

Article

A Residual Resource Fitness-Based Genetic Algorithm for a Fog-Level Virtual Machine Placement for Green Smart City Services

Sanjoy Choudhury ^{1,2}, Ashish Kumar Luhach ^{3,*}, Joel J. P. C. Rodrigues ^{4,5} , Mohammed AL-Numay ⁶ , Uttam Ghosh ⁷  and Diptendu Sinha Roy ¹ 

¹ Department of Computer Science and Engineering, National Institute of Technology Meghalaya, Shillong 793003, India

² Satyendra Nath Bose National Centre for Basic Sciences, Kolkata 700106, India

³ Department of Electrical and Communication Engineering, The PNG University of Technology, Lae MP 411, Papua New Guinea

⁴ College of Computer Science and Technology, China University of Petroleum (East China), Qingdao 266555, China

⁵ Instituto de Telecomunicações, 6201-001 Covilhã, Portugal

⁶ Electrical Engineering Department, College of Engineering, King Saud University, P.O. Box 800, Riyadh 11421, Saudi Arabia

⁷ Department of Computer Science and Data Science, Meharry Medical College, Nashville, TN 37208, USA

* Correspondence: ashish.kumar@pnguot.ac.pg

Abstract: Energy efficient information and communication technology (ICT) infrastructure at all levels of a city's edifice constitutes a core requirement within the sustainable development goals. The ICT infrastructure of smart cities can be considered in three levels, namely the cloud layer infrastructure, devices/sensing layer infrastructure, and fog layer infrastructure at the edge of the network. Efficiency of a data-centre's energy infrastructure is significantly affected by the placement of virtual machines (VMs) within the data-centre facility. This research establishes the virtual machine (VM) placement problem as an optimisation problem, and due to its adaptability for such complicated search issues, this paper applies the genetic algorithm (GA) towards the VM placement problem solution. When allocating or reallocating a VM, there is a large quantity of unused resources that might be used, however these resources are inefficiently spread over several different active physical machines (PMs). This study aims to increase the data-centre's efficiency in terms of both energy usage and time spent on maintenance, and introduces a novel fitness function to streamline the process of computing the fitness function in GAs, which is the most computationally intensive component in a GA. A standard GA and first fit decreasing GA (FFD-GA) are applied on benchmark datasets to compare their relative performances. Experimental results obtained using data from Google data-centres demonstrate that the proposed FFD-GA saves around 8% more energy than a standard GA while reducing the computational overhead by approximately 66%.

Keywords: fog computing; virtual machines; genetic algorithm; fitness function; energy efficiency



Citation: Choudhury, S.; Luhach, A.K.; Rodrigues, J.J.P.C.; AL-Numay, M.; Ghosh, U.; Sinha Roy, D. A Residual Resource Fitness-Based Genetic Algorithm for a Fog-Level Virtual Machine Placement for Green Smart City Services. *Sustainability* **2023**, *15*, 8918. <https://doi.org/10.3390/su15118918>

Academic Editor: Gerardo Maria Mauro

Received: 29 March 2023

Revised: 12 May 2023

Accepted: 16 May 2023

Published: 1 June 2023



Copyright: © 2023 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (<https://creativecommons.org/licenses/by/4.0/>).

1. Introduction

Virtualisation at the edge of the network has recently gained adequate research attention. This has paved the path for fog data-centres with augmentation capability to accommodate a huge variety of user applications [1]. While data-centres are essential for the smooth functioning of fog services, they also have a significant impact on the environment due to the large amounts of energy required to keep them running [2].

Data-centre energy consumption is a key concern [3] since IT equipment, especially PMs, consume as much as 60% of the total energy, even when a data-centre is idle. The high cost, poor environmental implications, and reduced dependability of computing

equipment [4] stem from computer servers' disproportionately high energy requirement. Ineffective virtual machine positioning and the distribution of frequent PMs across several customers waste valuable resources and use unnecessary energy [4,5]. These facts have motivated significant research focus on power-aware solutions for virtual machine placement in data-centres to decrease energy consumption.

However, despite significant advancements in energy efficiency across all industrial sectors, the overall energy consumption of computer systems has not decreased significantly [3]. It is estimated that anywhere from 10–30% of servers in data-centres are idle at any given time point. Therefore, green cloud data-centres mandate adherence to best practices for energy-aware resource management. Whereas most existing cloud data-centres use only around 20–50% of their available resources, virtualisation, as a technology, has allowed for much improved resource utilisation, thus resulting in improved energy efficiency [6]. Evolutionary algorithms such as GA or heuristic approaches such as FFD [7] may be used to find an optimal configuration for virtual machines. Although FFD is computationally lightweight, it cannot be employed as an efficient way for saving energy in data-centres. Moreover, a GA with standard fitness computations is computationally too inefficient to provide suitable solutions within acceptable time frames, especially for large-scale data-centres. The improper placement of VMs with trivial light-weight computations can seriously affect the performance and energy efficiency of fog data-centres. Thus, accelerating GAs in fog data-centres would require a viable method for intelligently organising VMs in the hosts (PMs), and incorporating FFD for evaluating the fitness function for a GA allows for both of these benefits to be leveraged simultaneously.

The computational efficiency of fitness functions can greatly affect the overall performance in a GA since they are computed at every epoch continuously, and even a slight reduction in computational complexity can have significant performance benefits [8]. Furthermore, in [5], the effects of the parameters on the overall performance have been studied in depth.

This paper introduces a method for incorporating a GA for the VM placement issue that can be applicable for data-centres of any capacity by a computationally lightweight FFD-based fitness function framed upon the Taylor expansion, where VM allocation has been addressed as a restricted optimisation problem. The term “residual resource” refers to the unused capacity in operational physical machines within a data-centre. The key contribution of this paper includes the advancement of a new approach for deriving a fitness function for the proposed GA solution that can better arrange VMs towards achieving a reduced energy consumption and execution time. To achieve this objective, this study suggests an innovative method of a fitness function component for the GA using Taylor's expansion. Secondly, the proposed GA provides VM allocation decisions to PMs with the goal of maximising energy efficiency while minimising leftover resources within PMs. This packing of VMs to PMs finally results in a reduced overall energy consumption across a data-centre. Using traces from Google's data-centre data, this paper demonstrates that the proposed FFD-GA outperforms both stand-alone FFDs and traditional GAs.

The remainder of this paper is organised as follows: Section 2 details the prior literature and provides the impetus for this research. Section 3 outlines the criteria for a successful fitness function in a genetic algorithm, and justifies the suitability of the proposed fitness function. Section 4 explains the algorithm's architecture and complexity. Section 5 presents the simulations carried out to assess the performance of the proposed solution, including a brief description of the data used for the experiment. Section 6 presents the results to demonstrate the efficacy of the solution. Section 7 presents the conclusions of this research along with the scope for future works.

2. Related Works

To improve energy efficiency in server farms, many studies have been carried out over the last decade. The management of energy-conscious assets, the efficiency of the equipment, and the productivity of users can have a significant impact on the improvement of energy efficiency [7]. The author of [8] employs graph neural networks to estimate the resource consumption of various workloads.

The author of [9] has provided a paradigm for quantifying operational carbon emissions using location-based and time-specific marginal emissions data per energy unit. The work presented natural language processing and computer vision applications with a broad range of model sizes, including the pre-training of a 6.1 billion parameter language model for evaluating carbon emission modelling.

The author of [10] has studied the limited energy utilisation for server farms using GAs for energy optimization [10]. In order to create a model that takes into account the superior features of GAs and a combinational ordering first fit (COFF) algorithm, in [11], a combined COFF-based genetic algorithm (COFF-GA) was proposed. Performance of the proposed COFF-GA was compared with a standard GA and a next fit GA. The research presented in [12] improved the robustness and speed of the GA by combining it with an infeasible arrangement fix and a local improvement technique. To increase data-centres' energy efficiency, a hybrid GA was proposed.

Furthermore, the authors of [13] employed a GA to limit the amount of physical machines used as well as data-centre migrations. Finally, they used the Grivon (SLA) virtual machine packing design to improve energy usage while addressing the requirement for administration level skills.

However, the plan of fitness function computation was not discussed in their paper but they did provide some improved results for some basic fitness functions for energy effectiveness. The authors of [14] proposed a cloud-based client side coordinated energy board to expand the energy productivity in a savvy local area. They introduced a two-level plan that depended on an energy centre and burden aggregators. Likewise, a data veiling component was created in light of the straight planning capacities to protect the privacy of the cloud server.

The authors of [15] fostered a distributed computing stage for the continuous estimation and verification of the energy performance. They applied molecule swarm streamlining and a multivariate relapse examination to make precise gauge principles. It helps the public authority in further developing energy productivity programs and the improvement of energy administration ventures.

This study [16] builds on previous work by offering models for the energy consumption of GPUs and CPU-GPU combinations that have been empirically verified using various GPU hardware types and GPU heavy workloads.

