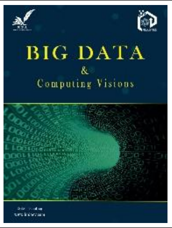
Big Data and Computing Visions

www.bidacv.com

Big. Data. Comp. Vis. Vol. 3, No. 3 (2023) 84–90.



Paper Type: Original Article

Exploring the Feasibility of a Data Center-Free Cloud Computing Framework Utilizing Underutilized Personal Computers

Ibrahim Mekawy[1], Alhanouf Alburaikan[2], Iman Atighi[3,*]

[1] Department of Mathematics, College of Science and Arts, Qassim University, Ar Rass, Saudi Arabia; im.mekawy@qu.edu.sa.
[2] Department of Mathematics, College of Arts and Sciences, Al-Badaya, Qassim University, Buraydah, Saudi Arabia; a.albrikan@qu.edu.sa.
[3] Department of Industrial Engineering, Kish Branch, Islamic Azad University, Kish, Iran; imanatighi4@gmail.com.

Citation:

Mekawy, I., Alburaikan, A., & Atighi, I. (2023). Exploring the feasibility of a data center-free cloud computing framework utilizing underutilized personal computers. *Big data and computing visions, 3*(3), 84-90.

Received: 05/03/2023 Reviewed: 08/04/2023 Revised: 19/05/2023 Accept: 21/06/2023

Abstract

The current landscape of Cloud Computing predominantly relies on closed data centers, housing a multitude of dedicated servers that cater to cloud services. However, an immense number of underutilized Personal Computers (PCs) are owned by individuals and organizations globally. These dormant resources can be harnessed to form an alternative cloud infrastructure, offering a wide array of cloud services, particularly focusing on infrastructure as a service. This innovative strategy, the "no data center" approach, complements the conventional data center-centric cloud provisioning model. In a research paper, the authors introduce their opportunistic Cloud Computing framework called cuCloud, which effectively utilizes the idle resources of underutilized PCs within a given organization or community. The success of their system serves as tangible evidence that the "no data center" concept is indeed feasible. Beyond conceptualization and philosophy, the authors' experimental findings strongly validate their approach.

Keywords: Cloud computing, Personal computers.

1 | Introduction

The current Cloud Computing services are based on the "data center" approach, where hundreds of thousands of dedicated servers are set up to give the services [1]. Setting up the data center for the cloud is expensive, and running the infrastructure needs expertise and a lot of resources, such as high power for cooling, redundant power for assured availability, etc. [2]. In addition to the vast number of servers used in data centers, billions of Personal Computers (PCs) are owned by individuals and organizations worldwide [3]. These PCs are mostly underutilized, usually used only a few hours per day. The authors suggest that we should treat the untapped CPU cycles and disk spaces of the great many underutilized PCs as precious assets, like monetary assets, to consolidate and reuse them for the good of society and individuals, just like how a credit union works [4].

Corresponding Author: imanatighi4@gmail.com

https://doi.org/10.22105/bdcv.2023.190168

The Credit Union Cloud Model (CUCM) is an alternative Cloud Computing provision model that provides cloud services (mainly IaaS) based on the "no data center" approach to provisioning Cloud Computing services for an institution, organization, or community [5]. With current public clouds, better-called vendor clouds provided by vendors based on dedicated data centers, the concern for security/safety and loss of control is the primary obstacle keeping traditional IT from moving to clouds [6]. The CUCM provides a feasible on-premise solution to Cloud Computing for institutions and organizations that highly care about cost and security. The key characteristic of CUCM is the "no data center" approach to provisioning Cloud Computing services for an institution, organization, or community [7]. Among many other benefits of CU clouds, affordability (almost no additional cost for acquiring and running an on-premise cloud infrastructure) is particularly appealing [8]. It can help an organization or business owner save up to 45% of the cost of a data center by eliminating the upfront purchase for the cloud servers, which would otherwise be necessary. In addition, the credit union cloud infrastructure does not need additional cooling systems, saving an additional 15% of the data center's cooling cost.

1.2 | Credit Union Cloud Model

The CUCM is a cloud provision model that aims to use idle/underutilized computers for cloud service provisioning [9]. It runs on existing infrastructures with excessive capacities, which are not specifically set up for supporting Cloud Computing. CU clouds run on existing infrastructures with excessive capacities, which are not specifically set up for supporting Cloud Computing [10]. The model allows PCs within an organization to join the "cloud credit union" and contribute their underutilized resources (CPU cycles or disk spaces) to the union's cloud resource pool to back up its cloud hardware [11].

