International Journal on Future Revolution in Computer Science & Communication Engineering (IJFRCSCE)
ISSN: 2454-4248 Volume: 8 Issue: 3
DOI: https://doi.org/10.17762/ijfrcsce.v8i3.2094
***Article Received**: 10 June 2022 **Revised**: 20 August 2022 **Accepted**: 25 August 2022 **Publication**: 15 September 2022*

Mobile Cloud IoT for Resource Allocation with Scheduling in Device- Device Communication and Optimization based on 5G Networks

Dr. Prakash Pise
Sinhgad Institute of Management,
SPPU PUNE UNIVERSITY
India
prakash.pise532@gmail.com

Abstract:
Internet of Things (IoT) is revolutionising technical environment of traditional methods as well as has applications in smart cities, smart industries, etc. Additionally, IoT enabled models' application areas are resource-constrained as well as demand quick answers, low latencies, and high bandwidth, all of which are outside of their capabilities. The above-mentioned issues are addressed by cloud computing (CC), which is viewed as a resource-rich solution. However, excessive latency of CC prevents it from being practical. The performance of IoT-based smart systems suffers from longer delay. CC is an affordable, emergent dispersed computing pattern that features extensive assembly of diverse autonomous methods. This research propose novel technique resource allocation and task scheduling for device-device communication in mobile Cloud IoT environment based on 5G networks. Here the resource allocation has been carried out using virtual machine based markov model infused wavelength division multiplexing. Task scheduling is carried out using meta-heuristic moath flame optimization with chaotic maps. So, by scheduling tasks in a smaller search space, system resources are conserved. We run simulation tests on benchmark issues and real-world situations to confirm the effectiveness of our suggested approach. The parameters measured here are resource utilization of 95%, response time of 89%, computational cost of 35%, power consumption of 38%, QoS of 85%.

Keywords: Internet of Things, mobile Cloud IoT, 5G networks, device-device communication, resource allocation, Task scheduling

1. Introduction:

Ground breaking season of PC labourer farms gave rise to the spectacular innovation of CC, which aids in putting an end to virtualization trends [1]. An "association" that connects programming, foundation as an aid, and platform as a service is how appropriate handling is described (PaaS). Everyone has a unique general thesis about business. Purpose of appropriated figuring is to design an application on a virtual asset of PCs that is used to serve clients regardless of the purchased model [2]. Additionally, as the cloud is based on the pillars of two fundamental foundations, such as CC and networking, Internet connectivity and infrastructure are crucial. The network can be utilised for CC and other applications in numerous cloud apps [3]. The cloud's infrastructure and capabilities are integrated with the QoS distribution network. As a result, more application service providers (ASPs) are aware of the difference between the operation and actual usage of necessary infrastructure as well as have made use of infrastructure that was leased from infrastructure providers. For instance, Force Square created the first cloud resource measurement resource by using Amazon EC2 Analytics for more than 5 million days while saving 53% of its value to fulfil measurable needs [4]. Each edge-node in a cyber-physical ecosystem is envisioned as an IoT device that may dynamically collaborate with other network nodes to carry out one or more user-assigned activities. Although these infrastructures as well as computer resources are given by CSP, IoT networks typically have restricted access to resources like processing power, storage, network bandwidth, and RAM. IoT devices' sensors generate an enormous amount of real-time data. Resource scheduling is the CC domain's greatest obstacle (RS). Using the proper hardware techniques and infrastructure, RS must be performed while maintaining at least satisfying levels of QoS [5]. According to a recent survey, the broker is often responsible for matching up requested end user tasks with the available virtualized hardware, which is typically done using a virtual machine (VM). The scheduling procedure is carried out by the broker during mapping. The process of assigning tasks to the appropriate VMs for implementation has grown increasingly difficult with the development of

International Journal on Future Revolution in Computer Science & Communication Engineering (IJFRCSCE)
ISSN: 2454-4248 Volume: 8 Issue: 3
DOI: https://doi.org/10.17762/ijfrcsce.v8i3.2094
***Article Received**: 10 June 2022 **Revised**: 20 August 2022 **Accepted**: 25 August 2022 **Publication**: 15 September 2022*

several resources that are now available. Few VMs may be under- or over-utilized when an ineffective scheduling procedure is used, and the implication of this situation results in a decline in the cloud system's overall performance. The RS issues that fall under the category of NP-hard optimization issues. It should be noted that the words cloudlet as well as task scheduling are also appropriated in contemporary computer science literature for process of matching submitted end-user tasks to available VMs [6].

The research contribution is as follows:

- To propose novel technique resource allocation and task scheduling for device-device communication in mobile Cloud IoT environment based on 5G networks
- The resource allocation has been carried out using virtual machine based markov model infused wavelength division multiplexing
- Task scheduling is carried out using meta-heuristic moath flame optimization with chaotic maps

2. Background study:

Over past few years, there is significant improvement in academic interest in SLA-based resource allocation [7] in cloud data centres. In work [8], a dynamic scheduling approach for context-aware, SOA-based applications' response time SLA was developed. In order to minimise taxing the service tier resources in accordance with a SLA measure, the authors of [9] suggested rate-limiting requests. The inventors of [10] employed a dynamic scheduling technique that is capable of providing SLAs for CPU service share in server clusters. Previous studies focused on IoT cloud allocation methodologies took into account execution speed and security groupings. A combinatorial auction system, for instance, was suggested by the author in [11] as a way to effectively allocate resources and decrease the penalty when there were execution time restrictions. A framework for fulfilling application needs for security in a cloud for processing IoT data was proposed in study [12]. Over past few years, there is significant improve in academic interest in SLA-based resource allocation in cloud data centres. In work [13], a dynamic scheduling technique for context-aware, SOA-based applications' response time SLA was developed. In order to prevent overtaxing the service tier resources as measured by a SLA metric, the developers of [14] have suggested rate-limiting requests. In their study [15], the authors devised a dynamic scheduling technique that can ensure SLAs for CPU service share in server clusters. Previous studies focused on IoT cloud allocation methodologies took into account execution speed and security groupings. For instance, work [16] suggested using a combinatorial auction system to effectively allocate resources and lower the penalty when there were execution time limits. In order to fulfil application requirements for security in a cloud for processing IoT data, author [17] presented a framework. In order of earliest finish time, tasks are sorted and opportunistically added to open processor idle time slots. In terms of robustness and performance, HEFT is rated as the top scheduling heuristic method out of twenty [18]. [19] presents a mixed-integer linear programming model to simulate offloading dependent jobs with time limitations in an IoT-fog system. IoT devices are disregarded in this work, which treats the cloud computing layer as if it were a CPU with infinite processing power. The top K new solutions are selected as the Pareto set after the new solutions are sorted by crowding distance. In [20], fuzzy dominance is used to enhance MOHEFT's performance, which maximises both makespan and cost index. Constrained optimization challenges in cloud fog platforms were related task scheduling problems that the author [21] formulated. It is suggested to handle these issues using an LBP-ACS strategy.