In [17], the authors described the preliminary findings from profiling virtual machines with regard to three power metrics, namely power, power efficiency, and energy, under various high performance conditions.

Figure 1 describes the various algorithm comparisons, including the different factors involved in various related studies on energy efficiency research.

In any case, none of these examinations has explored the calculation of fitness function capacities that significantly affects the GA efficiency. This examination intends to foster another fitness function capacity to apply a faster GA as a virtual asset to the board.

Author + Year	Based on	Resources considered	Strength	Weakness
Xia, X. et al. + 2021	Genetic Algorithm	CPU, memory	Reduced inter-communication cost, power consumption, execution time	Migration costs
Rbii, E. et al. + 2020	Constraint programming	CPU	Minimised total power consumption and number of PMs	Do not consider the execution time
Kozłowski, A. et al. + 2019	Constraint programming	CPU, memory	Reduced energy consumption and number of PM	More execution time
Li, Z. et al. + 2018	Genetic Algorithm	CPU, memory, bandwidth	Reduce data centre energy consumption	Increased computation time
Hallawi, H. et al. + 2017	Genetic Algorithm and Bin Packing	CPU, memory	Minimised the total number of PMs and resources wastage	More energy consumption
Jiang, J. et al. + 2015	Constraint Programming	CPU, memory	Reduced energy consumption in both PMs and communication network	Increased the computation workload
Li, Z. et al. + 2017	Constraint Programming	CPU and network bandwidth	Reduced energy consumption and network traffic	More VM migration cost
Yu, L. et al. + 2016	Genetic Algorithm	CPU, Network, Storage	Energy-efficient, Near-optimal solution	Asymmetric bucket code
Mastelic, T. et al. + 2014	Bin Packing	CPU	Reduced relocation's cost	Used more number of bins
Bagheri, Z. et al. + 2014	Genetic Algorithm	CPU, Network, Storage	Reduced energy consumption, Minimum resources wastage	Increased computation time
Yamada, N. et al. + 2014	Bin Packing	CPU	Reduced number of PMs and energy consumption	Slow execution
Chen, Q. et al. + 2011	Constraint Programming	CPU	Reduced migration cost, Reduced number of APMs	Slow execution

Figure 1. Comparison of various algorithms for energy efficiency research [3,6,10,12,13,17–23].

2.1. VM Configuration of an Energy Consumption Model

A VM arrangement is a method for selecting the ideal PM for a set of supplied [1]. On occasion, the PMs associated with server farms might not be fully employed, and this affects energy proficiency. The sensible position of VMs is significant for further developing energy effectiveness and limiting the quantity of dynamic PMs in cloud server farms [16]. In order to set up VMs correctly on a PM, one must be aware of the physical machine capacity, virtual machine prerequisites, and know how to resolve resource conflicts reliably with data-centre methods [18]. The computer processes' usage rate in a PM, P_j is computed as follows:

$$u_j = p_j^{w_{cpu}} / p_j^{cpu} \quad (1)$$

The overall PM usage is identical to the rundown of all designated VM uses. Subsequently, the overall usage of a j th PM is enumerated as follows:

$$u_j = \sum_{i=1}^{nvj} u_{ij} \quad (2)$$

When a PM's full potential usage is anticipated, its energy cost may be calculated. Figure 2 shows a typical CPU power model for an Intel-Xeon processor. It should be emphasised that the magnitude of the CPU consumption can be used to specifically depict the server power utilisation.

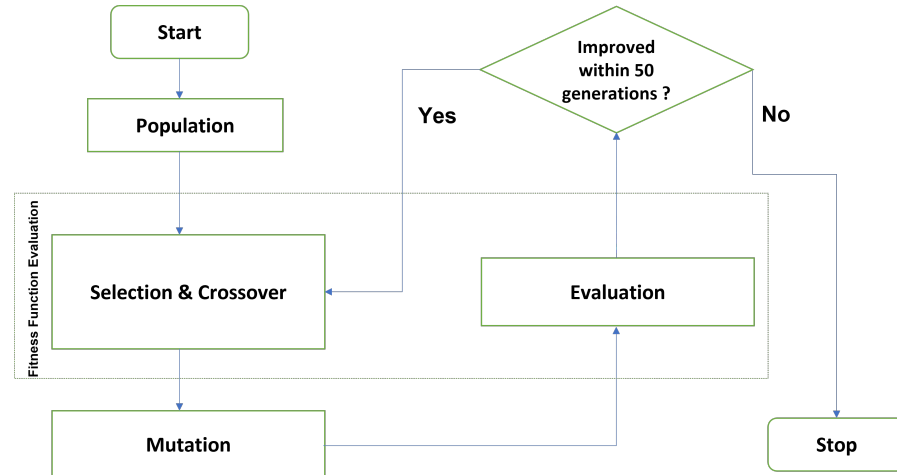


Figure 2. Schematic diagram of a genetic algorithm.

The following may be carried out quantitatively to plan a CPU power model:

$$p^{cpu} = p^{max} - (p^{max} - p^{min}) / e^{au} \quad (3)$$

where p^{min} is the base power when CPU usage is zero and p^{max} is the max working power when the CPU utilisation is 100 percent, which addresses the CPU use and shows consistency, as a result of which the power model's status is established in Figure 2. Each and every data-centre's PM has a large number of virtual machines, and these VMs routinely operate during various scheduled durations. As a result, the server data-centre's total all-in electricity cost is calculated as follows:

$$p_{total} = \sum_{j=1}^{np} p_j \quad (4)$$

Therefore, during the course of the suggested time periods, a PM's duties alter. Every VM has an initiation and end time, and it frequently operates in a different time period. As a result, different VMs operate in a comparable PM for different amounts of time. Each VM in this study is taken to have a comparable long-term duty in order to reduce the complexity of the problem. Finally, the CPU utilisation remains largely steady throughout all time intervals. Considering nT time spans over all activated VMs, the energy consumed by PMs can be evaluated in terms of individual PMs as follows:

$$e_j = \sum_{k=1}^{nt} p_{jk} t_{jk} \quad (5)$$

2.2. Constraints Plan

The improved standard focuses on assigning a basic total number of hosts assigned to all virtual machines, including projects. Therefore, idle PMs may be powered back on in the event that additional servers are required. This research makes an attempt to minimise energy usage over time while considering CPU prerequisites in an effort to improve the fog data-centres' energy efficiency. As a result, the streamlining elements restrict the resources that are available while decreasing the fitness function computation. With consideration for all PMs that contain VMs, an energy usage model for a server data-centre may be

created. Equation (6) illustrates how it combines the processing power model with the all-encompassing energy model for all dynamic PMs. As a result, a virtual resource planner can determine the forced augmentation issue by taking into account the VM configuration as follows:

$$\begin{cases} \min E = \sum_{j=1}^{np} \sum_{k=1}^n T(p_j^m ax - (p_j^m ax - p_j^m in) / e^{au_{jk}}) t_{jk} \\ s.t. u_{jk} = \sum_{i=1}^{nvj} u_{ijk} \in u_{v1}, u_{v2}, \dots, u_{nv} \\ 0 \leq u^m in \leq u_{jk} \leq u^{max} \leq 100 \end{cases} \quad (6)$$

2.3. Genetic Algorithm

As a result, a GA begins by producing the initial population, which generates better approximation solutions via generations utilising the rule of survival of the fittest. Each generation selects an individual on the basis of the compatibility of multiple solutions in precise challenging areas. A new population reflecting an innovative handful of decisions is produced when diverse solutions are found [24]. A population in a genetic algorithm is a combination of probable solution points. A generation represents an algorithmic iteration. GAs produce new offspring through a combination of selection, crossover, and mutation [20].

GAs just necessitate fitness function values related to every person in order to successfully look for endlessly better designs. Since the fitness function esteem is determined more than once in every GA age, any improvement of fitness function estimation will result in the impressively efficient generalisation of the genetic algorithm [21]. The essential benefit of utilising a GA to take care of NP-complete designation issues is that it considers a quicker convergence via looking for the arrangement space in different bearings. At the point when the GA has ended, a reasonable and further developed arrangement is always accomplished once a plausible arrangement has been procured. Figure 2 portrays the stream graph of how a fitness function in a GA functions.

The information structure described in [25] was used in this work. According to the addresses, timing, and asset circumstances, VMs are divided up into PMs in this information structure, and each PM has a specific purpose. Along these lines, we do not have to know which PMs gather which VMs. As indicated by Figure 2, a typical fitness function is shown here underneath.

$$\begin{cases} g(u_1, \dots, u_{np}; n_p) = \sum_{j=1}^{np} \sum_{k=1}^n p(p_j^m ax - (p_j^m ax - p_j^m in) / e^{au_{jk}}) \\ n_v = \sum_{j=1}^{np} n_j = \sum_{i=1}^{nvj} u_{ij} \in [0, 1], u_j \in [0, 1] \\ i = 1, \dots, n_{vj}, j = 1, \dots, n_p \end{cases} \quad (7)$$

2.4. Research Motivations

A GA has been successfully applied for optimal planning and operations in a wide variety of resource management domains, including server farm management. A key challenge in applying a GA in such variegated domains is the diversity in which the GA functions and parameters are tuned for effective deployment and functioning. However, the fitness functions that are often designed for efficient performance become computationally complex. In server farms, the VM scenario algorithm assigns different assets, such as PMs, to VMs in a manner that restricts the total amount of hosts by turning off certain servers [1].

