Number of Core	Size of CPU	Size of RAM
High-CPU Medium Instance	Single	2500 MIPS	0.85 GB
Extra Large Instance	Single	2000 MIPS	3.75 GB
Small Instance	Single	1000 MIPS	1.7 GB
Micro Instance	Single	500 MIPS	613 MB

PVMRT, $\hat{T}_{V_j}$ can be different from one VM to another (i.e., heterogeneous). *VMRT* is crucial, as it influences the overall performance of a DVMC algorithm. Therefore, for better estimation of performance of *PRTDVMC* in real Cloud scenario, $\hat{T}_{V_j}$values have been drawn from *VMRT* traces of real Cloud, namely Nectar Cloud hosting over thousands of VMs spread across different CDC located in eight different cities of Australia [28]. A user's past *VMRT* records are grouped together and is used as input to the RLR based *VMRT* predictor model to obtain *PVMRT*. To regulate prediction error, we have incorporated safety margin, articulated previously in Section 5.1.3. At the outset, VMs are provided with the resources defined by the VM types. However, during the lifetime, VMs utilize less resources according to the workload data, yielding spaces for dynamic consolidation. The workload data also reflects traces of real Cloud workload traffic, originated through CoMon project, a monitoring infrastructure for PlanetLab [29]. Table 5.3 represents the simulation parameters.

Table 5.3 Simulation Parameters

Name of the parameter	**Value**
Number of PMs in the CDC	800
θ_{max} for both *PRTDVMC* and THR-MMT	0.8
Number of run for every DVMC algorithm with each single day of PlanetLab workload data	2
Simulation run time for every run	1 hour CloudSim simulation clock time

5.7.2 Workload Data

Using workload traces of a real system in experiments is extremely important to make a simulation-based evaluation applicable for real world [45]. Therefore, we have utilized real could based workload traces, such as PlanetLab and Nectar. Real cloud based PlanetLab VMs of different days represent different types of workload, as the utilization distribution vary from one day to another. The number of VMs also vary from one day to another. For empirical evaluation, randomly 10 different days of PlanetLab workload as presented in Table 5.4 has been selected to compare the performance of *PRTDVMC* and THR-MMT in conjunction with diverse SVMC algorithm.

Table 5.4 Characteristics of PlanetLab Data (CPU Utilization)

Day	Number of VMs	Mean (%)	St. dev. (%)	Quartile 1	Quartile 2	Quartile 3
3 March	1052	12.31	17.09	2%	6%	15%
6 March	898	11.44	16.83	2%	5%	13%
9 March	1061	10.70	15.57	2%	4%	13%
22 March	1516	9.26	12.78	2%	5%	12%
25 March	1078	10.56	14.14	2%	6%	14%
3 April	1463	12.39	16.55	2%	6%	17%
9 April	1358	11.12	15.09	2%	6%	15%
11 April	1233	11.56	15.07	2%	6%	16%
12 April	1054	11.54	15.15	2%	6%	16%
20 April	1033	10.43	15.21	2%	4%	12%

In order to vary *VMRT* of these VMs, each VM is assigned with a randomly chosen *PVMRT* generated from different users' past *VMRT* records who previously created VMs in real Nectar Cloud [28]. A uniformly distributed random variable has been used to randomly select *PVMRT*. To explain further, records of multiple past *VMRT* having same user id are grouped together and used to generate *PVMRT* of that particular user. Earlier in Section 5.1.2, we have elaborately discussed on obtaining *PVMRT* utilising past *VMRT* records of a user through RLR model. To regulate prediction error, safety margin has been invested in *PVMRT* generation process, as elucidated in Section 5.1.3. Different *PVMRT* for different users have hence been generated, which is distributed across different PlanetLab VMs. Thus, a truly heterogeneous workload data featuring heterogeneous utilization and heterogeneous *VMRT* has been developed for performance testing. In Fig. 5.1, the histogram of logarithm of *PVMRT* of Nectar VMs is delineated.

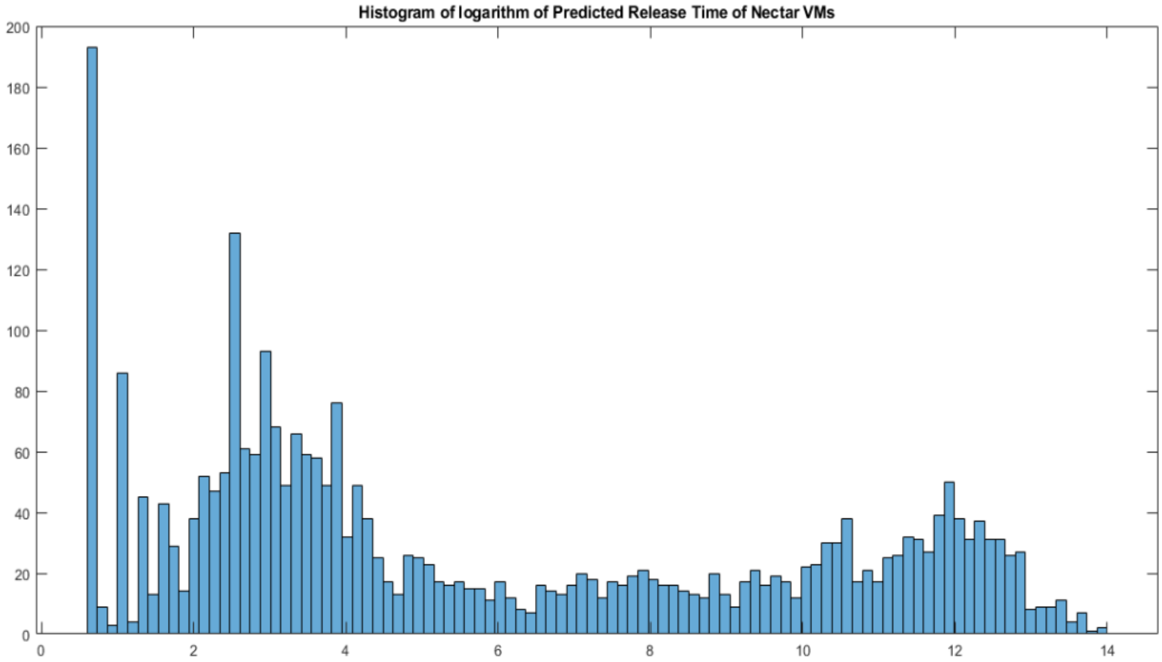

Fig. 5.1 Histogram of Logarithm of Predicted Release Time (in Second) of Nectar VMs

In the following subsection, we have brought forth the outcome of experiments and result analysis.

5.7.3 Simulation Results and Analysis

We have simulated the performance of *PRTDVMC* and THR-MMT using CloudSim in terms of mean CDC energy consumption and mean total number of VM migration under four diverse SVMC approaches – FF, BF, NF and RS. As part of experiments, real Cloud based PlanetLab workload of two different months – March and April and randomly chosen five different days of each of those months - 3 March, 6 March, 9 March, 22 March, 25 March, 3 April, 9 April, 11 April, 12 April and 20 April featuring different number of VMs and diverse distributions of time variant CPU utilization have been used. In Table 5.4, we have articulated the characteristics of different days of PlanetLab workload. For more accurate evaluation of system performance under real Cloud workload scenario, which is heterogeneous in terms of VMRT, a uniformly distributed random variable has been used to draw finishing time of different VMs (i.e., VMRT of different VMs) representing a day's PlanetLab workload from the distribution of *PVMRT* of thousands of VMs resided in real Cloud, namely Nectar Cloud. In Fig. 5.1, we have delineated the *PVMRT* distribution of thousands of VMs from Nectar Cloud. In the following section, the performance comparison of *PVMRT* and THR-MMT in terms of CDC energy consumption has been presented.

5.7.3.1 Energy Consumption

In Table 5.5 and Table 5.6, we have presented members of set of mean CDC energy consumption for *PRTDVMC* under different SVMC algorithm, $\bar{E}^{P}_{SV} = \{\bar{E}^{P}_{SV,PL}\}_{|PlanetLab|}$ and those for THR-MMT, $\bar{E}^{TMMT}_{SV} = \{\bar{E}^{TMMT}_{SV,PL}\}_{|PlanetLab|}$ for different days of PlanetLab workload with workload finishing time drawn from *PVMRT* distribution of Nectar Cloud.

Table 5.5 Mean CDC Energy Consumption by *PRTDVMC* under Diverse SVMC Algorithms for Different Days of PlanetLab Workload

CDC Energy Consumption (kW) by *PRTDVMC*				
	$\bar{E}^{P}_{FF}$	$\bar{E}^{P}_{BF}$	$\bar{E}^{P}_{NF}$	$\bar{E}^{P}_{RS}$
3 March	4.85	4.925	4.875	4.765
6 March	4.21	4.335	4.205	4.105
9 March	4.735	4.74	4.7	5.03
22 March	6.635	7.105	6.475	6.725
25 March	4.735	5.005	4.89	5.05
3 April	6.75	6.58	6.63	6.62
9 April	6.21	6.175	6.27	6.12
11 April	5.62	5.61	5.44	6.095
12 April	4.88	4.715	4.88	4.845
20 April	4.65	4.745	4.75	4.85

Table 5.6 Mean CDC Energy Consumption by THR-MMT under Diverse SVMC Algorithms for Different Days of PlanetLab Workload

CDC Energy Consumption (kW) by THR-MMT				
	$\bar{E}^{TMMT}_{FF}$	$\bar{E}^{TMMT}_{BF}$	$\bar{E}^{TMMT}_{NF}$	$\bar{E}^{TMMT}_{RS}$
3 March	6.045	5.77	5.68	5.425
6 March	4.735	4.995	4.785	4.825
9 March	5.555	5.06	5.715	6.075
22 March	6.425	7.075	6.46	6.325
25 March	5.16	5.715	5.69	5.9
3 April	7.115	6.855	6.985	6.48
9 April	6.625	6.965	6.96	6.9
11 April	6.27	6.655	6.17	7.085
12 April	5.88	5.16	5.475	5.49
20 April	5.23	5.475	5.01	5.295

Let, SV denotes the index to denote different element of the set $SVMC = \{SVMC_{SV}\}_{|4|} = \{\mathrm{FF}, \mathrm{BF}, \mathrm{NF}, \mathrm{RS}\}$ and PL denotes the index referring different element of the set

$PlanetLab = \{PlanetLab_{PL}\}_{|10|} = \left\{\begin{matrix} \text{3 March, 6 March, 9 March,} \\ \text{22 March, 25 March, 3 April,} \\ \text{9 April, 11 April, 12 April,} \\ \text{20 April} \end{matrix}\right\}$. Let, $X_{SV}^{\bar{E}} = \{X_{SV,PL}^{\bar{E}}\}_{|PlanetLab|}$ (5.17) denotes the set presenting difference between mean energy consumption by THR-MMT and mean energy consumption by *PRTDVMC* under a particular SVMC algorithm for different days of PlanetLab workload. In other words, $X_{SV}^{\bar{E}}$ represents the minimization of mean energy consumption proffered by *PRTDVMC* compare to THR-MMT for diverse PlanetLab workloads under a particular SVMC algorithm.

$$X_{SV}^{\bar{E}} = \{X_{SV,PL}^{\bar{E}}\}_{|PlanetLab|} = \{\bar{E}_{SV,PL}^{TMMT} - \bar{E}_{SV,PL}^{P}\}_{|PlanetLab|} \tag{5.17}$$

The equations to obtain $X_{FF}^{\bar{E}}$, $X_{BF}^{\bar{E}}$, $X_{NF}^{\bar{E}}$ and $X_{RS}^{\bar{E}}$ are presented in (5.18), (5.19), (5.20) and (5.21), respectively.

$$X_{FF}^{\bar{E}} = \{X_{FF,PL}^{\bar{E}}\}_{|PlanetLab|} = \{\bar{E}_{FF,PL}^{TMMT} - \bar{E}_{FF,PL}^{P}\}_{|PlanetLab|} \tag{5.18}$$

$$X_{BF}^{\bar{E}} = \{X_{BF,PL}^{\bar{E}}\}_{|PlanetLab|} = \{\bar{E}_{BF,PL}^{TMMT} - \bar{E}_{BF,PL}^{P}\}_{|PlanetLab|} \tag{5.19}$$

$$X_{NF}^{\bar{E}} = \{X_{NF,PL}^{\bar{E}}\}_{|PlanetLab|} = \{\bar{E}_{NF,PL}^{TMMT} - \bar{E}_{NF,PL}^{P}\}_{|PlanetLab|} \tag{5.20}$$

$$X_{RS}^{\bar{E}} = \{X_{RS,PL}^{\bar{E}}\}_{|PlanetLab|} = \{\bar{E}_{RS,PL}^{TMMT} - \bar{E}_{RS,PL}^{P}\}_{|PlanetLab|} \tag{5.21}$$

In Table 5.7, we have presented the minimization of mean energy consumption proffered by *PRTDVMC* compare to THR-MMT for diverse PlanetLab workloads under different SVMC algorithms.

Table 5.7 Minimization of Mean Energy Consumption Proffered by *PRTDVMC* compared to THR-MMT under Different SVMC Algorithms with Different Days of PlanetLab Workload

Minimization of mean CDC Energy Consumption (kW) by *PRTDVMC*				
	$X_{FF}^{\bar{E}}$	$X_{BF}^{\bar{E}}$	$X_{NF}^{\bar{E}}$	$X_{RS}^{\bar{E}}$
3 March	1.195	0.845	0.805	0.66
6 March	0.525	0.66	0.58	0.72
9 March	0.82	0.32	1.015	1.045
22 March	-0.21	-0.03	-0.015	-0.4
25 March	0.425	0.71	0.8	0.85
3 April	0.365	0.275	0.355	-0.14
9 April	0.415	0.79	0.69	0.78
11 April	0.65	1.045	0.73	0.99
12 April	1	0.445	0.595	0.645
20 April	0.58	0.73	0.26	0.445

Fig. 5.2 highlights the performance comparison between *PRTDVMC* and THR-MMT in terms of CDC energy consumption in conjunction with four different SVMC algorithms for different days of PlanetLab data of March 2011, while Fig. 5.3, shows the performance comparison for different days of April, 2011. Based on our experimental results, we have undertaken diverse critical statistical tests as presented in the following section.

- **Normality Testing**

From Fig. 5.2 and Fig. 5.3, we can view that *PRTDVMC* outperforms THR-MMT in terms mean CDC energy consumption. One can argue that such improvement obtained by *PRTDVMC* compared to THR-MMT has appeared by chance. Hence, for rebuttal, it needs to be checked if the improvement exhibited by *PRTDVMC* is statistically significant or not. Statistical significance tests can be either parametric or non-parametric. Parametric tests are more powerful than non-parametric tests. Before Parametric test normality testing is required. The ability to accurately determine if a data sample is originated from a non-normal distribution, referred to as power, is referred to as the strength of a normality test [112]. Chi-square test for normality is not as powerful and unsuitable for small data samples. Power of K-S test is low [113] and is sensitive to extreme values, which is regulated by Lilliefors correction [114]. The S-W normality test is regarded as more powerful than the K-S test even after the Lilliefors correction [115] and recommend as the best option for testing the normality of data [112].

Test statistics for the S-W normality test with $\bar{E}_{SV}^{TMMT}$ and $\bar{E}_{SV}^{P}$, referred to as $SW_{\bar{E}}^{TMMT,SV}$ and $SW_{\bar{E}}^{P,SV}$ respectively can be calculated from (5.22) and (5.23) [116, 117].

$$SW_{\bar{E}}^{TMMT,SV} = \left(\left(\sum_{w=1}^{|w|} a_w \cdot \left(\bar{E}_{(|w|+1-w)}^{TMMT,SV} - \bar{E}_{(w)}^{TMMT,SV}\right)\right)^2\right) / \left(\sum_{w=1}^{|w|}\left(\bar{E}_{(w)}^{TMMT,SV} - \bar{\bar{E}}_{SV}^{TMMT}\right)^2\right) \tag{5.22}$$

$$SW_{\bar{E}}^{P,SV} = \left(\left(\sum_{w=1}^{|w|} a_w \cdot \left(\bar{E}_{(|w|+1-w)}^{P,SV} - \bar{E}_{(w)}^{P,SV}\right)\right)^2\right) / \left(\sum_{w=1}^{|w|}\left(\bar{E}_{(w)}^{P,SV} - \bar{\bar{E}}_{SV}^{P}\right)^2\right) \tag{5.23}$$

$$\bar{\bar{E}}_{SV}^{TMMT} = \left(\sum_{w=1}^{|w|} \bar{E}_{(w)}^{TMMT,SV}\right) / (|w|) \tag{5.24}$$

$$\bar{\bar{E}}_{SV}^{P} = \left(\sum_{w=1}^{|w|} \bar{E}_{(w)}^{P,SV}\right) / (|w|) \tag{5.25}$$

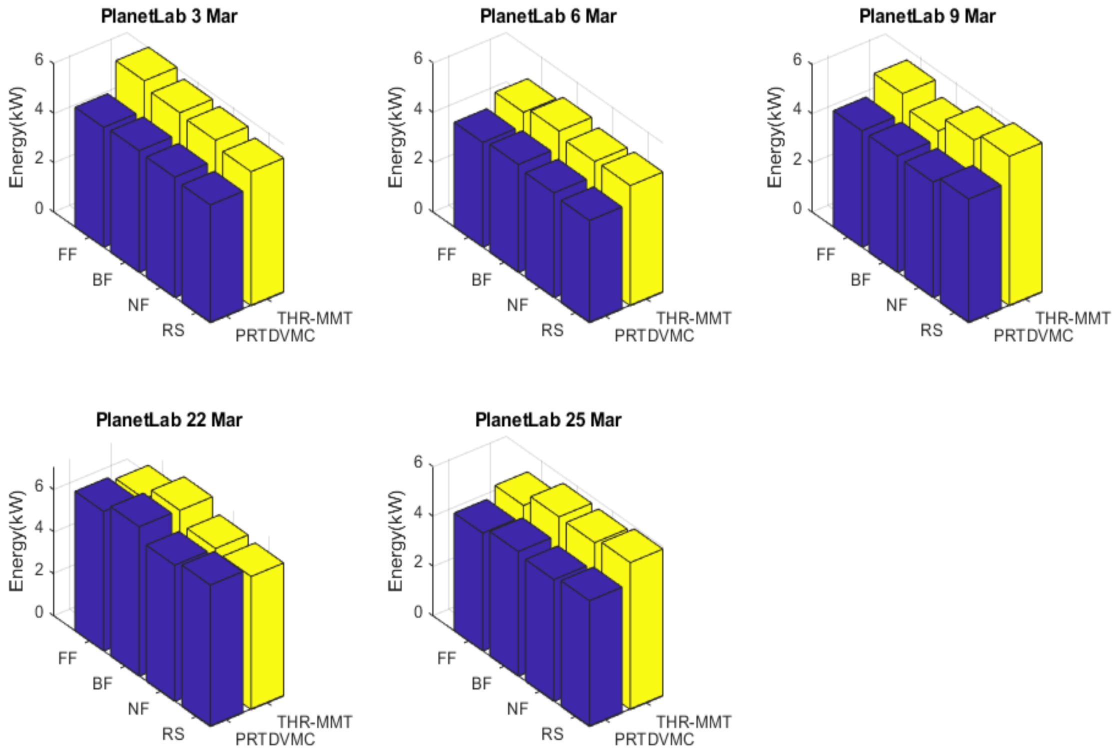

Fig. 5.2 Comparison of Mean CDC Energy Consumption between *PRTDVMC* and THR-MMT with PlanetLab Workload of March under Diverse SVMC Algorithms

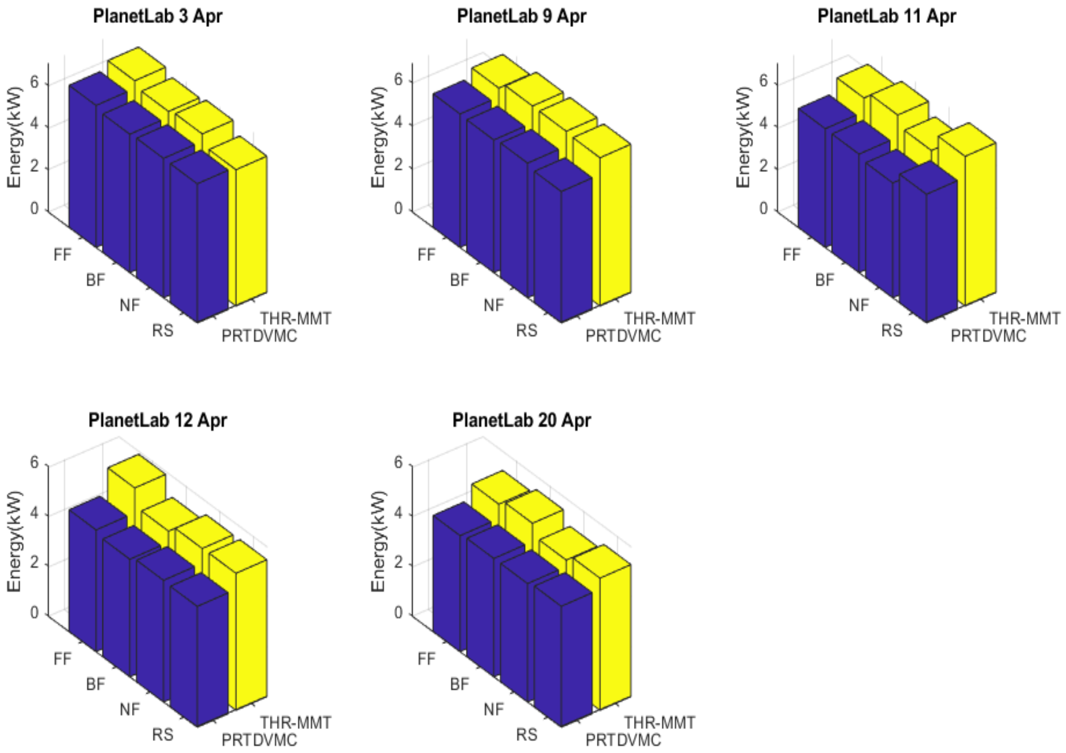

Fig. 5.3 Comparison of Mean CDC Energy Consumption between *PRTDVMC* and THR-MMT with PlanetLab Workload of April under Diverse SVMC Algorithms

Similarly, test statistics for the S-W normality test with $\bar{E}_{FF}^{TMMT}$, $\bar{E}_{FF}^{P}$, $\bar{E}_{BF}^{TMMT}$, $\bar{E}_{BF}^{P}$, $\bar{E}_{NF}^{TMMT}$, $\bar{E}_{NF}^{P}$, $\bar{E}_{RS}^{TMMT}$ and $\bar{E}_{RS}^{P}$, denoted by $SW_{\bar{E}}^{TMMT,FF}$, $SW_{\bar{E}}^{P,FF}$, $SW_{\bar{E}}^{TMMT,BF}$, $SW_{\bar{E}}^{P,BF}$, $SW_{\bar{E}}^{TMMT,NF}$, $SW_{\bar{E}}^{P,NF}$, $SW_{\bar{E}}^{TMMT,RS}$ and $SW_{\bar{E}}^{P,RS}$ can be calculated through (5.26), (5.27), (5.30), (5.31), (5.34), (5.35), (5.38) and (5.39). a_w weights are available in Shapiro-Wilk Table [118]. Different $\bar{E}_{(w)}^{TMMT,SV}$ and $\bar{E}_{(w)}^{P,SV}$ values are presented in Table 5.8 and Table 5.9, respectively.