3. Proposed Mobile Cloud IoT for resource allocation with scheduling:

This section discuss novel technique resource allocation and task scheduling for device-device communication in mobile Cloud IoT environment based on 5G networks. Here the resource allocation has been carried out using virtual machine based fuzzy rules infused wavelength division multiplexing. Task scheduling is carried out using meta-heuristic moath flame optimization with chaotic maps. The system architecture is shown in figure-1.

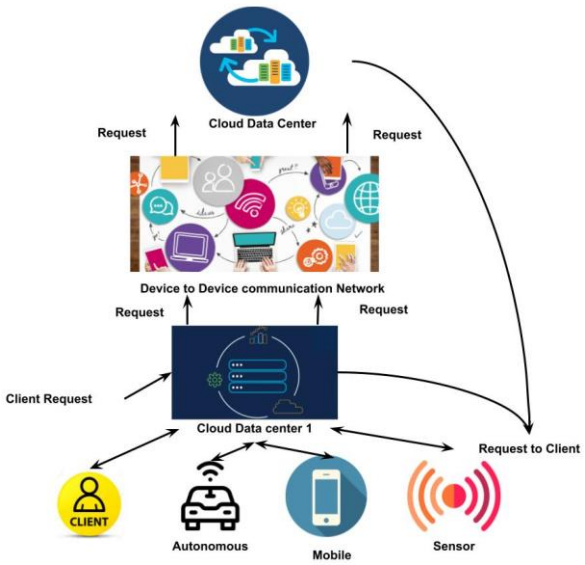

Figure-1 Proposed Mobile cloud IoT environment

International Journal on Future Revolution in Computer Science & Communication Engineering (IJFRCSCE)
ISSN: 2454-4248 Volume: 8 Issue: 3
DOI: https://doi.org/10.17762/ijfrcsce.v8i3.2094
Article Received: 10 June 2022 Revised: 20 August 2022 Accepted: 25 August 2022 Publication: 15 September 2022

Network Model:

It is assumed that all resource nodes in a cloud computing system must communicate with one another. Therefore, the overall cost of network connections should be determined for each solution to the RA problem. It is presumed that every resource communicated with every other resource. These messages' overall costs are assessed and taken into consideration as an objective function. the Tc symbol stands for the function called total cost. The suggested algorithm makes an effort to reduce this function. Tc is calculated using equation (1).

$$T_c = \frac{\sum_{j=1}^{|V_s|}\left(d_j^r \times d^\delta\right)}{p} \quad (1)$$

As a result, we specify three service events in the system model: 1) A user identified by the letter el submits a request to the cloud for security service number l; 2) A user, represented by the symbol eh, submits a request to the cloud for security service eh; and 3) When a security service transaction is complete, linked VIs are released, as indicated by the symbol ef. Nl and Nh, respectively, represent number of security services l and h that are now being provided in the cloud. Therefore, system state is given as: $S = \{s \mid s = \langle \hat{s}, e\rangle\}$

where $\hat{s} = \langle N_l, N_h\rangle, e \in \{e_l, e_h, e_f\}$, and $0 \le \alpha_l N_l + \alpha_h N_h \le x(s,a) - \tau(s,a)y(s,a), K$.

Based on service revenues and operating costs, the system's net return can be assessed by eqn (2):

$$x(s,a) - \tau(s,a)\mathrm{y(s,a)} \quad (2)$$

Where x(s, a) represents cloud's net lump sum earnings when decision an is made at current state s, y(s, a) represents service holding cost rate when decision an is made and the cloud is in state s, and τ (s, a) represents anticipated service time from current state s to next state. It is calculated by eqn (3)

$$x(s,a) = \begin{cases} 0, & a_{\langle \hat{s},e\rangle} = 0, \\ R_l, & a_{\langle \hat{s},e_l\rangle} = 1, \\ R_h, & a_{\langle \hat{s},e_h\rangle} = 1, \end{cases} \quad (3)$$

where Rl and Rh are cloud's earnings when requests for l and h security services are approved, respectively. The occupied cloud resources are proportional to service holding cost rate y(s, a), which is given by eqn (4)

$$y(s,a) = \begin{cases} \alpha_l N_l + \alpha_h N_h, & a_{\langle \hat{s},e\rangle} = 0, \\ \alpha_l (N_l + 1) + \alpha_h N_h, & a_{\langle \hat{s},e_l\rangle} = 1 \\ \alpha_l N_l + \alpha_h (N_h + 1), & a_{\langle \hat{s},e_h\rangle} = 1 \end{cases} \quad (4)$$

A decision epoch is point in time when any of occurrences, such as the arrival of a service request l or h or completion of a security service and release of VIs' allotted resources, occurs. An exponential distribution describes the amount of time that passes between two decision points. Given present state s and the action a, denote τ (s, a) as anticipated time interval between 2 decision epochs by eqn (5).

$$\tau(s,a) = \begin{cases} \left[\gamma + a_{\langle \hat{s},e_l\rangle}\mu_l\right]^{-1}, & e = e_l \\ \left[\gamma + a_{\langle \hat{s},e_h\rangle}\mu_h\right]^{-1}, & e = e_h \\ \gamma^{-1}, & e = e_f \end{cases} \quad (5)$$

Virtual Machine Based Markov Model Infused Wavelength Division Multiplexing In Resource Allocation:

Virtual machine allocation is defined as the assignment of a collection of VMs to a collection of physically located machines in a data centre. The term "virtualization" refers to the ability to execute several operating system instances on a single physical machine, maximising use of the hardware. Multiple users can share the computer resource pool in accordance with the actual demand thanks to the usage of virtualization technology. Therefore, the VM allocation system requires to be adjusted to meet needs for resource utilisation while minimising user costs. Figure 2 displays how VM resources are allocated in CC.

