CLOUD BASED COST DEDUCTIVE IMPLEMENTATION OF DATA CENTERS

MS. NANNAPANENI SRAVANI[1], MS. SK.AYISHA BEGUM[2*]

1 Final Year MCA Student, QIS College of Engineering and Technology, Ongole

2* Assistant Professor, MCA Dept., QIS College of Engineering and Technology, Ongole

Abstract: *We focus on the issue of dealing with the power conditions of the servers in a Cloud Data Center (CDC) to together limit the power utilization and the upkeep costs got from the variety of intensity (and thus of temperature) on the servers' CPU. More in detail, we think about a lot of virtual machines (VMs) and their prerequisites regarding CPU and memory over a lot of Time Slot (TSs). We at that point display the expended power by considering the VMs preparing costs on the servers, the expenses for exchanging information between the VMs, and the expenses for moving the VMs over the servers. Likewise, we utilize a material-based weariness model to register the support costs expected to fix the CPU, as a result of the variety after some time of the server control states. In the wake of itemizing the issue plan, we structure a unique calculation, called Maintenance and Electricity Costs Data Center (MECDC), to understand it. Our outcomes, got more than a few situations from a genuine CDC, demonstrate that MECDC to a great extent beats two reference calculations, which rather either focus on the heap adjusting or the vitality utilization of the servers.*

Keywords: Cloud Storage, Cloud Data Center, Maintenance and Electricity Costs Data Center.

I. INTRODUCTION

Data centers (DCs) have turned into a key part of the Information and Communication Technology (ICT) area. Verifiably, misusing DCs for registering undertakings goes back to the primary portion of the nineteenth century, when diverse conspicuous analysts characterized the idea of worldwide mind [1], [2], with the objective of giving all encompassing methods for learning. From that point forward, the staggering development in the ICT division, incorporating the upgrades in Hard Ware (HW) producing, just as the practically unending highlights given by Soft Ware (SW), have totally changed the likelihood of misusing DCs for figuring purposes. These days, DCs are broadly spread worldwide to continue an assortment of utilizations, for example, web perusing, gushing, superior quality recordings, and distributed storage. Of course, DCs for the most part embrace the distributed computing worldview [3], [4], as indicated by which the virtualized applications (and whole working frameworks) keep running over a lot of disseminated physical servers, which might be even situated in various mainlands. Consequently, the administration of a Cloud Data Center (CDC) is a part of crucial significance for the DC proprietor (which is alluded as a substance supplier from here on). In a period where the measure of figuring data is always developing [5], an essential requirement for a substance supplier is to effectively oversee CDCs. Aside from the fixed costs, which are identified with the establishment of CDCs gear [6], a major stress for a substance supplier is the means by which to manage the CDCs control utilization and the related power costs [7]. In this unique circumstance, the substance supplier needs to confront the extensive measure of intensity devoured by its very own CDCs. Accordingly; the diminishing of intensity utilization in CDCs has been customarily a hotly debated issue [8]. In accordance with this pattern, distinctive works (see e.g., [9], [10]) focus on the decrease of intensity for the servers in a CDC through the administration of their capacity states. Among them, the utilization of a Sleep Mode (SM) state to a subset of servers is an exceptionally encouraging methodology so as to spare vitality [11], [12]. More in detail, because of the way that the traffic from clients isn't steady and by and large shifts over the distinctive hours of the day, it is conceivable in a CDC to put diverse servers in SM, and to focus the clients' traffic on a subset of servers, which stay in an Active Mode (AM). Thusly, a decrease of intensity and, subsequently, a decrease of the related power costs paid by the substance supplier are accomplished.

In spite of the fact that the utilization of SM can guarantee lower power costs contrasted with the case in which every one of the servers are constantly fueled on, the advances among SM and AM, particularly when they are connected over times of a while and years, will in general negatively affect the support costs paid by the substance supplier [13]. More in

detail, when the server is placed in SM, a brief decline in the temperature of its segments (particularly for CPU and recollections) is watched [14]. In particular, the temperature drops from entirely high qualities (regularly higher than 70◦-80◦ [Celsius]) to the room temperature, which is normally cooled and kept around 20◦ [Celsius]. Then again, the contrary impact on the temperature is seen when the server goes from SM to AM. The variety of temperature on the gadgets segments, particularly when it is rehashed after some time, will in general present warm weariness impacts [15], [16]. This marvel is like the mechanical weariness experienced by a plane fuselage, subject to lodge pressurization and depressurization over various flights, which may crumble it in the long haul [17].