<u>FF</u>

$$SW_{\bar{E}}^{TMMT,FF} = \left(\left(\sum_{w=1}^{|w|} a_w \cdot \left(\bar{E}_{(|w|+1-w)}^{TMMT,FF} - \bar{E}_{(w)}^{TMMT,FF}\right)\right)^2\right) \Big/ \left(\sum_{w=1}^{|w|}\left(\bar{E}_{(w)}^{TMMT,FF} - \bar{\bar{E}}_{FF}^{TMMT}\right)^2\right) \tag{5.26}$$

$$SW_{\bar{E}}^{P,FF} = \left(\left(\sum_{w=1}^{|w|} a_w \cdot \left(\bar{E}_{(|w|+1-w)}^{P,FF} - \bar{E}_{(w)}^{P,FF}\right)\right)^2\right) \Big/ \left(\sum_{w=1}^{|w|}\left(\bar{E}_{(w)}^{P,FF} - \bar{\bar{E}}_{FF}^{P}\right)^2\right) \tag{5.27}$$

$$\bar{\bar{E}}_{FF}^{TMMT} = \left(\sum_{w=1}^{|w|} \bar{E}_{(w)}^{TMMT,FF}\right) / (|w|) \tag{5.28}$$

$$\bar{\bar{E}}_{FF}^{P} = \left(\sum_{w=1}^{|w|} \bar{E}_{(w)}^{P,FF}\right) / (|w|) \tag{5.29}$$

<u>BF</u>

$$SW_{\bar{E}}^{TMMT,BF} = \left(\left(\sum_{w=1}^{|w|} a_w \cdot \left(\bar{E}_{(|w|+1-w)}^{TMMT,BF} - \bar{E}_{(w)}^{TMMT,BF}\right)\right)^2\right) \Big/ \left(\sum_{w=1}^{|w|}\left(\bar{E}_{(w)}^{TMMT,BF} - \bar{\bar{E}}_{BF}^{TMMT}\right)^2\right) \tag{5.30}$$

$$SW_{\bar{E}}^{P,BF} = \left(\left(\sum_{w=1}^{|w|} a_w \cdot \left(\bar{E}_{(|w|+1-w)}^{P,BF} - \bar{E}_{(w)}^{P,BF}\right)\right)^2\right) \Big/ \left(\sum_{w=1}^{|w|}\left(\bar{E}_{(w)}^{P,BF} - \bar{\bar{E}}_{BF}^{P}\right)^2\right) \tag{5.31}$$

$$\bar{\bar{E}}_{BF}^{TMMT} = \left(\sum_{w=1}^{|w|} \bar{E}_{(w)}^{TMMT,BF}\right) / (|w|) \tag{5.32}$$

$$\bar{\bar{E}}_{BF}^{P} = \left(\sum_{w=1}^{|w|} \bar{E}_{(w)}^{P,BF}\right) / (|w|) \tag{5.33}$$

<u>NF</u>

$$SW_{\bar{E}}^{TMMT,NF} = \left(\left(\sum_{w=1}^{|w|} a_w \cdot \left(\bar{E}_{(|w|+1-w)}^{TMMT,NF} - \bar{E}_{(w)}^{TMMT,NF}\right)\right)^2\right) \Big/ \left(\sum_{w=1}^{|w|}\left(\bar{E}_{(w)}^{TMMT,NF} - \bar{\bar{E}}_{NF}^{TMMT}\right)^2\right) \tag{5.34}$$

$$SW_{\bar{E}}^{P,NF} = \left(\left(\sum_{w=1}^{|w|} a_w \cdot \left(\bar{E}_{(|w|+1-w)}^{P,NF} - \bar{E}_{(w)}^{P,NF}\right)\right)^2\right) \Big/ \left(\sum_{w=1}^{|w|}\left(\bar{E}_{(w)}^{P,NF} - \bar{\bar{E}}_{NF}^{P}\right)^2\right) \tag{5.35}$$

$$\bar{\bar{E}}_{NF}^{TMMT} = \left(\sum_{w=1}^{|w|} \bar{E}_{(w)}^{TMMT,NF}\right) / (|w|) \tag{5.36}$$

$$\bar{\bar{E}}_{NF}^{P} = \left(\sum_{w=1}^{|w|} \bar{E}_{(w)}^{P,NF}\right) / (|w|) \tag{5.37}$$

<u>RS</u>

$$SW_{\bar{E}}^{TMMT,RS} = \left(\left(\sum_{w=1}^{|w|} a_w \cdot \left(\bar{E}_{(|w|+1-w)}^{TMMT,RS} - \bar{E}_{(w)}^{TMMT,RS}\right)\right)^2\right) / \left(\sum_{w=1}^{|w|}\left(\bar{E}_{(w)}^{TMMT,RS} - \bar{\bar{E}}_{RS}^{TMMT}\right)^2\right) \tag{5.38}$$

$$SW_{\bar{E}}^{P,RS} = \left(\left(\sum_{w=1}^{|w|} a_w \cdot \left(\bar{E}_{(|w|+1-w)}^{P,RS} - \bar{E}_{(w)}^{P,RS}\right)\right)^2\right) / \left(\sum_{w=1}^{|w|}\left(\bar{E}_{(w)}^{P,RS} - \bar{\bar{E}}_{RS}^{P}\right)^2\right) \tag{5.39}$$

$$\bar{\bar{E}}_{RS}^{TMMT} = \left(\sum_{w=1}^{|w|} \bar{E}_{(w)}^{TMMT,RS}\right) / (|w|) \tag{5.40}$$

$$\bar{\bar{E}}_{RS}^{P} = \left(\sum_{w=1}^{|w|} \bar{E}_{(w)}^{P,RS}\right) / (|w|) \tag{5.41}$$

Table 5.8 Sorted Mean CDC Energy Consumption (kWh) by THR-MMT under diverse SVMC Algorithms for Different Days of PlanetLab Workload

Sorted mean CDC Energy Consumption (kWh) by THR-MMT				
w	$\bar{E}_{(w)}^{TMMT,FF}$	$\bar{E}_{(w)}^{TMMT,BF}$	$\bar{E}_{(w)}^{TMMT,NF}$	$\bar{E}_{(w)}^{TMMT,RS}$
1	4.74	5	4.79	4.83
2	5.16	5.06	5.01	5.3
3	5.23	5.16	5.48	5.43
4	5.56	5.48	5.68	5.49
5	5.88	5.72	5.69	5.9
6	6.05	5.77	5.72	6.08
7	6.27	6.66	6.17	6.33
8	6.43	6.86	6.46	6.48
9	6.63	6.97	6.96	6.9
10	7.12	7.08	6.99	7.09

Table 5.9 Sorted Mean CDC Energy Consumption (kWh) by THR-MMT under diverse SVMC Algorithms for Different Days of PlanetLab Workload

Sorted mean CDC Energy Consumption (kWh) by *PRTDVMC*				
w	$\bar{E}_{(w)}^{P,FF}$	$\bar{E}_{(w)}^{P,BF}$	$\bar{E}_{(w)}^{P,NF}$	$\bar{E}_{(w)}^{P,RS}$
1	4.21	4.34	4.21	4.11
2	4.65	4.72	4.7	4.77
3	4.74	4.74	4.75	4.85
4	4.74	4.75	4.88	4.85
5	4.85	4.93	4.88	5.03
6	4.88	5.01	4.89	5.05
7	5.62	5.61	5.44	6.1
8	6.21	6.18	6.27	6.12
9	6.64	6.58	6.48	6.62
10	6.75	7.11	6.63	6.73

The S-W normality test software has been collected from [106]. The Null Hypothesis for the S-W normality tests with data samples of $\bar{E}_{FF}^{TMMT}$, $\bar{E}_{FF}^{P}$, $\bar{E}_{BF}^{TMMT}$, $\bar{E}_{BF}^{P}$, $\bar{E}_{NF}^{TMMT}$, $\bar{E}_{NF}^{P}$, $\bar{E}_{RS}^{TMMT}$

and $\bar{E}_{RS}^{P}$, is that data is normally distributed. For distribution of $\bar{E}_{FF}^{TMMT}$, $\bar{E}_{FF}^{P}$, $\bar{E}_{BF}^{TMMT}$, $\bar{E}_{BF}^{P}$, $\bar{E}_{NF}^{TMMT}$, $\bar{E}_{NF}^{P}$, $\bar{E}_{RS}^{TMMT}$ and $\bar{E}_{RS}^{P}$, corresponding p values are found as 0.98, 0.09, 0.1, 0.17, 0.55, 0.09, 0.9 and 0.26 respectively, which are not smaller than critical value, α as 0.05. Hence, no strong evidence could be found to reject Null Hypothesis that elements of $\bar{E}_{FF}^{TMMT}$, $\bar{E}_{FF}^{P}$, $\bar{E}_{BF}^{TMMT}$, $\bar{E}_{BF}^{P}$, $\bar{E}_{NF}^{TMMT}$, $\bar{E}_{NF}^{P}$, $\bar{E}_{RS}^{TMMT}$ and $\bar{E}_{RS}^{P}$ have come from normal distribution. Since, the normality test highlights that elements of $\bar{E}_{FF}^{TMMT}$, $\bar{E}_{FF}^{P}$, $\bar{E}_{BF}^{TMMT}$, $\bar{E}_{BF}^{P}$, $\bar{E}_{NF}^{TMMT}$, $\bar{E}_{NF}^{P}$, $\bar{E}_{RS}^{TMMT}$ and $\bar{E}_{RS}^{P}$ are normally distributed, therefore condition to perform parametric hypothesis test is satisfied. In the following section, we have presented parametric hypothesis testing.

- **Parametric Hypothesis Testing, Test Error and Confidence Interval**

In the previous section, through the S-W normality test we have shown that elements of $\bar{E}_{FF}^{TMMT}$, $\bar{E}_{FF}^{P}$, $\bar{E}_{BF}^{TMMT}$, $\bar{E}_{BF}^{P}$, $\bar{E}_{NF}^{TMMT}$, $\bar{E}_{NF}^{P}$, $\bar{E}_{RS}^{TMMT}$ and $\bar{E}_{RS}^{P}$ follow normal distribution, which meets the prior condition of parametric tests. Elements of $X_{FF}^{\bar{E}}$, $X_{BF}^{\bar{E}}$, $X_{NF}^{\bar{E}}$ and $X_{RS}^{\bar{E}}$ articulated in Table 5.7 refer to the fact that energy consumption by *PRTDVMC* is numerically lower compare to THR-MMT for different workload scenarios as presented in experiments. One might still argue that such improvement is a random incident and has happened by chance. To refute such argument, parametric hypothesis test needs to be undertaken to prove the improvement of mean CDC energy consumption by *PRTDVMC* compared to THR-MMT is not a random incident, as the improvement is rather statistically significant. Our Sample size is 10 (i.e., $|PlanetLab|$), which is less than 30 and means of no more than two DVMC algorithms (i.e., THR-MMT and *PRTDVMC*) would be compared. Hence, among different parametric tests the two tail *t-test* is chosen, instead of Z-test, F-test and ANOVA. Based on the data samples, the *t-test* can be grouped into three categories: One sample, Two Independent Samples and Paired Samples *t-test*. For a specific combination of SV and PL, corresponding $\bar{E}_{SV,PL}^{TMMT}$ and $\bar{E}_{SV,PL}^{P}$ has a relationship, as $\bar{E}_{SV,PL}^{TMMT}$ and $\bar{E}_{SV,PL}^{P}$ refers to $\bar{E}_{CDC}$ for THR-MMT and *PRTDVMC* respectively, under a specific conjunction of SVMC algorithm and PlanetLab workload scenario. Therefore, the paired two tail *t-test* is performed.

Let, $\bar{X}_{SV}^{\bar{E}}$ and $\hat{\sigma}_{X_{SV}^{\bar{E}}}$ denote the mean and the standard deviation of sampling distribution of sample mean, $X_{SV}^{\bar{E}}$. The equation to calculate $X_{SV}^{\bar{E}}$ is presented in (5.17). Hence, through utilising (5.17), $\bar{X}_{SV}^{\bar{E}}$ can be calculated through (5.42) and $\hat{\sigma}_{X_{SV}^{\bar{E}}}$ can be calculated from (5.43).

$$\bar{X}_{SV}^{\bar{E}} = \left(\sum_{PL=1}^{|PlanetLab|} X_{SV,PL}^{\bar{E}}\right)/(|PlanetLab|) \tag{5.42}$$

$$\hat{\sigma}_{X_{sv}^{\bar{E}}} = \sqrt{\frac{\left(\left(\sum_{PL=1}^{|PlanetLab|}\left(\left(X_{SV,PL}^{\bar{E}} - \bar{X}_{SV}^{\bar{E}}\right)^{2}\right)\right)/\left((|PlanetLab|) - 1\right)\right)}{(|PlanetLab|)}} \tag{5.43}$$

$$t_{\bar{X}_{SV}^{\bar{E}}} = \left(\left(\bar{X}_{SV}^{\bar{E}}\right) - 0\right)/\left(\hat{\sigma}_{X_{sv}^{\bar{E}}}\right) \tag{5.44}$$

Similarly, the mean and the standard deviation of sampling distribution of sample mean $X_{FF}^{\bar{E}}$, $X_{BF}^{\bar{E}}$, $X_{NF}^{\bar{E}}$ and $X_{RS}^{\bar{E}}$ denoted by $\bar{X}_{FF}^{\bar{E}}$, $\hat{\sigma}_{X_{FF}^{\bar{E}}}$, $\bar{X}_{BF}^{\bar{E}}$, $\hat{\sigma}_{X_{BF}^{\bar{E}}}$, $\bar{X}_{NF}^{\bar{E}}$, $\hat{\sigma}_{X_{NF}^{\bar{E}}}$, $\bar{X}_{RS}^{\bar{E}}$, $\hat{\sigma}_{X_{RS}^{\bar{E}}}$ can be calculated from (5.45), (5.46), (5.48), (5.49), (5.51), (5.52), (5.54) and (5.55), respectively. Values of $X_{FF}^{\bar{E}}$, $X_{BF}^{\bar{E}}$, $X_{NF}^{\bar{E}}$ and $X_{RS}^{\bar{E}}$ are presented in Table 5.7.

FF

$$\bar{X}_{FF}^{\bar{E}} = \left(\sum_{PL=1}^{|PlanetLab|} X_{FF,PL}^{\bar{E}}\right)/(|PlanetLab|) \tag{5.45}$$

$$\hat{\sigma}_{X_{FF}^{\bar{E}}} = \sqrt{\frac{\left(\left(\sum_{PL=1}^{|PlanetLab|}\left(\left(X_{FF,PL}^{\bar{E}} - \bar{X}_{FF}^{\bar{E}}\right)^{2}\right)\right)/\left((|PlanetLab|) - 1\right)\right)}{(|PlanetLab|)}} \tag{5.46}$$

$$t_{\bar{X}_{FF}^{\bar{E}}} = \left(\left(\bar{X}_{FF}^{\bar{E}}\right) - 0\right)/\left(\hat{\sigma}_{X_{FF}^{\bar{E}}}\right) \tag{5.47}$$

BF

$$\bar{X}_{BF}^{\bar{E}} = \left(\sum_{PL=1}^{|PlanetLab|} X_{BF,PL}^{\bar{E}}\right)/(|PlanetLab|) \tag{5.48}$$

$$\hat{\sigma}_{X_{BF}^{\bar{E}}} = \sqrt{\frac{\left(\left(\sum_{PL=1}^{|PlanetLab|}\left(\left(X_{BF,PL}^{\bar{E}} - \bar{X}_{BF}^{\bar{E}}\right)^{2}\right)\right)/\left((|PlanetLab|) - 1\right)\right)}{(|PlanetLab|)}} \tag{5.49}$$

$$t_{\bar{X}_{BF}^{\bar{E}}} = \left(\left(\bar{X}_{BF}^{\bar{E}}\right) - 0\right)/\left(\hat{\sigma}_{X_{BF}^{\bar{E}}}\right) \tag{5.50}$$

NF

$$\bar{X}_{NF}^{\bar{E}} = \left(\sum_{PL=1}^{|PlanetLab|} X_{NF,PL}^{\bar{E}}\right)/(|PlanetLab|) \tag{5.51}$$

$$\hat{\sigma}_{X_{NF}^{\bar{E}}} = \sqrt{\frac{\left(\left(\sum_{PL=1}^{|PlanetLab|}\left(\left(X_{NF,PL}^{\bar{E}} - \bar{X}_{NF}^{\bar{E}}\right)^{2}\right)\right)/\left((|PlanetLab|) - 1\right)\right)}{(|PlanetLab|)}} \tag{5.52}$$

$$t_{\bar{X}_{NF}^{\bar{E}}} = \left(\left(\bar{X}_{NF}^{\bar{E}}\right) - 0\right)/\left(\hat{\sigma}_{X_{NF}^{\bar{E}}}\right) \tag{5.53}$$

RS

$$\bar{X}_{RS}^{\bar{E}} = \left(\sum_{PL=1}^{|PlanetLab|} X_{RS,PL}^{\bar{E}}\right)/(|PlanetLab|) \tag{5.54}$$

$$\hat{\sigma}_{X_{RS}^{\bar{E}}} = \sqrt{\frac{\left(\left(\sum_{PL=1}^{|PlanetLab|}\left(\left(X_{RS,PL}^{\bar{E}} - \bar{X}_{RS}^{\bar{E}}\right)^{2}\right)\right)/\left((|PlanetLab|) - 1\right)\right)}{(|PlanetLab|)}} \tag{5.55}$$

$$t_{\bar{X}_{RS}^{\bar{E}}} = \left(\left(\bar{X}_{RS}^{\bar{E}}\right) - 0\right) / \left(\hat{\sigma}_{X_{RS}^{\bar{E}}}\right) \tag{5.56}$$

Since, we aim to prove that for every single SVMC algorithm, the improvement of $\bar{E}_{CDC}$ by *PRTDVMC* compared to THR-MMT is not a random occurrence and rather statistically significant, therefore, the null hypothesis for $\bar{E}_{CDC}$ with a particular SVMC algorithm, $H_0^{\bar{E},SV}$ is $\bar{\bar{E}}_{SV}^{P} = \bar{\bar{E}}_{SV}^{TMMT}$, which we aim to disprove. In other words, the null hypothesis is that there is no difference between those two means (i.e., $\bar{\bar{E}}_{SV}^{P}$ and $\bar{\bar{E}}_{SV}^{TMMT}$), as we aim to prove otherwise. Therefore, the alternative hypothesis, $H_1^{\bar{E},SV}$ is $\bar{\bar{E}}_{SV}^{P} < \bar{\bar{E}}_{SV}^{TMMT}$. Meaning of notations $\bar{\bar{E}}_{SV}^{P}$ and $\bar{\bar{E}}_{SV}^{TMMT}$ is described in Nomenclature Table, as $\bar{\bar{E}}_{SV}^{P}$ and $\bar{\bar{E}}_{SV}^{TMMT}$ can be derived utilising (5.24)and (5.25). Similarly, the null hypothesis for $\bar{E}_{CDC}$ with FF, BF, NF and RS algorithms, denoted by $H_0^{\bar{E},FF}$, $H_0^{\bar{E},BF}$, $H_0^{\bar{E},NF}$ and $H_0^{\bar{E},RS}$ are presented in (5.57), (5.58), (5.59) and (5.60), respectively.