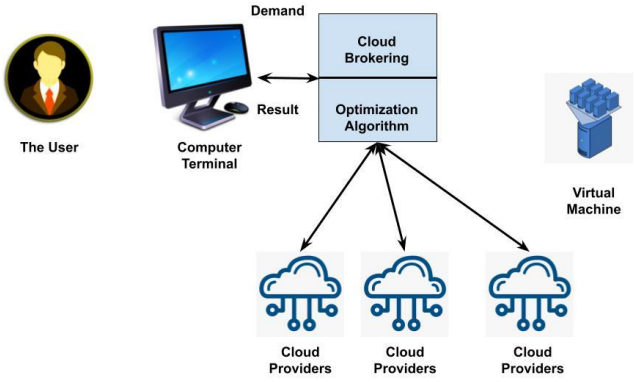

Figure-2 VM resource allocation

For a state $s = \langle \hat{s}, e\rangle$where $\hat{s} = \langle N_l, N_h\rangle, e \in \{e_l, e_h, e_f\}$, and action $a = 0$ the next state $j_1 = \langle N_l, N_h, e_l\rangle, j_2 = \langle N_l, N_h, e_h\rangle, j_3 = \langle N_l - 1, N_h, e_f\rangle (N_1 \ge 1)$, and $j_4 = \langle N_l, N_h - 1, e_f\rangle (N_h \ge 1)$. q(j|s, a) are given as by eqn (6)

$$q(j \mid s,a) = \begin{cases} \lambda_l \tau(s,a), & j = j_1 \\ \lambda_h \tau(s,a), & j = j_2 \\ N_l \mu_l \tau(s,a), & j = j_3 \\ N_h \mu_h \tau(s,a), & j = j_4 \end{cases} \quad (6)$$

Note that $0 \le \alpha_l N_l + \alpha_h N_h \le K$ For the current state $s = \langle \hat{s}, e_l\rangle$ and the action a = 1, the next state can be $j_5 = \langle N_l + 1, N_h, e_l\rangle, j_6 = \langle N_l + 1, N_h, \overline{e_h}\rangle$

, $j_7 = \langle N_l, N_h, e_f\rangle$, and $j_8 = \langle N_l + 1, N_h - 1, e_f\rangle (N_h \ge 1)$ by eqn. (7)

Thus, q(j|s, a) is given as:

International Journal on Future Revolution in Computer Science & Communication Engineering (IJFRCSCE)
ISSN: 2454-4248 Volume: 8 Issue: 3
DOI: https://doi.org/10.17762/ijfrcsce.v8i3.2094
***Article Received**: 10 June 2022 **Revised**: 20 August 2022 **Accepted**: 25 August 2022 **Publication**: 15 September 2022*

$$q(j \mid s,a) = \begin{cases} \lambda_l \tau(s,a), & j = j_5 \\ \lambda_h \tau(s,a), & j = j_6 \\ (N_l+1)\mu_l \tau(s,a), & j = j_7 \\ N_h \mu_h \tau(s,a), & j = j_8 \end{cases} \quad (7)$$

Similarly, for state $s = \langle \hat{s}, e_h \rangle$ and action a = 1, the next state is $j_9 = \langle N_l, N_h{}' + 1, e_l \rangle, j_{10} = \langle N_l, N_h + 1, e_h \rangle$, $j_{11} = \langle N_l - 1, N_h + 1, e_f \rangle (N_l \geq 1)$, and $j_{12} = \langle N_l, N_h, e_f \rangle$ Thus, q(j|s, a) is given as eqn (8)

$$q(j \mid s,a) = \begin{cases} \lambda_l \tau(s,a), & j = j_9 \\ \lambda_h \tau(s,a), & j = j_{10} \\ N_l \mu_l \tau(s,a), & j = j_{11} \\ (N_h+1)\mu_h \tau(s,a), & j = j_{12} \end{cases} \quad (8)$$

The expected discounted reward during τ (s, a) satisfies the following conditions when the discounted reward model is applied by eqn (9):

$$z(s,a) = x(s,a) - y(s,a) E_{\frac{a}{s}} \left\{ \int_0^{\tau_1} e^{-\alpha t} dt \right\}$$

$$= x(s,a) - y(s,a) E_{\frac{a}{s}} \left\{ \frac{[1 - e^{-\alpha \tau_1}]}{\alpha} \right\}$$

$$= x(s,a) - \frac{y(s,a)\tau(s,a)}{1 + \alpha \tau(s,a)}$$

$$v(s) = \max_{a \in A} \left\{ z(s,a) + \lambda \sum_{j \in S} q(j \mid s,a) v(j) \right\} \quad (9)$$

where λ = (1 + ατ (s, a))−1. Let w be a finite constant, w = λl+λh+K ∗max(μl, μh) < ∞, and λ˜ = w/(w+α). Following uniformization, v(s) optimality equation can be found.,

$$\tilde{v}(s) = \max_{a \in \bar{A}} \left\{ \tilde{z}(s,a) + \bar{\lambda} \sum_{j \in S} \bar{q}(j \mid s,a) \tilde{v}(j) \right\}$$

where $\tilde{z}(s,a) \equiv z(s,a) \frac{1 + a\tau(s,a)}{(\alpha + w) r(s,a)}$, and

$$\tilde{q}(j \mid s,a) = \begin{cases} 1 - \frac{|1 - q(s|s,a)|}{\tau(s,a) w}, & j = s \\ \frac{q(j|s,a)}{\tau(s,a) w}, & j \neq s, \end{cases} \quad (10)$$

The goal of the model is to reduce the total suggested architecture's power consumption, which includes processing as well as networking power consumption of processing locations and network connecting these locations, as shown in equation below eqn (11).

$$\sum_{n \in P_N} P_n + \sum_{n \in P_N} \mathcal{P}_n \quad (11)$$

The power used by processors for processing and the power used to transmit traffic across the network make up the two terms in the objective equation. Be aware that each processing node choice and the OW mobile units are connected by a separate route in our network topology. According to equation, the power consumption of this route is indicated by a single value by eqn (12). In order to add across processing nodes, the second term in equation (11) is used. Pn is the symbol for the processor power usage.

$$P_n = \sum_{k \in K} X_{kn} E_n \; \forall n \in PN \quad (12)$$

where Xkn represents the workload needed to complete job k, measured in MIPS and assigned to processing node n. En is node processor's energy in watts per MIPS, calculated using node's maximal processing power. Given as the networking power usage, Pn by eqn (13)

$$\mathcal{P}_n = \sum_{k \in K} \delta_{kn} F_{ks} \Psi_n \; \forall n \in PN, s \in SN \quad (13)$$

where Fks is task data rate demand (in Mbps) produced by source node s, and δkn is a binary variable that determines assignment of task k to processing node Ψn. Between source as well as designated processing node, n is the sum of all nodes' power per Mbps. The power per Mbps number for OW link is determined using specific wavelength colour (RYGB) that is being used for the connection.