Correspondingly, the HW gear, when it is liable to substantial temperature changes, will in general increment its disappointment rate. More in IEEE Transactions on Sustainable Computing Year:2018 IEEE Transactions detail, weakness (and split) impacts are experienced, for instance, by the bind joints interfacing the CPU/recollections to the motherboard [18]. As a result, a server subject to visit AM/SM advances will encounter disappointment occasions all the more regularly, contrasted with the case in which it is in every case left in AM, subsequently expanding the related support costs so as to fix as well as supplant the fizzled segments. In the most pessimistic scenario, the support costs will be considerably bigger than the power spared from the use of SM, accordingly delivering a fiscal misfortune to the substance supplier [13]. This setting represents a few difficulties: What is the effect of the support costs on the absolute expenses? Is it advantageous to use the tradeoff between power utilization and upkeep costs? How to ideally define the issue? How to structure an effective calculation to handle it? The objective of this paper is to reveal insight into these issues. More in detail, we first present a basic (yet viable) model to figure the upkeep costs, given the variety after some time of the power states for a lot of servers. Also, we embrace a nitty gritty model to process the power devoured by the CDC. In particular, our capacity demonstrate considers the CPU-related power expenses of the servers, the expenses for exchanging information among the servers, and the expenses for moving the Virtual Machines (VMs) running on the servers.

Subsequent to defining the issue of together diminishing the CDC power utilization and the related support costs, we propose another calculation, called Maintenance Energy Costs Data Center (MECDC), to handle it. Our outcomes, acquired more than a few situations from a genuine CDC, obviously demonstrate that our answer can shrewdly use the tradeoff among support and power costs so as to give financial reserve funds to the substance supplier. Then again, we demonstrate that different systems, either focusing on the VMs load adjusting, or the servers vitality utilization, will in general eminently increment the complete expenses. To the best of our insight, none of the past works in the CDC look into field has led a comparable investigation.

In spite of the fact that the outcomes announced in this paper are promising, we call attention to those different expenses than the ones considered here may build the upkeep bill. In particular, the expense of normal updates, due to HW/SW redesigns, may affect the support costs paid by the substance supplier. What's more, the selection of sustainable power sources may likewise shift the power bill. Both these issues, which are not considered in this work, can be conceivably included our structure.

II RELATED WORK

In the following, we briefly discuss the main literature in CDC related to our work. We first describe solutions targeting the management of energy and/or electricity in CDCs. Then, we move our attention to researches targeting the management of CDC Failures.

Energy and Elecricity Management in CDCs

Features such as electricity, power, as well as computing and network management tasks are addressed in [19], [20], [21], [22], [23], [24], [25], [26], [27], [28], [29]. A detailed overview of energy consumption models in Data Center (DC) is provided by Dayarathna et al. in [19]. In this context, several works target the management of a CDC by: i) providing algorithms for VM live migrations [20], [21], [22], ii) considering distributed server/CDC applications [23], [24], [25], [26], iii) focusing on business process management [27], and iv) detailing memory and storage management solutions [28], [29].

Focusing on the aspect of VM live migrations, Voorsluys et al. [20] adopt live migration of VMs, with the goal of reducing energy in the CDC while guaranteeing the performance to applications. However, this work does not consider the server maintenance costs. Moreover, the costs of VM migration and data transferring between VMs in a CDC environment are not taken into account. Liu et al. in [22] present a cost-aware learned knowledge method and an adaptive network bandwidth management, by applying VM live migration estimation to achieve power saving in the CDC. Soni et al. in [23] derive computing cost models for the CDC such that they try to cover the VMs' over/under loadings based on priority and states. Indeed, their proposed algorithm is able to manage load distribution among various applications running in each VM. Bi et al. in [25] present a queue-aware multi-tier application model inside the CDC. In addition, they compute the number of servers that must be allotted to each tier in order to meet the response time per application per server. They also consider the CPU resources per-VM in the CDC.

However, a live VM migration is not performed. Finally, Han et al. in [26] present an adaptive cost-aware elasticity method in order to scale up/down multi-tier cloud applications to meet run-time varying application demands. Nevertheless, the complexity of the proposed model in computational management is quadratic per-application. Focusing on the memory and storage management, Song et al. in [29] employ power performance information to estimate the desired storage and memory parameters in order to preserve energy and costs in the CDC. It is important to note that their quasi-analytical performance modeling can be accurate, but it requires a deep understanding of each individual application running on the VM and the server. Therefore, a consistent amount of preliminary information is needed and, as a consequence, the pre-processing time of the problem may sensibly increase. 2.2 Failure Management in CDCs Server failure is recognized as an important cost component for the cloud, see e.g. Greenberg et al. [30]. Therefore, different works target the reduction of the impact of the failure events by proposing efficient DC architectures. In particular, Guo et al. propose Dcell [31], a scalable and recursive architecture which is also fault-tolerant. Greenberg et al. [32] present VL2, a scalable and flexible DC network which is tolerant to failures experienced by networking equipment. Guo et al. [33] details BCube, an architecture for modular DCs, which is able to guarantee a graceful performance degradation as the server failure rate increases. Moreover, according to Kliazovich et al. [34], when the DC temperatures are not kept within their operational limits the HW reliability is decreased, thus bringing to a potential violation of Service Level Agreements (SLAs). In addition, the optimization of thermal states and cooling system operation is recognized as a challenge by Beloglazov et al. [10]. A detailed analysis of failures in a DC is performed by Gill et al. [35]. However, the work is mainly focused on network devices and not on servers like in our case. Eventually, a characterization of the HW components of the servers in terms of reliability is performed by Vishwanath et al. [36]. In particular, this work reports that the failure in one of the server HW components is a common event experienced in large DCs. In [12] Zhang et al. advocate the need of taking availability into consideration while mapping VMs. In this context, Fan et al. [31] explore the problem of mapping service function chains with guaranteed availability. Finally, Jhawar and Piuri [10] propose an approach to measure the effectiveness of fault tolerance mechanisms in Infrastructure as a Service (IaaS) cloud, by also providing a solution to select the best mechanism satisfying the users requirements.