$$H_0^{\bar{E},FF} = \left(\bar{\bar{E}}_{FF}^{TMMT} - \bar{\bar{E}}_{FF}^{P}\right) = \bar{X}_{FF}^{\bar{E}} = 0 \tag{5.57}$$

$$H_0^{\bar{E},BF} = \left(\bar{\bar{E}}_{BF}^{TMMT} - \bar{\bar{E}}_{BF}^{P}\right) = \bar{X}_{BF}^{\bar{E}} = 0 \tag{5.58}$$

$$H_0^{\bar{E},NF} = \left(\bar{\bar{E}}_{NF}^{TMMT} - \bar{\bar{E}}_{NF}^{P}\right) = \bar{X}_{NF}^{\bar{E}} = 0 \tag{5.59}$$

$$H_0^{\bar{E},RS} = \left(\bar{\bar{E}}_{RS}^{TMMT} - \bar{\bar{E}}_{RS}^{P}\right) = \bar{X}_{RS}^{\bar{E}} = 0 \tag{5.60}$$

Consequently, the alternative hypothesis for $\bar{E}_{CDC}$ with FF, BF, NF and RS algorithms, denoted by $H_1^{\bar{E},FF}$, $H_1^{\bar{E},BF}$, $H_1^{\bar{E},NF}$ and $H_1^{\bar{E},RS}$, which we are aiming to accept are presented in (5.61), (5.62), (5.63) and (5.64), respectively.

$$H_1^{\bar{E},FF} = \bar{\bar{E}}_{FF}^{TMMT} > \bar{\bar{E}}_{FF}^{P} \tag{5.61}$$

$$H_1^{\bar{E},BF} = \bar{\bar{E}}_{BF}^{TMMT} > \bar{\bar{E}}_{BF}^{P} \tag{5.62}$$

$$H_1^{\bar{E},NF} = \bar{\bar{E}}_{NF}^{TMMT} > \bar{\bar{E}}_{NF}^{P} \tag{5.63}$$

$$H_1^{\bar{E},RS} = \bar{\bar{E}}_{RS}^{TMMT} > \bar{\bar{E}}_{RS}^{P} \tag{5.64}$$

Let, $t_{\bar{X}_{SV}^{\bar{E}}}$ denote the test statistic of the two tail paired *t-test* with $\bar{X}_{SV}^{\bar{E}}$, which can be calculated through utilising equations (5.42) to (5.44). Similarly, test statistics of the two tail paired *t-test* with $\bar{X}_{FF}^{\bar{E}}$, $\bar{X}_{BF}^{\bar{E}}$, $\bar{X}_{NF}^{\bar{E}}$ and $\bar{X}_{RS}^{\bar{E}}$ denoted by $t_{\bar{X}_{FF}^{\bar{E}}}, t_{\bar{X}_{BF}^{\bar{E}}}, t_{\bar{X}_{NF}^{\bar{E}}}$ and $t_{\bar{X}_{RS}^{\bar{E}}}$ can be derived through (5.45) to (5.56). Values of $t_{\bar{X}_{FF}^{\bar{E}}}, t_{\bar{X}_{BF}^{\bar{E}}}, t_{\bar{X}_{NF}^{\bar{E}}}$ and $t_{\bar{X}_{RS}^{\bar{E}}}$ are found as 1.5, 1.8, 1.9 and 1.18, respectively, as corresponding p values are 0.0005, 0.0001, 0.00095 and 0.002, which are lower than critical value, α as 0.05. Hence, we reject null hypothesis (5.57), (5.58), (5.59) and (5.60), as we accept alternative hypothesis (5.61), (5.62), (5.63) and (5.64). In other words, we have proved that for every single SVMC algorithm, mean of CDC energy consumption for THR-MMT and *PRTDVMC* are not equal, as in fact, the mean of improvement of CDC energy consumption by *PRTDVMC* to THR-MMT is statistically significant. As such, the performance improvement in terms of mean CDC energy consumption by *PRTDVMC* compared to THR-MMT is not a random incident. In the following we have presented the logical explanation of steps constituting the *t-test*.

For easier understanding of the readers, we have first shown that the sampling distributions of sample mean, $\bar{E}_{FF}^{TMMT}$, $\bar{E}_{FF}^{P}$, $\bar{E}_{BF}^{TMMT}$, $\bar{E}_{BF}^{P}$, $\bar{E}_{NF}^{TMMT}$, $\bar{E}_{NF}^{P}$, $\bar{E}_{RS}^{TMMT}$ and $\bar{E}_{RS}^{P}$ follow normal distribution. From those sampling distributions of sample mean, we have next determined the respective mean of those sampling distributions of sample mean, denoted by $\bar{\bar{E}}_{FF}^{TMMT}$, $\bar{\bar{E}}_{FF}^{P}$, $\bar{\bar{E}}_{BF}^{TMMT}$, $\bar{\bar{E}}_{BF}^{P}$, $\bar{\bar{E}}_{NF}^{TMMT}$, $\bar{\bar{E}}_{NF}^{P}$, $\bar{\bar{E}}_{RS}^{TMMT}$ and $\bar{\bar{E}}_{RS}^{P}$, respectively. If we had repeated experiments more and more to obtain distributions of such mean as $\bar{\bar{E}}_{FF}^{TMMT}$, $\bar{\bar{E}}_{FF}^{P}$, $\bar{\bar{E}}_{BF}^{TMMT}$, $\bar{\bar{E}}_{BF}^{P}$, $\bar{\bar{E}}_{NF}^{TMMT}$, $\bar{\bar{E}}_{NF}^{P}$, $\bar{\bar{E}}_{RS}^{TMMT}$ and $\bar{\bar{E}}_{RS}^{P}$, then the resulting distributions would also follow normal distribution, since, $\bar{E}_{FF}^{TMMT}$, $\bar{E}_{FF}^{P}$, $\bar{E}_{BF}^{TMMT}$, $\bar{E}_{BF}^{P}$, $\bar{E}_{NF}^{TMMT}$, $\bar{E}_{NF}^{P}$, $\bar{E}_{RS}^{TMMT}$ and $\bar{E}_{RS}^{P}$ follow normal distribution. Furthermore, since, a distribution generated through subtracting two corresponding elements of two different normal distributions is also a normal distribution, hence, if we had generated such distributions as $\bar{\bar{E}}_{FF}^{P} - \bar{\bar{E}}_{FF}^{TMMT}$ or $\bar{X}_{FF}^{\bar{E}}$, $\bar{\bar{E}}_{BF}^{P} - \bar{\bar{E}}_{BF}^{TMMT}$ or $\bar{X}_{BF}^{\bar{E}}$, $\bar{\bar{E}}_{NF}^{P} - \bar{\bar{E}}_{NF}^{TMMT}$ or $\bar{X}_{NF}^{\bar{E}}$ and and $\bar{\bar{E}}_{RS}^{P} - \bar{\bar{E}}_{RS}^{TMMT}$ or $\bar{X}_{RS}^{\bar{E}}$ we would have observed that each of those distributions also follows normal distribution.

For any normal distribution, the mean holds the highest frequency or highest probability of appearance. The further away a value is from the mean, the lower is its respective probability of occurrence. If a value comes from a normal distribution, it is easy to determine its respective probability of appearance, if its distance away from the mean in terms of standard deviation is known. In our case, we wanted to establish that mean CDC energy consumption of THR-MMT is greater than that of *PRTDVMC*. Hence, we assumed the opposite as null hypothesis that two respective means of sampling distribution of sample mean are not different, shown in (5.57), (5.58), (5.59) and (5.60). In other words, as per the null hypothesis, mean of such normal distributions as $\bar{X}_{FF}^{\bar{E}}$, $\bar{X}_{BF}^{\bar{E}}$, $\bar{X}_{NF}^{\bar{E}}$ and $\bar{X}_{RS}^{\bar{E}}$ are 0. Next, assuming that the null hypothesis is true, we have calculated our test statistics that how many standard deviation far away is our obtained mean CDC energy consumption decrease (i.e., $\bar{X}_{FF}^{\bar{E}}$, $\bar{X}_{BF}^{\bar{E}}$, $\bar{X}_{NF}^{\bar{E}}$ and $\bar{X}_{RS}^{\bar{E}}$) from the mean of its respective distribution (i.e., 0), so that corresponding probability or *p* values can be obtained. Hence, equations (5.47), (5.50), (5.53) and (5.56) are introduced to calculate test statistics $t_{\bar{X}_{FF}^{\bar{E}}}$,

$t_{\bar{X}^{\bar{E}}_{BF}}$, $t_{\bar{X}^{\bar{E}}_{NF}}$ and $t_{\bar{X}^{\bar{E}}_{RS}}$, respectively. Based on the distance away from the distribution mean in terms of standard deviation, we have found the respective probability of occurrence, aka *p* value of our obtained mean CDC energy consumption decreases by *PRTDVMC*. For instance, assuming that the null hypothesis, $H_0^{\bar{E},FF}$ as true, we have found the respective probability for 0.634 kW to appear as $\bar{X}^{\bar{E}}_{FF}$ is 0.2%, which is less than 5%. Despite a very small probability of 0.2%, as we still have managed to receive 0.634 kW as $\bar{X}^{\bar{E}}_{FF}$, therefore, we can argue that this is a strong case that the assumption of $H_0^{\bar{E},FF}$ itself is not true and hence, shall be rejected. Similarly, based on the respective low *p* values, $H_0^{\bar{E},BF}$, $H_0^{\bar{E},NF}$ and $H_0^{\bar{E},RS}$are rejected and the alternative hypotheses are accepted.

- **Test Error and Confidence Interval (CI)**

One noteworthy point is that no statistical test including the *t-test* is error free. Errors of the *t-test* are two types: Type I error and Type II error. Type I error refers to the total probability of falsely rejecting the null hypothesis while it was true, and Type II error refers to the total probability of falsely rejecting alternative hypothesis while it was true. *p* value refers to the probability to obtain the test statistic assuming that the null hypothesis is true. Based on *p* values of 0.0005, 0.0001, 0.00095 and 0.002, we have rejected null hypotheses $H_0^{\bar{E},FF}$, $H_0^{\bar{E},BF}$, $H_0^{\bar{E},NF}$ and $H_0^{\bar{E},RS}$. Therefore, the probability to falsely reject $H_0^{\bar{E},FF}$, $H_0^{\bar{E},BF}$, $H_0^{\bar{E},NF}$ and $H_0^{\bar{E},RS}$ while those were true, aka Type I errors are 0.0005, 0.0001, 0.00095 and 0.002, respectively.

It is also important to consider that the mean improvement of CDC energy consumption by *PRTDVMC* compared to THR-MMT in the presence of FF, BF, NF and RS, denoted by $\bar{X}^{\bar{E}}_{FF}$, $\bar{X}^{\bar{E}}_{BF}$, $\bar{X}^{\bar{E}}_{NF}$ and $\bar{X}^{\bar{E}}_{RS}$ are not population mean, rather sample mean. Earlier, we have shown that the sampling distributions of sample mean, $\bar{E}^{TMMT}_{FF}$, $\bar{E}^{P}_{FF}$, $\bar{E}^{TMMT}_{BF}$, $\bar{E}^{P}_{BF}$, $\bar{E}^{TMMT}_{NF}$, $\bar{E}^{P}_{NF}$, $\bar{E}^{TMMT}_{RS}$ and $\bar{E}^{P}_{RS}$ follow normal distribution. Since, a distribution generated through subtracting two corresponding elements of two different normal distributions is also a normal distribution, therefore, such distributions as $X^{\bar{E}}_{FF}$ (5.18), $X^{\bar{E}}_{BF}$ (5.19), $X^{\bar{E}}_{NF}$ (5.20) and $X^{\bar{E}}_{RS}$ (5.21) would also follow normal distribution. Consequently, the probability is 95% that $\bar{X}^{\bar{E}}_{FF}$ (5.45), $\bar{X}^{\bar{E}}_{BF}$ (5.48), $\bar{X}^{\bar{E}}_{NF}$ (5.51) and $\bar{X}^{\bar{E}}_{RS}$ (5.54), which we have obtained through our experiment would be within the distance of ± 2 standard deviation from respective population mean. Hence, we can also make a statement that there is 95% probability that respective population mean would be within ± 2 standard deviation distance of sample mean $\bar{X}^{\bar{E}}_{FF}$, $\bar{X}^{\bar{E}}_{BF}$, $\bar{X}^{\bar{E}}_{NF}$ and $\bar{X}^{\bar{E}}_{RS}$. If we add the distance (i.e.,

± 2 standard deviation distance) with the value of mean (i.e., $\bar{X}_{FF}^{\bar{E}}$, $\bar{X}_{BF}^{\bar{E}}$, $\bar{X}_{NF}^{\bar{E}}$ and $\bar{X}_{RS}^{\bar{E}}$), a range can be obtained and the probability would be 95% for finding population mean within such range. Such range is referred to as Confidence Interval (CI). A common way of stating CI of a mean is we are 95% confident that the population mean would be within that range.

The standard deviation, which is referred with respect to CI is not sample standard deviation, rather population standard deviation. Since, we do not know the population mean, population standard deviation is unknown. The way to encounter such issue is to use Standard Error, SE instead, which is very close approximate of population standard deviation, as shown in (5.65).

$$SE = (Standard\ Deviation)/\left(\sqrt{Sample\ Size}\right) \tag{5.65}$$

Similarly, SE for distribution of $X_{FF}^{\bar{E}}$, $X_{BF}^{\bar{E}}$, $X_{NF}^{\bar{E}}$ and $X_{RS}^{\bar{E}}$, denoted by $SE_{X_{FF}^{\bar{E}}}$, $SE_{X_{BF}^{\bar{E}}}$, $SE_{X_{NF}^{\bar{E}}}$ and $SE_{X_{RS}^{\bar{E}}}$ can be calculated from (5.66), (5.67), (5.68) and (5.69), respectively.

$$SE_{X_{FF}^{\bar{E}}} = \left(\hat{\sigma}_{X_{FF}^{\bar{E}}}\right)/\left(\sqrt{|PlanetLab|}\right) \tag{5.66}$$

$$SE_{X_{BF}^{\bar{E}}} = \left(\hat{\sigma}_{X_{BF}^{\bar{E}}}\right)/\left(\sqrt{|PlanetLab|}\right) \tag{5.67}$$

$$SE_{X_{NF}^{\bar{E}}} = \left(\hat{\sigma}_{X_{NF}^{\bar{E}}}\right)/\left(\sqrt{|PlanetLab|}\right) \tag{5.68}$$

$$SE_{X_{RS}^{\bar{E}}} = \left(\hat{\sigma}_{X_{RS}^{\bar{E}}}\right)/\left(\sqrt{|PlanetLab|}\right) \tag{5.69}$$

We should carefully consider that distributions of such mean as $\bar{X}_{FF}^{\bar{E}}$, $\bar{X}_{BF}^{\bar{E}}$, $\bar{X}_{NF}^{\bar{E}}$ and $\bar{X}_{RS}^{\bar{E}}$ are not perfect normal distributions, rather t distribution. Let, t_{CI} denotes critical value for t interval for 95% CI. Hence, instead of ± 2 standard deviation distance, it would be more accurate to state that there is 95% probability for respective population mean to be found within ($\pm t_{CI} \cdot SE$) distance of sample mean $\bar{X}_{FF}^{\bar{E}}$, $\bar{X}_{BF}^{\bar{E}}$, $\bar{X}_{NF}^{\bar{E}}$ and $\bar{X}_{RS}^{\bar{E}}$. In other words, we are 95% confident that population mean would be within the range $\left[\bar{X}_{SV}^{\bar{E}} + \left(t_{CI} \cdot SE_{X_{SV}^{\bar{E}}}\right),\ \bar{X}_{SV}^{\bar{E}} - \left(t_{CI} \cdot SE_{X_{SV}^{\bar{E}}}\right)\right]$, aka 95% CI for $\bar{X}_{SV}^{\bar{E}}$. Similarly, CI for $\bar{X}_{FF}^{\bar{E}}$, $\bar{X}_{BF}^{\bar{E}}$, $\bar{X}_{NF}^{\bar{E}}$ and $\bar{X}_{RS}^{\bar{E}}$, denoted by $CI_{\bar{X}_{FF}^{\bar{E}}}$, $CI_{\bar{X}_{BF}^{\bar{E}}}$, $CI_{\bar{X}_{NF}^{\bar{E}}}$ and $CI_{\bar{X}_{RS}^{\bar{E}}}$ can be estimated from **Error! Reference source not found.**, **Error! Reference source not found.**, **Error! Reference source not found.** and **Error! Reference source not found.**, respectively. Values of t_{CI} are available in [148]. Fig. 5.4 presents values of $CI_{\bar{X}_{FF}^{\bar{E}}}$, $CI_{\bar{X}_{BF}^{\bar{E}}}$, $CI_{\bar{X}_{NF}^{\bar{E}}}$ and $CI_{\bar{X}_{RS}^{\bar{E}}}$.

$$CI_{\bar{X}_{FF}^{\bar{E}}} = \bar{X}_{FF}^{\bar{E}} \pm \left(t_{CI} \cdot SE_{X_{FF}^{\bar{E}}}\right) \tag{5.70}$$

$$CI_{\bar{X}_{BF}^{\bar{\bar{E}}}} = \bar{X}_{BF}^{\bar{\bar{E}}} \pm \left(t_{CI} \cdot SE_{X_{BF}^{\bar{\bar{E}}}}\right) \tag{5.71}$$

$$CI_{\bar{X}_{NF}^{\bar{\bar{E}}}} = \bar{X}_{NF}^{\bar{\bar{E}}} \pm \left(t_{CI} \cdot SE_{X_{NF}^{\bar{\bar{E}}}}\right) \tag{5.72}$$

$$CI_{\bar{X}_{RS}^{\bar{\bar{E}}}} = \bar{X}_{RS}^{\bar{\bar{E}}} \pm \left(t_{CI} \cdot SE_{X_{S}^{\bar{\bar{E}}}}\right) \tag{5.73}$$

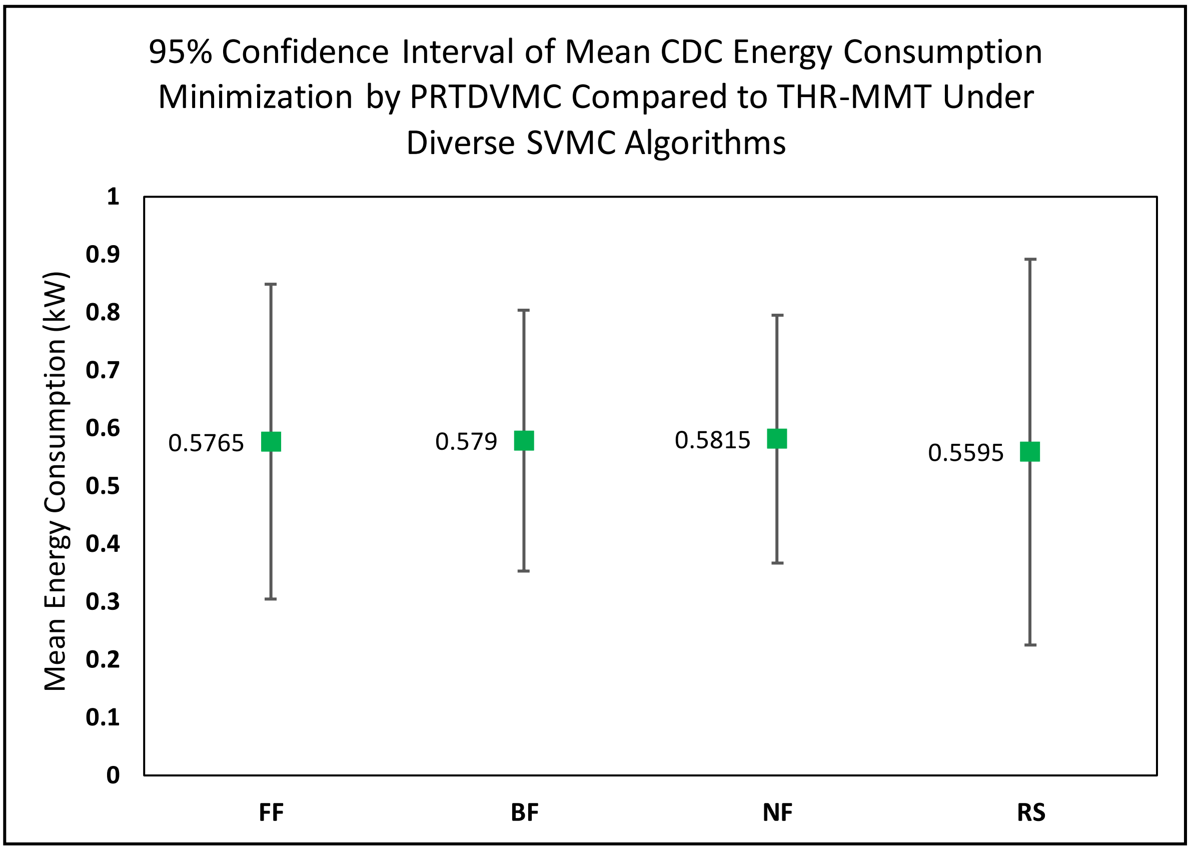

Fig. 5.4 Confidence Interval of Mean CDC Energy Consumption Minimization by *PRTDVMC* Compared to THR-MMT Under Diverse SVMC Algorithms

PRTDVMC is a multi-objective DVMC algorithm, which aims to minimize both CDC energy consumption and VM migration. In the following section, we have articulated experimental results in terms of VM migration.