1. Instead of working out the enormous energy recipe, we lessen the fitness function calculation by utilising a straightforward calculation to decrease the complete iteration of the GA in a timely manner;
2. Our review outlines the necessary conditions and numerical formulations to ensure a successful GA computation attempt;
3. This suggests a framework to measure and evaluate physical machines legacy holdings;

4. We reduce the complexity by limiting the number of generations (combination).

The purpose of this study is to encourage the use of an accelerated fitness function in a GA so as to reduce the amount of time spent on fitness function work calculations and, therefore, the amount of money spent on a server data-centre's execution of that work. In light of these considerations, this paper suggests a practical strategy for deploying VMs over a large variety of server configurations.

3. Analysis of the Prerequisites for the Fitness Function in a GA

This section examines the prerequisites of a fitness function in a genetic algorithm for the energy-effective fog data-centre via the use of virtual machine deployment. This will influence the advancement of another fitness function that is more straightforward than the standard fitness function Formula (7), with regards to energy utilisation and the management of time. Additionally, the numerical detailing of the fitness function work supports this study. For the sake of simplicity, the rundown of the documentation utilised in our research is presented in Abbreviations. The fitness function work assumes a basic part in directing the genetic algorithm to accomplish the best arrangements inside a more than adequate pursuit space. The phrase may also refer to the ability to assess the “fitness” or “greatness” of a recommended alternative problem when presented with several options. An awful fitness function's powers may effectively trap the GA in a near-ideal arrangement where it loses its revealing potential. In any case, great fitness function capacities will assist the GA with investigating the pursuit space more efficiently [26].

4. Essential Functioning Fitness Function Prerequisites

The prerequisites and characteristics for an effective fitness function are discussed in this section. We want to put these arrangements to the test and encourage the development of the most effective configuration of replies in order to deal with a specific problem. Therefore, each arrangement should be given a score to indicate how well it matches the prerequisites of the perfect arrangement. The test outcomes or findings from the tested arrangements are used toward the fitness function capability to obtain this score [27].

4.1. Fitness Function

A competing answer to a problem is used as input for the fitness function task, which then determines how suitable it is for the problem at hand. In a genetic algorithm, the estimation of fitness function is commonly rehashed; consequently, it ought to be adequately speedy. As indicated by our research, a fitness function capacity ought to have a truly multivariate capacity:

$$\begin{cases} f : x^{n_p+1} \rightarrow R, \\ x = (u_1, \dots, u_{n_p}; n_p), (u_1, \dots, u_{n_p}) \in r^{n_p} \\ u_j \in [0, 1], n_p \in n \end{cases} \quad (8)$$

4.1.1. Prerequisite I

In any case, lower-limited PMs need energy when they are not being used in a VM situation issue, as shown in Figure 3. Thus, a portion of the fitness function capacities should be maintained at foreordained values in certain circumstances. It ought to be underscored that energy utilisation can be depicted expressly as a size of CPU utilisation. Thus, the fitness function capacity ought to be limited and in favour of each and every quality as:

$$f(u_1, \dots, u_{n_p}, n_p) \geq \sum_{j=1}^{n_p} c_j \quad (9)$$

Formula (9) demonstrates the capacity for a virtual machine position that ought to exist. Equation (7) indicates that the base value of the normal fitness function, which is p_{np}^{min} , is also lower-limited. As a consequence of this, the primary fitness function need may be satisfied by the conventional fitness function.

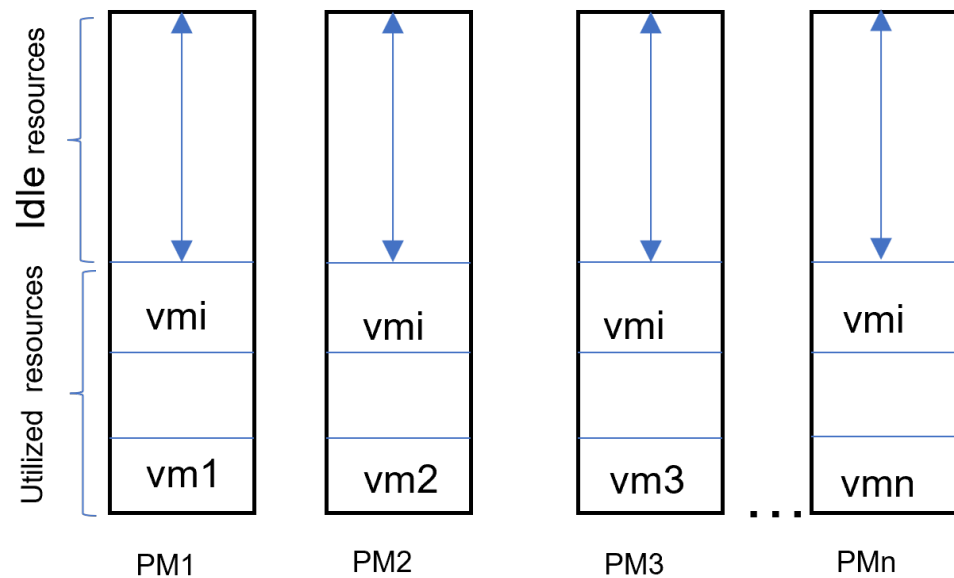


Figure 3. PMs and their utilisation and excess resources.

4.1.2. Prerequisite II

Monotonous growth as PM responsibilities grow.

For every single contribution to a fitness function, the second essential need is entirely monotonous. When solving a problem that involves the deployment of virtual machines, the fitness function is put to use to make an estimate of the amount of energy that is used in a fog data-centre. As physical machine energy consumption increases, the energy use rises in a monotone manner (See Figure 1). Consider the following two cases where the number of hosts in a virtual machine environment are rather comparable. The extension of one scenario's utilisation of the final PM indicates that the fitness function (energy proficiency) has progressed at that moment. To put it more precisely, we believe that the following circumstance should apply if we have a health study $f(u_1, \dots, u_{np}, n_p)$. For every $j = 1, \dots, n_p$,

$$f(u_1, \dots, u_j + \Delta u_j, \dots, u_{np}, n_p) > f(u_1, \dots, u_j, u_{np}, n_p) \quad (10)$$

Which demonstrates that as the responsibility builds, a worthy fitness function ought to increment monotonically. Additionally, n_p is the precise number of PMs, while u_j addresses the full integration of VMs into the j th PM. As a result, for the typical fitness function, we should have:

$$\begin{aligned} & \sum_{j=1}^{np} (p_j^m ax - (p_j^m ax - p_j^m in) e^{-\alpha u_j + \Delta u_j}) \\ & > \sum_{j=1}^{np} (p_j^m ax - (p_j^m ax - p_j^m in) e^{-\alpha u_j}) \end{aligned} \quad (11)$$

It is no different for some other $PM(j = 1, \dots, np^{-1})$.

$$\begin{aligned}
& \sum_{j=1}^{np} (p_j^m ax - (p_j^m ax - p_j^m in) e^{-\alpha u_j + \Delta u_j}) \\
& - \sum_{j=1}^{np} (p_j^m ax - (p_j^m ax - p_j^m in) e^{-\alpha(\Delta u_{np}) + u_{np}}) \\
& = \sum_{j=1}^{np} (p_j^m ax - (p_j^m ax - p_j^m in) e^{-\alpha u_j + \Delta u_j}) - \\
& \sum_{j=1}^{np} (p_j^m ax - (p_j^m ax - p_j^m in) e^{-\alpha u_{np}}) = \\
& - (p_j^m ax - (p_j^m ax - p_j^m in) e^{-\alpha(\Delta u_{np})}) =
\end{aligned} \tag{12}$$

where $p_{np}^{max} > p_{np}^{min}$, α , Δu_{np} and u_{np} represent a positive value.

It is also applicable to any other $PM(j = 1; ::; np - a)$.

4.1.3. Prerequisite III

Consistently growing as the number of PMs increases.