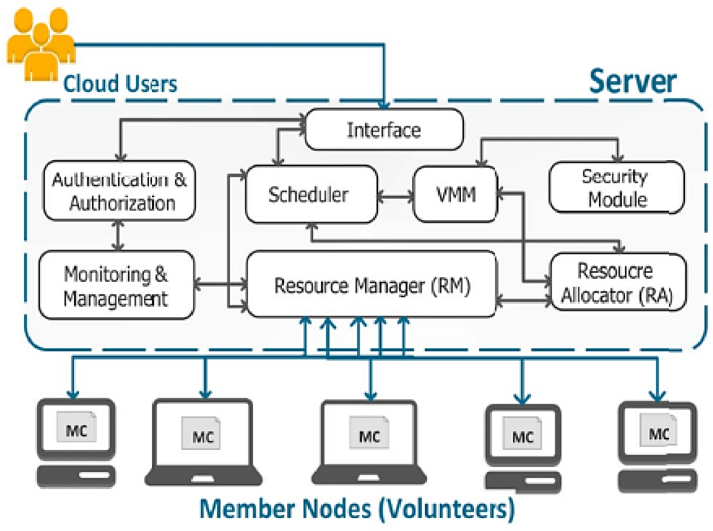


Fig. 1. CUCM/cu cloud architecture.

The architecture of Cisco Unified Communications Manager (CUCM) is a client/server architecture with member nodes as clients and dedicated management machine(s) as server(s) [12]. The server has different components, such as an Interface, Authentication and Authorization module, Resource Manager (RM), Resource Allocator (RA), Scheduler module, Virtual Machine Manager (VMM), security module, and monitoring and management module. The Interface is the first port of communication between CU Cloud and its users/clients [13]. The RM has the global picture of the cloud infrastructure's resources as a whole. The RA component selects a list of suitable member nodes for the deployment of Virtual Machines (VMs) according to the resource requirements of the cloud customer, the Service Level Agreement (SLA), and the availability as well as reliability profiles of the member nodes [14]. The Scheduler module accepts user requests and allocates or denies the requested resources in consultation with the VMM and the RA. The VMM component handles the deployment of VMs on member nodes [15]. The security module handles the security of the VMs. The monitoring and management module gives fine-grained resource information about the resources of the CU Cloud system.

The Membership Controller (MC) is a critical piece of software that resides on each member node that contributes resources to the CU Cloud system. The MC monitors the resource usage at a member node and decides the node's membership status [16]. Active status indicates enough resources to meet the need for a minimum VM, while inactive status indicates the unavailability of such resources. The MC collects and sends information to the server about the types and quantities of the available resources (CPU, RAM, Hard Disk) for contribution to the resource pool of CU Cloud periodically. The MC has three components: a sensor component, a reporter component, and a virtual environment monitor component. The sensor component monitors the resource usage of processes on a member node and gives that information to the reporter component. The Reporter component decides the node's membership status in the cloud infrastructure based on the sensed information. If the resource (RAM, CPU, Hard Disk) availability is above/below some threshold value, it will send a message to the RM on the server indicating that it is either an active or inactive member. The virtual environment monitor component manages the VMs deployed on the member node.

2 | Implementation and Experimentation

2.1 | Implementation

CU Cloud is a cloud management system and platform for delivering IaaS Services. It is expected to fulfill all the characteristics of Cloud Computing, such as elasticity, metering service, multitenancy, etc. As the first step of the project, a preliminary version of CUCM (called cuCloud) has been implemented using Apache CloudStack, which is an open-source IaaS platform that manages and orchestrates various resources, including storage, network, and computers to build a public or private IaaS compute cloud. CloudStack has management server(s) that can manage tens of thousands of physical servers installed in geographically distributed data centers. The Management Server of CloudStack communicates with the compute nodes (physical servers) through the hypervisors (Xen, KVM, Hyper-V, etc) installed on the machines. Since CloudStack is an IaaS system that is developed to manage dedicated data centers with only dedicated hosts, the management server of CloudStack needs to be modified in such a way that it can handle the non-dedicated member nodes that physically back up the cuCloud system . A special component (called the AdHoc component) was developed and integrated into the CloudStack management server to form the cuCloud management server. On the other side, SIGAR (system information Gatherer and reporter) was used for the sensor component of member controllers that reside on member nodes. The number of CPU cores, RAM capacity, idle CPU percentage, free memory percentage, and available free hard disk space are sensed/gathered and passed to the reporter module of MC. Each instance of the MC running on a member node continuously senses the resource usage of the processes on the member host.