EXISTING SYSTEM

Focusing on the memory and storage management, ”Unified performance and power modeling of scientific workloads” employ power performance information to estimate the desired storage and memory parameters in order to preserve energy and costs in the CDC. It is important to note that their quasi-analytical performance modeling can be accurate, but it requires a deep understanding of each individual application running on the VM and the server. Therefore, a consistent amount of preliminary information is needed and, as a consequence, the pre-processing time of the problem may sensibly increase.

III PROPOSED SYSTEM

This context poses several challenges: What is the impact of the maintenance costs on the total costs? Is it beneficial to leverage the tradeoff between electricity consumption and maintenance costs? How to optimally formulate the problem? How to design an efficient algorithm to tackle it? The goal of this paper is to shed light on these issues. More in detail, we first present a simple (yet effective) model to compute the maintenance costs, given the variation over time of the power states for a set of servers. In addition, we adopt a detailed model to compute the power consumed by the CDC. Specifically, our power model takes into account the CPU-related electricity costs of the servers, the costs for transferring data among the servers, and the costs for migrating the Virtual Machines (VMs) running on the servers. After formulating the problem of jointly reducing the CDC electricity consumption and the related maintenance costs, we propose a new algorithm, called Maintenance Energy Costs Data Center (MECDC), to tackle it.

METHODOLOGY

Cloud Data Center Architecture

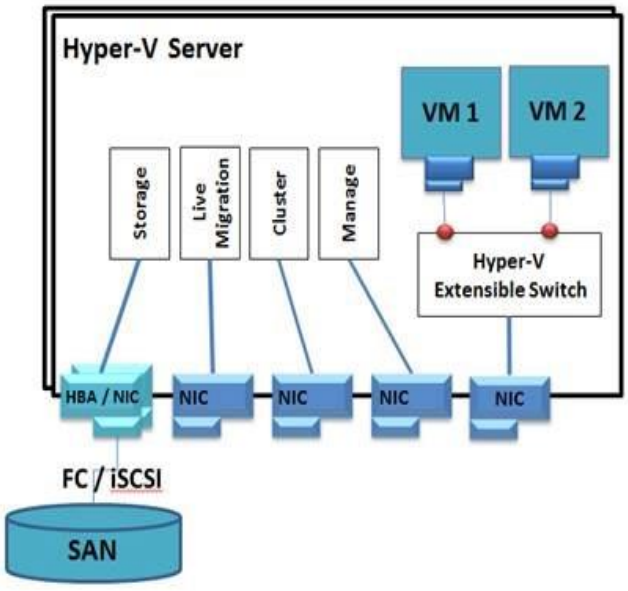


Fig. 1 reports the main building blocks of the considered CDC architecture. More in detail, the CDC is composed of

VMs, hypervisors, Physical Servers (PSs), switches and management entities. Each VM is hosted in a PS. The set of VMs in a PS is managed by an hypervisor. Moreover, the PSs are grouped in Pods. The interconnection between PSs in the same Pod is realized by means of a redundant set of switches and physical links. In addition, a DC network, again composed of switches and physical links, provides connectivity between the different Pods. Moreover, a centralized network manager (top left part of the figure) is then in charge of managing the set of networking devices, e.g., by providing software-defined functionalities. Finally, an allocation manager (mid left part of the figure) distributes the VMs over the PSs, by ensuring that each VM receives the required amount of CPU and memory from the PS hypervisor.

$$\phi_s^{TOT}(t) \triangleq \phi_s^{AM}\left(1-\frac{\tau_s^{SM}(t)}{\tau^{ALL}(t)}\right)+\phi_s^{SM}\cdot\frac{\tau_s^{SM}(t)}{\tau^{ALL}(t)}+\frac{\eta_s}{N_s^F} \quad (1)$$