5.7.3.2 VM Migration

VM Migration causes QoS degradation, consumes network bandwidth and raises networking equipment related energy consumption. However, DVMC consolidates VMs in lesser number of PMs through VM migration. Hence, concomitant regulation of both VM migration and CDC energy consumption is extremely challenging. Table 5.10 and Table 5.11, presents members of set of mean total number of VM migration for *PRTDVMC* under different SVMC algorithm,

$\bar{\psi}_{SV}^{P} = \{\bar{\psi}_{SV,PL}^{P}\}_{|PlanetLab|}$ and those for THR-MMT, $\bar{\psi}_{SV}^{TMMT} = \{\bar{\psi}_{SV,PL}^{TMMT}\}_{|PlanetLab|}$ respectively for different days of PlanetLab workload with workload finishing time drawn from *PVMRT* distribution of Nectar Cloud (Fig. 5.1).

Table 5.10 Mean Total VM Migration by *PRTDVMC* under Diverse SVMC Algorithms for Different Days of PlanetLab Workload

Mean Total Number of VM Migration by *PRTDVMC*				
	$\bar{\psi}_{FF}^{P}$	$\bar{\psi}_{BF}^{P}$	$\bar{\psi}_{NF}^{P}$	$\bar{\psi}_{RS}^{P}$
3 March	543	533	544	507
6 March	442	465	449	443
9 March	514	516	525	544
22 March	694	746	697	701
25 March	499	507	536	525
3 April	733	711	705	729
9 April	646	653	653	667
11 April	599	605	577	644
12 April	522	517	508	516
20 April	481	503	488	486

Table 5.11 Mean Total VM Migration by THR-MMT under Diverse SVMC Algorithms for Different Days of PlanetLab Workload

Mean Total Number of VM Migration by THR-MMT				
	$\bar{\psi}_{FF}^{TMMT}$	$\bar{\psi}_{BF}^{TMMT}$	$\bar{\psi}_{NF}^{TMMT}$	$\bar{\psi}_{RS}^{TMMT}$
3 March	1256	1180	1161	1133
6 March	975	1086	1017	1045
9 March	1178	1084	1198	1253
22 March	1443	1492	1427	1436
25 March	1121	1165	1160	1244
3 April	1519	1436	1475	1397
9 April	1388	1388	1454	1456
11 April	1305	1339	1278	1422
12 April	1254	1116	1147	1161
20 April	1112	1128	1084	1105

Let, $X_{SV}^{\bar{\psi}} = \left\{X_{SV,PL}^{\bar{\psi}}\right\}_{|PlanetLab|}$ (5.74) denotes the set presenting difference between mean total number of VM migration by THR-MMT and mean total number of VM migration by *PRTDVMC* under a particular SVMC algorithm for different days of PlanetLab workload. In other words, $X_{SV}^{\bar{\psi}}$ represents the minimization of mean total number of VM migration proffered by *PRTDVMC* compare to THR-MMT for diverse PlanetLab workloads under a particular SVMC algorithm.

$$X_{SV}^{\bar{\psi}} = \left\{X_{SV,PL}^{\bar{\psi}}\right\}_{|PlanetLab|} = \{\bar{\psi}_{SV,PL}^{TMMT} - \bar{\psi}_{SV,PL}^{P}\}_{|PlanetLab|} \tag{5.74}$$

Similarly, equations to obtain minimization of mean total number of VM migration proffered by *PRTDVMC* compare to THR-MMT under FF, BF, NF and RS, denoted by $X_{FF}^{\bar{\psi}}$, $X_{BF}^{\bar{\psi}}$, $X_{NF}^{\bar{\psi}}$ and $X_{RS}^{\bar{\psi}}$ are presented in (5.75), (5.76), (5.77) and (5.78), respectively.

$$X_{FF}^{\bar{\psi}} = \left\{X_{FF,PL}^{\bar{\psi}}\right\}_{|PlanetLab|} = \{\bar{\psi}_{FF,PL}^{TMMT} - \bar{\psi}_{FF,PL}^{P}\}_{|PlanetLab|} \tag{5.75}$$

$$X_{BF}^{\bar{\psi}} = \left\{X_{BF,PL}^{\bar{\psi}}\right\}_{|PlanetLab|} = \{\bar{\psi}_{BF,PL}^{TMMT} - \bar{\psi}_{BF,PL}^{P}\}_{|PlanetLab|} \tag{5.76}$$

$$X_{NF}^{\bar{\psi}} = \left\{X_{NF,PL}^{\bar{\psi}}\right\}_{|PlanetLab|} = \{\bar{\psi}_{NF,PL}^{TMMT} - \bar{\psi}_{NF,PL}^{P}\}_{|PlanetLab|} \tag{5.77}$$

$$X_{RS}^{\bar{\psi}} = \left\{X_{RS,PL}^{\bar{\psi}}\right\}_{|PlanetLab|} = \{\bar{\psi}_{RS,PL}^{TMMT} - \bar{\psi}_{RS,PL}^{P}\}_{|PlanetLab|} \tag{5.78}$$

In Table 5.12, we have presented the reduction of mean total number of VM Migration proffered by *PRTDVMC* compare to THR-MMT for diverse PlanetLab workloads under different SVMC algorithms.

Table 5.12 Minimization of Mean Total VM Migration Proffered by *PRTDVMC* compared to THR-MMT under Different SVMC Algorithms with Different Days of PlanetLab Workload

Minimization of Mean Total Number of VM Migration by *PRTDVMC*				
	$X_{FF}^{\bar{\psi}}$	$X_{BF}^{\bar{\psi}}$	$X_{NF}^{\bar{\psi}}$	$X_{RS}^{\bar{\psi}}$
3 March	713	647	617	625
6 March	533	621	567	601
9 March	664	568	673	709
22 March	748	745	730	734
25 March	621	658	624	719
3 April	786	725	770	668
9 April	742	735	801	789
11 April	705	734	701	778
12 April	732	599	639	644
20 April	630	625	596	619

Fig. 5.5 and Fig. 5.6 delineate the performance comparison between *PRTDVMC* and THR-MMT in terms of mean total number of VM migration in combination with four different SVMC algorithms for different days of PlanetLab data.

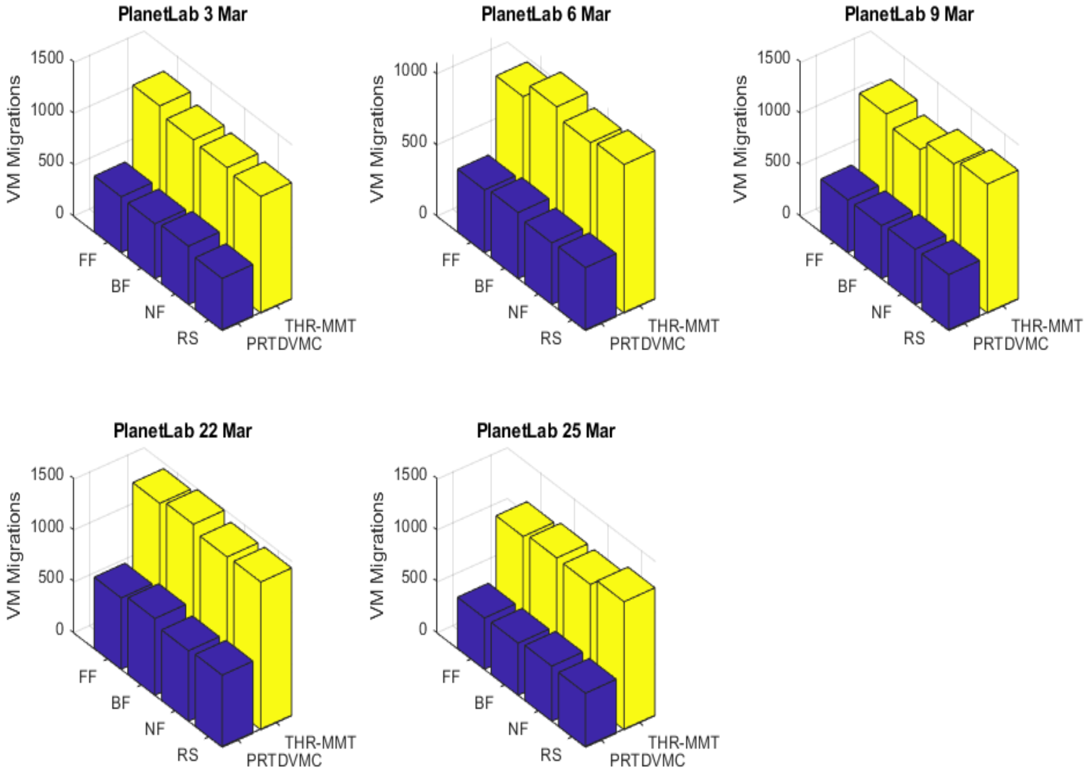

Fig. 5.5 Comparison of Mean Total Number of VM Migration caused by *PRTDVMC* and THR-MMT with PlanetLab Workload of March under Diverse SVMC Algorithms

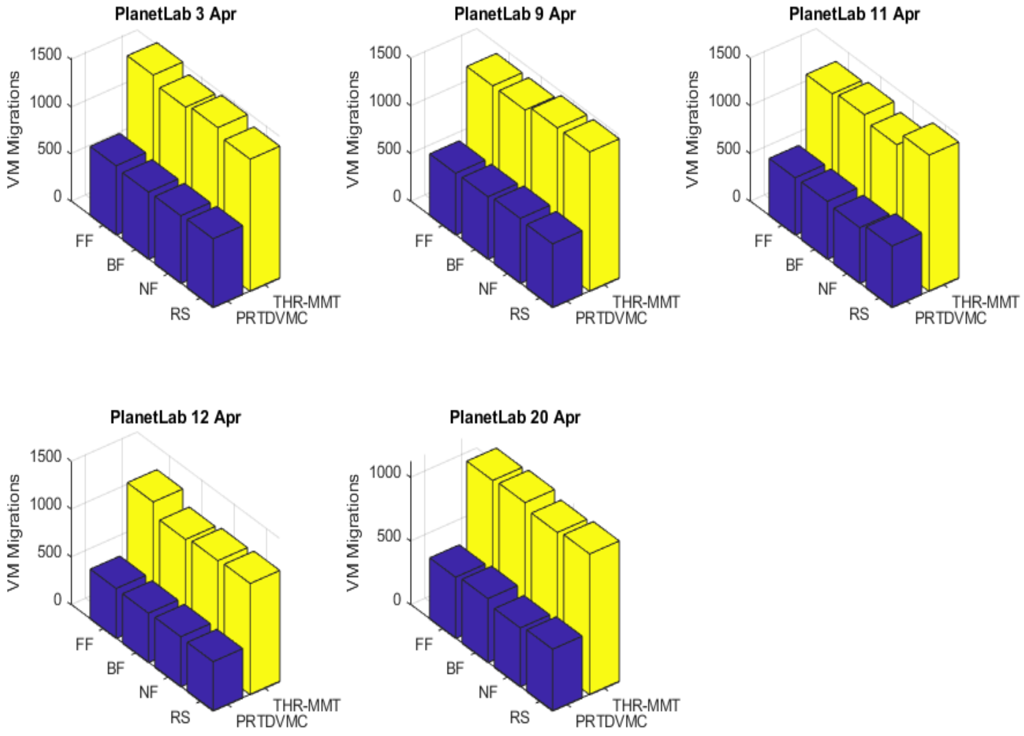

Fig. 5.6 Comparison of Mean Total Number of VM Migration caused by *PRTDVMC* and THR-MMT with PlanetLab Workload of April under Diverse SVMC Algorithms

Based on our experimental results, we have undertaken diverse critical statistical tests as presented in the following section.

- **Normality Testing**

From Fig. 5.5 and Fig. 5.6, we can view that *PRTDVMC* outperforms THR-MMT in terms mean total number of VM migration, $\bar{\psi}$. One might deny such improvement exhibited by empirical evaluations stating that the improvement by *PRTDVMC* compared to THR-MMT is merely a random occurrence. To refute such argument, it is essential to verify through parametric test if the improvement exhibited by *PRTDVMC* is statistically significant or not. Prior Parametric test normality testing is required. Test statistics for the S-W normality test with $\bar{\psi}_{SV}^{TMMT}$ and $\bar{\psi}_{SV}^{P}$, referred to as $SW_{\bar{\psi}}^{TMMT,SV}$ and $SW_{\bar{\psi}}^{P,SV}$respectively can be calculated through utilizing (5.79) to (5.82) [116, 117].

$$SW_{\bar{\psi}}^{TMMT,SV} = \left(\left(\sum_{w=1}^{|w|} a_w \cdot \left(\bar{\psi}_{(|w|+1-w)}^{TMMT,SV} - \bar{\psi}_{(w)}^{TMMT,SV}\right)\right)^2\right) / \left(\sum_{w=1}^{|w|}\left(\bar{\psi}_{(w)}^{TMMT,SV} - \bar{\bar{\psi}}_{SV}^{TMMT}\right)^2\right) \tag{5.79}$$

$$SW_{\bar{\psi}}^{P,SV} = \left(\left(\sum_{w=1}^{|w|} a_w \cdot \left(\bar{\psi}_{(|w|+1-w)}^{P,SV} - \psi_{(w)}^{P,SV}\right)\right)^2\right) / \left(\sum_{w=1}^{|w|}\left(\bar{\psi}_{(w)}^{P,SV} - \bar{\bar{\psi}}_{SV}^{P}\right)^2\right) \tag{5.80}$$

$$\bar{\bar{\psi}}_{SV}^{TMMT} = \left(\sum_{w=1}^{|w|} \bar{\psi}_{(w)}^{TMMT,SV}\right) / (|w|) \tag{5.81}$$

$$\bar{\bar{\psi}}_{SV}^{P} = \left(\sum_{w=1}^{|w|} \bar{\psi}_{(w)}^{P,SV}\right) / (|w|) \tag{5.82}$$

Test statistics for the S-W normality test with $\bar{\psi}_{FF}^{TMMT}$, $\bar{\psi}_{FF}^{P}$, $\bar{\psi}_{BF}^{TMMT}$, $\bar{\psi}_{BF}^{P}$, $\bar{\psi}_{NF}^{TMMT}$, $\bar{\psi}_{NF}^{P}$, $\bar{\psi}_{RS}^{TMMT}$ and $\bar{\psi}_{RS}^{P}$, denoted by $SW_{\bar{\psi}}^{TMMT,FF}$, $SW_{\bar{\psi}}^{P,FF}$, $SW_{\bar{\psi}}^{TMMT,BF}$, $SW_{\bar{\psi}}^{P,BF}$, $SW_{\bar{\psi}}^{TMMT,NF}$, $SW_{\bar{\psi}}^{P,NF}$, $SW_{\bar{\psi}}^{TMMT,RS}$ and $SW_{\bar{\psi}}^{P,RS}$ can also be calculated through (5.83) to (5.98) like thereof. a_w weights are available in Shapiro-Wilk Table [118]. Different $\bar{\psi}_{(w)}^{TMMT,SV}$ and $\bar{\psi}_{(w)}^{P,SV}$ values are presented in Table 5.13 and Table 5.14, respectively.

FF

$$SW_{\bar{\psi}}^{TMMT,FF} = \left(\left(\sum_{w=1}^{|w|} a_w \cdot \left(\bar{\psi}_{(|w|+1-w)}^{TMMT,FF} - \bar{\psi}_{(w)}^{TMMT,FF}\right)\right)^2\right) / \left(\sum_{w=1}^{|w|}\left(\bar{\psi}_{(w)}^{TMMT,FF} - \bar{\bar{\psi}}_{FF}^{TMMT}\right)^2\right) \tag{5.83}$$

$$SW_{\bar{\psi}}^{P,FF} = \left(\left(\sum_{w=1}^{|w|} a_w \cdot \left(\bar{\psi}_{(|w|+1-w)}^{P,FF} - \bar{\psi}_{(w)}^{P,FF}\right)\right)^2\right) / \left(\sum_{w=1}^{|w|}\left(\bar{\psi}_{(w)}^{P,FF} - \bar{\bar{\psi}}_{FF}^{P}\right)^2\right) \tag{5.84}$$

$$\bar{\bar{\psi}}_{FF}^{TMMT} = \left(\sum_{w=1}^{|w|} \bar{\psi}_{(w)}^{TMMT,FF}\right)/(|w|) \tag{5.85}$$

$$\bar{\bar{\psi}}_{FF}^{P} = \left(\sum_{w=1}^{|w|} \bar{\psi}_{(w)}^{P,FF}\right)/(|w|) \tag{5.86}$$

BF

$$SW_{\bar{\psi}}^{TMMT,BF} = \tag{5.87}$$

$$\left(\left(\sum_{w=1}^{|w|} a_w \cdot \left(\bar{\psi}_{(|w|+1-w)}^{TMMT,BF} - \psi_{(w)}^{TMMT,BF}\right)\right)^2\right) / \left(\sum_{w=1}^{|w|}\left(\bar{\psi}_{(w)}^{TMMT,BF} - \bar{\bar{\psi}}_{BF}^{TMMT}\right)^2\right)$$

$$SW_{\bar{\psi}}^{P,BF} = \left(\left(\sum_{w=1}^{|w|} a_w \cdot \left(\bar{\psi}_{(|w|+1-w)}^{P,BF} - \bar{\psi}_{(w)}^{P,BF}\right)\right)^2\right) / \left(\sum_{w=1}^{|w|}\left(\bar{\psi}_{(w)}^{P,BF} - \bar{\bar{\psi}}_{BF}^{P}\right)^2\right) \tag{5.88}$$

$$\bar{\bar{\psi}}_{BF}^{TMMT} = \left(\sum_{w=1}^{|w|} \bar{\psi}_{(w)}^{TMMT,BF}\right)/(|w|) \tag{5.89}$$

$$\bar{\bar{\psi}}_{BF}^{P} = \left(\sum_{w=1}^{|w|} \bar{\psi}_{(w)}^{P,BF}\right)/(|w|) \tag{5.90}$$

NF

$$SW_{\bar{\psi}}^{TMMT,NF} = \tag{5.91}$$

$$\left(\left(\sum_{w=1}^{|w|} a_w \cdot \left(\bar{\psi}_{(|w|+1-w)}^{TMMT,NF} - \bar{\psi}_{(w)}^{TMMT,NF}\right)\right)^2\right) / \left(\sum_{w=1}^{|w|}\left(\bar{\psi}_{(w)}^{TMMT,NF} - \bar{\bar{\psi}}_{NF}^{TMMT}\right)^2\right)$$

$$SW_{\bar{\psi}}^{P,NF} = \left(\left(\sum_{w=1}^{|w|} a_w \cdot \left(\bar{\psi}_{(|w|+1-w)}^{P,NF} - \bar{\psi}_{(w)}^{P,NF}\right)\right)^2\right) / \left(\sum_{w=1}^{|w|}\left(\bar{\psi}_{(w)}^{P,NF} - \bar{\bar{\psi}}_{NF}^{P}\right)^2\right) \tag{5.92}$$

$$\bar{\bar{\psi}}_{NF}^{TMMT} = \left(\sum_{w=1}^{|w|} \bar{\psi}_{(w)}^{TMMT,NF}\right)/(|w|) \tag{5.93}$$

$$\bar{\bar{\psi}}_{NF}^{P} = \left(\sum_{w=1}^{|w|} \bar{\psi}_{(w)}^{P,NF}\right)/(|w|) \tag{5.94}$$

RS

$$SW_{\bar{\psi}}^{TMMT,RS} = \tag{5.95}$$

$$\left(\left(\sum_{w=1}^{|w|} a_w \cdot \left(\bar{\psi}_{(|w|+1-w)}^{TMMT,RS} - \bar{\psi}_{(w)}^{TMMT,RS}\right)\right)^2\right) / \left(\sum_{w=1}^{|w|}\left(\bar{\psi}_{(w)}^{TMMT,RS} - \bar{\bar{\psi}}_{RS}^{TMMT}\right)^2\right)$$

$$SW_{\bar{\psi}}^{P,RS} = \left(\left(\sum_{w=1}^{|w|} a_w \cdot \left(\bar{\psi}_{(|w|+1-w)}^{P,RS} - \bar{\psi}_{(w)}^{P,RS}\right)\right)^2\right) / \left(\sum_{w=1}^{|w|}\left(\bar{\psi}_{(w)}^{P,RS} - \bar{\bar{\psi}}_{RS}^{P}\right)^2\right) \tag{5.96}$$

$$\bar{\bar{\psi}}_{RS}^{TMMT} = \left(\sum_{w=1}^{|w|} \bar{\psi}_{(w)}^{TMMT,RS}\right)/(|w|) \tag{5.97}$$

$$\bar{\bar{\psi}}_{RS}^{P} = \left(\sum_{w=1}^{|w|} \bar{\psi}_{(w)}^{P,RS}\right)/(|w|) \tag{5.98}$$

Table 5.13 Sorted Mean CDC Energy Consumption (kWh) by THR-MMT under Diverse SVMC Algorithms for Different Days of PlanetLab Workload

Sorted Mean Total Number of VM Migration by THR-MMT				
w	$\bar{\psi}_{(w)}^{TMMT,FF}$	$\bar{\psi}_{(w)}^{TMMT,BF}$	$\bar{\psi}_{(w)}^{TMMT,NF}$	$\bar{\psi}_{(w)}^{TMMT,RS}$
1	976	1085	1017	1045
2	1112	1087	1084	1105
3	1121	1116	1147	1133
4	1178	1129	1161	1161
5	1254	1166	1161	1244
6	1256	1180	1199	1254
7	1305	1340	1279	1397
8	1389	1389	1428	1422
9	1443	1436	1454	1436
10	1520	1492	1476	1457

Table 5.14 Sorted mean CDC Energy Consumption (kWh) by *PRTDVMC* under Diverse SVMC Algorithms for Different Days of PlanetLab Workload

Sorted Mean Total Number of VM Migration by *PRTDVMC*				
w	$\bar{\psi}_{(w)}^{P,FF}$	$\bar{\psi}_{(w)}^{P,BF}$	$\bar{\psi}_{(w)}^{P,NF}$	$\bar{\psi}_{(w)}^{P,RS}$
1	442	465	450	444
2	482	503	488	486
3	500	508	508	508
4	514	516	526	517
5	522	517	537	525
6	543	533	544	545
7	600	606	578	644
8	646	653	653	668
9	695	711	697	702
10	734	747	706	729

The S-W normality testing software has been acquired from [106]. The Null Hypothesis for the S-W normality tests with data samples of $\bar{\psi}_{FF}^{TMMT}$, $\bar{\psi}_{FF}^{P}$, $\bar{\psi}_{BF}^{TMMT}$, $\bar{\psi}_{BF}^{P}$, $\bar{\psi}_{NF}^{TMMT}$, $\bar{\psi}_{NF}^{P}$, $\bar{\psi}_{RS}^{TMMT}$ and $\bar{\psi}_{RS}^{P}$, is that data is normally distributed, as corresponding p values are found as 0.98, 0.46, 0.08, 0.12, 0.31, 0.37, 0.3 and 0.33 respectively, which are not smaller than critical value, α as 0.05. Hence, no strong evidence could be found to reject Null Hypothesis that elements of $\bar{\psi}_{FF}^{TMMT}$, $\bar{\psi}_{FF}^{P}$, $\bar{\psi}_{BF}^{TMMT}$, $\bar{\psi}_{BF}^{P}$, $\bar{\psi}_{NF}^{TMMT}$, $\bar{\psi}_{NF}^{P}$, $\bar{\psi}_{RS}^{TMMT}$ and $\bar{\psi}_{RS}^{P}$ have come from normal distribution. Hence, the condition is met to perform parametric hypothesis test with $\bar{\psi}_{FF}^{TMMT}$,

$\bar{\psi}^{P}_{FF}$, $\bar{\psi}^{TMMT}_{BF}$, $\bar{\psi}^{P}_{BF}$, $\bar{\psi}^{TMMT}_{NF}$, $\bar{\psi}^{P}_{NF}$, $\bar{\psi}^{TMMT}_{RS}$ and $\bar{\psi}^{P}_{RS}$. In the following section, we have presented parametric hypothesis testing.

