

Efficient fault tolerant cost optimized approach for scientific workflow via optimal replication technique within cloud computing ecosystem

Asma Anjum, Asma Parveen

Department of Computer Science and Engineering, Khaja Banda Nawaz College of Engineering, Kalaburagi, India

Article Info

Article history:

Received Dec 16, 2022

Revised Jan 24, 2023

Accepted Mar 10, 2023

Keywords:

Cloud computing

CyberShake

Fault tolerance

Makespan

Optimal replication technique with fault tolerance and cost minimization

ABSTRACT

Cloud computing is one of the dispersed and effective computing models, which offers tremendous opportunity to address scientific issues with big scale characteristics. Despite having such a dynamic computing paradigm, it faces several difficulties and falls short of meeting the necessary quality of services (QoS) standards. For sustainable cloud computing workflow, QoS is very much required and need to be addressed. Recent studies looked on quantitative fault-tolerant programming to reduce the number of copies while still achieving the reliability necessity of a process on the heterogeneous infrastructure as a service (IaaS) cloud. In this study, we create an optimal replication technique (ORT) about fault tolerance as well as cost-driven mechanism and this is known as optimal replication technique with fault tolerance and cost minimization (ORT-FTC). Here ORT-FTC employs an iterative-based method that chooses the virtual machine and its copies that have the shortest makespan in the situation of specific tasks. By creating test cases, ORT-FTC is tested while taking into account scientific workflows like CyberShake, laser interferometer gravitational-wave observatory (LIGO), montage, and sipt. Additionally, ORT-FTC is shown to be only slightly improved over the current model in all cases.

This is an open access article under the [CC BY-SA](#) license.



Corresponding Author:

Asma Anjum

Department of Computer Science and Engineering, Khaja Banda Nawaz College of Engineering

Kalaburagi, India

Email: asmacs13@gmail.com

1. INTRODUCTION

The use of "pay as you go" is an option with cloud computing services, which are regarded as the most efficient commercial framework for computation by providing consumers with both a computing platform and computing resources. Additionally, this virtual computing paradigm gives users the freedom to present providers with their quality of services (QoS) requirements [1]–[5]. Additionally, recent advances in cloud computing (CC) have prompted a significant expansion of workflow applications across a number of disciplines, including astrophysics along with astronomy as well as bioinformatics, in order to assess applications in light of CC platforms. Additionally, the CC-properties prototypes that includes dynamic resource allocation as well as storage resources. Further, these features can be taken advantage via effective scheduling, which addresses the given specific issues covered later within the segment to perform the efficient system performance [6]–[9]. Figure 1 shows the directed acyclic graph (DAG) model. The main aim for workflow scheduling tends to maximize the given heterogeneous cloud paradigm; in this context, users concentrate for the QoS that includes the charge and deadline execution, when submitting the workflow requirements. Additionally, the growing need for computation as well as services in task scheduling

applications brings with it issues with energy consumption, deadline pressure, time-frame minimization, and cost cutting. Therefore, DAG is used to model workflows; DAG is a pipeline model where the given node is defined as task as well as the edge provides the link between the tasks [10]–[12].

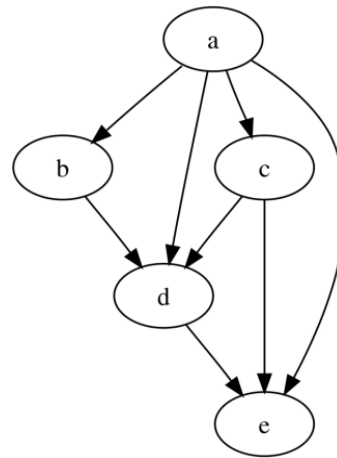


Figure 1. DAG model

The epigenetic condition of cells in human being is mapped by the epigenomics method, a highly pipelined biology application. The majority of the chores have minimal I/O as well as high CPU consumption. In order to detect gravitational waves, the laser interferometer gravitational-wave observatory (LIGO) workflow uses many CPU-intensive tasks that demand a lot of memory. Using the Pegasus project's generator, we may create workflows with a variety of job counts, and these processes are present in DAX format (DAG in XML). The DAX tasks' completing times depends on Intel Core 2 quad-core processor running at 2.4 GHz, which has a processing power of roughly equal to 8 ECUs (2.33 4/1.2 8). We take into account three sizes for each of these workflows: Small comprises 50 jobs, Medium comprises 200 tasks, as well as Large comprises 1,000 tasks. Additionally, 20 distinct instances with the similar structure but varying communication and compute workloads are constructed for each size. Researchers that are interested in green computing and want to reduce costs generally develop energy-aware scheduling; most likely, dynamic voltage and frequency scaling (DVFS) is utilized as the mechanism to reduce energy. Energy-aware mechanisms, however, neglect fault tolerance and cost in favor of just minimizing energy use. To handle enormous data on clouds, scientific workflows need many resources. Real-time cloud services also demand a variety of computational capabilities, which raises the risk of transitory failure. Additionally, a growth in failures and complexity has a negative impact on resource management, leading to QoS issues, particularly with regard to dependability requirements [13]–[15]. While serving the requests of the clients, the servers have to balance the load of the requests from several clients. When the servers are clustered, the main original server is being scaled out horizontally. Moreover, Cloud provides various service; infrastructure service is one of the popular and usable resources that provides the capabilities to re-release or pre-configure the virtual machines from the cloud infrastructure. Moreover, diverse requirement of computing can be fulfilled through the cloud computing as it provides the on demand scalable service including resource and platform [16]–[19]. Additionally, cost minimization is crucial because an ideal cost reflects the effectiveness of the model [20], [21]. The proposed fault-tolerant technique is very efficient way to enhance the dependability of any workflow as well as replication that is primary backup, which seems to be one of the vital software regarding fault-tolerant methods that is applied to meet the given reliability needs. Those fault-tolerant techniques, which already exist, applies either fixed backup for every primary to tolerate simultaneous failures depending as per the active replication method. This may meet the reliability requirements but can lead to unnecessary redundancy and cost, or apply one backup in each main to accept one fault depending on the passive replication scheme, which cannot resist potentially numerous failures [22]–[24]. The following factors further emphasize the value of research work.