If the resource utilization at the member node is below a certain threshold, the MC instance sends an "active" message to the AdHoc component on the cuCloud management server and an "inactive" message otherwise. The AdHoc component will update the resource base of CloudStack based on the message it accepts from the MCs. The other components of CloudStack remain unchanged in this preliminary version of cuCloud. Due to the full self-autonomy of member nodes, type I hypervisors like Xen or Hyper-V cannot be used. Instead, a virtualization solution that runs along with other applications on the member nodes is needed. Therefore, KVM (Kernel-based VM) was chosen. It introduces virtualization by augmenting the traditional kernel and user modes of Linux with a new process mode called guest with its own kernel and user modes and answers for code execution from guest operating systems. This characteristic of KVM allows VMs to run along with local applications on member nodes simultaneously.

2.2 | Experimentation

To test the feasibility of CUCM, two sets of experiments were conducted using one server and four client machines. As discussed in the implementation subsection, the modified CloudStack version 4.9.0

was used. The Management Server of CloudStack was installed on a machine with 8 GB of RAM, an Intel 8 Core i7 2.4 GHz CPU, and a hard disk of 250GB. Each computing node had 8 GB of RAM, intel 4 cores i3 3.1 GHz CPU, and 250GB hard disk capacity. All the machines ran Ubuntu 14.04, were connected to a 16Gbps switch, and supported intel hardware virtualization (VT-x). For the first set of experiments, a dedicated CloudStack infrastructure was set up using one management server and four compute nodes. The performance and usage of CPU and RAM were measured using well-known benchmarks, LINPACK and STREAM, respectively.

The second set of experiments was done on a modified CloudStack, renamed cuCloud, as it implements our CUCM model. The cuCloud also assumed one management server node and four-member nodes with which the benchmarks were run to gather the same set of measurements for comparison. A member node would join cuCloud infrastructure if its CPU idle percentage were greater than or equal to 70%. Five scenarios were used to compare the performance of CloudStack assuming dedicated machines (or data center) vis–a-vis our cuCloud relying on contributing member compute nodes. The experiments with cuCloud were conducted while the local users were still using the machines.

Scenario 1. On dedicated CloudStack infrastructure, one small VM instance (1 vCPU, 512MB RAM, 20GB HD) is deployed on one of the computing nodes.

Scenario 2. On dedicated CloudStack infrastructure, one medium (2 vCPU, 1GB RAM, 20GB HD) and two small VM instances are deployed on one of the compute nodes, and on one of the instances, the performance measurement tasks are carried out while the other two instances are set busy with 40% and 60% (CPU usage).

Scenario 3. Same as *Scenario 1*, except it is on cuCloud non-dedicated infrastructure.

Scenario 4. Same as *Scenario 2* but on cuCloud. Here, we purposefully make the member node that hosts the benchmark tasks busy to cause live migration of the VM with the performance measurement tasks.

Scenario 5. Same as *Scenario 4*, but we purposefully induced two live migrations of the VM that hosts the performance tasks.

The LINPACK benchmark was set up with matrix dimensions 5000x5000 and was targeted at CPU-intensive computations involving large linear equations. An average of 10 executions of the LINPACK benchmark was gathered. The experimental results are depicted in *Fig. 2* and *Fig. 3*. As shown in *Fig. 2*; there is almost no difference between running a task on a CloudStack dedicated compute node and on a cuCloud shared member node as long as there is enough resource to execute the task. However, as shown in *Fig. 3*, when there are one or more migrations, the tasks running on cuCloud might take longer. It is obviously a consequence of the induced migration of the VMs. From our experimental result, the performance gap between one migration and two migrations does not seem obvious (12.64 vs 12.71 seconds), which may not be generalized.

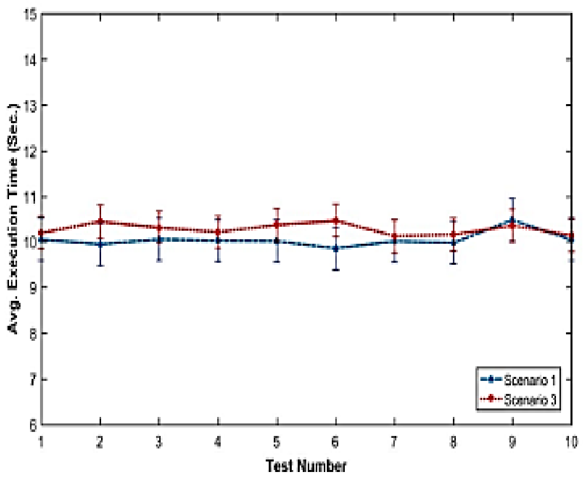


Fig. 2. LINPACK: dedicated Vs CU-Cloud.