- **Parametric Hypothesis Testing**

Sampling distribution of sample mean, such as $X^{\bar{\psi}}_{FF}$, $X^{\bar{\psi}}_{BF}$, $X^{\bar{\psi}}_{NF}$ and $X^{\bar{\psi}}_{RS}$ articulated in Table 5.12 refer to the fact that mean total number of VM migration by *PRTDVMC* is numerically lower compare to THR-MMT for different workload scenarios as presented in experiments. It can be argued that such improvement is a random incident and has happened by chance. To refute such argument, we have performed parametric hypothesis test to prove that the minimization of mean total number of VM Migration by *PRTDVMC* compared to THR-MMT is statistically significant and hence, not a random incident, as the improvement is rather. Sample size (i.e., $|PlanetLab|$) is 10, which is less than 30 and means of no more than two DVMC algorithms (i.e., THR-MMT and *PRTDVMC*) would be compared. Hence, among different parametric tests the two tail *t-test* is chosen, instead of Z-test, F-test and ANOVA. Based on the data samples, the *t-test* can be classified into three categories: One sample, Two Independent Samples and Paired Samples *t-test*. For a specific combination of SV and PL, corresponding $\bar{\psi}^{TMMT}_{SV,PL}$ and $\bar{\psi}^{P}_{SV,PL}$ has a relation, as $\bar{\psi}^{TMMT}_{SV,PL}$ and $\bar{\psi}^{P}_{SV,PL}$ refers to mean total number of VM migration for THR-MMT and *PRTDVMC* respectively, under a particular combination of SVMC algorithm and PlanetLab workload scenario. Therefore, the paired two tail *t-test* is chosen.

Previously, we have explained the logical progression and rationale of steps related to the *t-test*. Let, $\bar{X}^{\bar{\psi}}_{SV}$ and $\hat{\sigma}_{X^{\bar{\psi}}_{SV}}$ denote the mean and the standard deviation of sampling distribution of sample mean, $X^{\bar{\psi}}_{SV}$. $\bar{X}^{\bar{\psi}}_{SV}$ can be calculated through (5.99) and $\hat{\sigma}_{X^{\bar{\psi}}_{SV}}$ can be calculated from (5.100).

$$\bar{X}^{\bar{\psi}}_{SV} = \left(\sum_{PL=1}^{|PlanetLab|} X^{\bar{\psi}}_{SV,PL}\right) \Big/ (|PlanetLab|) \tag{5.99}$$

$$\hat{\sigma}_{X^{\bar{\psi}}_{sv}} = \sqrt{\frac{\left(\left(\sum_{PL=1}^{|PlanetLab|}\left(\left(X^{\bar{\psi}}_{SV,PL} - \bar{X}^{\bar{\psi}}_{SV}\right)^{2}\right)\right) \Big/ \left((|PlanetLab|) - 1\right)\right)}{(|PlanetLab|)}} \tag{5.100}$$

$$t_{\bar{X}^{\bar{\psi}}_{SV}} = \left(\bar{X}^{\bar{\psi}}_{SV}\right) \Big/ \left(\hat{\sigma}_{X^{\bar{\psi}}_{sv}}\right) \tag{5.101}$$

Similarly, the mean and the standard deviation of sampling distribution of sample mean $X_{FF}^{\bar{\psi}}$, $X_{BF}^{\bar{\psi}}$, $X_{NF}^{\bar{\psi}}$ and $X_{RS}^{\bar{\psi}}$ denoted by $\bar{X}_{FF}^{\bar{\psi}}$, $\hat{\sigma}_{X_{FF}^{\bar{\psi}}}$, $\bar{X}_{BF}^{\bar{\psi}}$, $\hat{\sigma}_{X_{BF}^{\bar{\psi}}}$, $\bar{X}_{NF}^{\bar{\psi}}$, $\hat{\sigma}_{X_{NF}^{\bar{\psi}}}$, $\bar{X}_{RS}^{\bar{\psi}}$ and $\hat{\sigma}_{X_{RS}^{\bar{\psi}}}$ can be calculated from (5.102), (5.103), (5.105), (5.106), (5.108), (5.109), (5.111) and (5.112), respectively as values of $X_{FF}^{\bar{\psi}}$, $X_{BF}^{\bar{\psi}}$, $X_{NF}^{\bar{\psi}}$ and $X_{RS}^{\bar{\psi}}$ are articulated in Table 5.12.

FF

$$\bar{X}_{FF}^{\bar{\psi}} = \left(\sum_{PL=1}^{|PlanetLab|} X_{FF,PL}^{\bar{\psi}}\right) \Big/ \left(|PlanetLab|\right) \tag{5.102}$$

$$\hat{\sigma}_{X_{FF}^{\bar{\psi}}} = \sqrt{\frac{\left(\left(\sum_{PL=1}^{|PlanetLab|}\left(\left(X_{FF,PL}^{\bar{\psi}} - \bar{X}_{FF}^{\bar{\psi}}\right)^{2}\right)\right) \Big/ \left(\left(|PlanetLab|\right) - 1\right)\right)}{\left(|PlanetLab|\right)}} \tag{5.103}$$

$$t_{\bar{X}_{FF}^{\bar{\psi}}} = \left(\bar{X}_{FF}^{\bar{\psi}}\right) \Big/ \left(\hat{\sigma}_{X_{FF}^{\bar{\psi}}}\right) \tag{5.104}$$

BF

$$\bar{X}_{BF}^{\bar{\psi}} = \left(\sum_{PL=1}^{|PlanetLab|} X_{BF,PL}^{\bar{\psi}}\right) \Big/ \left(|PlanetLab|\right) \tag{5.105}$$

$$\hat{\sigma}_{X_{BF}^{\bar{\psi}}} = \sqrt{\frac{\left(\left(\sum_{PL=1}^{|PlanetLab|}\left(\left(X_{BF,PL}^{\bar{\psi}} - \bar{X}_{BF}^{\bar{\psi}}\right)^{2}\right)\right) \Big/ \left(\left(|PlanetLab|\right) - 1\right)\right)}{\left(|PlanetLab|\right)}} \tag{5.106}$$

$$t_{\bar{X}_{BF}^{\bar{\psi}}} = \left(\bar{X}_{BF}^{\bar{\psi}}\right) \Big/ \left(\hat{\sigma}_{X_{BF}^{\bar{\psi}}}\right) \tag{5.107}$$

NF

$$\bar{X}_{NF}^{\bar{\psi}} = \left(\sum_{PL=1}^{|PlanetLab|} X_{NF,PL}^{\bar{\psi}}\right) \Big/ \left(|PlanetLab|\right) \tag{5.108}$$

$$\hat{\sigma}_{X_{NF}^{\bar{\psi}}} = \sqrt{\frac{\left(\left(\sum_{PL=1}^{|PlanetLab|}\left(\left(X_{NF,PL}^{\bar{\psi}} - \bar{X}_{NF}^{\bar{\psi}}\right)^{2}\right)\right) \Big/ \left(\left(|PlanetLab|\right) - 1\right)\right)}{\left(|PlanetLab|\right)}} \tag{5.109}$$

$$t_{\bar{X}_{NF}^{\bar{\psi}}} = \left(\bar{X}_{NF}^{\bar{\psi}}\right) \Big/ \left(\hat{\sigma}_{X_{NF}^{\bar{\psi}}}\right) \tag{5.110}$$

RS

$$\bar{X}_{RS}^{\bar{\psi}} = \left(\sum_{PL=1}^{|PlanetLab|} X_{RS,PL}^{\bar{\psi}}\right) \Big/ \left(|PlanetLab|\right) \tag{5.111}$$

$$\hat{\sigma}_{X_{RS}^{\bar{\psi}}} = \sqrt{\frac{\left(\left(\sum_{PL=1}^{|PlanetLab|}\left(\left(X_{RS,PL}^{\bar{\psi}} - \bar{X}_{RS}^{\bar{\psi}}\right)^{2}\right)\right)/\left(\left(|PlanetLab|\right) - 1\right)\right)}{\left(|PlanetLab|\right)}} \tag{5.112}$$

$$t_{\bar{X}_{RS}^{\bar{\psi}}} = \left(\bar{X}_{RS}^{\bar{\psi}}\right)/\left(\hat{\sigma}_{X_{RS}^{\bar{\psi}}}\right) \tag{5.113}$$

Since, we aim to prove that for every single SVMC algorithm, the minimization of mean total number of VM migration, $\bar{\psi}$ by *PRTDVMC* compared to THR-MMT is not a random occurrence and rather statistically significant, therefore, the null hypothesis for $\bar{\psi}$ with a particular SVMC algorithm, $H_0^{\bar{\psi},SV}$ is $\bar{\bar{\psi}}_{SV}^{P} = \bar{\bar{\psi}}_{SV}^{TMMT}$, which we aim to reject. In other words, the null hypothesis is that there is no difference between those two means (i.e., $\bar{\bar{\psi}}_{SV}^{P}$ and $\bar{\bar{\psi}}_{SV}^{TMMT}$), as we aim to prove otherwise. Hence, the alternative hypothesis, $H_1^{\bar{\psi},SV}$ is $\bar{\bar{\psi}}_{SV}^{P} < \bar{\bar{\psi}}_{SV}^{TMMT}$. $\bar{\bar{\psi}}_{SV}^{P}$ and $\bar{\bar{\psi}}_{SV}^{TMMT}$ can be derived utilising (5.81) and (5.82), respectively. Similarly, the null hypothesis for $\bar{\psi}$ with FF, BF, NF and RS algorithms, denoted by $H_0^{\bar{\psi},FF}$, $H_0^{\bar{\psi},BF}$, $H_0^{\bar{\psi},NF}$ and $H_0^{\bar{\psi},RS}$ are presented in (5.114), (5.115), (5.116) and (5.117), respectively.

$$H_0^{\bar{\psi},FF} = \left(\bar{\bar{\psi}}_{FF}^{TMMT} - \bar{\bar{\psi}}_{FF}^{P}\right) = \bar{X}_{FF}^{\bar{\psi}} = 0 \tag{5.114}$$

$$H_0^{\bar{\psi},BF} = \left(\bar{\bar{\psi}}_{BF}^{TMMT} - \bar{\bar{\psi}}_{BF}^{P}\right) = \bar{X}_{BF}^{\bar{\psi}} = 0 \tag{5.115}$$

$$H_0^{\bar{\psi},NF} = \left(\bar{\bar{\psi}}_{NF}^{TMMT} - \bar{\bar{\psi}}_{NF}^{P}\right) = \bar{X}_{NF}^{\bar{\psi}} = 0 \tag{5.116}$$

$$H_0^{\bar{\psi},RS} = \left(\bar{\bar{\psi}}_{RS}^{TMMT} - \bar{\bar{\psi}}_{RS}^{P}\right) = \bar{X}_{RS}^{\bar{\psi}} = 0 \tag{5.117}$$

Consequently, the alternative hypothesis for $\bar{\psi}$ with FF, BF, NF and RS algorithms, denoted by $H_1^{\bar{E},FF}$, $H_1^{\bar{E},BF}$, $H_1^{\bar{E},NF}$ and $H_1^{\bar{E},RS}$, which we are aiming to accept are presented in (5.118), (5.119), (5.120) and (5.121), respectively.

$$H_1^{\bar{\psi},FF} = \bar{\bar{\psi}}_{FF}^{TMMT} > \bar{\bar{\psi}}_{FF}^{P} \tag{5.118}$$

$$H_1^{\bar{\psi},BF} = \bar{\bar{\psi}}_{BF}^{TMMT} > \bar{\bar{\psi}}_{BF}^{P} \tag{5.119}$$

$$H_1^{\bar{\psi},NF} = \bar{\bar{\psi}}_{NF}^{TMMT} > \bar{\bar{\psi}}_{NF}^{P} \tag{5.120}$$

$$H_1^{\bar{\psi},RS} = \bar{\bar{\psi}}_{RS}^{TMMT} > \bar{\bar{\psi}}_{RS}^{P} \tag{5.121}$$

Let, $t_{\bar{X}_{SV}^{\bar{\psi}}}$ denotes the test statistic of the two tail paired *t-test* with $\bar{X}_{SV}^{\bar{\psi}}$, which can be calculated through utilising equations from (5.99) to (5.101). Similarly, test statistics of the two tail paired *t-test* with $\bar{X}_{FF}^{\bar{\psi}}$, $\bar{X}_{BF}^{\bar{\psi}}$, $\bar{X}_{NF}^{\bar{\psi}}$ and $\bar{X}_{RS}^{\bar{\psi}}$ denoted by $t_{\bar{X}_{FF}^{\bar{\psi}}}$, $t_{\bar{X}_{BF}^{\bar{\psi}}}$, $t_{\bar{X}_{NF}^{\bar{\psi}}}$ and $t_{\bar{X}_{RS}^{\bar{\psi}}}$ can be derived through

(5.102) to (5.113). Values of $t_{\bar{X}^{\bar{\psi}}_{FF}}$, $t_{\bar{X}^{\bar{\psi}}_{BF}}$, $t_{\bar{X}^{\bar{\psi}}_{NF}}$ and $t_{\bar{X}^{\bar{\psi}}_{RS}}$ are found as 9.11, 10.34, 8.7 and 10.23, respectively, as corresponding p values are 1.78×10^{-10}, 5.73×10^{-11}, 2.67×10^{-10} and 6×10^{-11}, which are lower than critical value, α as 0.05. Hence, we reject null hypothesis (5.114), (5.115), (5.116) and (5.117), as we accept alternative hypothesis (5.118), (5.119), (5.120) and (5.121). In other words, we have proved that for every single SVMC algorithm, mean total number of VM migration for THR-MMT and *PRTDVMC* are not equal, as in fact, the mean of improvement is statistically significant. Hence, the performance improvement in terms of minimizing mean total number of VM migration by *PRTDVMC* compared to THR-MMT is not a random incident. Nevertheless, no statistical test, including the *t-test* is error free. Furthermore, the mean minimization of VM migration by *PRTDVMC* compared to THR-MMT, which we have obtained through empirical evaluation is not population mean. In the following section we have addressed such issues.

- **Test Error and Confidence Interval (CI)**

Previously, we have discussed on classifications of errors related to the *t-test*. Type I error refers to falsely rejecting null hypothesis while it is actually true and Type II error refers to falsely rejecting alternative null hypothesis while it is actually true. p value refers to the probability of obtaining the test statistic given that the null hypothesis is true. Based on p values, 1.78×10^{-10}, 5.73×10^{-11}, 2.67×10^{-10} and 6×10^{-11} as found in the earlier section, we have rejected null hypothesis, $H_0^{\bar{\psi},FF}$, $H_0^{\bar{\psi},BF}$, $H_0^{\bar{\psi},NF}$ and $H_0^{\bar{\psi},RS}$. Hence, respective total probability of rejecting such null hypotheses as $H_0^{\bar{\psi},FF}$, $H_0^{\bar{\psi},BF}$, $H_0^{\bar{\psi},NF}$ and $H_0^{\bar{\psi},RS}$, even though those were true, aka Type I error are 1.78×10^{-10}, 5.73×10^{-11}, 2.67×10^{-10} and 6×10^{-11}, respectively.

One additional point to note is that the improvement in terms of minimizing mean total number of VM migration by *PRTDVMC* compared to THR-MMT in the presence of FF, BF, NF and RS, denoted by $\bar{X}^{\bar{\psi}}_{FF}$, $\bar{X}^{\bar{\psi}}_{BF}$, $\bar{X}^{\bar{\psi}}_{NF}$ and $\bar{X}^{\bar{\psi}}_{RS}$ are not population mean, rather sample mean. Earlier it has been articulated that the sampling distributions of sample mean, $\bar{\psi}^{TMMT}_{FF}$, $\bar{\psi}^{P}_{FF}$, $\bar{\psi}^{TMMT}_{BF}$, $\bar{\psi}^{P}_{BF}$, $\bar{\psi}^{TMMT}_{NF}$, $\bar{\psi}^{P}_{NF}$, $\bar{\psi}^{TMMT}_{RS}$ and $\bar{\psi}^{P}_{RS}$ follow normal distribution. Since, a distribution generated through subtracting two corresponding elements of two different normal distributions is also a normal distribution, therefore, such distributions as $X^{\bar{\psi}}_{FF}$ (5.75), $X^{\bar{\psi}}_{BF}$ (5.76), $X^{\bar{\psi}}_{NF}$ (5.77) and $X^{\bar{\psi}}_{RS}$ (5.78) also follow normal distribution. Consequently, the probability is 95% that such sample mean as $\bar{X}^{\bar{\psi}}_{FF}$ (5.102), $\bar{X}^{\bar{\psi}}_{BF}$ (5.105), $\bar{X}^{\bar{\psi}}_{NF}$ (5.108) and $\bar{X}^{\bar{\psi}}_{RS}$ (5.111), which we have obtained through our experiment would be within the distance of ± 2 standard deviation from respective population mean, given that the standard deviation is population standard deviation. Hence, we can also make a statement that there is 95% probability that respective population mean would be within ± 2 population standard deviation distance of sample mean $\bar{X}^{\bar{\psi}}_{FF}$ (5.102), $\bar{X}^{\bar{\psi}}_{BF}$ (5.105), $\bar{X}^{\bar{\psi}}_{NF}$ (5.108) and $\bar{X}^{\bar{\psi}}_{RS}$ (5.111). If we add the distance (i.e., ± 2 population standard deviation distance) with the value of mean (i.e., $\bar{X}^{\bar{\psi}}_{FF}$, $\bar{X}^{\bar{\psi}}_{BF}$, $\bar{X}^{\bar{\psi}}_{NF}$ and $\bar{X}^{\bar{\psi}}_{RS}$), a range, aka CI can be obtained and the probability would be 95% for finding population mean within such range.