The following restrictions are placed on the model: Constrained processing allocation by eqn (14)

$$\alpha X_{kn} \geq \delta_{kn} \; \forall k \in K, n \in PN$$

$$X_{knn} \leq \alpha \delta_{kn} \; \forall k \in K, n \in PN$$

$$\sum_{n \in PN} \delta_{kn} = 1 \; \forall k \in K \quad (14)$$

assures that one processing node will be given to each of the task k tasks. Constrained processing node capacity by eqn (15)

$$\sum_{k \in K} X_{kt} \leq C_{nt} \; \forall n \in PN \quad (15)$$

Every task k given to a processing node n does not go above that node's processing limit. Link capacity limitation by eqn (16)

$$\sum_{k \in K} \sum_{S \in SN} \lambda_{ij}^{k \in d} \leq L_{ij} \; \forall i \in N, j \in Nm_i, i \neq j.$$
$$d \in PN \quad (16)$$

The amount of task k traffic travelling from source s to processing node d does not go beyond limit of any link connecting nodes I and j. Constraint on flow conservation by eqn (17)

$$\sum_{\substack{j \in Nmi \\ i \neq j}} \lambda_{ij}^{ksd} - \sum_{\substack{j \in Nmi \\ i \neq j}} \lambda_{ji}^{ksd} = \begin{cases} L_{ksd} & \text{if } i = s \\ -L_{ksd} & \text{if } i = d \\ 0 & \text{otherwise} \end{cases} \quad \forall k \in K, s \in SN, d \in PN, i, j \in N. \quad (17)$$

The quadratic term's multiplication of the continuous variable by the binary variable was linearized using the following constraints (17) through (18), where non-negative linearization variable $\phi_{m,u,f}^{a,b,\lambda} = \gamma_{u,f}^{a,\lambda} S_{m,f}^{b,\lambda}$ is is introduced.

$$\phi_{m,\lambda,f}^{a,b\lambda} \geq 0$$

$$\phi_{m,\lambda,f}^{a,b\lambda} \leq \beta S_{m,f}^{b,\lambda} \; \forall u, m \in \mathcal{U}, \forall a, b \in \mathcal{A}, \forall \lambda \in \mathcal{W}, \forall f \in \mathcal{B} \; (u \neq \mathrm{m}, a \neq b) \quad (18)$$

where β is a large number, so that β >>γ .

$$\phi_{m,u,f}^{a,b,\lambda} \leq \gamma_{u,f}^{a,\lambda} \; \forall u, m \in \mathcal{U}, \forall a, b \in \mathcal{A}, \forall \lambda \in \mathcal{W}, \forall f \in \mathcal{B} \; (u \neq \mathrm{m}, a \neq b)$$

$$\phi_{m,u,f}^{a,b,\lambda} \geq \beta S_{m,f}^{b,\lambda} + \gamma_{u,f}^{a,\lambda} - \beta \; \forall u, m \in \mathcal{U}, \forall a, b \in \mathcal{A}, \forall \lambda \in \mathcal{W}, \forall f \in \mathcal{B} \; (u \neq \mathrm{m}, a \neq b). \quad (19)$$

Equation (19) can be recast as Equation (20) by substituting this linearization variable for quadratic term.

$$\phi_{m,u,f}^{a,b,\lambda} \geq \beta S_{m,f}^{b,\lambda} + \gamma_{u,f}^{a,\lambda} - \beta \; \forall u, m \in \mathcal{U}, \forall a, b \in \mathcal{A} \quad (20)$$

International Journal on Future Revolution in Computer Science & Communication Engineering (IJFRCSCE)
ISSN: 2454-4248 Volume: 8 Issue: 3
DOI: https://doi.org/10.17762/ijfrcsce.v8i3.2094
***Article Received**: 10 June 2022 **Revised**: 20 August 2022 **Accepted**: 25 August 2022 **Publication**: 15 September 2022*

Modulation format switching is not always an option for impaired demands, though, as there may not always be available spectrum on the route links to increase bandwidth. Be aware that the MFS may refuse a request if there are no more available spectrum resources or if all viable modulation switching options have been exhausted without satisfying the physical layer constraint.

Meta-Heuristic Moth Flame Optimization With Chaotic Maps In Task Scheduling:

A well-known SI algorithm called the moth-flame optimization (MFO) was motivated by the nighttime spiral migration of moths. This behaviour is derived from moths' ability to navigate by keeping a stable inclination to the moon, which allows them to travel vast distances in a straight path. If light source is reasonably close to moths, though, this sensible navigation system transforms into a lethal spiral path in that direction. The MFO algorithm is described as being made up of moths and flames in the quick summary. Moths are referred to as search agents in Equation (1), where N is total number of moths and M(t) is their organisational matrix that they use to search across the D-dimensional search space by eqn (21).

$$M(t) = \begin{bmatrix} m_{1,1} & m_{1,2} & \cdots & m_{1,D} \\ m_{2,1} & m_{2,2} & \cdots & m_{2,D} \\ \vdots & \vdots & \vdots & \vdots \\ m_{N,1} & m_{N/2} & \cdots & m_{N,D} \end{bmatrix} \quad (21)$$

Additionally, as seen below in the array OM (t), the fitness of the relevant moth is kept by eqn (22).

$$OM(t) = \begin{bmatrix} OM_1(t) \\ OM_2(t) \\ \vdots \\ OM_N(t) \end{bmatrix} \quad (22)$$

However, the best spots found by moths are flames, which are kept in a matrix identical to F (t) together with their fitness values b indicates shape of logarithmic spiral, illustrates how moths spirally move around their corresponding flames by eqn (23).

$$\alpha X_{kn} \geq \delta_{kn} \; \forall k \in K, n \in PN \quad (23)$$

Equation (5), where t finds current number of iterations, while N and Max define maximum number of iterations, shows that as the algorithm iterates, the number of flames reduces to allow for greater exploitation. It shows the maximum iterations and the total number of flames, respectively by eqn (24).