Focusing on the tasks performed by the allocation manager, this element is in charge of running the proposed VMs' allocation algorithm, which is able to leverage the tradeoff between elec-tricity costs and maintenance costs by acting on the PSs power states. In our work, we assume that time is discretized in Time Slots (TSs), and that the allocation algorithm is run for every TS. Given: i) a current TS τ and the corresponding VMs requests in terms of CPU and memory;[1] ii) the power state of the PSs (AM or SM) and the allocation of VMs at the previous TS $\tau - 1$; the allocation manager computes the allocation of VMs for TS τ . Eventually, the allocation manager notifies the PSs that need to be put in AM/SM for the current TS. In case a PS was in AM at previous TS and needs to be put in SM at current TS, the allocation manager interacts with the PS operating system to gracefully halt the machine.

$$C_E^{TR}(t) = K_E \cdot \delta(t) \sum_{s\in S}\Big[O_s(t)\cdot P_s^{TR-IF} + \\ + \sum_{\substack{\sigma\in S\\ \sigma\neq s}}\sum_{m,n\in M} d_{mn}^{s\sigma}(t)\cdot P_{s\sigma}^{TR-NET}\Big], \quad [\$]$$

Data Owner:

In this module, the data owner registers to the particular cloud servers (cs1, cs2, cs3) with valid user details & logs in. After logged in data owner will browse the data file and sends to the particular cloud servers (cs1, cs2, cs3). And data owner can adds the remote users with their valid details.

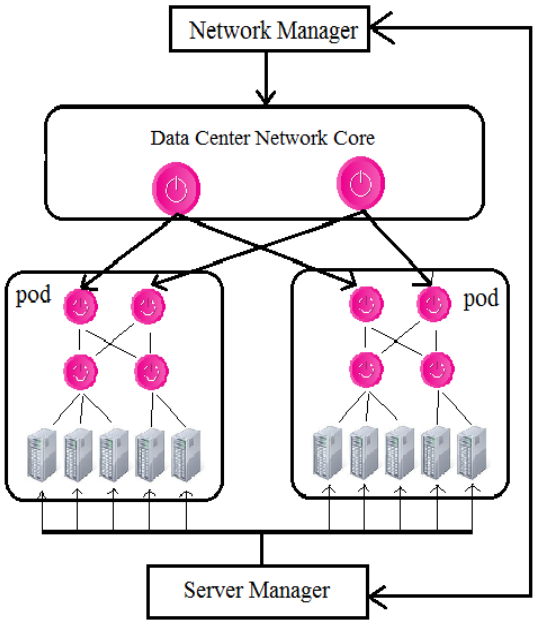


Fig: System Model

Data Centre

The data centre manages all cloud servers and load distributor is to provide the files storage services for all cloud servers. And it has the activities like assign energy to the Multicore Server Processor (MSP) of each cloud servers and View energy of MSP nodes.

Cloud Servers (cs1, cs2, cs3)

In these modules, the cloud server receives data file from data owner and stores on it. And when remote user's requests for the data file to particular cloud servers at that time these cloud servers will provide the data file by without changing file content. And cloud servers also do some operations like add data owner, view cloud users, view all data owners, view all owner files, view attacker details & unblock users.

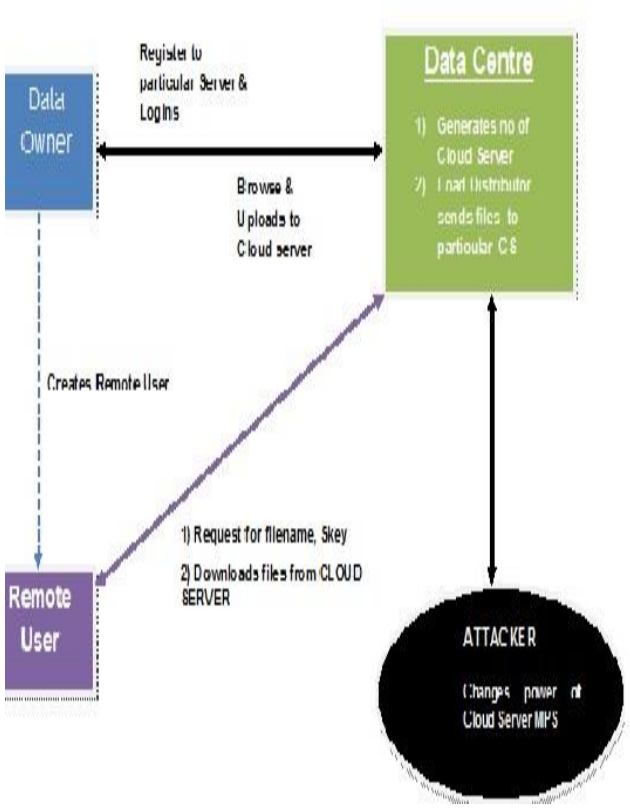
Remote User (End User)

In this module, the receiver can receive the data file from the particular cloud servers. Before this the data owner will adds the remote users to particular cloud servers with the valid end user details & after remote user will logged in and download the files. Data Owner will send data file to cloud servers via data centre & load distributor. The remote users receive the file by without changing the File Contents. Users may receive particular data files within the network only.