Previously, we have explained the reason of population standard deviation being unknown and incorporation of SE instead, which is very close approximate of population standard deviation, as showed in (5.65). Similarly, SE for distribution of $X^{\bar{\psi}}_{FF}$, $X^{\bar{\psi}}_{BF}$, $X^{\bar{\psi}}_{NF}$ and $X^{\bar{\psi}}_{RS}$, denoted by $SE_{X^{\bar{\psi}}_{FF}}$, $SE_{X^{\bar{\psi}}_{BF}}$, $SE_{X^{\bar{\psi}}_{NF}}$ and $SE_{X^{\bar{\psi}}_{RS}}$ can be calculated from (5.122), (5.123), (5.124) and (5.125), respectively.

$$SE_{X^{\bar{\psi}}_{FF}} = \left(\hat{\sigma}_{X^{\bar{\psi}}_{FF}}\right)/\left(\sqrt{|PlanetLab|}\right) \tag{5.122}$$

$$SE_{X^{\bar{\psi}}_{BF}} = \left(\hat{\sigma}_{X^{\bar{\psi}}_{BF}}\right)/\left(\sqrt{|PlanetLab|}\right) \tag{5.123}$$

$$SE_{X^{\bar{\psi}}_{NF}} = \left(\hat{\sigma}_{X^{\bar{\psi}}_{NF}}\right)/\left(\sqrt{|PlanetLab|}\right) \tag{5.124}$$

$$SE_{X^{\bar{\psi}}_{RS}} = \left(\hat{\sigma}_{X^{\bar{\psi}}_{RS}}\right)/\left(\sqrt{|PlanetLab|}\right) \tag{5.125}$$

One noteworthy point is that such distributions as $\bar{X}^{\bar{\psi}}_{FF}$, $\bar{X}^{\bar{\psi}}_{BF}$, $\bar{X}^{\bar{\psi}}_{NF}$ and $\bar{X}^{\bar{\psi}}_{RS}$ are not perfect normal distributions, rather *t* distribution. Hence, instead of ± 2 population standard deviation distance, it would be more accurate to state that there is 95% probability for respective population mean to be found within $(\pm t_{CI} \cdot SE)$ distance of such mean as $\bar{X}^{\bar{\psi}}_{FF}$, $\bar{X}^{\bar{\psi}}_{BF}$, $\bar{X}^{\bar{\psi}}_{NF}$ and $\bar{X}^{\bar{\psi}}_{RS}$. In other words, we are 95% confident that population mean would be within the range $\left[\bar{X}^{\bar{\psi}}_{SV} + \left(t_{CI} \cdot SE_{X^{\bar{\psi}}_{SV}}\right),\ \bar{X}^{\bar{\psi}}_{SV} - \left(t_{CI} \cdot SE_{X^{\bar{\psi}}_{SV}}\right)\right]$, aka 95% CI for $\bar{X}^{\bar{\psi}}_{SV}$. Similarly, CI for $\bar{X}^{\bar{\psi}}_{FF}$, $\bar{X}^{\bar{\psi}}_{BF}$, $\bar{X}^{\bar{\psi}}_{NF}$ and $\bar{X}^{\bar{\psi}}_{RS}$, denoted by $CI_{\bar{X}^{\bar{\psi}}_{FF}}$, $CI_{\bar{X}^{\bar{\psi}}_{BF}}$, $CI_{\bar{X}^{\bar{\psi}}_{NF}}$ and $CI_{\bar{X}^{\bar{\psi}}_{RS}}$ can be estimated from (5.126), (5.127), (5.128) and (5.129), respectively. Values of t_{CI} are available in [148]. Fig. 5.7 presents values of $CI_{\bar{X}^{\bar{\psi}}_{FF}}$, $CI_{\bar{X}^{\bar{\psi}}_{BF}}$, $CI_{\bar{X}^{\bar{\psi}}_{NF}}$ and $CI_{\bar{X}^{\bar{\psi}}_{RS}}$.

$$CI_{\bar{X}^{\bar{\psi}}_{FF}} = \bar{X}^{\bar{\psi}}_{FF} \pm \left(t_{CI} \cdot SE_{X^{\bar{\psi}}_{FF}}\right) \tag{5.126}$$

$$CI_{\bar{X}^{\bar{\psi}}_{BF}} = \bar{X}^{\bar{\psi}}_{BF} \pm \left(t_{CI} \cdot SE_{X^{\bar{\psi}}_{BF}}\right) \tag{5.127}$$

$$CI_{\bar{X}^{\bar{\psi}}_{NF}} = \bar{X}^{\bar{\psi}}_{NF} \pm \left(t_{CI} \cdot SE_{X^{\bar{\psi}}_{NF}}\right) \tag{5.128}$$

$$CI_{\bar{X}^{\bar{\psi}}_{RS}} = \bar{X}^{\bar{\psi}}_{RS} \pm \left(t_{CI} \cdot SE_{X^{\bar{\psi}}_{RS}}\right) \tag{5.129}$$

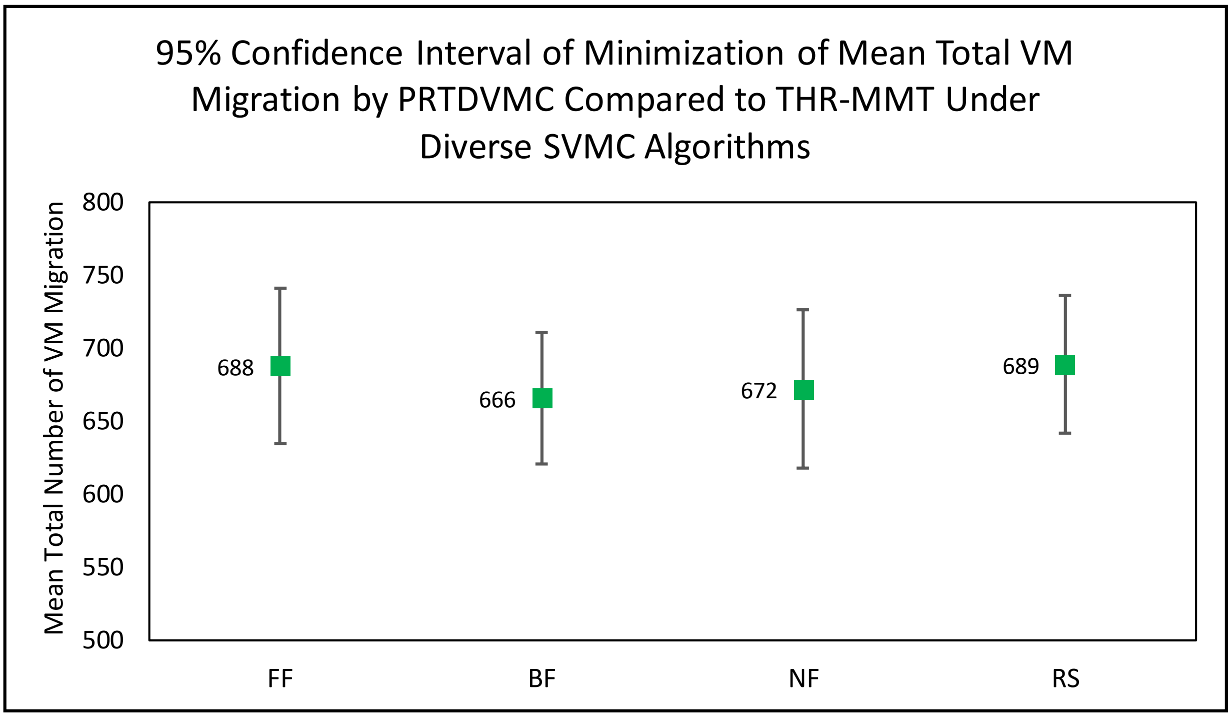

Fig. 5.7 Confidence Interval of Mean Minimization of Total Number of VM Migration by *PRTDVMC* Compared to THR-MMT Under Diverse SVMC Algorithms

5.8 Result Analysis

Experimental results reveal several critical aspects as discussed in the following.

Observation 1: From empirical evaluation as portrayed in Fig. 5.4 and Fig. 5.7, we can observe that *PRTDVMC* edges out THR-MMT in terms of minimizing mean CDC energy consumption and mean total number of VM migration alike. The differences between two algorithms are that the former one takes heterogeneous *VMRT* and highly energy proportional PMs into consideration during VM consolidation decision process, while the later one does not. THR-MMT aims to minimize energy consumption by consolidating VMs in as few PMs as possible, whereas *PRTDVMC* not only consolidates VMs in lesser number of PMs, but also reduces *PMRT* of PMs. Reduction of both elements – active duration of a PM and total number of active PMs is more effective for minimizing energy consumption than reduction of only one element - total number of active PMs.

Observation 2: From Fig. 5.2 and Fig. 5.3, we can note that, 35 out of 40 times of performance comparisons between *PRTDVMC* and THR-MMT, the former one has been found as superior in terms of mean CDC energy consumption. In one such cases as PlanetLab 22 March, THR-MMT outperformed *PRTDVMC* with all SVMC algorithms and in one such case as PlanetLab 3 April, THR-MMT outperformed *PRTDVMC* with RS. This observation indicates that with such NP-hard problem as DVMC, no algorithm, including *PRTDVMC* can guarantee that it would always find optimal solution for any input set in every possible scenario compared to rest of the algorithms. As such, in practise, a target algorithm is run multiple times with different days of data and then performance is compared with that of rest of the algorithms. From our experimental results delineated in Fig. 5.4, considering performance under different days of workload, *PRTDVMC* minimizes mean of mean CDC energy consumption by a minimum of 0.5595 kW (i.e., 9.4%) with RS and a maximum of 0.5815 kW (i.e., 10%) with NF compared to THR-MMT.

Observation 3: From Fig. 5.5 and Fig. 5.6, we can note that, 40 out of 40 times of performance comparisons between *PRTDVMC* and THR-MMT, the former one has been found as superior in terms of mean total number of VM migration. Consideration of state-of-the-art highly energy proportional PMs energy-efficiency characteristics into VM consolidation decision process has resulted into such improved performance by *PRTDVMC*.

Observation 4: Results of the *t-tests* prove that the improved performance by *PRTDVMC* compared to THR-MMT is statistically significant. Furthermore, as articulated in Fig. 5.4 and Fig. 5.7 superiority of *PRTDVMC* to THR-MMT has been found as true in terms of population mean.

Observation 5: From we can observe that CI is higher for RS compared to the rest of the greedy heuristic approaches. Two factors are associated with causing variations in CI - number of samples and standard deviation. Number of samples was same for all the methods. However, contrast to greedy heuristic based approaches, the random nature of RS in choosing initial destination PM has brought forth more disperse in standard deviation resulting into a bigger CI.

Observation 6: Performance of a DVMC algorithm may change with the change in SVMC algorithm with which it is coupled together. Both *PRTDVMC* and THR-MMT are extensively examined with four diverse SVMC algorithm under diverse workload scenarios. From our study as illustrated in Fig. 5.4 and Fig. 5.7, we have found that *PRTDVMC* has outperformed THR-MMT with every SVMC algorithm.

The summary of this chapter is articulated in the following section.

5.9 Summary

Major drawback with the release time based DVMC algorithms is that such algorithms assume IAAS users/PAAS providers would either be aware or be able to estimate VMRT and would provide such information to CDC owner/IAAS provider. Hence, the overall system become non-automated, as CDC owners/IAAS providers cannot apply these algorithms without receiving VMRT information of all VMs from IAAS users/PAAS providers. Furthermore, estimation of VMRT by IAAS users begets additional investment burden on IAAS users to acquire skills and facilities in order to manage and operate large volume of past usage data. To mitigate such burden, IAAS providers would need to provide incentives to IAAS users, which would increase business operational cost and would also reduce profit margin.

To eradicate these caveats, for *PRTDVMC*, instead of receiving VMRT from IAAS users, we have embodied LR and RLR based two different VMRT predictor models to be directly used by IAAS providers to generate *PVMRT* through utilizing users' past usage records. Since, as part of billing process, IAAS providers always need to manage and operate users' usage records, hence, obtaining *PVMRT* to be used in *PRTDVMC* through LR and RLR based VMRT predictor models do not impose any unprecedented additional cost for IAAS users. Moreover, advantage of LR and RLR based VMRT predictor models is that no data analyst is required to generate global function that fit a model to the data of a user [142]. Hence, *PRTDVMC* is an automated cost-effective release time based DVMC algorithm, since dependency on IAAS users and requirement of associated incentives as well as spending after data analysts have been eliminated. To the best of our knowledge, *PRTDVMC* is pioneer in introducing automated release time based DVMC algorithm.

We have simulated performance of *PRTDVMC* in combination with diverse SVMC algorithms under real Cloud based heterogeneous workload and compared that with a notable existing DVMC algorithm, namely THR-MMT. Experimental results show the superiority of *PRTDVMC* to THR-MMT under every SVMC algorithm in terms of both mean CDC energy consumption and mean total number of VM migration. Improvement of performance has also been found as statistically significant. In the next chapter, we have presented the overall conclusion of the dissertation.

6 Conclusions and Future Directions

The motivations and objectives of the thesis have been outlined in Chapter 1. To meet the objectives, we have proposed three diverse DVMC algorithms in Chapter 3, Chapter 4 and Chapter 5 that exploits predicted VMRT and VMRT information as received from CSUs. The characteristics of the proposed DVMC algorithms are validated through simulation results. The contributions of this thesis are summarised in Section 6.1 and possible future works are articulated in Section 6.2.

6.1 Conclusions

Energy-efficiency is one of the most important performance metrics for a CDC. DVMC proffers reduced energy consumption through migrating out VMs, which are scattered across multiple under-utilized PMs and placing those VMs into fewest possible PMs. A major challenge for researchers is to design a DVMC algorithm that simultaneously meets two requirements – reduction of energy consumption and regulation of VM migration.

To address this challenge, researchers have proposed a wide range of DVMC algorithms in the literature. Our extensive literature study presented in Chapter 2 shows that while utilization of CSU provided information is prevalent in many aspects of Cloud resource management system, such as resource scheduling, resource reservation and reservation-based pricing scheme, its application in regulating CDC energy consumption is a novel research direction. The challenges herein are twofold. First, finding the right benign information to be received from a CSU, which can complement the energy-efficiency of CDC. Second, smart application of such information to significantly reduce the energy consumption of CDC. In Chapter 3 of this dissertation, we have undertaken a research to overcome these challenges. Outcome of our research is the addition of a DVMC algorithm in the literature, namely *RTDVMC*, first of its kind, to be used for minimization of CDC energy consumption via utilizing CSU provided information. The information to be received from a CSU is VMRT – the time in future when a VM would be released. Generally, a better decision can be made, if future information can be known in advance. Cloud is a multi-tenant environment. VMs are assigned with different workload resulting into heterogeneous VMRT. Existing DVMC algorithms assume that VMRT of all VMs are homogeneous and hence, are not developed considering heterogeneous VMRT. In contrast, *RTDVMC* takes VM consolidation decision based on the prior received VMRT information from the owner of the VM itself. Therefore, in real Cloud scenario with heterogeneous VMRT, *RTDVMC* excels in performance compared to existing literature.

Simulation models for both *RTDVMC* and existing promising STDVMC and ATDVMC algorithms have been developed using a Cloud oriented discrete event simulation software, namely, CloudSim under different days of real Cloud workload, namely PlanetLab. It is evident from the results that our proposed algorithm significantly outperforms alternative algorithms in terms of minimizing mean CDC energy consumption. Chapter 3 concludes with two key findings. First, our experimental outcome show that consideration of VMRT in VM consolidation decision process can reduce the mean CDC energy consumption. Second, from our empirical evaluation, STDVMC algorithms are found as more energy-efficient than ATDVMC algorithms.

Existing DVMC algorithms attempt to consolidate maximum possible number of VMs in minimum possible number of PMs, based on the underlying assumption that maximum energy-efficiency is achievable with maximum load on PMs. However, state-of-the-art highly energy proportional PMs being on the horizon, such assumption has become invalid. Studies show that with state-of-the-art PMs, energy-efficiency rather drops beyond 70% load level. Furthermore, the increased energy-efficiency through *RTDVMC* comes at a cost of increased VM migration. The harm of excessive VM migration is huge. Increased VM migration causes increase of network traffic resulting into many overheads, such as degraded QoS, also interpreted as SLA violation, increased energy consumption by networking equipment, additional operational cost and so forth. It is critical to understand that concomitant minimization of energy consumption through DVMC and VM migration are confronting objectives, since VM consolidation is impossible without VM migration. In addition, there is a probability that such CSU provided information as VMRT would turn out as inaccurate. Inaccurate VMRT would negatively affect the overall performance. *RTDVMC* is not designed to withstand inaccurate VMRT. We have further extended our research to Chapter 4 to address these research challenges.

Taking all the findings of Chapter 3 into consideration, a novel multi-objective STDVMC algorithm, namely *SRTDVMC* has been proposed in Chapter 4. Both algorithms - *RTDVMC* and *SRTDVMC* are STDVMC algorithms and consider heterogeneous VMRT, and hence, are more energy-efficient. The key differences between *RTDVMC* and *SRTDVMC* are twofold. First, *SRTDVMC* assumes that VMRT information received from CSU might not be completely accurate and hence, adopts the stochastic VMRT approach instead of original CSU provided VMRT as applied in *RTDVMC*. Second, *SRTDVMC* only migrates VMs, if the benefit is greater than the associated cost, while *RTDVMC* does not consider the cost of VM migration. *RTDVMC* keeps migrating VMs based on the principle that the higher is the number of VMs consolidated in a single PM, the better. However, such assumption is not true for state-of-the-art PMs. *SRTDVMC* on the other hand, considers the difference between potential drop of energy consumption of the source PM and potential rise of energy consumption of the source PM. If the drop of energy consumption is greater than the rise of energy consumption, only then the VM is migrated. Thus, *SRTDVMC* solves two problems at the same time – first, unlike existing DVMC algorithms, it does not lose energy-efficiency in the presence of state-of-the-art PMs and second, number of VM migration is also restricted. Both algorithms have been simulated with real Cloud based PlanetLab workload coupled with heterogeneous VMRT distribution drawn from real Cloud, namely Nectar Cloud.

Experimental results have unveiled that *SRTDVMC* is superior to *RTDVMC* in terms of minimizing both mean CDC energy consumption and mean total number of VM migration. The chapter is concluded with three key findings. First, our simulation results show that VMRT impacts energy consumption and VM migration. Bases on that results, considering the presence of heterogeneous VMRT in DVMC algorithm has been suggested. Second, such working principal of existing DVMC algorithms that maximum energy-efficiency is achievable at maximum load on PM is found as false for state-of-the-art PMs, resulting into performance inefficiencies. Third, our experimental results show that prior VM migration, if corresponding cost and benefit are considered, then concomitant optimization of both aspects - reduction of energy consumption and VM migration can be attained.

The key assumption with *RTDVMC* and *SRTDVMC* algorithms is that CSUs would be aware of VMRT and would provide that information to CDC owner/IAAS provider. Nonetheless, in real Cloud, such users as pay-as-you-go users might not always be able to provide VMRT information in advance to IAAS providers. Furthermore, in order to generate VMRT, PAAS providers/IAAS users would need to estimate potential resource demand. To generate potential resource demand, large volume of records related to previous time-variant resource demand would be needed. Many PAAS providers do not have the skill to maintain and operate large database and unable to afford the associated cost. In such case, IAAS providers/CDC owners would need to provide financial incentives to PAAS providers, which increases business operational cost.

To eliminate the burden of IAAS users to generate VMRT, in Chapter 5, we have proposed a novel release time based and highly energy proportional PMs aware DVMC algorithm, namely *PRTDVMC*, which uses *PVMRT*. Instead of receiving VMRT from IAAS users, two diverse regression-based predictor models, namely LR and RLR have been proposed to be directly used by IAAS provider/CDC owner to generate *PVMRT* through utilizing users' past usage records. It is important to mention that as part of billing process, IAAS providers always need to manage and operate users' usage records. Hence, obtaining *PVMRT* to be used in *PRTDVMC* through LR and RLR based VMRT predictor models do not impose any additional cost for IAAS providers. An additional advantage with LR and RLR models is that no data analyst is required to generate global function that fit a model to the data of a user. Therefore, the spending after data analyst is also avoided. Nonetheless, *PRTDVMC* is not limited to LR and RLR models, as it is a *PVMRT* based generic DVMC algorithm, since any model to generate *PVMRT* can be incorporated in *PRTDVMC*.

Compared to *RTDVMC* and *SRTDVMC*, which IAAS provider/CDC owner cannot apply in CDC before receiving VMRT information from CSUs, *PRTDVMC* is a completely automated solution, since dependency of IAAS provider on IAAS users in obtaining VMRT information has been removed. We hence classify *PRTDVMC* as automated release time based DVMC technique and *RTDVMC* and *SRTDVMC* as non-automated release time based DVMC technique. Performance of a DVMC algorithm changes with the change in SVMC algorithm. Therefore, *PRTDVMC* is simulated under diverse SVMC algorithms with real Cloud based heterogeneous workload and compared with a notable existing DVMC algorithm, namely THR-MMT. Experimental results illustrate the superiority of *PRTDVMC* to THR-MMT under every SVMC algorithm in terms of both mean CDC energy consumption and mean total number of VM migration. There is scope for further research to enhance the performance of our proposed DVMC algorithms as well as energy-efficient management of CDC. These are presented in the following section.

6.2 Future Directions

This thesis primarily explored and evaluated various ideas for energy-efficiency in Cloud computing though modelling and simulation. In addition, some of the proposed techniques need to better manage QoS parameters and resource utilization threshold settings. To overcome these limitations, we proposed several new research directions as discussed below:

- **Embodiment of Advanced Machine Learning Techniques**

Accurate estimation of VMRT information plays an important role in minimizing energy consumption through release time based DVMC algorithm. To explain further, if input VMRT value (i.e., VMRT value given as input in the system) is greater/lower than the true VMRT value (i.e., the authentic time when the VM would truly be removed from CDC), then such decisions as source PM selection, migrating VMs selection and destination PM selection taken on the basis of VMRT value would also be inaccurate, resulting into inefficient performance. Diverse research pathways can be examined in this regard. Advanced machine learning based techniques, neural networks and so forth can be embodied in automated release time based DVMC algorithms to generate *PVMRT*. Each technique would consist its own set of advantages and disadvantages. It would be an interesting research problem to measure the change in system performance with the change in accuracy of input VMRT.