$$Flame_{Num}(t) = round(N - t \times \frac{N-1}{MAxIT}) \quad (24)$$

The MFO method is a three-tuple that is defined as follows and approximates global optimal of optimization issues by eqn (25):

$$\text{MFO} = (I, P, T) \quad (25)$$

The following is the function's methodical model by eqn (26):

$$I: \emptyset \rightarrow \{M, OM\} \quad (26)$$

The moths are moved throughout the search space by the P function, which is the primary function. This method was given the M matrix and eventually returns its modified one by eqn (27).

$$P: M \rightarrow M \quad (27)$$

When the termination requirement is satisfied, the T function returns true; otherwise, it returns false by eqn (28).

$$T: M \rightarrow \{ \text{ true, false } \} \quad (28)$$

By randomly shifting the position of Mi, the RM operator expands the scope of research. In addition, the GM operator is introduced to use enhanced moths stored in the guiding archive to direct the population toward promising zones. A dimensionally aware switch between these two operators is also advantageous for the migration strategy, as shown in eqn (7), where δ stands for current size of guiding archive by eqn (29).

$$M_i(t+1) = \begin{cases} \text{RM operator} & \delta < D \\ \text{GM operator} & \delta \geq D \end{cases} \quad (29)$$

Let's say that there is a finite number of unlucky moths (M1, M2,..., Mi,...) such that OMi (t) > OMi (t 1). As a result, in this operator, Mi is parents in crossover defined by eqns (8) and (9), where is a random number in range [0, 1], and Mr is a randomly generated moth (Mr). The crossover creates two children, and the best offspring is chosen by comparing it to the other offspring and, if it can outperform the OMi, adding its position to the guiding archive (t). By stochastically relocating unlucky moths to find potential places in initial iterations, the RM operator satisfies the necessity for exploration by eqn (30).

$$\begin{aligned} \text{Offspring}_{\,1} &= \alpha \times M_i + (1-\alpha) \times M_r \\ \text{Offspring}_{\,2} &= \alpha \times M_r + (1-\alpha) \times M_i \end{aligned} \quad (30)$$

When size of GM equals size of issue variables, GM operator is used to modify position of the unlucky moth, Mi. By using crossover described in eqns (10) and (11), in which LMr is a chosen lucky moth from guiding archive, the GM modifies position of Mi. Similar to the RM operator, position of Mi (t + 1) is updated and added to guiding archive if new offspring achieves a better position than Mi (t) by eqn (31).

$$\begin{aligned} \text{Offspring}_{\,1} &= \alpha \times M_i + (1-\alpha) \times M_r \\ \text{Offspring}_{\,2} &= \alpha \times M_r + (1-\alpha) \times M_i \end{aligned} \quad (32)$$

Due to its characteristics of non-repetition, ergodicity, and dynamicity, chaos theory is one of the most effective methods for addressing the premature convergence problem. After a specific number of repetitive operations carried out by the same chaotic function, chaos is a stochastic methods

International Journal on Future Revolution in Computer Science & Communication Engineering (IJFRCSCE)
ISSN: 2454-4248 Volume: 8 Issue: 3
DOI: https://doi.org/10.17762/ijfrcsce.v8i3.2094
***Article Received**: 10 June 2022 **Revised**: 20 August 2022 **Accepted**: 25 August 2022 **Publication**: 15 September 2022*

in non-linear deterministic method that ultimately renders numerical sequences of two closed beginning values useless. Operation of the proposed task scheduling is straightforward as well as easy to attain an optimal solution. For each iteration, we define r' and p using a chaotic map. Since other specifications values directly depend on values of r' and p, we can use chaotic sequence to speed up algorithm convergence while also preventing premature convergence. These maps behave differently when starting at a point of 0.7. Based on range of chaotic map, it is also feasible to choose any integer as the starting point between [0, 1] and [1, 1]. However, the initial value has a substantial impact on the fluctuation pattern in some of these maps. Figure 3 depicts the HMFO-CM flowchart.

Flowchart for HMFO-CM:

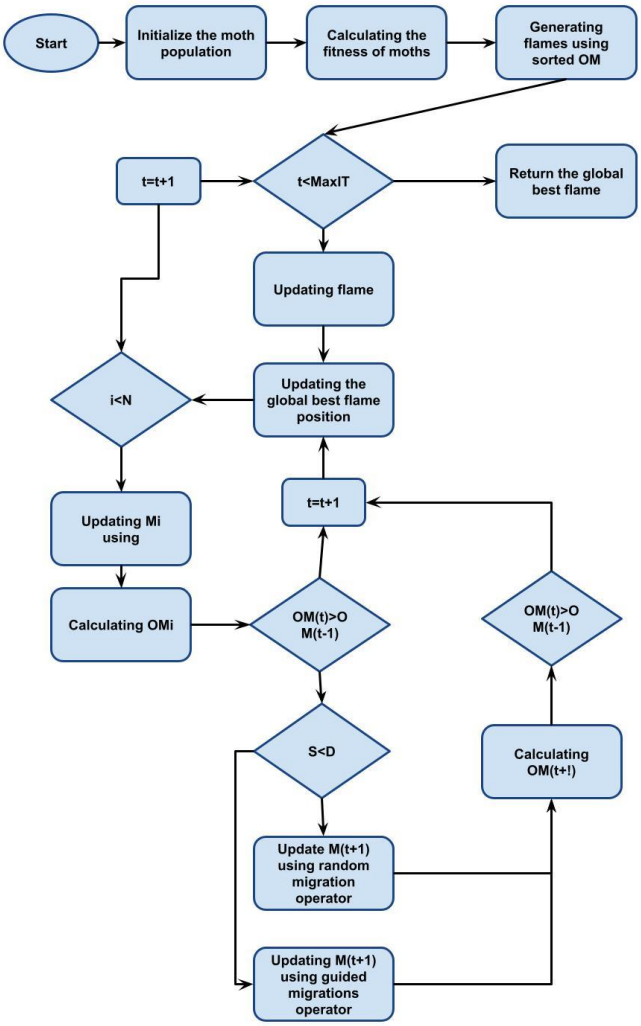

Figure 3. Overall flow chart

The IoT applications are executed in large part thanks to the mobile computing paradigm, which is an extension of CC. The system designer anticipates deployment of fog nodes with specific computing and storage resources at edge of the network, between IoT as well as cloud layer, in this hierarchical paradigm (see Fig. 4). The mobile layer reduces the time it takes for data from IoT devices to and from cloud layer to be transmitted, allowing IoT applications to use nearby fog resources as infrastructure to offload and carry out their IoT duties in real-time. Additionally, new computational and storage limits apply to the completion of IoT tasks. Additionally, when offloading IoT tasks to fog or cloud resources, there are many QoS criteria that need to be considered. These factors are significant from both the standpoint of the system designer and the end-users of IoT applications. For example, real-time feature is crucial as a QoS specification that affects IoT applications' end users, or the energy consumption of fog nodes is a QoS parameter that matters to system designers. Goal is to provide a scheduler for dynamically arriving IoT jobs while taking into account a variety of goals for the QoS parameter optimization for both end users and system designers.