Furthermore, a fitness function must have a diminishing fitness work value when more servers are added. With a growing number of data-centre physical machines (PMs), the fitness function value should drop, since this creates a VM location problem. Assume that the quantity of PMs increased by one; therefore, for every PM, the fitness function expands and brings more energy consumption to the server farms. To keep the complete usage similar, the use of this recently added PM will diminish u_{np} by Δu_{np} and we ought to have

$$f(u_1, \dots, u_{np} - \Delta u_{np}, \dots, u_{np}, n_{p1} + 1) > f(u_1, \dots, u_j, u_{np}, n_p) \tag{13}$$

With a single server addition, we found the following for the GA's standard fitness function:

$$\begin{aligned}
& g(u_1, \dots, u_{np} - \Delta u_{np}, \Delta u_{np}, n_{p+1}) \\
& = \sum_{j=1}^m ax (p_j^m ax - (p_j^m ax - p_j^m in) e^{\alpha u_j}) \\
& + p_{np}^m ax - (p_{np}^m ax - p_{np}^m in) e^{-\alpha(u_{np}) - \Delta u_{np}} \\
& + p_{np+1}^m ax - (p_{np+1}^m ax - p_{np+1}^m in) e^{-\alpha \Delta u_{np}}, \\
& \text{and } g(u_1, \dots, u_{np}, n_p) = \sum_{j=1}^{n_p} -1 - (p_j^m ax - p_j^m in) e^{-\alpha u_j} + p_{np}^m ax - (p_{np}^m ax - p_{np}^m in) e^{-\alpha u_{np}}, \\
& \text{where } 1 \geq u_{np} > u_{np} - \Delta u_{np} \geq 0.
\end{aligned}$$

Assume that when the no. of servers has increased by one, $p_{np}^{max} = p_{np+1}^{max}$ and $p_{np}^{min} = p_{np+1}^{min}$. As stated in the concave function definition, g_0 is a concave one. With $g_0 = p_{np}^m in > 0$, g_0 is also a sub-additive function as follows: $0 \leq x_1 < x_2 \leq 1$, $g_0 x_1 + g_0 x_2 \geq g_0(x_1 + x_2)$; therefore we can obtain:

$$\begin{aligned}
& g(u_1, \dots, u_{np} - \Delta u_{np}, n_p + 1) - g(u_1, \dots, u_{np}, n_p) \\
& = g_0(u_{np} - \Delta u_{np}) + g_0(\Delta u_{np} - g_0(u_{np})) > 0
\end{aligned} \tag{14}$$

This demonstrates that as the total amount of physical machines has grown, so too has the value of the fitness work, and thus, the energy utilisation. The conventional fitness function now clearly fits the third fitness function requirement.

4.1.4. Prerequisite IV

Increasing with an increasingly fair distribution of duties. In a VM situation issue, a fitness function capacity ought to be more terrible (greater) when the conveyance of the responsibility is all the more terrible in any event for a similar number of servers. In a similar vein, the fitness function's capacity and energy consumption both rise when each PM in a server farm is equally responsible for its own power and cooling systems, even if we save more money by utilising more PMs. An improvement in the fitness function

value and greater efficiency in the use of energy are both the result of a more uniformly distributed workload. This means that the fourth condition for a conventional fitness function is currently enough for our needs.

4.2. The Suggested Fitness Function

The recommended fitness function task, which is essentially simpler than the usual one, Equation (7), is presented in this section after taking into consideration the needs analysis provided in Section 3. To measure the overall practicability of the prospective arrangements in a certain application, a fitness function task is a specific target activity. The fitness function estimation essentially dials back the GA as it must be executed a great deal of times in each instance. In such a manner, we plan the VM situation issue by applying to address the problem of energy efficiency in cloud data-centres; therefore, a fitness function for the GA is required.

The residual resources are the unused resources that may be accessed via the PMs in the data-centre's server farms [28]. Working on the convenience of the remaining assets in the cloud server farm altogether affects the expanding energy productivity and execution time. Assuming that the leftover assets are put away in less dynamic PMs, such resources can be proficiently utilised for provisioning new VM assignment demands.

Moreover, expanded resource usage has been acquired by diminishing the leftover assets. This suggests that when PMs are utilised more frequently, fewer dynamic PMs are present. In this study, the Taylor extension is used to promote a different fitness function computation that is straightforward and eco-friendly in a cloud data-centre's virtual machine configurations. Appropriately, the proposed measure of the remaining assets diminishes, prompting expanded energy utilisation in the cloud data-centre.

4.2.1. Prerequisite I

The suggested fitness function must be lower-limited, which is the first prerequisite. Limiting a fitness function's capacity is a necessary condition, as stated by Equation (9). Similarly, arguments in favour of this criterion are met by the suggested fitness function, since there is a unique numerical representation for every value in the fitness function's domain. The inactive state power failure of the PM's components causes a restricted and usually unique power scope of a PM at 30 percent. This indicates that a PM is permitted to use up to 70% of its maximum power, even when it is completely dormant [5]. The suggested fitness function only allows for u_j to fall between 0 and 1, meaning that the j th PM is always used within that range. The amount of power used by the PM may be represented directly in terms of CPU size. The minimum amount of idle resources in our VM placement problem is the one at which CPU usage is at its lowest. In addition, the largest amount of idle resources is available when the CPU is not being used at all. As with other lower-limited fitness functions, the one presented here is:

That maximum CPU utilisation u_j corresponds to a point when the fitness function has a lower-limit esteem. We said above that a PM's greatest power drain occurs when it is doing nothing at all (See Figure 3). Equation (16) indicates that the proposed fitness function has a minimum number as well as a minimum value.

4.2.2. Prerequisite II

Mandatory second prerequisite: the estimated fitness function task will drop with time as the PM's duties grow. Our suggested fitness function satisfies the second prerequisite of a general fitness function. According to Equation (10), the use of PMs should be in wrinkle with respect to energy consumption (input). However, in our suggested fitness function, the values that matter are those that reflect a sustained energy expenditure across time. Therefore, in a VM position problem, we may assume that the consumption of energy grows steadily with each additional use of PMs. Figure 3: (for illustration purposes only).

This means that when demand increases, idle resources are depleted, resulting in higher server farm energy efficiency [19]. Depending on the results of the fitness function,

the suggested exercise regime should either expand monotonically with u_j or decrease monotonically with u_j . Thus, we might prove beyond a reasonable doubt the fact that the proposed fitness function meets the following condition.

4.2.3. Prerequisite III

The anticipated fitness function expands monotonically while the total number of physical machines continues to rise. The third condition for a fitness function in a virtual machine position issue in a server is that the work value must increase in a manner that is monotonically proportional to the number of PMs (n_p). Assume the absolute pursuit is uniform in our virtual machine layout technique. With the same level of consumption and an increase in PMs, the predicted fitness function deteriorates. Then, we will be able to obtain the following.

The standard fitness function prerequisite III dictates that as the number of physical machines (PMs) in a server increases, the value of the suggested fitness function ought to decrease as the number grows (expand). In our virtual machine position procedure, the overall use is steady and if we assume that the quantity of PMs in a single circumstance increases by one, so too do we see a similar usage for every PM, the fitness function esteem (energy utilisation) increases. Accordingly, it is obvious that this is accomplished by the suggested fitness function necessity of three.

4.2.4. Prerequisite IV

The suggested fitness function improves as more responsibilities are shared among participants. A fitness function's first and most basic prerequisite is that, for a given number of potential members (PMs), the fitness work value should decrease as the usage spreads. The use of PMs is especially critical when dealing with the virtual machine (VM) state of affairs in a cloud-based data-centre.

We demonstrated that the fitness function esteem decreases and the remaining assets of PMs increase when utilisation is distributed evenly among them in a VM scenario, leading to a decrease in energy efficiency. With the same amount of PMs used in the server farm, the fitness function becomes wrinkly. When the energy efficiency of a server farm increases in tandem with the increased use of each PM in a virtual machine (VM) environment, the situation is reversed. As a result, the suggested fitness function satisfies the prerequisites of the gold standard fitness function. In this paper, we use the Taylor extension to provide a novel approach to estimating the fitness function that reduces energy consumption and facilitates verification in distributed cloud computing infrastructures.

4.3. Using the Taylor Expansion to Derive the Proposed Fitness Function

Additionally, from a quantitative standpoint, there is significant support for the suggested fitness function. The Taylor development theory is applied to the average fitness function capacity, ignoring greater levels of well being.

Request terms bring about accelerated fitness functions. Appropriately, the approached fitness function recipe can be obtained from the standard fitness function referenced in (7) as given below, $g(g_1, \dots, u_{np}; n_p) = \sum_{j=1}^n p_j^m ax - (P_j^m ax - P_j^m in) / e^{\alpha u_j}$. Consequently, the Taylor expansion shows that the average fitness function is about $g(g_1, \dots, u_{np}; n_p) = \sum_{j=1}^n p_j^m ax - \sum_{j=1}^n p_j^m (p_j^m ax - p_j^m in) + \sum_{j=1}^n p_j^m (p_j^m ax - p_j^m in) \alpha u_j - \frac{1}{2} \sum_{j=1}^n p_j^m \alpha^2 (p_j^m ax - p_j^m in) u_j^2$.