- **Release Time based ATDVMC Algorithm**

VM consolidation minimizes energy consumption through hosting multiple VMs in a PM. Nonetheless, as the number of VMs hosted in a PM grows, resource contention may arise, resulting into poor QoS. Therefore, VM consolidation comes with a trade-off between energy consumption minimization through resource over-subscription and subsequent QoS degradation. ATDVMC algorithm monitors the change in QoS degradation. With the rise/drop of QoS degradation, the maximum resource utilization threshold based on which a PM is marked as *O-UPM* is dropped/raised. If the maximum resource utilization threshold value is set low, then resource contention is minimized, resulting into improved QoS. Nonetheless, a low threshold value causes resource under-utilization, leading towards declined energy-efficiency. Future studies can be undertaken on embodiment of ATDVMC in release time DVMC algorithm to regulate QoS degradation due to resource over-subscription.

- **Release Time Based Predictive DVMC Algorithm**

Non-predictive DVMC algorithm consolidates VMs based on present resource utilization. Predictive DVMC algorithm, on the other hand, estimates predicted resource utilization (i.e., potential resource utilization in future) and consolidates VMs based on predicted resource utilization. Benefit of predictive DVMC algorithm is improved QoS. Performance of predictive DVMC algorithm depends upon accurate estimation of predicted resource utilization. A wide range of prediction methods are available in the literature to generate predicted resource utilization. Release time based predictive DVMC algorithm can be explored in future to concomitantly minimize QoS degradation and energy consumption.

- **Application of Meta-Heuristics in Release Time Based DVMC Algorithm**

  Greedy heuristics based DVMC algorithms are more widespread in the literature. Our proposed release time based DVMC algorithms are greedy heuristic based approaches. However, greedy heuristic may fall into local minima/local maxima because of narrowing its search space. Meta-heuristics avoids local minima/local maxima, explore much wider search space, which might lead to a better solution. Researchers have proposed diverse meta-heuristics, such as Ant Colony Optimization, Evolutionary Algorithm, Simulated Annealing and so forth. Determining the degree of variation in system efficiency for variation in meta-heuristics would be an interesting research problem to explore in future.

- **Data Center Cooling Aware Release Time based DVMC Algorithm for Geo-distributed Cloud**

  In this dissertation, we have focused on DVMC algorithm for energy-efficient management of CDC that minimizes number of active PMs through maximizing resource utilization of active PMs. It is important to note that heat dissipation of a PM increases with the increase in resource utilization. Because of consolidating more VMs in fewer PMs, resource utilization of those PMs hosting more VMs go high. Such PMs in which VMs are consolidated then become extra hot, which may create hotspots in CDC. A significant portion of total energy usage in CDC is spent after CDC cooling. Hotspots in CDC increases the energy spending of CDC for cooling purpose. Therefore, minimizing energy spending after PMs of CDC through DVMC and minimizing energy spending after CDC cooling through reducing hotspots are confronting objectives. Future studies can be undertaken in designing hotspot aware release time based DVMC algorithm.

- **Network Energy Aware Release Time based DVMC Algorithm for Geo-distributed Cloud**

  Our research focus was on minimizing server related energy through release time based DVMC algorithm. Nevertheless, consideration of network related energy is imperative in order to achieve energy-efficient CDC. Furthermore, with the growing popularity of Cloud, many CSPs, such as Nectar Cloud have geo-distributed CDCs. Research can be undertaken in future to design network energy aware release time based DVMC algorithm for geo-distributed Cloud.

- **Application of Proposed DVMC Algorithms in Fog/Edge Computing**

  To reduced latency of services for Internet of Things (IoT) applications, Fog/Edge computing model deploys micro-Data Centres near the edge of networks [149]. Techniques proposed in this thesis can be extended to these micro-Data Centres so that services can be delivered with minimal energy.

- **Experimental Evaluations under Real Testbed**

  As part of future research, various techniques proposed in this thesis can be practically implemented in real world software systems such as OpenStack Neat [150] for energy-efficient management of Data Centres in Cloud Computing environments.

We conclude that with rapid adoption of Cloud computing in hosting a wide variety of applications and its ubiquitous usage, the need energy-efficient use of Cloud infrastructure will become even more critical. Hence, it will attract many new research investigations and investments.

References

[1] L. Kleinrock, “A vision for the Internet,” *ST Journal of Research,* vol. 2, no. 1, pp. 4-5, 2005.

[2] R. Buyya, C. S. Yeo, S. Venugopal, J. Broberg, and I. Brandic, “Cloud computing and emerging IT platforms: Vision, hype, and reality for delivering computing as the 5th utility,” *Future Generation computer systems,* vol. 25, no. 6, pp. 599-616, 2009.

[3] J. M. Kaplan, W. Forrest, and N. Kindler, *Revolutionizing data center energy efficiency*, Technical report, McKinsey & Company, 2008.

[4] A. Beloglazov, “Energy-efficient management of virtual machines in data centers for cloud computing,” dissertation, Department of Computing and Information Systems, The University of Melbourne, Melbourne, AU, 2013.

[5] J. Koomey, “Growth in data center electricity use 2005 to 2010,” *A report by Analytical Press, completed at the request of The New York Times,* vol. 9, 2011.

[6] I. Gartner Estimates, “Industry Accounts for 2 Percent of Global CO2 Emissions,” *press release*, 2007.

[7] J. G. Koomey. "Estimating total power consumption by servers in the US and the world". Researchgate.net. https://www.researchgate.net/publication/228365136_Estimating_Total_Power_Consumption_by_Servers_in_the_US_and_the_World. (Accessed Oct. 6, 2020).

[8] R. Buyya, A. Beloglazov, and J. Abawajy, “Energy-efficient management of data center resources for cloud computing: A vision, architectural elements, and open challenges,” *arXiv preprint arXiv:1006.0308,* 2010.

[9] Data driven. Advancing a sustainable future (Mar. 20, 2018). Accessed Oct. 6, 2020. [Online Video]. Available: https://www.youtube.com/watch?v=vqKWMfB_lJ0.

[10] M. Smolaks. "Google data center chief: go green because your customers care." datacenterdynamics.com. http://www.datacenterdynamics.com/content-tracks/design-build/google-data-center-chief-go-green-because-your-customerscare/91942.fullarticle. (Accessed Oct. 6, 2020).

[11] "3M: The future of data centers will depend on cooling technologies." datacenterdynamics.com. https://www.datacenterdynamics.com/en/news/3m-the-future-of-data-centers-will-depend-on-cooling-technologies/. (Accessed Oct. 6, 2020).

[12] A. Shehabi, S. J. Smith, N. Horner, I. Azevedo, R. Brown, J. Koomey, E. Masanet, D. Sartor, M. Herrlin, and W. Linter, *Uniter States Data Center Energy Usage Report.*, Lawrence Berkley National Laboratory, California, 2016.

[13] N. Backbill. "The Energy Of The Cloud," http://large.stanford.edu/courses/2016/ph240/brackbill2/. (Accessed Oct. 6, 2020).

[14] Q. Hardy. "Google Says It Will Run Entirely on Renewable Energy in 2017." nytimes.com. https://www.nytimes.com/2016/12/06/technology/google-says-it-will-run-entirely-on-renewable-energy-in-2017.html. (Accessed Oct. 6, 2020).

[15] M. Avgerinou, P. Bertoldi, and L. Castellazzi, “Trends in Data Centre Energy Consumption under the European Code of Conduct for Data Centre Energy Efficiency,” *Energies,* vol. 10, no. 10, pp. 1470, 2017.

[16] "Code of Conduct for Energy Efficiency in Data Centres." ec.europa.eu. https://ec.europa.eu/jrc/en/energy-efficiency/code-conduct/datacentres. (Accessed Oct. 6, 2020).

[17] A. Mark, B. Paolo, B. John, N. Liam, R. Andre, and T. Robert, *2018 Best Practice Guidelines for the EU Code of Conduct on Data Centre Energy Efficiency*, 2018.

[18] M. Dabbagh, B. Hamdaoui, M. Guizani, and A. Rayes, “Toward energy-efficient cloud computing: Prediction, consolidation, and overcommitment,” *IEEE Network*, vol. 29, no. 2, pp. 56-61, 2015.

[19] L. A. Barroso, and U. Hölzle, “The case for energy-proportional computing,” *IEEE Computer*, vol. 40, 2007.

[20] A. Vashistha and P. Verma, "A Literature Review and Taxonomy on Workload Prediction in Cloud Data Center" in *2020 10th Int. Conf. Confluence*, Noida India, 2020, pp. 415-420, doi: 10.1109/Confluence47617.2020.9057938.

[21] F. Qiu, B. Zhang and J. Guo, "A deep learning approach for VM workload prediction in the cloud" in *2016 17th IEEE/ACIS Int. Conf. SNPD*, Shanghai China, 2016, pp. 319-324, doi: 10.1109/SNPD.2016.7515919.

[22] J. Kumar, and A. K. Singh, “Workload prediction in cloud using artificial neural network and adaptive differential evolution,” *Future Generation Computer Systems*, vol. 81, pp. 41-52, 2018.

[23] M. Dabbagh, B. Hamdaoui, M. Guizani and A. Rayes, "Efficient datacenter resource utilization through cloud resource overcommitment" in *2015 IEEE Conf. INFOCOM WKSHPS*, Hong Kong, 2015, pp. 330-335, doi: 10.1109/INFCOMW.2015.7179406.

[24] R. Ghosh and V. K. Naik, "Biting Off Safely More Than You Can Chew: Predictive Analytics for Resource Over-Commit in IaaS Cloud" in *2012 IEEE 5th Int. Conf. Cloud*, Honolulu HI USA, 2012, pp. 25-32, doi: 10.1109/CLOUD.2012.131.

[25] E. Pinheiro, R. Bianchini, E. V. Carrera, and T. Heath, "Load balancing and unbalancing for power and performance in cluster-based systems" in *Workshop COLP’01*, Barcelona, Spain, 2001, pp. 182-195. Available: https://doi.org/doi:10.7282/t3-agfw-yt73.

[26] "Migration with vMotion." vmware.com. https://pubs.vmware.com/vsphere-60/index.jsp?topic=%2Fcom.vmware.vsphere.vcenterhost.doc%2FGUID-D19EA1CB-5222-49F9-A002-4F8692B92D63.html&resultof=%22vmotion%22%20. (Accessed Oct. 6, 2020).

[27] R. N. Calheiros, R. Ranjan, A. Beloglazov, C. A. De Rose, and R. Buyya, “CloudSim: a toolkit for modeling and simulation of cloud computing environments and evaluation of resource provisioning algorithms,” *Software: Practice and experience*, vol. 41, no. 1, pp. 23-50, 2011.

[28] "NECTAR CLOUD." nectar.org.au. https://nectar.org.au/research-cloud/. (Accessed Oct. 6, 2020).

[29] K. Park, and V. S. Pai, “CoMon: a mostly-scalable monitoring system for PlanetLab,” *ACM SIGOPS Operating Systems Review*, vol. 40, no. 1, pp. 65-74, 2006.

[30] J. Hopkin. "VMware ESX Server [Image on internet]." Accessed 02/09/2016; http://ostatic.com/vmware-esx-server/screenshot/1.

[31] Z. Á. Mann, “Allocation of virtual machines in cloud data centers—a survey of problem models and optimization algorithms,” *ACM Computing Surveys (CSUR)*, vol. 48, no. 1, pp. 11, 2015.

[32] I. Pietri, and R. Sakellariou, “Mapping virtual machines onto physical machines in cloud computing: A survey,” *ACM Computing Surveys (CSUR)*, vol. 49, no. 3, pp. 49, 2016.

[33] S. Kaur, and S. Bawa, "A review on energy aware VM placement and consolidation techniques" in *2016 ICICT*, Coimbatore India, 2016, pp. 1-7, doi: 10.1109/INVENTIVE.2016.7830219.

[34] E. S. Madhan, and S. Srinivasan, "Energy aware data center using dynamic consolidation techniques: A survey" in *Proc. IEEE ICCCS14,* Chennai India, 2014, pp. 043-045, doi: 10.1109/ICCCS.2014.7068165.

[35] F. L. Pires, and B. Barán, "A Virtual Machine Placement Taxonomy" in *2015 15th IEEE/ACM Int. Symp. CCGRID,* Shenzhen China, 2015, pp. 159-168, doi: 10.1109/CCGrid.2015.15.

[36] R. Ranjana, and J. Raja, "A survey on power aware virtual machine placement strategies in a cloud data center." *Journal of Network and Computer Applications,* vol. 66, pp. 106-127, 2016.

[37] A. Varasteh, and M. Goudarzi, “Server consolidation techniques in virtualized data centers: A survey,” *IEEE Systems Journal,* vol. 11, no. 2, pp. 772-783, 2015.

[38] R. W. Ahmad, A. Gani, S. H. A. Hamid, M. Shiraz, A. Yousafzai, and F. Xia, “A survey on virtual machine migration and server consolidation frameworks for cloud data centers,” *Journal of Network and Computer Applications,* vol. 52, pp. 11-25, 2015.

[39] A. Choudhary, S. Rana, and K. J. Matahai, “A Critical Analysis of Energy Efficient Virtual Machine Placement Techniques and its Optimization in a Cloud Computing Environment,” *Procedia Computer Science,* vol. 78, pp. 132-138, 2016/01/01, 2016.

[40] M. Masdari, S. S. Nabavi, and V. Ahmadi, “An overview of virtual machine placement schemes in cloud computing,” *Journal of Network and Computer Applications,* vol. 66, pp. 106-127, 2016.

[41] Z. Usmani, and S. Singh, “A Survey of Virtual Machine Placement Techniques in a Cloud Data Center,” *Procedia Computer Science,* vol. 78, pp. 491-498, 2016/01/01, 2016.

[42] M. H. Ferdaus, and M. Murshed, "Energy-aware virtual machine consolidation in IaaS cloud computing," *Cloud Computing,* pp. 179-208: Springer, 2014.

[43] L. Chen, H. Shen, and S. Platt, "Cache contention aware Virtual Machine placement and migration in cloud datacenters" in *2016 IEEE 24th ICNP,* Singapore, 2016, pp. 1-10, doi: 10.1109/ICNP.2016.7784447.

[44] L. C. Jersak, and T. Ferreto, "Performance-aware server consolidation with adjustable interference levels" in *Proc. 31st Annu. ACM SAC’16,* Pisa Italy, 2016, pp. 420-425. Available: https://dl.acm.org/doi/10.1145/2851613.2851625.

[45] A. Beloglazov, and R. Buyya, “Optimal online deterministic algorithms and adaptive heuristics for energy and performance efficient dynamic consolidation of virtual machines in cloud data centers,” *Concurrency and Computation: Practice and Experience,* vol. 24, no. 13, pp. 1397-1420, 2012.

[46] F. Ahamed, S. Shahrestani, and B. Javadi, "Security Aware and Energy-Efficient Virtual Machine Consolidation in Cloud Computing Systems" in *2016 IEEE Trustcom,* Tianjin China, 2016, pp. 1516-1523, doi: 10.1109/TrustCom.2016.0236.

[47] D. Deng, K. He, and Y. Chen, "Dynamic virtual machine consolidation for improving energy efficiency in cloud data centers" in *2016 4th Int. Conf. CCIS,* Beijing China, 2016, pp. 366-370, doi: 10.1109/CCIS.2016.7790285.

[48] G. B. Fioccola, P. Donadio, R. Canonico, and G. Ventre, "Dynamic Routing and Virtual Machine Consolidation in Green Clouds" in *2016 IEEE Int. Conf. CloudCom,* Luxembourg City Luxembourg, 2016, pp. 590-595, doi: 10.1109/CloudCom.2016.0102.

[49] S. S. Masoumzadeh, and H. Hlavacs, “A Gossip-Based Dynamic Virtual Machine Consolidation Strategy for Large-Scale Cloud Data Centers” in *Proc. 3rd Int. Workshop*

ARMS-CC'16, Chicago IL USA, 2016, pp. 28-34. Available: https://dl.acm.org/doi/abs/10.1145/2962564.2962565.

[50] M. Khelghatdoust, V. Gramoli, and D. Sun, "GLAP: Distributed Dynamic Workload Consolidation through Gossip-Based Learning" in *2016 IEEE Int. Conf. CLUSTER*, Taipei Taiwan, 2016, pp. 80-89, doi: 10.1109/CLUSTER.2016.24.

[51] A. Beloglazov, J. Abawajy, and R. Buyya, "Energy-aware resource allocation heuristics for efficient management of data centers for cloud computing," *Future generation computer systems,* vol. 28, no. 5, pp. 755-768, 2012.

[52] F. Farahnakian, A. Ashraf, T. Pahikkala, P. Liljeberg, J. Plosila, I. Porres, and H. Tenhunen, "Using ant colony system to consolidate vms for green cloud computing," *IEEE Transactions on Services Computing,* vol. 8, no. 2, pp. 187-198, 2015.

[53] F. Farahnakian, R. Bahsoon, P. Liljeberg, and T. Pahikkala, "Self-Adaptive Resource Management System in IaaS Clouds" in *2016 IEEE 9th Int. Conf. CLOUD,* San Francisco CA USA, 2016, pp. 553-560, doi: 10.1109/CLOUD.2016.0079.

[54] F. Farahnakian, P. Liljeberg, and J. Plosila, "LiRCUP: Linear regression based CPU usage prediction algorithm for live migration of virtual machines in data centers" in *2013 39th Euromicro Conf. SEAA*, Santander Spain, 2013, pp. 357-364, doi: 10.1109/SEAA.2013.23.

[55] G. E. I. Selim, M. A. El-Rashidy, and N. A. El-Fishawy, "An efficient resource utilization technique for consolidation of virtual machines in cloud computing environments" in *2016 33rd NRSC*, Aswan Egypt, 2016, pp. 316-324, doi: 10.1109/NRSC.2016.7450844.

[56] H. Abdi, "Multiple correlation coefficient," *Encyclopedia of measurement and statistics*. pp. 648-651, 2007. http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.444.5689&rep=rep1&type=pdf.

[57] C. Yan, Z. Li, X. Yu, and N. Yu, "Bayesian networks-based selection algorithm for virtual machine to be migrated" in *2016 IEEE Int. Conf. BDCloud-SocialCom-SustainCom*, Atlanta GA USA, 2016, pp. 573-578, doi: 10.1109/BDCloud-SocialCom-SustainCom.2016.89.

[58] A. S. Tanenbaum, *Distributed operating systems*: Pearson Education India, 1995.

[59] M. H. Ferdaus, M. Murshed, R. N. Calheiros, and R. Buyya, "Virtual machine consolidation in cloud data centers using ACO metaheuristic" in *EURO-PAR*, 2014, pp. 306-317. Available: https://doi.org/10.1007/978-3-319-09873-9_26

[60] H. Inkwon, and M. Pedram, "Hierarchical, Portfolio Theory-Based Virtual Machine Consolidation in a Compute Cloud," *IEEE Transactions on Services Computing,* vol. 11, no. 1, pp. 63-77, 2018.

[61] L. Ma, H. Liu, Y. W. Leung, and X. Chu, "Joint VM-Switch Consolidation for Energy Efficiency in Data Centers" in *2016 IEEE GLOBECOM*, Washington DC USA, 2016, pp. 1-7, doi: 10.1109/GLOCOM.2016.7841944.

[62] A. Marcel, P. Cristian, P. Eugen, P. Claudia, T. Cioara, I. Anghel, and S. Ioan, "Thermal aware workload consolidation in cloud data centers" in *2016 IEEE 12th Int. Conf. ICCP*, Cluj-Napoca Romania, 2016, pp. 377-384, doi: 10.1109/ICCP.2016.7737177.

[63] R. Nasim, J. Taheri, and A. J. Kassler, "Optimizing Virtual Machine Consolidation in Virtualized Datacenters Using Resource Sensitivity" in *2016 IEEE Int. Conf. CloudCom*, Luxembourg City Luxembourg, 2016, pp. 168-175.

[64] F. Farahnakian, T. Pahikkala, P. Liljeberg, J. Plosila, N. T. Hieu, and H. Tenhunen, “Energy-aware VM Consolidation in Cloud Data Centers Using Utilization Prediction Model,” *IEEE Transactions on Cloud Computing*, vol. 7, no. 2, pp. 524-536, 2019.

[65] J. A. Pascual, T. Lorido-Botrán, J. Miguel-Alonso, and J. A. Lozano, “Towards a greener cloud infrastructure management using optimized placement policies,” *Journal of Grid Computing*, vol. 13, no. 3, pp. 375-389, 2015.

[66] F. Farahnakian, T. Pahikkala, P. Liljeberg, J. Plosila, N. T. Hieu, and H. Tenhunen, “Energy-aware vm consolidation in cloud data centers using utilization prediction model”, *IEEE Transactions on Cloud Computing*, vol. 7, no. 2, pp. 524-536, 2019.

[67] A. Marotta, and S. Avallone, "A Simulated Annealing Based Approach for Power Efficient Virtual Machines Consolidation" in *2015 IEEE 8th Int. Conf. CLOUD*, New York USA, 2015, pp. 445-452, doi: 10.1109/CLOUD.2015.66.

[68] A. Jobava, A. Yazidi, B. J. Oommen, and K. Begnum, "Achieving Intelligent Traffic-Aware Consolidation of Virtual Machines in a Data Center Using Learning Automata." *Journal of Computational Science*, vol. 24, no. 4, pp. 290-312, 2018.