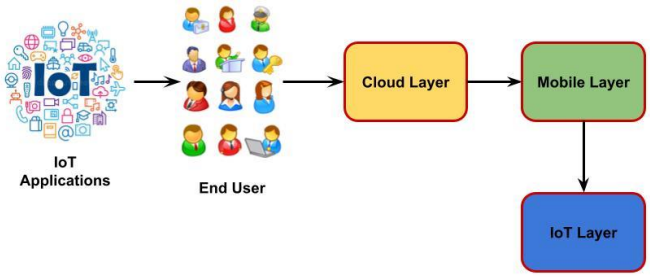

Figure 4. 1 Hierarchical mobile-based architecture model

4. Experimental results:

We develop our Nash equilibrium method in Matlab and conduct experiments on a DELL T3400 workstation to illustrate viability of our strategy. Through a linear combination of actions on many components, such as CPU utilisation, Wi-Fi module status, LCD brightness, etc., we are able to derive the power model of mobile applications. We just use one factor, t, to represent overall activities of application because our goal is not a thorough power analysis for mobile devices. Assuming a mobile device's maximum power consumption is 2.4W and the application's actual power is $P_{app} = \alpha \times 2.4W$ We presume that comprehensive application profiling has predetermined the amount of compute that will be offloaded. We presum that the servers are diverse and that their operational frequencies range from [2.0GHz to 3.0GHz]. We utilise the Intel Xeon processor-based server power model. This model states that server power is a linear function of CPU frequency at maximum utilisation. At 2.0GHz, it is 200W, and at 3.0GHz, it is 240W. The total system power, including static power,

International Journal on Future Revolution in Computer Science & Communication Engineering (IJFRCSCE)
ISSN: 2454-4248 Volume: 8 Issue: 3
DOI: https://doi.org/10.17762/ijfrcsce.v8i3.2094
***Article Received**: 10 June 2022 **Revised**: 20 August 2022 **Accepted**: 25 August 2022 **Publication**: 15 September 2022*

is represented by these values. Based on information provided, consider that our server's static power is 100W when it is at rest. Server power and VM system utilisation β have a linear relationship. For simplicity, we set $\beta = \alpha$ in our trials. All things considered, we presumptively believe that server power is a linear function against both running frequency as well as usage. Server uses 100W to 140W more electricity while operating at full capacity than when it is idle. For delay method, consider that all servers, while operating at 2.0GHz, outperform mobile devices by a factor of. Its speedup ratio is $sM/2$ while moving at speed s, or $g(s_j) = 2/(s_j M)$. When N mobile devices offload to server, consider slowness factor is $h(N) = N^{\gamma}, \gamma \geq 1, \gamma$ which takes into account the overhead of context switching. Device I will have to wait $10 \times (sM/2) \times \mathrm{N}\ \gamma$ seconds to receive results if it offloads a computation that will take it 10 seconds to complete to a server with a speed of s that is shared by N mobile devices γ. During our test, γ is set at 1.05.

Evaluation metrics:

- Resource Utilization: It speaks of the total amount of resources that a job uses to finish its execution. Resource Utilization (RU) is denoted by the following by eqn (33)

$$R_U = R_{avl} - R_{mu} \quad (33)$$

where Rnu stands for unused resources and Ravl stands for available resources. Our particular work's processor and memory utilisation are examples of resource usage. When the other two procedures are merged, the percentage of a specific approach is always higher. A graphic representation of percentage usage of resources utilising different resource allocation systems is shown in Table 1 and Figure 5. It demonstrates that the proposed technique has the highest resource utilisation for a variety of task sizes.

Table-1 Comparison of percentage utilization of resources

Task Size	MOHEFT	LBP-ACS	MCIoT_RA_5G
20	65	69	71
40	68	72	75
60	71	74	79
80	73	78	81
100	75	81	83
120	79	85	85
140	81	88	89
160	83	91	93
180	89	92	94
200	91	93	95

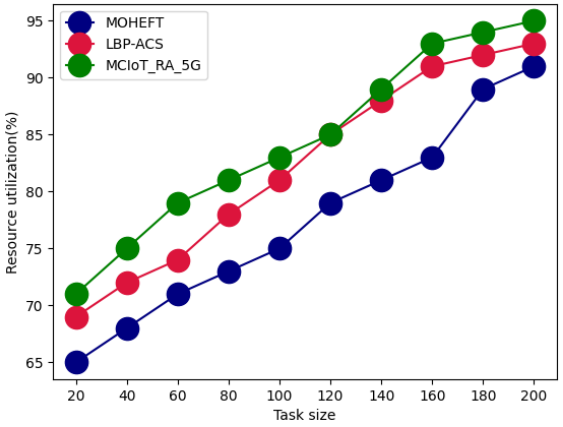

Figure-5 Comparison of percentage utilization of resources

- Response Time: The amount of time that passes between a task being launched and being finished is referred to as the response time of a task. The following can be used to express the reaction time TSRes by eqn (34):

$$TS_{\mathrm{Res}} = TS_{CT} - TS_{AT} \quad (34)$$

where TSAT is the task's arrival time and TSCT is the task's completion time. Our particular work's processor and memory utilisation are examples of resource usage. When the other two procedures are merged, the percentage of a specific approach is always higher. The maximum response time is shown in Figure 6 and Table-2. This is essentially the method that is suggested in our study to turn off passive prime ministers. It takes part in the system of resource use.