Attacker

Attacker is one who makes changes the Multicore Server Processor power sizes of particular cloud server. And all

attackers' details stored in particular cloud servers with their all details such as attacker name, Ip address, MSP node and size.



ALGORITHEM

MECDC algorithmic program DESCRIPTION Since the OMEC drawback is incredibly difficult to be solved even for instances of little size, we tend to propose the upkeep and Electricity Costs knowledge Center (MECDC) algorithmic program to much tackle it. The most intuitions of the projected approach area unit twofold: i) we don't contemplate all the toxic shock syndrome together} together, however rather we tend to focus on every single TS,4 and ii) we tend to guarantee a possible resolution which ensures the constraints (15)-(17) in every TS. As a result, the MECDC algorithmic program is consecutive endure every TS. Specifically, for each TS τ, we tend to use the answer computed for TS τ − one as input for the single-period drawback related to TS τ . The solution for τ is then passed as input to {the resolution|the answer} of the matter associated with the sequent TS τ + one then on till we tend to reach τ = |T |.

Algorithm 1 Pseudo-Code of the MECDC algorithm

Input: $t, \gamma_m(t), \mu_m(t), D_{mn}(t)$, prev_power_state_s
Output: $x_{sm}(t)$,curr_power_state_s

```
1: Phase 1: Select an admissible VMs to PSs allocation.
2: prev_VM_assigned=read_conf(x_sm(t − 1));
3: curr_VM_assigned=prev_VM_assigned;
4: curr_power_state_s=prev_power_state_s;
5: [curr_CPU_s, curr_mem_s, flag_check]=comp_CPU_mem(curr_VM_assignment,γ_m(t),μ_m(t));
6: if flag_check==false then
7:    s_sorted=sort(curr_CPU_s,'descend');
8:    VM_sorted=sort(curr_CPU_VM,'ascend');
9:    for vm=1:|M| do
10:      if check_CPU_mem(curr_VM_assigned,VM_sorted[vm],γ_m(t),μ_m(t))==false then
11:         stop_condition=0;
12:         for server=1:|S| do
13:            curr_s=s_sorted[server];
14:            if stop_condition < 1 then
15:               if curr_s != curr_VM_assigned[VM_sorted[vm]] then
16:                  if check_VM_to_server(curr_VM_assigned,curr_s,VM_sorted[vm],γ_m(t),μ_m(t))==true then
17:                     [curr_CPU_s, curr_mem_s, curr_VM_assigned]=VM_to_server(curr_VM_assigned,curr_s,VM_sorted[vm],γ_m(t), μ_m(t));
18:                     stop_condition=1;
19:                  end if
20:               end if
21:            end if
22:         end for
23:      end if
24:   end for
25: end if
26: f1_VM_assigned=curr_VM_assigned;
27: [curr_total_costs,curr_power_state_s]=compute_total_costs(prev_VM_assigned,curr_VM_assigned,t,γ_m(t),μ_m(t),D_mn(t),curr_power_state_s);
28: f1_total_costs=curr_total_costs;
29: f1_power_state_s=curr_power_state_s;
30: Phase 2: Refinement of the VMs' allocation to reduce the costs for current TS.
31: s_sorted=sort(curr_CPU_s,'ascend');
32: for server=1:|S| do
33:    if curr_power_state_s[server,]] > 0 then
34:       [tmp_VM_assigned, flag_bin]=adaptive_bin_packing(curr_VM_assigned,s_sorted[server],γ_m(t),μ_m(t));
35:       if flag_bin == 1 then
36:          [tmp_total_costs, tmp_power_state_s]=compute_total_costs(prev_VM_assigned,tmp_VM_assignment,t,γ_m(t),μ_m(t),D_mn(t),
   curr_power_state_s);
37:          if tmp_total_costs < curr_total_costs then
38:             curr_VM_assigned=tmp_VM_assigned;
39:             curr_total_costs=tmp_total_costs;
40:             curr_power_state_s[server,]=0;
41:          end if
42:       end if
43:    end if
44: end for
45: Phase 3: Adjustment of the VMs' allocation to limit the increase of the future costs.
46: [all_on_total_costs, all_on_power_state_s, all_on_VM_assigned]=compute_total_costs_all_on(prev_VM_assigned,t,γ_m(t), μ_m(t),D_mn(t),
   f1_power_state_s);
47: if curr_total_costs > all_on_total_costs × ζ then
48:    curr_VM_assignment=all_on_VM_assignment;
49:    curr_power_state_s=all_on_power_state_s;
50: end if
51: x_sm(t)=write_conf(curr_VM_assignment);
```