As a result, the suggested fitness function's Taylor expansion may be found in the manner shown below.

$$f(u_1, \dots, u_{np}; n_p) = \sum_{j=1}^n p_j^m in - \sum_{j=1}^n p_j^m (p_j^m ax - p_j^m in) u_j^2 \quad (15)$$

In our VM circumstance issue, Condition (11), which derives from Equation (15), expresses the normalised lingering assets of PMs. It implies that when the usage expands, the

standardised lingering assets monotonically diminish. Therefore, our proposed normalised fitness function is introduced in Equation (12),

$$r_{total} = \sum_{j=1}^n p(1 - u_j^2), \quad (16)$$

where r_{total} alludes to the complete normalised lingering assets of PMs. In the equivalent, the mathematical definition of our suggested fitness function job Equation (15) decreases when consumption grows monotonically.

Method for estimating the power consumption of an Intel Xeon-based system's CPU [19] is a demonstration of a claimed fitness function. Standardised lingering assets may be identified by the fact that the Y-pivot has changed from 0 to 1. Whenever u is 1, the suggested fitness function reduces the remaining asset to zero, and whenever u is 0, the remaining asset increases to one. The leftover resources diminish when the use is increased and there is just a single lingering comparison asset for every usage. This implies that the proposed fitness function $1 - u^2$ is monotonically decreasing.

Equation (16) may be used to construct VM scenario designs, whereas Equation (15) can be used to obtain the energy use in Equation (7). The primary operation itemisation target for this test is Equation (16), which encourages improved asset usage.

Low-power efficiency in cloud data-centres within a shorter time frame (in terms of both PMs and years), offer rapid fitness function computation and absolute GA execution.

Moreover, the proposed fitness functions obviously satisfy the numerical necessities as a whole, and the third section has established new imperatives.

4.4. Simplifying the Proposed Fitness Function Even Further

We found a comparable fitness function from Taylor's fitness feature from a previous fitness function. In order to make further rearrangements, Formula (15) represents our new fitness function.

5. Design and Complexity of the Algorithms

Here we describe a calculation for the suggested fitness function for the genetic algorithm from a virtual machine situation in a data-centre. The accompanying calculation is intended to further develop the fitness function calculation in the genetic algorithm for the blueprint of a given virtual machine situation for the data-centre. The output of the computation is a virtual machine scenario scheme that details how each virtual machine will be used along with its purported PM. This computation yields the virtual machine's situation plan. Each virtual machine's usage is added to its linked physical machine when a physical machine's use list has been configured. Consequently, a breakdown of a physical machine's total CPU consumption is made available. In the unlikely event that the physical machine's use is not zero, it will then be possible to calculate the PM's remaining resources.

The following VMs are given all of the duties, and additional virtual machines are added to the PMs via the GA. Once Algorithm 1 has provided the workload distribution and the VM deployment strategy, the energy consumption may be calculated using Algorithm 2.

We utilised common setups Table 1 to the pick parameter settings since this post is not about parameter settings. Our GA's population is made up of 64 individuals. A reduced population size should result in a faster convergence; however, the opposite is true for big population numbers. Furthermore, our GA uses an elite developed using FFD [9] to minimise the total number of generations.

Upon obtaining the virtual machine's position strategy, the whole energy use of a virtual machine scenario plan may then be determined using standard energy utilisation Equation (7). While both the normal GA and our GA are typically complex, our GA just has to complete a simple assessment of the amount of fitness function: Equation (20), as opposed to performing the enormous standard fitness function computation Formula (7). In this way, the intricacy of the suggested fitness function is written in the following manner:

n_v represents the absolute number of virtual machines and n_p addresses the quantity of dynamic physical machines in virtualised data-centres. Subsequently, our proposed fitness function makes temporal complexity that is linear. The intricacy of the fitness function computation has been diminished in this concentration by utilising a straightforward fitness function for leftover resources; therefore, decreasing the energy utilisation and reducing the quantity of physical machines and production.

Algorithm 1: Formulation of a Fitness Function Computation

Input: A strategy for deploying virtual machines that details where and how each one will be used.

Output: The sum of the plan's potential leftover resources.

Initialisation: PM usage set with no items.

Do this for each virtual machine in the specified strategy.

Increase the use of the located PM by including this VM.

Change the status of any applicable active PMs.

end for

Do the following for each instance of using PMs that are shown in the PM uses table:

Use of PMs, If 0, then

To what extent may PMs be used once all other resources have been exhausted?

Include this PM's leftover resource when calculating the sum of all leftovers.

end if

end for

Return the sum of the plan's potential leftover resources.

Algorithm 2: Evaluation of the Power Consumption

Input: We have a whole new strategy for deploying our virtual machines, complete with details on how and where each one will be used.

Output: Total amount of energy needed for this strategy.

Initialisation: PM usage set with no items.

Do this for each virtual machine in the specified strategy.

Increase the use of the located PM by including this VM.

Increase the use of the located PM by including this VM.

end for

Do the following for each instance of using PMs that are listed in the PM uses table:

If there is no use of PMs, then

Please use the following Equation (6) to determine the power draw of this PM.

Take into account the PM's energy needs in addition to the plan's potential fitness benefits.

end if

end for

Return the total amount of energy used.

The dynamic PMs are all expected to have VMs loaded with computational undertakings, and the server farm's complete control throughout the evaluation and forecasting phases of the hypothetical property distribution uses the suggested fitness function for the genetic algorithm. This exploration analyses various sizes of server farms for carrying out replication tests. Our re-enactment tests are carried out on a work area computer. The PC is furnished with Intel Core2 i7-9650, 2.341 gigahertz and 32 GB RAM 2866 megahertz. It runs on a Windows operating system and Eclipse 4.25. The Java development kit was used to create our program in 1.9.0.171. The recreation results are applied to assess the exhibition of the suggested fitness formula procedure for using the genetic algorithm in computing

data-centres. Additionally, a collection of one-time approaching tasks is assembled using the task information that was taken from the cluster traces provided by Google. A normal comparison of the referred trials is used to introduce the result.

Additionally, VMs ought to be allocated to the approaching processing undertakings first. In this way, we really want to incorporate Google’s information from all assignments into VMs utilising Algorithm 3. Given that VMs are assigned tasks, these new VMs ought to be added to PMs via the GA. Algorithm 2 may then be used to calculate the energy usage after receiving the job as well as the VM’s organisation plan from Algorithm 1.

Algorithm 3: Virtual Machine (VM) Task Allocation

I/P: Applications for all tasks.

O/P: A strategy for assigning applications to virtual machines.

Initialisation: A set of vacant virtual computers are employed for each work for each task that is available.

A VM of the proper size should be given this job.

Add this VM to the existing group of VMs.

end for

Perform the assignment plan conversion on the VM set.

Return the assignment plan for the application.

Sixty-four percent makes up our genetic algorithm’s population. Our genetic algorithm combines a kind of parochialism that is created using FFD [7] in the promotion edition to reduce the overall age distribution. Elitism has no impact on the evaluation of a candidate’s level of fitness during testing. Each person is taken into account while creating a placement strategy for the VM. In the activity that requires decision, the competition determination approach [22] is chosen. The competition group should first be randomly chosen, and then its size should be changed. For the hybrid activity [29], which is 0.5 in this paper, a uniform hybrid is used. Every trait is arbitrarily chosen in the uniform hybrid, either from the first or second parent [30]. The change activity is utilised to maintain the hereditary variety, starting with one age of a populace and then moving onto the next one. Using this strategy, each VM has more opportunities to be transformed into a superior PM. In this manner, the versatile change plot with a mutation pace of 0.015 is utilised [31]. At last, the GA ends when more than 50 ages play out with no upgrades.

Table 1. Parameter table.

Parameters	Scale
Count	55
Dimensions	13
Exclusiveness	true
Equal Tariff	0.5
Mutation Rate	0.015
Runs	100
Maximum Iterations	1000

6. Dataset

In order to mimic the computing workloads for the design of our VM scenario, we use the publicly accessible Google cluster-usage traces, which come from a non-virtualised server. Cluster-usage traces from Google should be combined with forethought in re-enactment research. Even though every assignment and occupation is listed, the borders and arrangements of the projects, the asset solicitations, and the asset part in the traces are all accurate. A month’s worth of server farm data are included in the Google cluster-usage traces [23]. Within one estimating period, which is 5 min in Google’s cluster-usage traces, incoming tasks (responsibility) should be split amongst VMs and then put into PMs. As information records are enormous for a period of a month in a server farm, this article only

evaluates information records for a period of around 24 h. Table 2 lists the precise Average. The small, medium, and large scope informational indexes are first designed with generally identical PMs without any accountability, but then they take into account the power model depicted in Formula (3), which pursues the path promised in [19].

Table 2. Parameter table.