- a. For meeting the reliability requirements, here we establish a design and build a fault-tolerant as well as effective mechanism. The proposed methodology is called optimal replication technique with fault tolerance and cost minimization (ORT-FTC).
- b. ORT-FTC employs an iterative selection process to choose virtual machines and attainable duplicates with the least amount of redundancy.

- c. The cost parameter is taken into account while evaluating ORT-FTC; typically, as the virtual machine crashes and more resources are required to deal with the failure, leading to an increase the cost.
- d. In addition, scientific process is considered to demonstrate the effectiveness of paradigm, and four instances with a specific number of virtual machines are constructed to test the model.
- e. A comparison is made, and the proposed methodology shows to be more effective than the current paradigm.

2. LITERATURE SURVEY

One of the key study areas in cloud computing is the scheduling and minimization of scientific workflows. Several other topics have been investigated, including the exploration of diverse workloads, workflows, platforms, and scheduling mechanisms. There are several cases such as makespan as well as energy consumption, cost or reliability and the multi- objective for all minimization targets, this section concentrates on many pertinent jobs related to fault tolerance along with cost minimization. A powerful technique against missing information was developed in [22], where it was recognized that workflow scheduling is now more difficult in the presence of potential resource failure. A failure-aware technique was also suggested in [23] utilizing a Markov chain-based resource availability methodology. The model, however, had a significant degree of dependency, thus [24] created a second dependent model with replication strategy and additional schema included in situation of further failures. Given that resource allocation in clouds is highly reliant, Tao *et al.* [25] presented work queue using replication, or world quality report (WQR), to address the substantial performance penalties that occur from this. Fault tolerance is typically achieved using one of two distinct techniques, either passive replication or active replication [26], [27]. Additionally, the fault tolerance approach can be thought of as a cost-saving improvement to reliability [28]–[31].

When the primary replica fails, passive replication typically refers to the backup replica that is conducted while taking into consideration the virtual machine; therefore, passive replication acceptance through backup alone is rather difficult [32], [33]. Additionally, the cost of adopting passive replication is higher, and redundancy caused by replication is uncertain. The primary replication is reproduced a predetermined number of occasions in every virtual machine in which the task can be effectively finished in the event of active replication. The MaxRe algorithm was first presented in [34], and it generates both cost and redundancy. Here, the reliability requirements for each task are comparable, and each task's need is mathematically equivalent to the defined reliability requirement's square root, where n is assumed to be the given workflow. In contrast to the previous mechanism, Zhao *et al.* [28] offers an optimal resources methodology for confirmation of reliability requirement which minimizes reliability demands, but it does so at a high expense in terms of reduced reliability requirements as well as redundancy. To further apply quantitative-based replication, Zhao *et al.* [28] proposed the engine room resource management (ERRM) technique, which uses an iterative-based approach. For smaller workflows, this model does provide low-cost redundancy minimization, but when applied to large workflows; it fails terribly, making it a bad efficiency model. Despite the fact that evolutionary as well as multi objective techniques and energy utilization are not addressed, their testing results demonstrate that their method greatly minimizes failure events when compared to other existing load balance techniques [35]–[37]. Improved non-dominated sorting genetic algorithm (NSGA-III) algorithm with the addition of an intelligent fault tolerance mechanism to maximize system usage.

3. PROPOSED METHODOLOGY

Because of the huge number of servers and components that are loaded down with workloads, cloud computing generally has significant failure rates. Additionally, these failures might limit the availability of virtual machines, but this problem can be resolved by using the best fault tolerance method. As a result, this component of the study shows the mathematical model of the suggested ORT-FTC methodology, this seeks to provide the best dependability requirement and further reduce costs.