We analyze the time quality of MECDC. Specializing in the first section (lines 2-25), the computation of C.P.U. and memory requirements on every note is completed in O(|M| · |S|) iterations. Similarly, checking if a given VM may be migrated to a given PS (lines 16), additionally because the VM migration (line 17), may be done in O(|M|·|S|) iterations. The procedure is then perennial for every VM and every note within the worst case. As a result, the overall quality of section one is O(|M|2 · |S|2).

Algorithm 2 Pseudo-Code of the Adaptive Bin Packing function

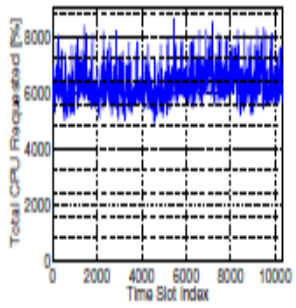
Input: curr_VM_assigned,curr_s,$\gamma_m(t)$,$\mu_m(t)$

Output: curr_VM_assigned,flag_bin

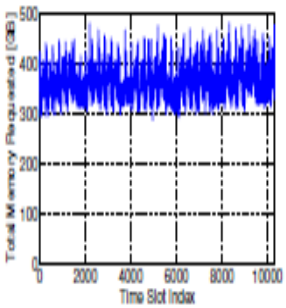
```
[VM_sorted]=sort(γm(t),'ascend');
[curr_CPU_s, curr_mem_s, flag_check]=comp_CPU_mem (curr_VM_assignment,γm(t),μm(t));
num_VM_to_move=comp_VM_to_move(curr_VM_assignment,curr_s);
power_state_s=comp_power_state_s(curr_VM_assignment);
VM_moved=0;
for vm=1:|M| do
    if curr_VM_assigned[VM_sorted[vm]]==curr_s then
        [curr_CPU_s, curr_mem_s, flag_check]=comp_CPU_mem(curr_VM_assignment,γm(t),μm(t));
        s_sorted=sort(curr_CPU_s,'descend');
        flag_moved_VM=0
        for server=1:|S| do
            if (s_sorted[server]! =curr_s) and (flag_moved_VM==0) and (power_state_s[s_sorted[server]]>0) then
                curr_VM_assignment[VM_sorted[vm]]=s_sorted[server];
                [curr_CPU_s, curr_mem_s,flag_check]= comp_CPU_mem(curr_VM_assignment,γm(t),μm(t));
                if flag_check<1 then
                    curr_VM_assignment[VM_sorted[vm]]=curr_s;
                else
                    flag_moved_VM=0;
                    VM_moved++;
                end if
            end if
        end for
    end if
end for
if VM_moved=num_VM_to_move then
    flag_bin=1;
else
    flag_bin=0;
end if
```

Virtual Machines Parameters

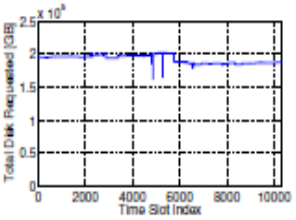
The thought-about parameters for every VM m and for every TS t include: the requested central processor γm (t), ii) the requested memory μm(t), iii) the quantity of information Dmn(t) changed by the VM m to every other VM n ∈ M. so as to retrieve such parameters, we have considered the trace Materna-3, that reports real measurements of a bureau collected from TU earthenware. The trace includes the log files of 547 VMs, that area unit wont to deploy a CDC dedicated to business intensive applications. every VM log reports a collection of knowledge collected for every TS, including: i) the central processor needs (both in terms of central processor.



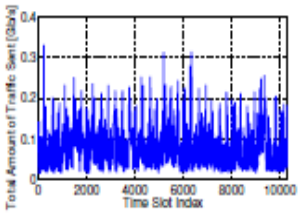
(a) Total CPU Requested



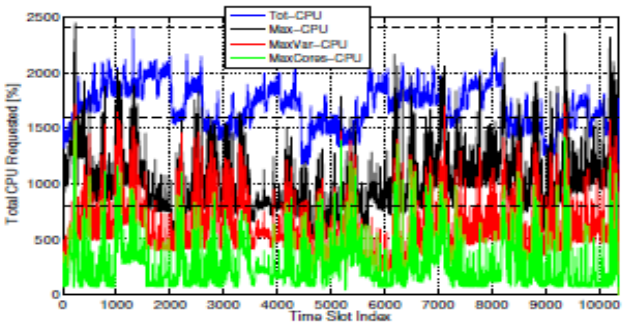
(b) Total Memory Requested



(c) Total Disk Requested



(d) Total Network Traffic Sent



V RESULT

We have targeted the matter of put together managing the upkeep prices and also the electricity consumption in a very authority. once showing that ever-changing the ability states of PSs has a bearing on each the failure management prices, still because the energy consumption, we've developed the OMEC drawback, with the goal of put together managing the for mentioned prices.