Cluster Trace (G)	Small	Medium	Large
Assignment of Virtual Machines	1200	2400	4500
Physical Machines	800	1000	2000

7. Results

The evaluation outcomes of all relevant experiments are covered in this section, taking into account datasets of various sizes (small, medium, and large). Apart from the extent of the informative index's contribution, all parameters in all sizes are comparable: the GA with fitness function energy calculation Formula (6), the GA with our novel fitness function calculation Formula (20), and the calculation of FFD as force seat characteristics. The standard GA and FFD complete our GA when it is over, giving their VM scenario options. Their presentation is recorded and decided upon. Below is a display of the results.

The time for fitness function computation: The flow paper's examination centres around lessening the fitness function computation time. Figure 4 represents what amount of time the fitness function requires when contrasted with the fitness function season of the standard GA and the one with the proposed fitness function.

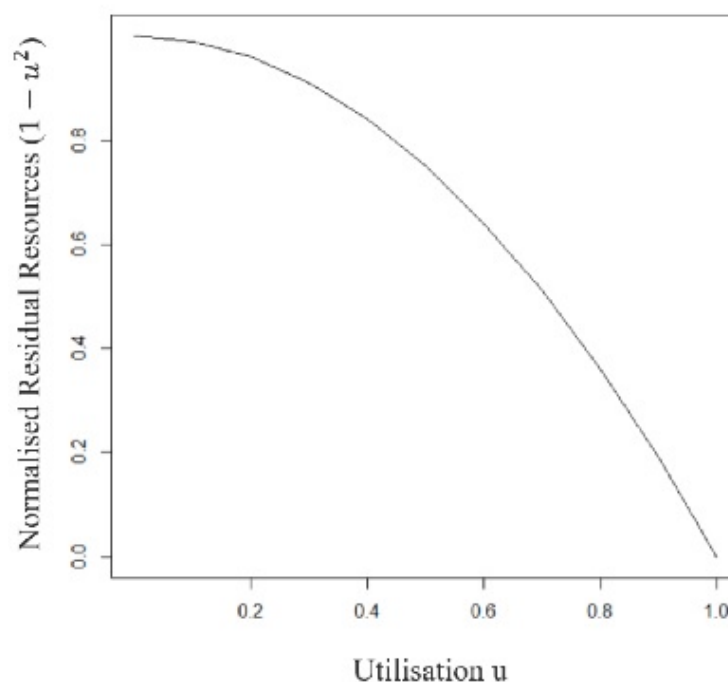


Figure 4. Taylor expansion, a new approach to the fitness function.

As such, we want to cut down on the amount of time needed to estimate the fitness function in order to speed up the GA. Therefore, the suggested fitness function capacity may drastically reduce the time required in order to compute the fitness function, in comparison to the regular fitness function. In addition, compared to the traditional GA, the suggested fitness function shortens the duration of its overall execution.

7.1. The Normal GA Execution Time in Little, Medium, and Large Scopes

In contrast to the conventional fitness function job, Figure 5 shows that the proposed fitness function's absolute execution time for the GA is lowered by 66% for the large scope informative index. Our updated fitness function cuts down on the GA's absolute calculation time. This affirms that our progression in energy streamlining is successful. The new structure turns out to be quicker than the standard GA. In this manner, the results in Figure 6 demonstrate that our system is able to maintain a controlled and appropriate calculation time.

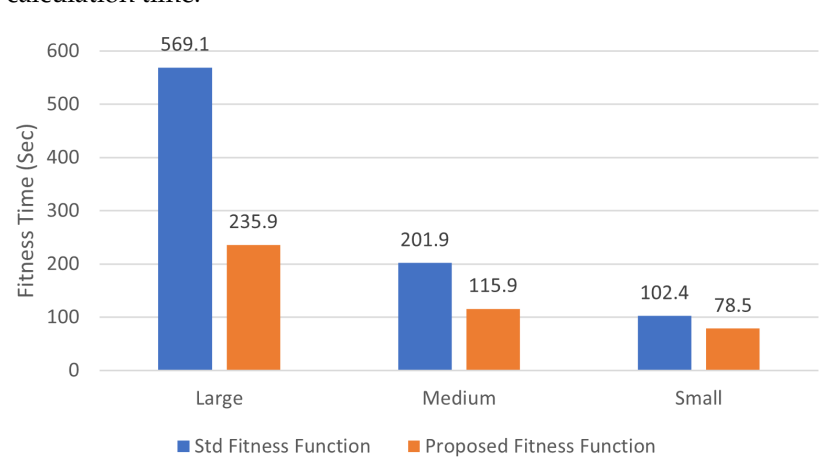


Figure 5. Fitness time comparison.

7.2. The Typical Number of Active PMs in Each Scope Size, Including Small, Medium, and Large

The results of the tests of various sizes in a cloud data-centre during a new development stage with the anticipated fitness function are given in Figure 6 for the number of dynamic PMs. As demonstrated in Figure 6, the GA with the recommended fitness function often utilises fewer PMs than the regular GA and FFD. We start with 3000 PMs in the vast scope informative index. Next, when we conduct our GA, we only use 1086 PMs. Unlike a typical GA, we can turn off a far larger number of PMs.

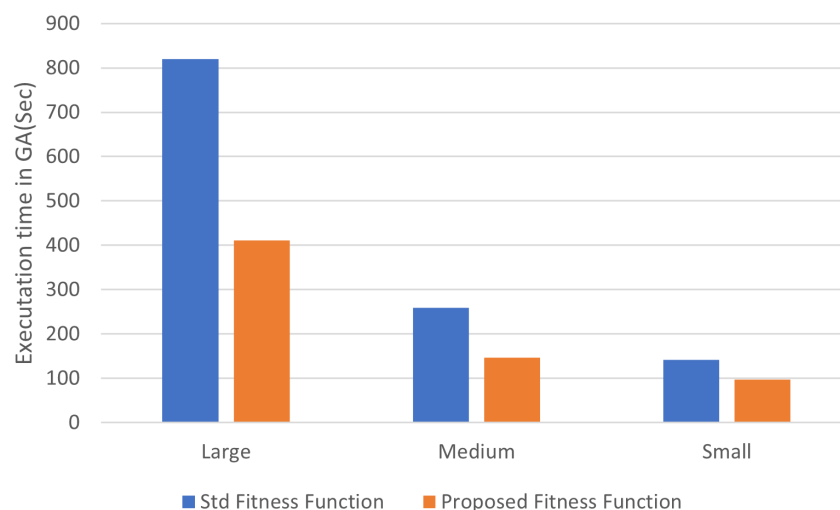


Figure 6. Average execution time for the GA.

7.3. In Small, Medium, and Large Scopes, the Typical Generational Count

In a genetic algorithm, energy improvement is possible with fewer generations. The amount of genetic algorithm ages in a virtual machine placement scheme may be reduced thanks to the suggested fitness function capability, as shown in Figure 4.

The genetic algorithm calculates the fitness function score across all ages. In this article, compared to the traditional genetic algorithm, the age range implemented for the proposed fitness function of the genetic algorithm is demonstrably smaller. According to these findings in Figures 7–9 our energy improvement procedure’s VM scenario plan is far superior to the FFD and traditional GA.

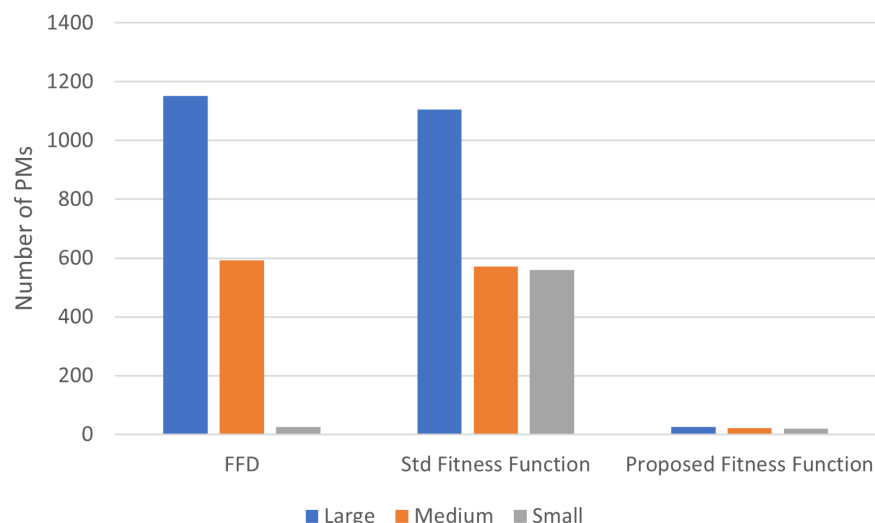


Figure 7. Average number of PMs.