3.1. Preliminaries

Let's have a look at a certain setup of virtual machines, represented by the variable G , where $G = \{G_0, G_1, \dots, G_m\}$; additionally, this configuration includes many parameters, such as cost as well as memory and the total count of virtual machines along with memory. Additionally, when taking into account the virtual machine set, a virtual machine occurrence can be designed by applying the variable W , this W can be mentioned as $W = \{W_0, W_1, \dots, W_h\}$, where W_h directly involves the examples of virtual machine with a particular configuration. It is also important to keep in mind that ORT-FTC is made for a machine that uses parallel processing.

3.2. Workflow modelling

Consider a complex, scientific, and large workflow model including variable A, that is further explained as $A=(X,G)$, where both the semi-variables represent given task set and their dependencies. The dependencies are frequently seen in complex, scientific, as well as large workflow models. Additionally, we initialize a few other factors related to the task, such as $a(w)$, $b(w_x)$, $c(w_x)$ and $d(w_x)$; for example, if the task from v is completed in consideration of the resources w , then the workflow expenses model might be mathematically calculated by the following as displayed in (1). In addition, bandwidth resources can be designed as, mentioned in (2).

$$\mathfrak{C}(w_x, w_y) = \begin{cases} \frac{ed_wt_{wx}}{band_{k,l}} & \text{if } W_l \text{ is not equal to } W_m \\ 0 & \text{if } W_l \text{ is equal to } W_m \end{cases} \quad (1)$$

$$\mathfrak{N}_{l,m} = opt(\mathfrak{N}(\rho(W_l)), \mathfrak{N}(\rho(W_m))) \quad (2)$$

3.3. Task modelling with respect to fault tolerance

Let us suppose, every task as per the set of tasks is assigned the similar timeframe to be completed. Along with the interval parameter of the given task may be computed mathematically as displayed in the following equation, where n_p is the ideal fault tolerance level and i_p is the frequency of operation. Mathematical modelling is given in (3). The scenario is optimum when there is never a failure because of the equation, which can be given as, in (4). Furthermore, assuming task w_x is allocated to resource V_l , ideal execution can be defined as the following equation, where $Q(x_x, W_l)$ includes the total number of jobs as well as Q_l as overhead as displayed in (5). In the meanwhile, here we compute the length of interval through (6).

$$S_p = \left[\sqrt{\left((G_p G_p) (G_{tj_p})^{-1} - 1 \right)} \right] \quad (3)$$

$$\tau_n = G_p + S_p \times g_v \times j_p \quad (4)$$

$$\omega_{best}(x_x, W_l) = (\tau(x_x)) (wm((W_l)))^{-1} + R(x_x, W_l) \cdot R_l \quad (5)$$

$$\varepsilon(x_x, W_l) = wn(\varphi(W_l)) (\tau(x_x))^{-1} \cdot (Q(w_x, W_l) + 1)^{-1} \quad (6)$$

After the best case has been created, it is crucial to create the worst scenario, where the most errors will occur with the task and virtual machine that have already been specified; moreover, $R(x_x, W_l)$ implies the entire overhead, and the fault tolerance overhead will be given by $\varepsilon(x_x, W_l)$ as shown in (7). Therefore, intervals are optimized and their optimality is determined using the (8) for optimizing worst-case scenario. Additionally, error probability is created in order to accept the defect, further defining the task reliability as shown in (9).

$$FU_y(x_x, W_l) = ((X_x))^{-1} \cdot (wm(\tau(W_l)))^{-1} + seg(x_x, W_l) \cdot J_{max} + 2 \cdot R(x_x, W_l) \cdot R_l \quad (7)$$

$$R_{opt}(x_x, W_l) = \sqrt{\left(J_{max}(S_l)^{-1} (\tau(x_x)) \cdot (wn(\tau(W_l)))^{-1} \right) - 1} \quad (8)$$

$$(x, W_l, J) = J! \left(g^{-t.FU_y(x_x, W_l)} (\lambda_k.FU_y(x_x, W_l)^G) \right)^{-1} \quad (9)$$

Task reliability, on the other hand, is described as the likely state in which tasks are carried out even in the event of failure. As a result, the likelihood that tasks will be carried out successfully is $\xi_{succeed}(J, W_l) = g^{-t.ET_y(w_x, V_l)}$. Thus, reliability taking into account the job set is defined as displayed in (10).

$$S(x_x, W_l) = \sum_{J=0}^{J_{max}} (x_x, W_l, J_{max}) \cdot \xi_{succeed}(J, W_l) \quad (10)$$

3.4. Reliability requirement

Two types of system breakdown are present generally, i.e., permanent failure as well as transient failure. This research, however, only takes into account the second form of failure; as a result, we construct the reliability having respect to an occurrence in a specific time frame v . This can be represented in given (11). As

per the (11) v i.e., nu specifies the frequent failures within a given unit of time, on the other hand w_l is castoff to display the constant failures of given virtual machine; further the reliability of p_j within given time w_l is computed as (12). Further, the failure occurred without applying ORT-FTC model is given as in (13).