VI CONCLUSION

Since the OMEC downside is NPHard, within the future work we'll describe the MECDC rule, which can be designed to showing wisdom leverage the exchange between totally different prices, in addition as taking under consideration their future impact over time. Results, obtained over a collection of realistic eventualities, clearly show that MECDC continuously

needs systematically lower prices compared to the FFD and NFD reference algorithms. Moreover, we've conjointly shown that the entire prices obtained by MECDC are about to a edge. Additionally, the computation time, obtained from a situation within which there ar many VMs and by running the rule on a Desktop computer, is extremely low, i.e., but two [s] on the average.

VII. REFERENCES

[1] H. Wells, World Brain. Methuen & Co., London, 1938.

[2] V. Bush et al., "As we may think," The atlantic monthly, vol. 176, no. 1,pp. 101–108, 1945.

[3] M. Armbrust, A. Fox, R. Griffith, A. D. Joseph, R. Katz, A. Konwinski, G. Lee, D. Patterson, A. Rabkin, I. Stoica, et al., "A view of cloud computing," Communications of the ACM, vol. 53, no. 4, pp. 50–58, 2010.

[4] P. Mell, T. Grance, et al., "The NIST definition of cloud computing," 2011.

[5] M. Hilbert and P. L´opez, "The worlds technological capacity to store, communicate, and compute information," Science, vol. 332, no. 6025, pp. 60–65, 2011.

[6] C. D. Patel and A. J. Shah, "Cost model for planning, development and operation of a data center," Tech. Rep. HPL-2005-107(R.1), HP Laboratories Palo Alto, 2005.

[7] 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.

[8] K. Bilal, S. U. R. Malik, O. Khalid, A. Hameed, E. Alvarez, V. Wijaysekara, R. Irfan, S. Shrestha, D. Dwivedy, M. Ali, et al., "A taxonomy and survey on green data center networks," Future Generation Computer Systems, vol. 36, pp. 189–208, 2014.

[9] S. Srikantaiah, A. Kansal, and F. Zhao, "Energy aware consolidation for cloud computing," in Proceedings of the 2008 conference on Power aware computing and systems, vol. 10, pp. 1–5, San Diego, California, 2008.

[10] 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.

[11] C. Mastroianni, M. Meo, and G. Papuzzo, "Probabilistic consolidation of virtual machines in self-organizing cloud data centers," IEEE Transactions on Cloud Computing, vol. 1, no. 2, pp. 215–228, 2013.

[12] W. Fang, X. Liang, S. Li, L. Chiaraviglio, and N. Xiong, "VMPlanner: Optimizing virtual machine placement and traffic flow routing to reduce network power costs in cloud data centers," Computer Networks, vol. 57, no. 1, pp. 179–196, 2013.

[13] L. Chiaraviglio, P. Wiatr, P. Monti, J. Chen, J. Lorincz, F. Idzikowski, M. Listanti, and L. Wosinska, "Is green networking beneficial in terms of device lifetime?," IEEE Communications Magazine, vol. 53, no. 5, pp. 232–240, 2015.

[14] L. Chiaraviglio, N. Blefari-Melazzi, C. Canali, F. Cuomo, R. Lancellotti, and M. Shojafar, "A Measurement-Based Analysis of Temperature Variations Introduced by Power Management on Commodity Hardware," in International Conference on Transparent Optical Networks (ICTON 2017), 2017.

[15] D. Frear, D. Grivas, and J. Morris, "Thermal fatigue in solder joints," JOM, vol. 40, no. 6, pp. 18–22, 1988.

[16] W. Lee, L. Nguyen, and G. S. Selvaduray, "Solder joint fatigue models:review and applicability to chip scale packages," Microelectronics reliability, vol. 40, no. 2, pp. 231–244, 2000.

[17] R. P. G. Muller, "An experimental and analytical investigation on the fatigue behaviour of fuselage riveted lap joints. The significance of the rivet squeeze force, and a comparison of 2024-T3 and Glare 3," Delft University of Technology, 1995.

[18] J. Mi, Y.-F. Li, Y.-J. Yang, W. Peng, and H.-Z. Huang, "Thermal cycling life prediction of Sn-3.0 Ag-0.5 Cu solder joint using type-I censored data," The Scientific World Journal, vol. 2014, 2014.