Table-2 Comparison of response time

Task Size	MOHEFT	LBP-ACS	MCIoT_RA_5G
20	55	61	65
40	59	63	68
60	61	65	72
80	63	68	75
100	68	71	79
120	71	73	81
140	75	75	83
160	77	77	86
180	81	83	88
200	83	85	89

International Journal on Future Revolution in Computer Science & Communication Engineering (IJFRCSCE)
ISSN: 2454-4248 Volume: 8 Issue: 3
DOI: https://doi.org/10.17762/ijfrcsce.v8i3.2094
***Article Received**: 10 June 2022 **Revised**: 20 August 2022 **Accepted**: 25 August 2022 **Publication**: 15 September 2022*

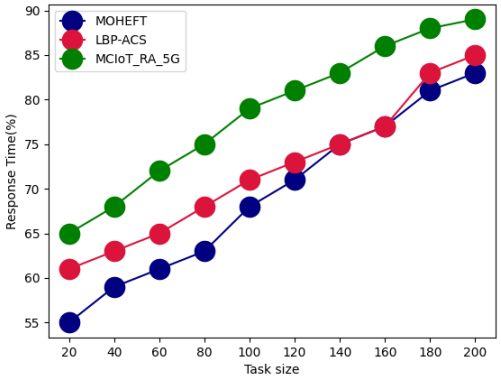

Figure 6. Comparison of response time

- Power Consumption: It can be characterised as the energy unit that each cloud server uses to distribute resources. An energy management module is used by management to minimise energy consumption in this specific task. The power consumption solutions used in real-time data centres, such as dynamic voltage, frequency, and resource sleep, are numerous, however they are insufficient for the virtualized environment. When compared to current methods, our suggested methodology produces greater outcomes for energy reduction. Table 3 and Figure 7 provide as evidence for this. Energy management strategy described in this research minimises primary energy consumption PM, external energy consumption PM, and internal communication PM.

Table-3 Comparison of Power consumption

Task Size	MOHEFT	LBP-ACS	MCIoT_RA_5G
20	71	65	61
40	68	63	59
60	65	61	55
80	62	58	51
100	61	56	49
120	58	54	45
140	55	51	42
160	52	48	41
180	51	44	40
200	48	42	38

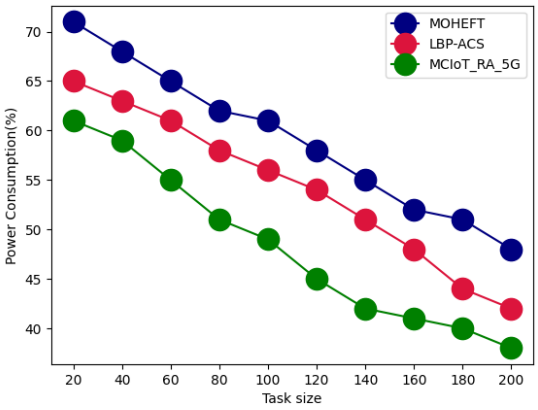

Figure 7: Comparison of power consumption

- QoS: QoS is description or measurement of a service's total performance, particularly the performance experienced by network users, whether it be a cloud computing service, a telephony service, or a computer network. The table-4 and figure-8 shows QoS for proposed technique in resource allocation with task scheduling.

Table-4 Comparison of QoS

Task Size	MOHEFT	LBP-ACS	MCIoT_RA_5G
20	45	52	59
40	46	55	62
60	48	59	65
80	51	61	67
100	53	63	72
120	55	66	75
140	59	71	79
160	61	73	81
180	63	75	83
200	65	78	85

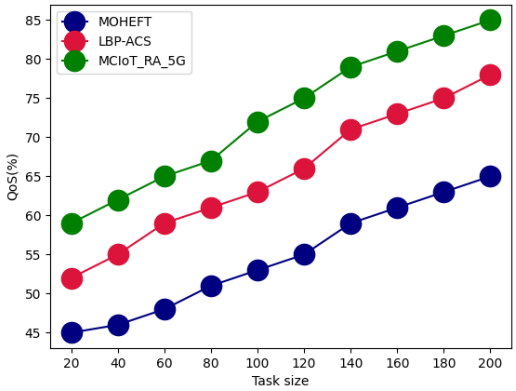

Figure 8. Comparison of QoS

International Journal on Future Revolution in Computer Science & Communication Engineering (IJFRCSCE)
ISSN: 2454-4248 Volume: 8 Issue: 3
DOI: https://doi.org/10.17762/ijfrcsce.v8i3.2094
***Article Received**: 10 June 2022 **Revised**: 20 August 2022 **Accepted**: 25 August 2022 **Publication**: 15 September 2022*

Table-5 Comparison of communication cost

Task Size	MOHEFT	LBP-ACS	MCIoT_RA_5G
20	71	65	61
40	68	63	59
60	65	61	55
80	61	55	51
100	58	53	49
120	55	51	45
140	53	45	44
160	51	42	41
180	49	41	38
200	45	38	35

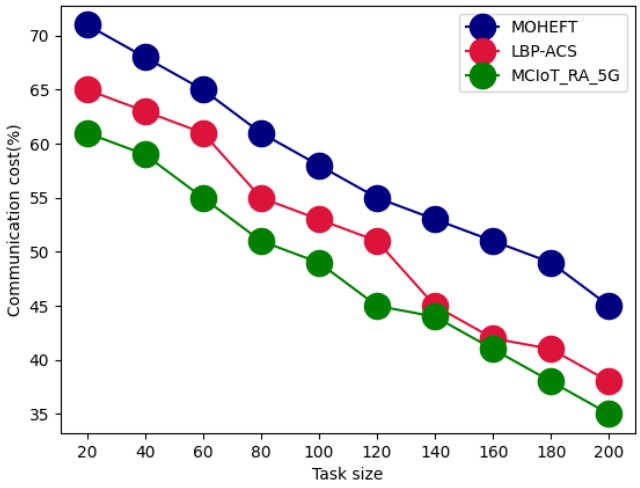

Figure 9. Comparison of communication cost

- Communication cost: The magnitude of the task's input determines the communication cost. Bytes are used to express this size. However, because we'll be utilising relational database operations as our examples, we'll frequently refer to size in terms of the number of tuples. A Markov chain is used to calculate the probability under the users, which represents the query rates for each user as shown in table 5 and figure 9.