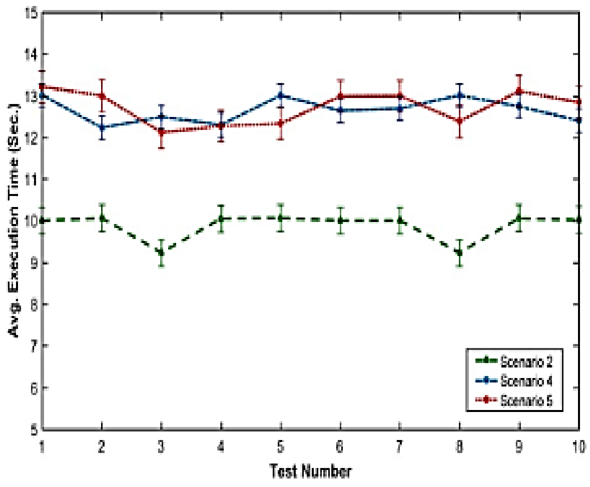


Fig. 3. LINPACK; dedicated Vs CU-Cloud with migration.

STREAM was designed mainly to measure the memory bandwidth using four operations, Add, Copy, Scale, and Triad, and was set up with an array of 2,000,000. *Fig. 4* depicts the average bandwidth usage of 10 trials of running STREAM with the 5 scenarios. As *Fig. 4* shows, the bandwidth usage of the four operations in the first 3 scenarios is almost the same, and VM migration noticeably causes an increase in bandwidth usage.

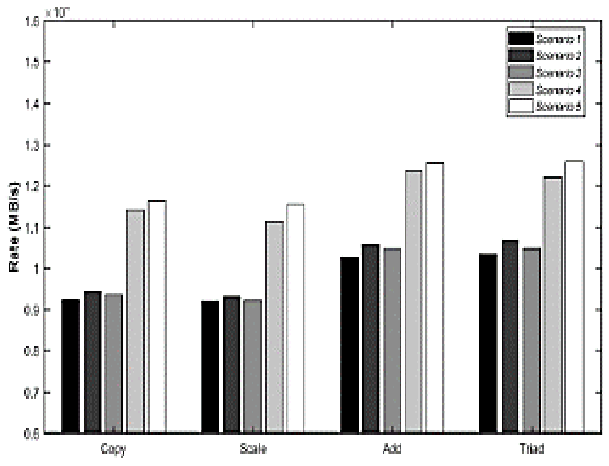


Fig. 4. STREAM bandwidth rate.

The preliminary implementation of CUCM and experiments with both cuCloud and CloudStack demonstrate that the concept of CUCM (i.e., no data center cloud solution) works and presents a promising alternative approach to Cloud Computing with many advantages.

3 | Related Works

Very few works have been done in the research direction that we embark on: the development of the Cloud Computing model based on spare resources of PCs as its resource base. One noticeable work is the so-called ad hoc cloud reported in [9], where in various research issues related to cloud provisioning using general-purpose computers were explored. The proposed cloud infrastructure architecture consists of several components: one creates/destroys cloud elements; one monitors the effects of created cloud elements; one handles the quality of service issues; and one executes allocated tasks. The authors gave no actual implementation of their ad hoc cloud system. Another work in non-dedicated data center-based Cloud Computing is [10] that tried to investigate the feasibility, reliability and performance of ad hoc Cloud Computing infrastructures. The ad hoc cloud system, a client/server system, is based on the well-known VC system BOINC with a virtualization support called V-BOINC. The server component has three subcomponents: Cloud Interface, VM Service, and job service. The client is the one that accepts the jobs and executes them reliably. The research concluded that ad hoc cloud is feasible and a viable alternative to the current data center-based Cloud Computing systems. The authors mentioned nothing about the system's elasticity, multitenancy, etc. characteristics.

4 | Conclusion and Future Work

The CUCM aims at tapping into the underutilized computing resources available within an organization/community rather than dedicated servers and provides a promising alternative Cloud Computing solution for organizations and communities. Our work demonstrates that the "no data center" solution indeed works. Besides proving the concept, model, and philosophy of CUCM, our experimental study was highly encouraging - the "no data center" solution can gain highly competitive performance compared to its counterpart that depends on dedicated cloud servers.