$$\zeta(v) = f^{-vw} \quad (11)$$

$$T(p_j, w_l) = f^{-w_l v_i l} \quad (12)$$

$$1 - T(p_j, w_l) = 1 - f^{-w_l v_i l} \quad (13)$$

Further, by considering that every task holds a certain number of values that are duplicates, thus $num_h (num_h \leq |U|)$ defined as P_h . Further, we describe duplicate set p_h that is assumed as; $\langle p_h^1, p_h^2, \dots, p_h^{num_h} \rangle$ whereas p_h^1 is the absolute as well as others are duplicates. The total count of duplicates for the workflow is mentioned (14). Once o_h replica is established, it is witnessed that no failure occurred and along with that reliability is updated as (15).

$$num_{dup(H)} = \sum_{h=1}^{|O|} num_j \quad (14)$$

$$T(p_h) = 1 - \sum_{y=1}^{num_h} \left(1 - T(p_h^y, w_{pr(p_h^y)}) \right) \quad (15)$$

In the (15), $w_{pr(p_h^y)}$ indicates $o_{p_h^y}$

$$T(I) = \sum_{p_h \in O} T(p_h) \quad (16)$$

3.5. ORT-FTC cost modelling

While analyzing the proposed workflow model along with the provided virtual machine that is mentioned in the same section earlier results in difficulty is to allocate the replicas together along with the associated virtual machine regarding every individual task, that can be defined as: decreasing the cost executions with dependability. Additionally, execution costs are decreased and fault tolerance is guaranteed thanks to the designed reliability condition in the previous section. The following is the recommended allocation of duplicates (that are also called as backups) and virtual machines,

$$cost(i) = \sum_{p_j \in O} cost(o_{p_j}) \quad (17)$$

$$T(I) = \sum_{p_j \in O} (T(p_h) \text{ is less than or equivalent to } T_{rq}) \quad (18)$$

3.6. Reliability requirement for the individual task

For every individual job's dependability, firstly we define the requirements and then accomplish the requirements. At the initial level, we compute these requirements by applying the absolute reliability that is determined through the prior allocations as per (18). The i th task is denoted by $p_{sq(i)}$ in the (18). We then optimize the dependability requirements using the points provided. The upper constraint for the reliability requirements is job P_h that is supplied by the $\sqrt[|p|]{T_{rq}(I)}$, which we take into consideration when determining the operational needs for each individual task.

$$T_{rq}(p_{sq(i)}) = \sqrt[1-i+|O|]{T_{rq}(I) \left(\sum_{y=1}^{i-1} T(p_{sq(y)}) \right)^{-1}} \quad (19)$$

$$T_{URQ}(P_h) = \sqrt[|p|]{T_{rq}(I)} \quad (20)$$

The given set of task is provided as the unallocated given task $[p_{sq(1)}, p_{sq(2)}, \dots, p_{sq(i-1)}]$ as well as allotted task $[p_{sq(1)}, p_{sq(2)}, \dots, p_{sq(i-1)}]$ if the task given is p (sq(i)) for task I. Additionally, the ORT-FTC model presupposes that the each job within the workflow paradigm will be allocated along with a virtual machine as well as with a reliability parameter value. This is calculated using the method (19), which guarantees reliability. Consequently, the overall reliability need is calculated as given in (21). The (21) can be

used to calculate it for specific tasks as well, and in order to satisfy reliability requirements, duplicate copies and virtual machines with the shortest makespan are chosen iteratively. Additionally, we provide an algorithm that lowers costs and offers efficient fault tolerance regarding the workflow.

$$T_{rq}(I) = \sum_{y=1}^{i-1} \left(T(p_{sq(i)}) \right) \left(T(p_{sq(y)}) \right) \left(\sum_{z=i+1}^{[p]} T_{URQ} p_{sq(z)} \right) \quad (21)$$

$$T_{rq}(p_{sq(i)}) = \left(T_{rq}(I) \right) \left(\sum_{y=1}^{i-1} \left(T(p_{sq(i)}) \right) \left(T(p_{seq(y)}) \right) \left(\sum_{z=i+1}^{[p]} T_{URQ} p_{sq(z)} \right) \right)^{-1} \quad (22)$$

3.7. ORT-FTC process

The main goal of the algorithm is to delegate the need for reliability to sub division while taking into account every single individual task; in addition, the proposed algorithm, called ORT-FTC, reduces the operating cost by choosing duplicates as well as optimal virtual machine. On the other side here, optimal virtual machine are also the ones that have effective execution time; moreover, some redundancy has been witnessed; it was also noted that some of the multiple copies can be discarded. Table 1 displays the ORT-FTC algorithm Table 1. The ORT-FTC methodology schedules the tasks o_h through an order; the best order is determined using the (22), where w_i is the task's execution time as displayed in (22).