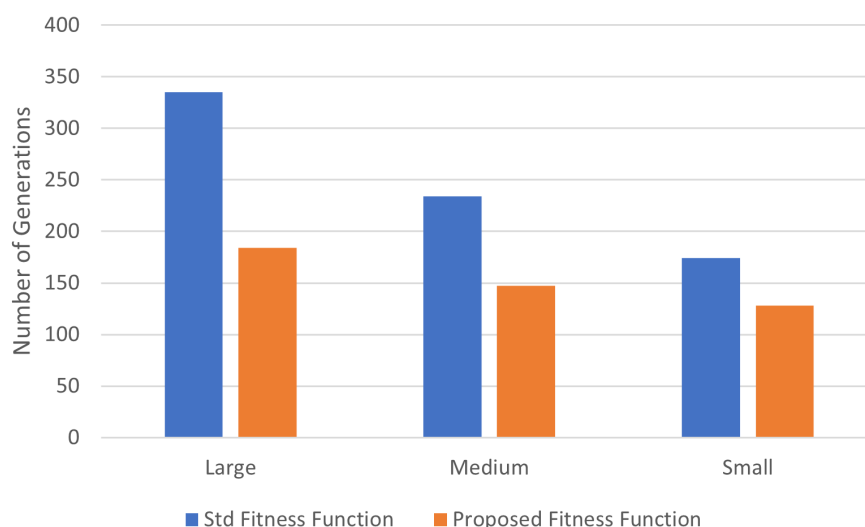


Figure 8. Average number of generations.

7.4. The Typical Energy Measurement in Small, Medium, and Large Scopes

The equation used in the representation of the conventional GA is Formula (7). With the use of the Taylor extension for the fitness task estimate equation, our GA aims to increase the energy efficiency in the cloud data-centre (15). This study used the traditional GA and FFD computation as a touchstone to compare our GA with a proposed fitness function capacity.

In the fog server data-centre, our GA’s energy usage and the typical GA are compared in Figure 10. However, our new technique has diminished the energy utilisation by about 5 percent, which contrasts with FFD.

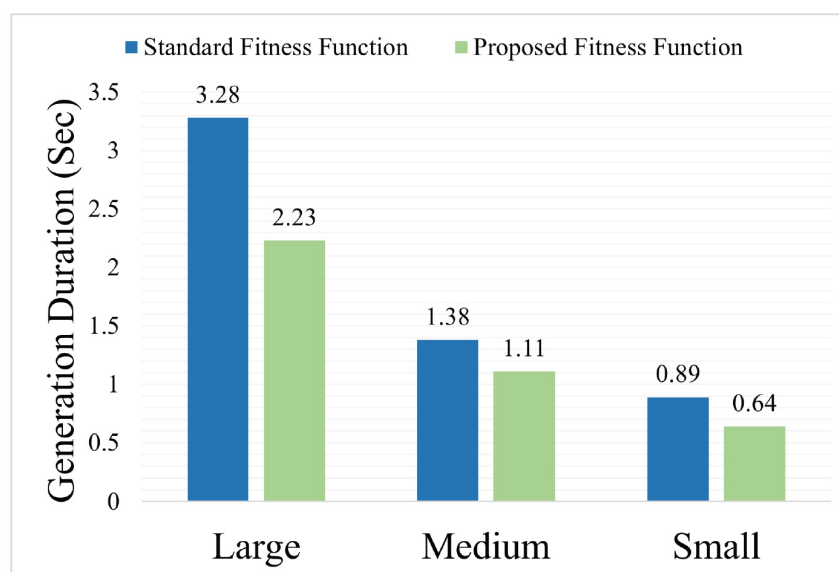


Figure 9. The typical lifespan of a generation.

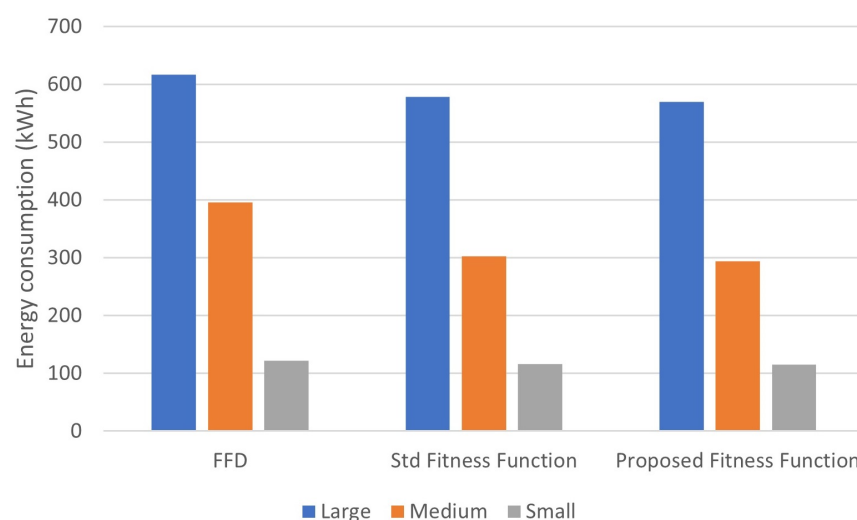


Figure 10. Scales and algorithms for calculating the data-centre power usage over the course of a full day.

8. Conclusions

To achieve greater energy efficiency in the fog layer's virtualised ICT infrastructure, this paper proposes a novel residual resource-based fitness function for developing an improved GA for the VM placement problem. The proposed residual resource-based fitness function is computationally much simpler than its conventional GA and FFD-GA counterparts, and is capable of helping reduce the number of required generations and active PMs, as well as the overall energy usage of the data-centre and the convergence time of the proposed GA. The experimental results obtained using Google data-centres' data demonstrate that the proposed GA saves around 8% of energy via-a-vis FFD-GA and it executes the algorithm 66% faster than the standard GA for virtualised data-centres.

As a consequence, reducing the total amount of PMs in data-centres in the cloud would improve energy efficiency. The favourable outcomes produced with the proposed technique demonstrate that our GA may be effectively employed in real-world data-centres. Because of its decreased energy consumption, it might be employed in sustainable computing centres.

Author Contributions: Conceptualization, S.C.; Methodology, J.J.P.C.R.; Software, D.S.R.; Validation, D.S.R.; Formal analysis, A.K.L., J.J.P.C.R., U.G. and D.S.R.; Investigation, A.K.L. and U.G.; Resources, M.A.-N.; Data curation, M.A.-N.; Writing—review & editing, S.C. All authors have read and agreed to the published version of the manuscript.

Funding: This research was supported by FCT/MCTES through national funds and, when applicable, was co-funded by EU funds under the Project UIDB/50008/2020; and by the Brazilian National Council for Scientific and Technological Development-CNPq, via Grant No. 313036/2020-9 and also supported by Researchers Supporting Project Number (RSP2023R150), King Saud University, Riyadh, Saudi Arabia.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: Data sharing is not applicable to this article as no datasets were generated or analysed during the current study.

Conflicts of Interest: The authors declare no conflict of interest.

Abbreviations

The following abbreviations are used in this manuscript:

Symbols	Meaning
p_{total}	Overall data-centre power
e_{total}	Overall consumption of energy
s_{total}	Total removed constants from energy total
e	Optimisation of energy
n_t	Time period
k	Time slot
p^{cpu}	The power of cpu
$f; g$	Fitness function
$f_0; g_0$	Defined function
x	Fitness function input
r_{total}	Overall residual resources
$p^{min};$	Base power
$p_j^{max}; p_j^{min}$	Max, min j th PMs power
$p_{np}^{max}; p_{np}^{min}$	Max, min PMs n th PM power
$p_{np+1}^{max}; p_{np+1}^{min}$	Max, min $(np + 1)$ th PM power
r	Fitness function output as real number
n	Approximate number of physical machines
$e_j; p_j$	j th PM's energy and power
t	Rate of change of the virtual machine's duration
t_j	Rate of change of the j th virtual machine's duration
u	Overall use of processor time
$u^{ij} j$	Use of the i th VM in j th PM
u_{jk}	Using the j th PM in the k th time slot
$u^{max}; u^{min}$	Use of the machine at its max and lowest capacity
$n_v; n_p$	Quantity of virtual and activated physical machines
$u_{nv}; u_{np}$	Use of the n th VMs with n th hosts
α	Constant representing of the distribution of the cpu speed
c_j	Parameter for the fitness function constant lower-bounded
$p_j^{wcpu}; p_j^{pcpu}$	Processing Time and CPU load for the j th PM
n_{vj}	Proportion of VMs running on the j th host
n_{vjk}	Nos. of VMs active in the j th period in time slot k th
$V; V_i$	Composed of every virtual machine and the i th VM
$\Delta u_j; \Delta u_{np}$	Using the j th and (np) th hosts