5. Conclusion

For IoT applications that are time-sensitive, a paradigm that is governed by cloud, fog, and edge computing may give a solution. Additionally, fog nodes typically offer greater data processing and repository capacity, which can be exploited to boost performance and lower connection and latency costs. This research proposed novel framework in mobile cloud computing based resource allocation and task scheduling integrated with IoT module. Here the resource allocation has been carried out using virtual machine based markov model infused wavelength division multiplexing. Task scheduling is carried out using meta-heuristic moath flame optimization with chaotic maps. The experimental results has been carried out in terms of resource ulitization of 95%, response time of 89%, computational cost of 35%, power consumption of 38%, QoS of 85%. By minimising the amount of time that user jobs need to be processed, service providers can increase their income through the efficient use of task scheduling and resource allocation of cloud infrastructure. Usefulness of the suggested model in a real-world scenario will be determined in future using a significant amount of actual data in a real cloud environment.

Reference:

[1]. Quasim, M. T. (2021). Resource management and task scheduling for IoT using mobile edge computing. *Wireless Personal Communications*, 1-18.

[2]. Aletri, O. Z., Alahmadi, A. A., Saeed, S. O., Mohamed, S. H., El-Gorashi, T. E. H., Alresheedi, M. T., & Elmirghani, J. M. (2020). Optimum resource allocation in optical wireless systems with energy-efficient fog and cloud architectures. *Philosophical Transactions of the Royal Society A*, *378*(2169), 20190188.

[3]. Ge, Y., Zhang, Y., Qiu, Q., & Lu, Y. H. (2012, July). A game theoretic resource allocation for overall energy minimization in mobile cloud computing system. In *Proceedings of the 2012 ACM/IEEE international symposium on Low power electronics and design* (pp. 279-284).

[4]. Liang, H., Huang, D., Cai, L. X., Shen, X., & Peng, D. (2011, April). Resource allocation for security services in mobile cloud computing. In *2011 IEEE Conference on Computer Communications Workshops (INFOCOM WKSHPS)* (pp. 191-195). IEEE.

[5]. Nadimi-Shahraki, M. H., Fatahi, A., Zamani, H., Mirjalili, S., Abualigah, L., & Abd Elaziz, M. (2021). Migration-based moth-flame optimization algorithm. *Processes*, *9*(12), 2276.

[6]. Mirjalili, S. (2015). Moth-flame optimization algorithm: A novel nature-inspired heuristic paradigm. *Knowledge-based systems*, *89*, 228-249.

[7]. Movahedi, Z., & Defude, B. (2021). An efficient population-based multi-objective task scheduling approach in fog computing systems. *Journal of Cloud Computing*, *10*(1), 1-31.

[8]. Mijuskovic, A., Chiumento, A., Bemthuis, R., Aldea, A., & Havinga, P. (2021). Resource management techniques for cloud/fog and edge computing: An evaluation framework and classification. *Sensors*, *21*(5), 1832.

[9]. Bal, P. K., Mohapatra, S. K., Das, T. K., Srinivasan, K., & Hu, Y. C. (2022). A Joint Resource Allocation, Security with Efficient Task Scheduling in Cloud Computing Using Hybrid Machine Learning Techniques. *Sensors*, *22*(3), 1242.

International Journal on Future Revolution in Computer Science & Communication Engineering (IJFRCSCE)
ISSN: 2454-4248 Volume: 8 Issue: 3
DOI: https://doi.org/10.17762/ijfrcsce.v8i3.2094
***Article Received**: 10 June 2022 **Revised**: 20 August 2022 **Accepted**: 25 August 2022 **Publication**: 15 September 2022*

[10]. Raj, R., Varalatchoumy, M., Josephine, V. L., Jegatheesan, A., Kadry, S., Meqdad, M. N., & Nam, Y. (2022). Evolutionary Algorithm Based Task Scheduling in IoT Enabled Cloud Environment.

[11]. Sangaiah, A. K., Hosseinabadi, A. A. R., Shareh, M. B., Bozorgi Rad, S. Y., Zolfagharian, A., & Chilamkurti, N. (2020). IoT resource allocation and optimization based on heuristic algorithm. *Sensors*, *20*(2), 539.

[12]. Wu, C. G., Li, W., Wang, L., & Zomaya, A. Y. (2021). An evolutionary fuzzy scheduler for multi-objective resource allocation in fog computing. *Future Generation Computer Systems*, *117*, 498-509.

[13]. Abohamama, A. S., El-Ghamry, A., & Hamouda, E. (2022). Real-Time Task Scheduling Algorithm for IoT-Based Applications in the Cloud–Fog Environment. *Journal of Network and Systems Management*, *30*(4), 1-35.

[14]. Dewangan, B. K., Agarwal, A., Venkatadri, M., & Pasricha, A. (2019). Self-characteristics based energy-efficient resource scheduling for cloud. *Procedia Computer Science*, *152*, 204-211.

[15]. Wu, C. G., Li, W., Wang, L., & Zomaya, A. Y. (2021). An evolutionary fuzzy scheduler for multi-objective resource allocation in fog computing. *Future Generation Computer Systems*, *117*, 498-509.

[16]. Praveenchandar, J., & Tamilarasi, A. (2021). Dynamic resource allocation with optimized task scheduling and improved power management in cloud computing. *Journal of Ambient Intelligence and Humanized Computing*, *12*(3), 4147-4159.

[17]. Kandan, M., Krishnamurthy, A., Selvi, S., Sikkandar, M. Y., Aboamer, M. A., & Tamilvizhi, T. (2022). Quasi oppositional Aquila optimizer-based task scheduling approach in an IoT enabled cloud environment. *The Journal of Supercomputing*, *78*(7), 10176-10190.

[18]. Ding, D., Fan, X., Zhao, Y., Kang, K., Yin, Q., & Zeng, J. (2020). Q-learning based dynamic task scheduling for energy-efficient cloud computing. *Future Generation Computer Systems*, *108*, 361-371.

[19]. Rjoub, G., Bentahar, J., Abdel Wahab, O., & Saleh Bataineh, A. (2021). Deep and reinforcement learning for automated task scheduling in large-scale cloud computing systems. *Concurrency and Computation: Practice and Experience*, *33*(23), e5919.

[20]. Singh, H., Bhasin, A., & Kaveri, P. R. (2021). QRAS: efficient resource allocation for task scheduling in cloud computing. *SN Applied Sciences*, *3*(4), 1-7.

[21]. Kanbar, A. B., & Faraj, K. H. A. (2022). Region aware dynamic task scheduling and resource virtualization for load balancing in IoT-fog multi-cloud environment. *Future Generation Computer Systems*.