The most significant aspect of our work so far is that by cuCloud, we have set up a platform and have a door widely open for many exciting new research issues for the future. The resource pool over which VMs run in cuCloud is not dedicated but shared with native users/tasks. We must devise a mechanism to provide cloud services reliably and efficiently while keeping the services from interfering with the native users/tasks at member nodes. Another requirement of CUCM is to have a robust, dynamic, and efficient resource management and provisioning mechanism. The resource management and provisioning module should consider the dynamic and unreliable nature of the member hosts that contribute resources to the resource pool of cuCloud. We also need to investigate novel and efficient scheduling algorithms that consider the availability, location, and reliability of the member nodes used by the system to deploy VMs. In addition, cuCloud requires strong security measures to ensure the security of member nodes from malicious cloud client processes and client VMs from malicious native users at member nodes.

References

[1] Hassan, N., Aazam, M., Tahir, M., & Yau, K. L. A. (2023). Floating Fog: extending fog computing to vast waters for aerial users. *Cluster computing*, *26*(1), 181–195. DOI:10.1007/s10586-022-03567-6

[2] Magotra, B., Malhotra, D., & Dogra, A. K. (2023). Adaptive computational solutions to energy efficiency in cloud computing environment using VM consolidation. *Archives of computational methods in engineering*, *30*(3), 1789–1818. DOI:10.1007/s11831-022-09852-2

[3] Whaiduzzaman, M., Haque, M. N., Rejaul Karim Chowdhury, M., & Gani, A. (2014). A study on strategic provisioning of cloud computing services. *Scientific world journal*, *2014*. DOI:10.1155/2014/894362

[4] Gao, F., Thiebes, S., & Sunyaev, A. (2018). Rethinking the meaning of cloud computing for health care: A taxonomic perspective and future research directions. *Journal of medical internet research*, *20*(7), e10041. DOI:10.2196/10041

[5] Sun, Y., & Zhang, N. (2017). A resource-sharing model based on a repeated game in fog computing. *Saudi journal of biological sciences*, *24*(3), 687–694. DOI:10.1016/j.sjbs.2017.01.043

[6] Liu, H., Li, S., & Sun, W. (2020). Resource allocation for edge computing without using cloud center in smart home environment: A pricing approach. *Sensors (Switzerland)*, *20*(22), 1–28. DOI:10.3390/s20226545

[7] Khani, H., & Khanmirza, H. (2019). Randomized routing of virtual machines in IaaS data centers. *PeerJ computer science*, *2019*(9), e211. DOI:10.7717/peerj-cs.211

[8] Yu, S., Gui, X., Lin, J., Tian, F., Zhao, J., & Dai, M. (2014). A security-awareness virtual machine management scheme based on Chinese wall policy in cloud computing. *The scientific world journal*, *2014*. DOI:10.1155/2014/805923

[9] Mohapatra, H., & Rath, A. K. (2022). IoE based framework for smart agriculture: Networking among all agricultural attributes. *Journal of ambient intelligence and humanized computing*, *13*(1), 407–424. DOI:10.1007/s12652-021-02908-4

[10] Detti, A., Nakazato, H., Navarro, J. A. M., Tropea, G., Funari, L., Petrucci, L., … Kanai, K. (2021). Viriot: A cloud of things that offers iot infrastructures as a service. *Sensors*, *21*(19), 6546. DOI:10.3390/s21196546

[11] Ala'anzy, M. A., Othman, M., Hanapi, Z. M., & Alrshah, M. A. (2021). Locust inspired algorithm for cloudlet scheduling in cloud computing environments. *Sensors*, *21*(21), 7308. DOI:10.3390/s21217308

[12] Isazadeh, A., Ziviani, D., & Claridge, D. E. (2023). Global trends, performance metrics, and energy reduction measures in datacom facilities. *Renewable and sustainable energy reviews*, *174*, 113149. DOI:10.1016/j.rser.2023.113149

[13] Tarafdar, A., Debnath, M., Khatua, S., & Das, R. K. (2020). Energy and quality of service-aware virtual machine consolidation in a cloud data center. *Journal of supercomputing*, *76*(11), 9095–9126. DOI:10.1007/s11227-020-03203-3

[14] Panda, S. K., & Sen, S. (2023). SRRA: a novel skewness-based algorithm for cloudlet scheduling. *2023 IEEE 23rd international conference on software quality, reliability, and security (QRS)* (pp. 772–781). IEEE. DOI: 10.1109/qrs60937.2023.00080

[15] Mohapatra, H., & Rath, A. K. (2021). A fault tolerant routing scheme for advanced metering infrastructure: an approach towards smart grid. *Cluster computing*, *24*(3), 2193–2211. DOI:10.1007/s10586-021-03255-x

[16] Perumal, K., Mohan, S., Frnda, J., & Divakarachari, P. B. (2022). Dynamic resource provisioning and secured file sharing using virtualization in cloud azure. *Journal of cloud computing*, *11*(1), 1–12. DOI:10.1186/s13677-022-00326-1