References

1. Bibri, S.E. The IoT for smart sustainable cities of the future: An analytical framework for sensor-based big data applications for environmental sustainability. *Sustain. Cities Soc.* **2018**, *38*, 230–253. [\[CrossRef\]](#)
2. Bouzguenda, I.; Alalouch, C.; Fava, N. Towards smart sustainable cities: A review of the role digital citizen participation could play in advancing social sustainability. *Sustain. Cities Soc.* **2019**, *50*, 101627. [\[CrossRef\]](#)
3. Xia, X.; Wu, X.; BalaMurugan, S.; Karuppiah, M. Effect of environmental and social responsibility in energy-efficient management models for smart cities infrastructure. *Sustain. Energy Technol. Assessments* **2021**, *47*, 101525. [\[CrossRef\]](#)
4. Amirtharaj, I.; Groot, T.; Dezfouli, B. Profiling and improving the duty-cycling performance of Linux-based IoT devices. *J. Ambient. Intell. Humaniz. Comput.* **2019**, *11*, 1967–1995. [\[CrossRef\]](#)
5. Hormozi, E.; Hu, S.; Ding, Z.; Tian, Y.-C.; Wang, Y.-G.; Yu, Z.-G.; Zhang, W. Energy-efficient virtual machine placement in data centres via an accelerated Genetic Algorithm with improved fitness computation. *Energy* **2022**, *252*, 123884. [\[CrossRef\]](#)
6. Kozłowski, A.; Sosnowski, J. Energy Efficiency Trade-Off Between Duty-Cycling and Wake-Up Radio Techniques in IoT Networks. *Wirel. Pers. Commun.* **2019**, *107*, 1951–1971. [\[CrossRef\]](#)
7. Mishra, J.; Sheetlani, J.; Reddy, K.H.K. Data center network energy consumption minimization: A hierarchical FAT-tree approach. *Int. J. Inf. Technol.* **2018**, *14*, 507–519. [\[CrossRef\]](#)
8. Yang, G.; Shin, C.; Lee, J.; Yoo, Y.; Yoo, C. Prediction of the Resource Consumption of Distributed Deep Learning Systems. *Proc. ACM Meas. Anal. Comput. Syst.* **2022**, *6*, 1–25.
9. Dodge, J.; Prewitt, T.; Combes, R.T.D.; Odmark, E.; Schwartz, R.; Strubell, E.; Luccioni, A.S.; Smith, N.A.; DeCario, N.; Buchanan, W. Measuring the Carbon Intensity of AI in Cloud Instances. In Proceedings of the 2022 ACM Conference on Fairness, Accountability, and Transparency, Seoul, Republic of Korea, 21–24 June 2022. [\[CrossRef\]](#)
10. Rbii, E.; Jemili, I. Leveraging SDN for smart city applications support. In *International Workshop on Distributed Computing for Emerging Smart Networks*; Springer: Berlin/Heidelberg, Germany, 2020; pp. 95–119.
11. Hussain, M.W.; Reddy, K.H.K.; Rodrigues, J.J.P.C.; Roy, D.S. An Indirect Controller-Legacy Switch Forwarding Scheme for Link Discovery in Hybrid SDN. *IEEE Syst. J.* **2020**, *15*, 3142–3149. [\[CrossRef\]](#)
12. Mastelic, T.; Oleksiak, A.; Claussen, H.; Brandic, I.; Pierson, J. M.; Vasilakos, A.V. Cloud computing: Survey on energy efficiency. *ACM Comput. Surv.* **2014**, *47*, 1–36. [\[CrossRef\]](#)
13. Yamada, N.; Takeshita, H.; Okamoto, S.; Sato, T. Using Optical-Approaches to Raise Energy Efficiency of Future Central and/or Linked Distributed Data Center Network Services. *Int. J. Netw. Comput.* **2014**, *4*, 209–222. [\[CrossRef\]](#) [\[PubMed\]](#)
14. Roy, D.S.; Behera, R.K.; Reddy, K.H.K.; Buyya, R. A Context-Aware Fog Enabled Scheme for Real-Time Cross-Vertical IoT Applications. *IEEE Internet Things J.* **2018**, *6*, 2400–2412. [\[CrossRef\]](#)
15. Behera, R.K.; Reddy, K.H.K.; Roy, D.S. A Novel Context Migration Model for Fog-Enabled Cross-Vertical IoT Applications. In *International Conference on Innovative Computing and Communications: Proceedings of ICICC 2019*; Springer: Singapore, 2019; pp. 287–295. [\[CrossRef\]](#)
16. Reddy, K.H.K.; Behera, R.K.; Chakrabarty, A.; Roy, D.S. A Service Delay Minimization Scheme for QoS-Constrained, Context-Aware Unified IoT Applications. *IEEE Internet Things J.* **2020**, *7*, 10527–10534. [\[CrossRef\]](#)
17. Chen, Q.; Grosso, P.; van der Veldt, K.; de Laat, C.; Hofman, R.; Bal, H. Profiling Energy Consumption of VMs for Green Cloud Computing. In Proceedings of the 2011 IEEE Ninth International Conference on Dependable, Autonomic and Secure Computing, Sydney, NSW, Australia, 12–14 December 2011; pp. 768–775. [\[CrossRef\]](#)
18. Reddy, K.H.K.; Luhach, A.K.; Pradhan, B.; Dash, J.K.; Roy, D.S. A genetic algorithm for energy efficient fog layer resource management in context-aware smart cities. *Sustain. Cities Soc.* **2020**, *63*, 102428. [\[CrossRef\]](#)
19. Li, Z.; Yan, C.; Yu, X.; Yu, N. Bayesian network-based Virtual Machines consolidation method. *Futur. Gener. Comput. Syst.* **2017**, *69*, 75–87. [\[CrossRef\]](#)
20. Hallawi, H.; Mehnen, J.; He, H. Multi-Capacity Combinatorial Ordering GA in Application to Cloud resources allocation and efficient virtual machines consolidation. *Futur. Gener. Comput. Syst.* **2017**, *69*, 1–10. [\[CrossRef\]](#)
21. Jiang, J.; Feng, Y.; Zhao, J.; Li, K. DataABC: A fast ABC based energy-efficient live VM consolidation policy with data-intensive energy evaluation model. *Future Gener. Comput. Syst.* **2017**, *74*, 132–141. [\[CrossRef\]](#)
22. Yu, L.; Chen, L.; Cai, Z.; Shen, H.; Liang, Y.; Pan, Y. Stochastic Load Balancing for Virtual Resource Management in Datacenters. *IEEE Trans. Cloud Comput.* **2016**, *8*, 459–472. [\[CrossRef\]](#)
23. Bagheri, Z.; Zamanifar, K. Enhancing energy efficiency in resource allocation for real-time cloud services. In Proceedings of the 7th International Symposium on Telecommunications (IST'2014), Tehran, Iran, 9–11 September 2014; pp. 701–706. [\[CrossRef\]](#)
24. Makaratzis, A.T.; Khan, M.M.; Giannoutakis, K.M.; Elster, A.C.; Tzovaras, D. GPU Power Modeling of HPC Applications for the Simulation of Heterogeneous Clouds. In *Parallel Processing and Applied Mathematics: 12th International Conference, PPAM 2017, Lublin, Poland, 10–13. September 2017, Revised Selected Papers, Part II 12*; Springer International Publishing: Berlin/Heidelberg, Germany, 2018; pp. 91–101. [\[CrossRef\]](#)
25. Li, Z.; Yan, C.; Yu, L.; Yu, X. Energy-aware and multi-resource overload probability constraint-based virtual machine dynamic consolidation method. *Futur. Gener. Comput. Syst.* **2018**, *80*, 139–156. [\[CrossRef\]](#)
26. Renuka, K.; Roy, D.S.; Reddy, K.H.K. An SDN empowered location aware routing for energy efficient next generation vehicular networks. *IET Intell. Transp. Syst.* **2021**, *15*, 308–319. [\[CrossRef\]](#)

27. Cao, G. Topology-aware multi-objective virtual machine dynamic consolidation for cloud datacenter. *Sustain. Comput. Inform. Syst.* **2019**, *21*, 179–188. [[CrossRef](#)]
28. Paulraj, G.J.L.; Francis, S.A.J.; Peter, J.D.; Jebadurai, I.J. A combined forecast-based virtual machine migration in cloud data centers. *Comput. Electr. Eng.* **2018**, *69*, 287–300. [[CrossRef](#)]
29. Haghighi, M. A.; Maeen, M.; Haghparsat, M. An energy-efficient dynamic resource management approach based on clustering and meta-heuristic algorithms in cloud computing IaaS platforms. *Wirel. Pers. Commun.* **2019**, *104*, 1367–1391. [[CrossRef](#)]
30. Verma, S.; Kawamoto, Y.; Kato, N. Energy-Efficient Group Paging Mechanism for QoS Constrained Mobile IoT Devices Over LTE-A Pro Networks Under 5G. *IEEE Internet Things J.* **2019**, *6*, 9187–9199. [[CrossRef](#)]
31. Park, K.; Pai, V.S. CoMon: A mostly-scalable monitoring system for PlanetLab. *ACM Sigops Oper. Syst. Rev.* **2016**, *40*, 65–74. [[CrossRef](#)]

Disclaimer/Publisher’s Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.