$$o_v(p_h) = y_h + o_h \in scc(o_h) \max\{e_{h,i} + o_v(m_j)\} \quad (22)$$

Table 1. ORT-FTC algorithm

| | |
|----------|---|
| Step 1: | Start |
| Step 2: | Input will be taken as the DAG data that is nodes set and execution time as well as communication time along with virtual machine sets the reliability necessity. Output will be the Reliability value as well as the cost appeared along with the schedule length |
| Step 3: | Sorting is done in sloping order. |
| Step 4: | <i>for</i> ($k = 1; k \leq O ; k++$) <i>do</i> compute $S_{rq}(O_{sq(k)})$ $A_{sq(k)} = 0$ $S_{o_{sq(k)}} = 0$ |
| Step 5: | Define a backup list $dup(o_{sq(k)})$ and store that in $o_{sq(k)}$ |
| Step 6: | <i>for</i> ($l = 1; l \leq V ; l++$) <i>do</i> Compute $S_{rq}(o_{sq(k)}, v_l)$ <i>for</i> $o_{sq(k)}$ Compute opt_finish_time <i>end for</i> |
| Step 7: | <i>while</i> ($S(o_{sq(k)})$ is less than $S_{rq}(o_{sq(k)})$) <i>do</i> choose replica $o_{sq(k)}^y$ and VM $v_{pr(o_{sq(k)}^y)}$ with optimal execution time $x_{seq(k),pr(o_{sg(j)}^x)}; num_{sq(k)}++$ |
| Step 8: | Place $o_{seq(k)}^y$ to the dup_list in downward order; further they discard the previous allocations $num_{sq(j)} = 0$ $S(o_{sq(k)}) = 0$ |
| Step 9: | <i>while</i> $S(o_{sq(k)})$ is less than $S_{rq}(o_{seq(k)})$ <i>do</i> choose duplicate $o_{sq(k)}^y$ and $v_{pr(o_{sq(k)}^y)}$ in this given list; also remove replicas $o_{seq(k)}^y$ from that given list and increment $num_{seq(k)}$ |
| Step 10: | compute $finish_time(o_{sq(k)}^y) = opt_finish_time(n_{sg(j)}^x, u_{pr(n_{sg(j)}^x)})$ |
| Step 11: | compute $S(o_{sq(k)})$ <i>end while</i> (step6) |
| Step 12: | <i>end for</i> |
| Step 13: | compute makespan, cost, reliability |

Additionally, ORT-FTC selects the replicas with the best virtual machine, and these virtual machines are set aside as well as sorted in the best way possible. Once the ideal virtual machines have been sorted then the ORT-FTC model cleans the previous allocations and moves on to assigning duplicates. Only the virtual machines with the highest order of dependability are selected by ORT-FTC, and the computation of duplication, execution costs, and optimal makespan is also done.

4. PERFORMANCE EVALUATION

One of the effective real-time computing models that is accessible, affordable, and can be accessible from anywhere over the internet is cloud-computing resources. It can be used for a wide range of purposes and has a number of advantages and disadvantages, includes workflow scheduling in scientific workflows with interdependent as well as independent operations. In this section of our research, it displays the simulation outcomes as well as the comprehensive comparisons of the proposed methodology. Today, workflow-scheduling problems are widely tested using simulation approaches to test novel routing algorithms. In this way, it is possible for academics to compute the performance of the algorithms, which is proposed as per the research in a predictable and skillful way. On a Windows 10 PC with an Intel(R) Core(TM) i5-8300H processor clocked at 2.30 GHz and 8 GB of RAM, the simulations are run using the Java coding environment. All the simulations are performed on a Windows 10 PC. This PC comprises the processor Intel(R) Core(TM) i5-8300H, which is clocked at 2.30 Hz along with 8 GB RAM. All the simulations are done by applying Java coding environment. Five scientific workflow requests are explored here along with huge amount of data as well as the computational characteristics in order to assess our suggested algorithms with a realistic workload: Montage (I/O intensive) along with CyberShake (data intensive) as well as Epigenomics (CPU intensive) and the LIGO (memory intensive), siph (CPU intensive).

4.1. Instance design

For evaluating the model, we created four separate instances, each of which contains a particular number of all virtual machines and a particular variant of the workflow. In addition, four workflows-CyberShake as well as Inspiral and Montage and the siph-are taken into consideration. For example, the CyberShake workflow is designed with a first instance of CyberShake 30 as well as 20 virtual machines, a second example of CyberShake 50 along with 40 virtual machines, and a third instance of CyberShake 100 with 60 virtual machines. Additionally, the fourth instance includes CyberShake 1,000 and 80 virtual machines. Similarly, there are 20, 40, 60, and 80 virtual machines for the four separate instances of Inspiral 30 along with Inspiral 50 as well as Inspiral 100 and 1,000, respectively. Additionally, the total number of all the virtual computers in the scenario of Montage and siph.

4.2. Work cost comparison

4.2.1. CyberShake

The earthquake science program CyberShake, which has high memory and CPU needs, is used to quantify earthquake hazards by merging massive datasets. Figure 2 compares four instances while taking into account the CyberShake workflow; in the first instance, the execution cost for the current model is 1,933.71 whereas the execution cost for the proposed model is 18,418.71. Similarly, the execution costs for the second and third instances of the current model are 21,451.33 and 26,222.62, respectively, while the cost of execution for ORT-FTC are 20,038.49 as well as 23,877.95. In the fourth example, the current model cost us 177,836.21, while the cost of the proposed model is 153,904.64.