[69] T. H. Nguyen, M. D. Francesco, and A. Yla-Jaaski, “Virtual Machine Consolidation with Multiple Usage Prediction for Energy-Efficient Cloud Data Centers,” *IEEE Transactions on Services Computing*, 2017.

[70] Q. Wu, F. Ishikawa, Q. Zhu, and Y. Xia, “Energy and migration cost-aware dynamic virtual machine consolidation in heterogeneous cloud datacenters,” *IEEE Transactions on Services Computing*, vol. 12, no. 4, pp. 550-563, 2019.

[71] A. Choudhary, M. C. Govil, G. Singh, L. K. Awasthi, E. S. Pilli and N. Kumar, "Improved Virtual Machine Migration Approaches in Cloud Environment" in *2016 IEEE Int. Conf. CCEM*, Bangalore India, 2016, pp. 17-24, doi: 10.1109/CCEM.2016.013.

[72] D. Grimes *et al.*, "Robust Server Consolidation: Coping with Peak Demand Underestimation" in *2016 IEEE 24th Int. Symp. MASCOTS*, London UK, 2016, pp. 271-276, doi: 10.1109/MASCOTS.2016.60.

[73] A. Kaur and M. Kalra, "Energy optimized VM placement in cloud environment" in *2016 6th Int. Conf. Confluence*, Noida India, 2016, pp. 141-145, doi: 10.1109/CONFLUENCE.2016.7508103.

[74] A. Montresor and M. Jelasity, "PeerSim: A scalable P2P simulator" in *2009 IEEE 9th Int. Conf. Peer-to-Peer Comput.*, Seattle WA USA, 2009, pp. 99-100, doi: 10.1109/P2P.2009.5284506.

[75] Y. Liu, "A Consolidation Strategy Supporting Resources Oversubscription in Cloud Computing" in *2016 IEEE 3rd Int. Conf. CSCloud*, Beijing China, 2016, pp. 154-162, doi: 10.1109/CSCloud.2016.21.

[76] H. Li, G. Zhu, C. Cui, H. Tang, Y. Dou, and C. He, “Energy-efficient migration and consolidation algorithm of virtual machines in data centers for cloud computing,” *Computing*, vol. 98, no. 3, pp. 303-317, 2016.

[77] X. Li, A. Ventresque, J. Murphy, and J. Thorburn, “SOC: Satisfaction-Oriented Virtual Machine Consolidation in Enterprise Data Centers,” *International Journal of Parallel Programming*, vol. 44, no. 1, pp. 130-150, 2016.

[78] J.-k. Dong, H.-b. Wang, Y.-y. Li, and S.-d. Cheng, “Virtual machine placement optimizing to improve network performance in cloud data centers,” *The Journal of China Universities of Posts and Telecommunications*, vol. 21, no. 3, pp. 62-70, 2014/06/01, 2014.

[79] Q. Zheng, R. Li, X. Li, N. Shah, J. Zhang, F. Tian, K.-M. Chao, and J. Li, “Virtual machine consolidated placement based on multi-objective biogeography-based optimization,” *Future Generation Computer Systems*, vol. 54, pp. 95-122, 2016.

[80] Z. Cao, and S. Dong, “An energy-aware heuristic framework for virtual machine consolidation in Cloud computing,” *The Journal of Supercomputing*, vol. 69, no. 1, pp. 429-451, 2014.

[81] D.-K. Kang, F. Alhazemi, S.-H. Kim, and C.-H. Youn, "Dynamic Virtual Machine Consolidation for Energy Efficient Cloud Data Centers" in *6th Int. Conf. CloudComp 2015, Daejeon South Korea, October 28-29, 2015*, pp. 70-80. Available: https://doi.org/10.1007/978-3-319-38904-2_8.

[82] L. Liu, S. Zheng, H. Yu, V. Anand, and D. Xu, “Correlation-based virtual machine migration in dynamic cloud environments,” *Photonic Network Communications*, vol. 31, no. 2, pp. 206-216, 2016.

[83] M. A. H. Monil, and R. M. Rahman, “VM consolidation approach based on heuristics, fuzzy logic, and migration control,” *Journal of Cloud Computing*, vol. 5, no. 1, 2016.

[84] C. A. Patel, and J. S. Shah, "Server Consolidation with Minimal SLA Violations" in *Proc. Int. Conf. on CIDM*, New Delhi India, 2016, pp. 455-462, doi: https://doi.org/10.1007/978-81-322-2731-1_43.

[85] M. H. Fathi, and L. M. Khanli, “Consolidating VMs in Green Cloud Computing Using Harmony Search Algorithm" in *Proc. ICIEB’18*, Singapore Singapore, 2018, pp. 146-151. Available: https://dl.acm.org/doi/abs/10.1145/3230348.3230369.

[86] A. Mosa, and R. Sakellariou, "Virtual machine consolidation for cloud data centers using parameter-based adaptive allocation" in *Proc. 5th European Conf. ECBS’17*, New York NY USA, 2017. Available: https://dl.acm.org/doi/10.1145/3123779.3123807.

[87] F. Farahnakian, T. Pahikkala, P. Liljeberg, J. Plosila, N. T. Hieu, and H. Tenhunen, “Energy-aware VM Consolidation in Cloud Data Centers Using Utilization Prediction Model,” *IEEE Transactions on Cloud Computing*, vol. 7, no. 2, pp. 524-536, 2019.

[88] D. Alsadie, E. J. Alzahrani, N. Sohrabi, Z. Tari and A. Y. Zomaya, "DTFA: A Dynamic Threshold-Based Fuzzy Approach for Power-Efficient VM Consolidation" in *2018 IEEE 17th Int. Symp. NCA*, Cambridge MA USA, 2018, pp. 1-9, doi: 10.1109/NCA.2018.8548162.

[89] K. Ajmera and T. Kumar Tewari, "Greening the Cloud Through Power-Aware Virtual Machine Allocation" in *2018 11th Int. Conf. Contemporary Computing (IC3)*, Noida, 2018, pp. 1-6, doi: 10.1109/IC3.2018.8530625.

[90] Y. Liu, X. Sun, W. Wei, and W. Jing, “Enhancing Energy-Efficient and QoS Dynamic Virtual Machine Consolidation Method in Cloud Environment,” *IEEE Access*, vol. 6, pp. 31224-31235, 2018.

[91] H. Wang, and H. Tianfield, “Energy-Aware Dynamic Virtual Machine Consolidation for Cloud Datacenters,” *IEEE Access*, vol. 6, pp. 15259-15273, 2018.

[92] Y. Chang, C. Gu and F. Luo, "Energy efficient virtual machine consolidation in cloud datacenters" in *2017 4th ICSAI*, Hangzhou China, 2017, pp. 401-406, doi: 10.1109/ICSAI.2017.8248325.

[93] S. B. Shaw, J. P. Kumar and A. K. Singh, "Energy-performance trade-off through restricted virtual machine consolidation in cloud data center" in *2017 Int. Conf. I2C2*, Coimbatore India, 2017, pp. 1-6, doi: 10.1109/I2C2.2017.8321783.

[94] D. Alsadie, Z. Tari, E. J. Alzahrani, and A. Y. Zomaya, "LIFE: A predictive approach for VM placement in cloud environments" in *2017 IEEE 16th Int. Symp. NCA*, 2017. pp. 1-8, doi: 10.1109/NCA.2017.8171338.

[95] M. A. H. Monil, and A. D. Malony, "QoS-Aware Virtual Machine Consolidation in Cloud Datacenter" in *2017 IEEE IC2E*, Vancouver BC Canada, 2017, pp. 81-87, doi: 10.1109/IC2E.2017.31.

[96] E. Arianyan, "Multi objective consolidation of virtual machines for green computing in Cloud data centers" in *2016 8th IST*, Tehran Iran, 2016, pp. 654-659, doi: 10.1109/ISTEL.2016.7881903.

[97] "SPECpower_ssj2008". spec.org. http://www.spec.org/power_ssj2008/results/res2011q1/power_ssj2008-20110124-00338.html. (Accessed Oct. 6, 2020).

[98] "SPECpower_ssj2008". spec.org. http://www.spec.org/power_ssj2008/results/res2011q1/power_ssj2008-20110124-00339.html. (Accessed Oct. 6, 2020).

[99] "SPECpower_ssj2008". spec.org. https://www.spec.org/power_ssj2008/results/res2017q4/power_ssj2008-20171010-00789.html. (Accessed Oct. 6, 2020).

[100] "SPECpower_ssj2008". spec.org. https://www.spec.org/power_ssj2008/results/res2017q4/power_ssj2008-20171010-00790.html. (Accessed Oct. 6, 2020).

[101] "SPECpower_ssj2008". spec.org. https://www.spec.org/power_ssj2008/results/res2017q4/power_ssj2008-20171009-00787.html. (Accessed Oct. 6, 2020).

[102] J. von Kistowski, J. Beckett, K.-D. Lange, H. Block, J. A. Arnold, and S. Kounev, "Energy efficiency of hierarchical server load distribution strategies" in *2015 IEEE 23rd Int. Symp. MASCOTS*, Atlanta GA USA, 2015, pp. 75-84, doi: 10.1109/MASCOTS.2015.11.

[103] D. Wong, and M. Annavaram, "Implications of high energy proportional servers on cluster-wide energy proportionality" in *2014 IEEE 20th Int. Symp. HPCA*, Orlando FL USA, 2014, pp. 142-153, doi: 10.1109/HPCA.2014.6835925.

[104] M. A. Khan, A. Paplinski, A. M. Khan, M. Murshed, and R. Buyya, "Dynamic Virtual Machine Consolidation Algorithms for Energy-Efficient Cloud Resource Management: A Review," *Sustainable Cloud and Energy Services*, pp. 135-165: Springer, 2018.

[105] "Amazon EC2 Instance Types." aws.amazon.com. https://aws.amazon.com/ec2/instance-types/. (Accessed Oct. 6, 2020).

[106] Z. Charles. "Real Statistics Using Excel." real-statistics.com. https://www.real-statistics.com/free-download/real-statistics-resource-pack/. (Accessed Oct. 6, 2020).

[107] The Cloud Computing and Distributed Systems (CLOUDS) Laboratory, and University of Melbourne. "CloudSim: A Framework For Modeling And Simulation Of Cloud Computing Infrastructures And Services [Internet]." 29/7/2016; http://cloudbus.org/cloudsim/.

[108] M. A. Khan, A. Paplinski, A. M. Khan, M. Murshed, and R. Buyya, "Exploiting User Provided Information In Dynamic Consolidation of Virtual Machines to Minimize Energy Consumption of Cloud Data Centers" in *3rd Int. Conf. FMEC*, Barcelona Spain, 2018, pp. 105-114, doi: 10.1109/FMEC.2018.8364052.

[109] J. v. Kistowski, H. Block, J. Beckett, K.-D. Lange, J. A. Arnold, and S. Kounev, "Analysis of the influences on server power consumption and energy efficiency for CPU-intensive workloads" in *Proc. 6th ACM/SPEC ICPE*, New York NY USA, 2015, pp. 223-234. Available: https://dl.acm.org/doi/abs/10.1145/2668930.2688057.

[110] J. Gustedt, E. Jeannot, and M. Quinson, “Experimental methodologies for large-scale systems: a survey,” *Parallel Processing Letters,* vol. 19, no. 03, pp. 399-418, 2009.

[111] X. Peng, B. Pernici, and M. Vitali, "Virtual Machine Profiling for Analyzing Resource Usage of Applications" in *SCC 2018*, pp. 103-118. Available: https://doi.org/10.1007/978-3-319-94376-3_7.

[112] A. Ghasemi, and S. Zahediasl, “Normality Tests for Statistical Analysis: A Guide for Non-Statisticians,” *International Journal of Endocrinology and Metabolism,* vol. 10, no. 2, pp. 486-489, April 20, 2012, 2012.

[113] H. C. Thode, *Testing For Normality,* Boca Raton: CRC Press, 2002.

[114] J. Peat, and B. Barton, *Medical statistics: A guide to data analysis and critical appraisal*: John Wiley & Sons, 2008.

[115] D. J. Steinskog, D. B. Tjøstheim, and N. G. Kvamstø, “A cautionary note on the use of the Kolmogorov–Smirnov test for normality,” *Monthly Weather Review,* vol. 135, no. 3, pp. 1151-1157, 2007.

[116] Z. Charles. "Shapiro-Wilk Original Test." real-statistics.com. https://www.real-statistics.com/tests-normality-and-symmetry/statistical-tests-normality-symmetry/shapiro-wilk-test/. (Accessed Oct. 6, 2020).

[117] M. E. Clapham. "9: Shapiro-Wilk Test." Accessed Oct. 6, 2020. [Online Video]. Available: https://www.youtube.com/watch?v=dRAqSsgkCUc.

[118] Z. Charles. "Shapiro-Wilk Tables." real-statistics.com. https://www.real-statistics.com/statistics-tables/shapiro-wilk-table/. (Accessed Oct. 6, 2020).

[119] M. Hinz, G. P. Koslovski, C. C. Miers, L. L. Pilla, and M. A. Pillon, “A Cost Model for IaaS Clouds Based on Virtual Machine Energy Consumption,” *Journal of Grid Computing,* vol. 16, no. 3, pp. 493-512, September 01, 2018.

[120] R. Nasim, and A. J. Kassler, "A robust Tabu Search heuristic for VM consolidation under demand uncertainty in virtualized datacenters" in *17th IEEE/ACM Int. Symp. CCGRID 2017*, Madrid Spain, 2017, pp. 170-180, doi: 10.1109/CCGRID.2017.35.

[121] H. Shen, and L. Chen, “CompVM: A Complementary VM Allocation Mechanism for Cloud Systems,” *IEEE/ACM Trans. Netw.,* vol. 26, no. 3, pp. 1348-1361, 2018.

[122] D. Alsadie, Z. Tari, E. J. Alzahrani, and A. Alshammari, "LIFE-MP: Online Virtual Machine Consolidation with Multiple Resource Usages in Cloud Environments" in *WISE 2018*, Dubai UAE, 2018, pp. 167-177. Available: https://doi.org/10.1007/978-3-030-02925-8_12.

[123] A. H. Borhani, T. Hung, B.-S. Lee, and Z. Qin, “Power-network aware VM migration heuristics for multi-tier web applications,” *Cluster Computing,* vol. 22, no. 3, pp. 757-782, December 01, 2018.

[124] H. Monshizadeh Naeen, E. Zeinali, and A. Toroghi Haghighat, “A stochastic process-based server consolidation approach for dynamic workloads in cloud data centers,” *The Journal of Supercomputing,* vol. 76, no. 3, pp. 1903-1930, 2020/03/01, 2020.

[125] N. Patel, and H. Patel, "An Efficient VM Selection Strategy for Minimization of Migration in Cloud Environment" in *SmartCom 2017,* Singapore: Springer, Singapore, 2017, pp. 92-101. Available: https://doi.org/10.1007/978-981-13-1423-0_11.

[126] M. Al-Tarazi, and J. M. Chang, “Network-aware energy saving multi-objective optimization in virtualized data centers,” *Cluster Computing,* vol. 22, no. 2, pp. 635-647, June 01, 2019.

[127] S. Bhattacherjee, R. Das, S. Khatua, and S. Roy, “Energy-efficient migration techniques for cloud environment: a step toward green computing,” *The Journal of Supercomputing*, pp. 1-29, March 26, 2019.

[128] H. Nashaat, N. Ashry, and R. Rizk, “Smart elastic scheduling algorithm for virtual machine migration in cloud computing,” *The Journal of Supercomputing*, vol. 75, no. 7, pp. 3842-3865, January 11, 2019.

[129] M. H. Sayadnavard, A. Toroghi Haghighat, and A. M. Rahmani, “A reliable energy-aware approach for dynamic virtual machine consolidation in cloud data centers,” *The Journal of Supercomputing*, vol. 75, no. 4, pp. 2126-2147, April 01, 2019.

[130] H. Xu, Y. Liu, W. Wei, and Y. Xue, “Migration Cost and Energy-Aware Virtual Machine Consolidation Under Cloud Environments Considering Remaining Runtime,” *International Journal of Parallel Programming*, vol. 47, no. 3, pp. 481-501, January 03, 2019.

[131] C.-T. Yang, S.-T. Chen, J.-C. Liu, Y.-W. Chan, C.-C. Chen, and V. K. Verma, “An energy-efficient cloud system with novel dynamic resource allocation methods,” *The Journal of Supercomputing*, vol. 75, no. 8, pp. 4408-4429, March 06, 2019.

[132] M. Abdel-Basset, L. Abdle-Fatah, and A. K. Sangaiah, “An improved Lévy based whale optimization algorithm for bandwidth-efficient virtual machine placement in cloud computing environment,” in *Cluster Computing*, vol. 22, no. 4, pp. 8319-8334, 2019/07/01, 2019.

[133] A. Aryania, H. S. Aghdasi, and L. M. Khanli, “Energy-Aware Virtual Machine Consolidation Algorithm Based on Ant Colony System,” *Journal of Grid Computing*, vol. 16, no. 3, pp. 477-491, September 01, 2018.

[134] L.-A. Galaviz-Alejos, F. Armenta-Cano, A. Tchernykh, G. Radchenko, A. Y. Drozdov, O. Sergiyenko, and R. Yahyapour, "Bi-objective Heterogeneous Consolidation in Cloud Computing," in *High Performance Computing*, vol. 796, pp. 384-398, 2018. Available: https://doi.org/10.1007/978-3-319-73353-1_27.

[135] J. Yan, H. Zhang, H. Xu, and Z. Zhang, “Discrete PSO-based workload optimization in virtual machine placement,” *Personal and Ubiquitous Computing*, vol. 22, no. 3, pp. 589-596, June 01, 2018.

[136] S. S. Alresheedi, S. Lu, M. Abd Elaziz, and A. A. Ewees, “Improved multiobjective salp swarm optimization for virtual machine placement in cloud computing,” *Human-centric Computing and Information Sciences*, vol. 9, no. 1, pp. 15-38, April 09, 2019.

[137] T. Kimura, T. Suzuki, K. Hirata, and M. Muraguchi, “Residual Capacity-Aware Virtual Machine Assignment for Reducing Network Loads in Multi-tenant Data Center Networks,” *Journal of Network and Systems Management*, vol. 27, no. 4, pp. 949-971, February 22, 2019.

[138] Z. Li, Y. Li, T. Yuan, S. Chen, and S. Jiang, “Chemical reaction optimization for virtual machine placement in cloud computing,” *Applied Intelligence*, vol. 49, no. 1, pp. 220-232, January 01, 2019.

[139] F. F. Moges, and S. L. Abebe, “Energy-aware VM placement algorithms for the OpenStack Neat consolidation framework,” *Journal of Cloud Computing*, vol. 8, no. 1, pp. 2-15, January 24, 2019.

[140] Kamran, and B. Nazir, “QoS-aware VM placement and migration for hybrid cloud infrastructure,” *The Journal of Supercomputing*, vol. 74, no. 9, pp. 4623-4646, September 01, 2018.

[141] W. S. Cleveland, “Robust locally weighted regression and smoothing scatterplots,” *Journal of the American statistical association,* vol. 74, no. 368, pp. 829-836, 1979.

[142] M. Natrella, "LOESS (aka LOWESS)," *NIST/SEMATECH e-Handbook of Statistical Methods*, 4, 2013.

[143] W. S. Cleveland, and C. Loader, "Smoothing by local regression: Principles and methods," in *Statistical theory and computational aspects of smoothing*, pp. 10-49: Springer, 1996. Available: https://doi.org/10.1007/978-3-642-48425-4_2.

[144] W. S. Cleveland, and S. J. Devlin, “Locally weighted regression: an approach to regression analysis by local fitting,” *Journal of the American statistical association,* vol. 83, no. 403, pp. 596-610, 1988.

[145] M. Dabbagh, B. Hamdaoui, M. Guizani and A. Rayes, "An Energy-Efficient VM Prediction and Migration Framework for Overcommitted Clouds," in *IEEE Transactions on Cloud Computing*, vol. 6, no. 4, pp. 955-966, 1 Oct.-Dec. 2018, doi: 10.1109/TCC.2016.2564403.

[146] C. Banton. "Moving Average, Weighted Moving Average, and Exponential Moving Average." investopedia.com. https://www.investopedia.com/ask/answers/071414/whats-difference-between-moving-average-and-weighted-moving-average.asp. (Accessed Oct. 6, 2020).

[147] A. Milton. "Simple, Exponential and Weighted Moving Averages." thebalance.com. https://www.thebalance.com/simple-exponential-and-weighted-moving-averages-1031196. (Accessed Oct. 6, 2020).

[148] "Appendix: Critical Value Tables." https://www.coconino.edu /resources /files/pdfs/academics/sabbatical-reports/kate-kozak/appendix_table.pdf. (Accessed Oct. 6, 2020)..

[149] R. Mahmud, K. Ramamohanarao, and R. Buyya, “Latency-aware application module management for fog computing environments,” *ACM Transactions on Internet Technology (TOIT)*, vol. 19, no. 1, pp. 1-21, 2018.

[150] A. Beloglazov, and R. Buyya, “OpenStack Neat: a framework for dynamic and energy-efficient consolidation of virtual machines in OpenStack clouds,” *Concurrency and Computation: Practice and Experience,* vol. 27, no. 5, pp. 1310-1333, 2015.