[19] M. Dayarathna, Y. Wen, and R. Fan, "Data center energy consumption modeling: A survey," IEEE Communications Surveys & Tutorials, vol. 18, no. 1, pp. 732–794, 2016.

[20] W. Voorsluys, J. Broberg, S. Venugopal, and R. Buyya, "Cost of virtual machine live migration in clouds: A performance evaluation," in IEEE International Conference on Cloud Computing, pp. 254–265, Springer, 2009.

[21] J. Stoess, C. Lang, and F. Bellosa, "Energy management for hypervisorbased virtual machines.," in USENIX annual technical conference, pp. 1–14, 2007.

[22] H. Liu, H. Jin, C.-Z. Xu, and X. Liao, "Performance and energy modeling for live migration of virtual machines," Cluster computing, vol. 16, no. 2, pp. 249–264, 2013.

[23] G. Soni and M. Kalra, "A novel approach for load balancing in cloud data center," in Advance Computing

Conference (IACC), 2014 IEEE International, pp. 807–812, IEEE, 2014.

[24] A. Beloglazov and R. Buyya, "Energy efficient resource management in virtualized cloud data centers," in Proceedings of the 2010 10th IEEE/ACM international conference on cluster, cloud and grid computing, pp. 826–831, IEEE Computer Society, 2010.

[25] J. Bi, Z. Zhu, R. Tian, and Q. Wang, "Dynamic provisioning modeling for virtualized multi-tier applications in cloud data center," in Cloud Computing (CLOUD), 2010 IEEE 3rd international conference on, pp. 370–377, IEEE, 2010.

[26] R. Han, M. M. Ghanem, L. Guo, Y. Guo, and M. Osmond, "Enabling cost-aware and adaptive elasticity of multi-tier cloud applications," Future Generation Computer Systems, vol. 32, pp. 82–98, 2014.

[27] J. L. Berral, R. Gavalda, and J. Torres, "Power-aware multi-data center management using machine learning," in Parallel Processing (ICPP), 2013 42nd International Conference on, pp. 858–867, IEEE, 2013.

[28] R. Ge, X. Feng, and K. W. Cameron, "Modeling and evaluating energy performance efficiency of parallel processing on multicore based power aware systems," in Parallel & Distributed Processing, 2009. IPDPS 2009. IEEE International Symposium on, pp. 1–8, IEEE, 2009.

[29] S. L. Song, K. Barker, and D. Kerbyson, "Unified performance and power modeling of scientific workloads," in Proceedings of the 1st International Workshop on Energy Efficient Supercomputing, p. 4, ACM, 2013.

[30] A. Greenberg, J. Hamilton, D. A. Maltz, and P. Patel, "The cost of a cloud: research problems in data center networks," ACM SIGCOMM computer communication review, vol. 39, no. 1, pp. 68–73, 2008.

[31] C. Guo, H. Wu, K. Tan, L. Shi, Y. Zhang, and S. Lu, "Dcell: a scalable and fault-tolerant network structure for data centers," in ACM SIGCOMM Computer Communication Review, vol. 38, pp. 75–86, ACM, 2008.

[32] A. Greenberg, J. R. Hamilton, N. Jain, S. Kandula, C. Kim, P. Lahiri, D. A. Maltz, P. Patel, and S. Sengupta, "VL2: a scalable and flexible data center network," in ACM SIGCOMM computer communication review, vol. 39, pp. 51–62, ACM, 2009.

[33] C. Guo, G. Lu, D. Li, H. Wu, X. Zhang, Y. Shi, C. Tian, Y. Zhang, and S. Lu, "BCube: a high performance, server-centric network architecture for modular data centers," ACM SIGCOMM Computer Communication Review, vol. 39, no. 4, pp. 63–74, 2009.

[34] D. Kliazovich, P. Bouvry, Y. Audzevich, and S. U. Khan, "GreenCloud: a packet-level simulator of energy-aware cloud computing data centers," in Global Telecommunications Conference (GLOBECOM 2010), 2010 IEEE, pp. 1–5, IEEE, 2010.

[35] P. Gill, N. Jain, and N. Nagappan, "Understanding network failures in data centers: measurement, analysis, and implications," in ACM SIGCOMM Computer Communication Review, vol. 41, pp. 350–361, ACM, 2011.

[36] K. V. Vishwanath and N. Nagappan, "Characterizing cloud computing hardware reliability," in Proceedings of the 1st ACM symposium on Cloud computing, pp. 193–204, ACM, 2010.

Authors Profile

Ms. **Nannapaneni Sravani** pursuing MCA 3rd year in Qis College and Engineering and Technology in Department of Master of Computer Applications, Ongole.

Ms. **Sk. Ayisha Begum** is currently working as an Assistant Professor in Department of Master of Computer Applications in QIS College of Engineering & Technology with the Qualification MCA.