4.2.2. LIGO flow analysis

Here, we use the LIGO workflow for generating as well as analyzing the gravitational waveforms of the collected data during coalescing of the compact binary systems. Figure 3 displays the workflow of LIGO. Figure 4 compares the execution costs of the old system versus ORT-FTC. The cost of the current model in the first instance, that is 19,957.79, even though the amount of ORT-FTC is 13,340.13. In the same way, the execution costs for the second and third instances are 35,468.35 and 63,426.63, correspondingly, while the costs for ORT-FTC are 23,705.17 and 42,400.89. In addition, the cost of running the fourth instance of the present model is 6,886,731.17, while the cost of running the ORT-FTC is 459,009.39.

4.2.3. Comparison of cost

The execution expenses of the current and planned ORT-FTC regarding the four examples are contrasted in Figure 4. Whereas the execution expenses regarding the ORT-FTC are 508.31 as well as 1,118.82, the execution expenses of the first and second occurrences are 736.37 as well as 1,628.23, respectively. The execution costs of the current model are 3,430.49 as well as 36,032.25, respectively, similar to the third and fourth cases, while the costs of the suggested model are 2,349.4 as well as 24,636.46.

4.2.4. Comparative analysis of makespan model

In mentioned Figure 5 at Appendix displays the makespan evaluation for each of the four scenarios. Makespan is typically described as the total time needed to accomplish the given task from the beginning to the end. First, the makespan of the existing model is 58.72 while the makespan of the ORT-FTC is 48.67; second, the makespan of the existing model is 97.84 while the makespan of the proposed paradigm is 82.13.

This is Similar to the second as well as third cases, the makespan of the present model is 125.93 and 100.84 respectively, but the makespan of the ORT-FTC model is 299.84 as well as 1,679.35 correspondingly.

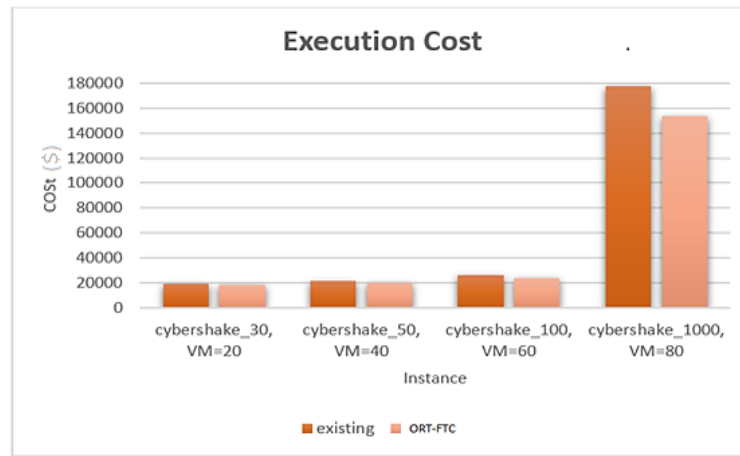


Figure 2. Compares four instances while taking into account the CyberShake workflow

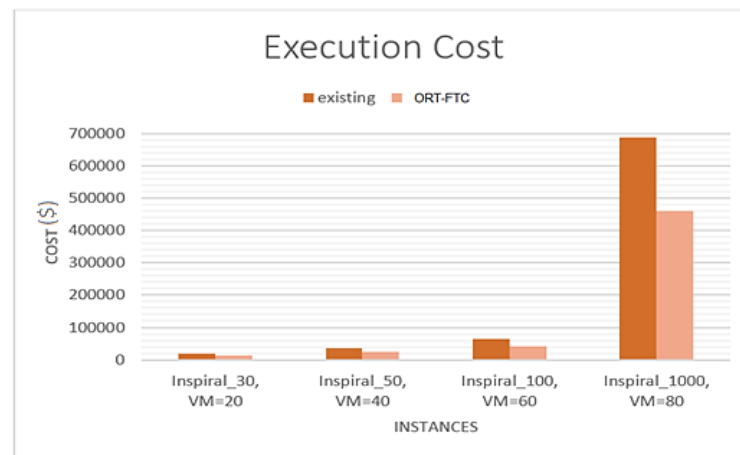


Figure 3. Comparison of execution cost regarding inspiral work

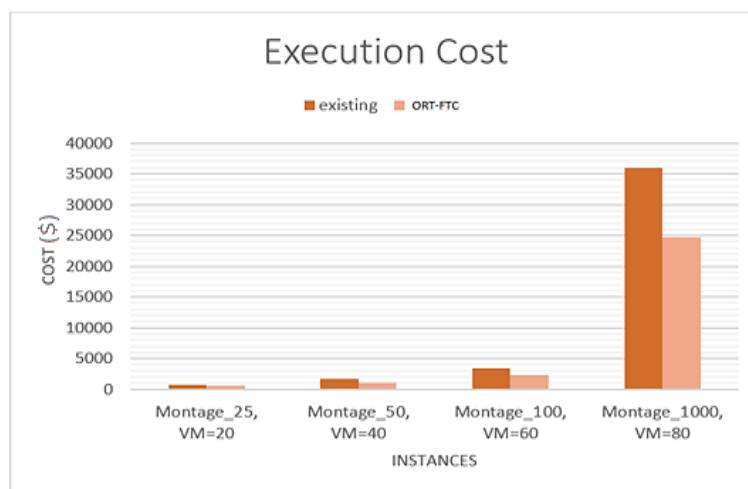


Figure 4. Comparison of montage cost

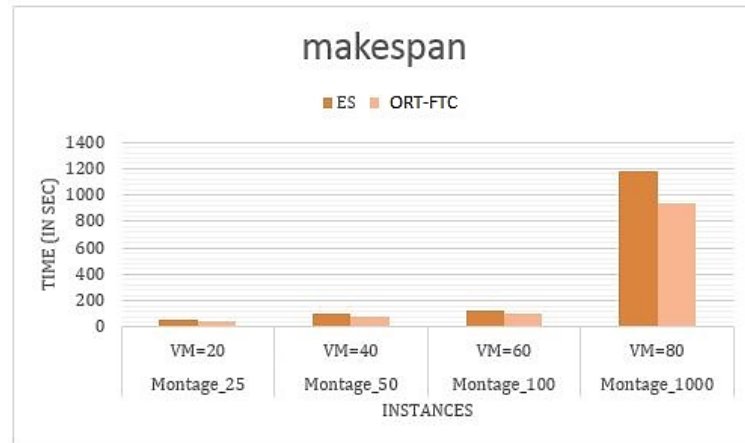


Figure 5. Comparison of makespan

4.2.5. Workflow of sipht

The automated search for Ribonucleic acid (sRNA) encoding genes to bacterial replicons using bioinformatics is done using the sipht methodology. Execution costs for the first and second instances of the current model are 16,668.77 and 35,056.42 correspondingly, whereas ORT-FTC execution costs are 11,121.29 and 23,386.35. Similar to the second and third instances, the suggested model's 34,833.32 and 347,999.82 execution costs are superior to those of the existing model at 52,215.16 and 5,216,668.61, respectively. Figure 6 shows the cost assessment analysis within existing model and proposed model.



Figure 6. Cost assessment analysis within existing model and proposed model

4.3. Comparative assessment

The ORT-improvisation fault tolerance and cost minimization (FTCs) while accounting for the CyberShake process; the ORT-FTC observes, respectively, 4.72% along with 6.58% and 8.94% as well as 13.45% for the first, second, third, as well as the fourth instances. Here ORT-improvisation FTCs with respect to the LIGO workflow; for each of the four cases, ORT-FTC achieves 33.15% as well as 33.16% along with 33.14% and with value of 93.33%, correspondingly. ORT-improvement FTCs over the current model when taking the montage workflow into account; for each of the four cases, ORT-FTC observes an improvement of 30.97% as well as 31.28% along with 31.51% and with value 31.62%, correspondingly. ORT-improvement FTCs over the current model when taking the montage workflow into account in relation to makespan; for each of the four cases, ORT-FTC records improvements of 17.11% as well as 16.05% along with 19.92%, and the value with 20.76%. Corresponding ORT-FTC witnesses an improvement of 33.28% for the three cases as well as 93.32% for the fourth instances as compared to the current model when taking the sipht workflow into account.

5. CONCLUSION

The widespread acceptance of cloud computing and its rising popularity have made it possible to deploy it in large-scale applications; in these situations, cloud environments are also chosen by scientific associations to ensure that workflows are implemented as intended. Despite being so dynamic, cloud computing has a greater failure rate. Failures can happen for a variety of reasons, and these failures lead to a virtual machine being unavailable to execute the task. The fault tolerance approach can be used to design a solution for this problem. As a result, the methodology ORT-FTC established in this research work aims to improve execution cost and reliability. In order to choose a virtual machine and available duplicates with the least amount of redundancy, ORT-FTC employs the duplication method. In addition, ORT-FTC not only minimizes the cost, on the other side it reduces the makes pan. For analyzing the proposed ORT-FTC methodology, we have established four random examples. Here all the instances comprise the particular workflow variation as well as virtual machines in certain number. In addition, the average instance's cost displays an improvisation of 8.42% as well as 48.19% and 31.34%, along with 48.29% for the corresponding CyberShake, Ligo, Montage, and siphT workflows. It's also crucial to note that an aggregate of 18.46% improvisation is shown in the case of the Montage workflow. Despite the fact that ORT-FTC has demonstrated superior fault tolerance as well as cost minimization over other mechanisms for a variety of scientific processes, it must be mentioned that this effectiveness of the model can be varied from one particular workflow to the next depending as per the workload as well as task complexity; correspondingly, future studies may consider this.

ACKNOWLEDGEMENTS

This research was supported by directorate of minority's fellowship for minority students, Bangalore, India. The grant number is order letter no--- DOM/FELLOWSHIP/CR-24/2018-2019.




REFERENCES

- [1] M. Armbrust *et al.*, "A view of cloud computing," *Communications of the ACM*, vol. 53, no. 4, pp. 50–58, Apr. 2010, doi: 10.1145/1721654.1721672.
- [2] H. Chen, X. Zhu, D. Qiu, L. Liu, and Z. Du, "Scheduling for workflows with security-sensitive intermediate data by selective tasks duplication in clouds," *IEEE Transactions on Parallel and Distributed Systems*, vol. 28, no. 9, pp. 2674–2688, Sep. 2017, doi: 10.1109/TPDS.2017.2678507.
- [3] M. A. S. Netto, R. N. Calheiros, E. R. Rodrigues, R. L. F. Cunha, and R. Buyya, "HPC cloud for scientific and business applications: Taxonomy, vision, and research challenges," *ACM Computing Surveys*, vol. 51, no. 1, pp. 1–29, Jan. 2018, doi: 10.1145/3150224.
- [4] G. B. Berriman, G. Juve, E. Deelman, M. Regelson, and P. Plavchan, "The application of cloud computing to astronomy: A study of cost and performance," in *Proceedings - 6th IEEE International Conference on e-Science Workshops, e-ScienceW 2010*, Dec. 2010, pp. 1–7, doi: 10.1109/eScienceW.2010.10.
- [5] Z. Lv and L. Qiao, "Analysis of healthcare big data," *Future Generation Computer Systems*, vol. 109, pp. 103–110, Aug. 2020, doi: 10.1016/j.future.2020.03.039.
- [6] J. Ekanayake, T. Gunarathne, and J. Qiu, "Cloud technologies for bioinformatics applications," *IEEE Transactions on Parallel and Distributed Systems*, vol. 22, no. 6, pp. 998–1011, Jun. 2011, doi: 10.1109/TPDS.2010.178.
- [7] Y. Liu, C. Yang, and Q. Sun, "Thresholds Based Image Extraction Schemes in Big Data Environment in Intelligent Traffic Management," *IEEE Transactions on Intelligent Transportation Systems*, vol. 22, no. 7, pp. 3952–3960, Jul. 2021, doi: 10.1109/TITS.2020.2994386.
- [8] Z. Lv and W. Xiu, "Interaction of Edge-Cloud Computing Based on SDN and NFV for Next Generation IoT," *IEEE Internet of Things Journal*, vol. 7, no. 7, pp. 5706–5712, Jul. 2020, doi: 10.1109/JIOT.2019.2942719.
- [9] Z. Lv and H. Song, "Mobile Internet of Things under Data Physical Fusion Technology," *IEEE Internet of Things Journal*, vol. 7, no. 5, pp. 4616–4624, May 2020, doi: 10.1109/JIOT.2019.2954588.
- [10] H. Chen, J. Wen, W. Pedrycz, and G. Wu, "Big Data Processing Workflows Oriented Real-Time Scheduling Algorithm using Task-Duplication in Geo-Distributed Clouds," *IEEE Transactions on Big Data*, vol. 6, no. 1, pp. 131–144, Mar. 2018, doi: 10.1109/tbdata.2018.2874469.
- [11] X. Zeng *et al.*, "SLA Management for Big Data Analytical Applications in Clouds: A Taxonomy Study," *ACM Computing Surveys*, vol. 53, no. 3, pp. 1–40, Jun. 2020, doi: 10.1145/3383464.
- [12] Z. Li, J. Ge, H. Hu, W. Song, H. Hu, and B. Luo, "Cost and energy aware scheduling algorithm for scientific workflows with deadline constraint in clouds," *IEEE Transactions on Services Computing*, vol. 11, no. 4, pp. 713–726, Jul. 2018, doi: 10.1109/TSC.2015.2466545.
- [13] X. Tang, "Reliability-Aware Cost-Efficient Scientific Workflows Scheduling Strategy on Multi-Cloud Systems," *IEEE Transactions on Cloud Computing*, vol. 10, no. 4, pp. 2909–2919, Oct. 2022, doi: 10.1109/TCC.2021.3057422.
- [14] Z. Li, V. Chang, H. Hu, H. Hu, C. Li, and J. Ge, "Real-time and dynamic fault-tolerant scheduling for scientific workflows in clouds," *Information Sciences*, vol. 568, pp. 13–39, Aug. 2021, doi: 10.1016/j.ins.2021.03.003.
- [15] X. Zhu, J. Wang, H. Guo, D. Zhu, L. T. Yang, and L. Liu, "Fault-Tolerant Scheduling for Real-Time Scientific Workflows with Elastic Resource Provisioning in Virtualized Clouds," *IEEE Transactions on Parallel and Distributed Systems*, vol. 27, no. 12, pp. 3501–3517, Dec. 2016, doi: 10.1109/TPDS.2016.2543731.
- [16] A. Anjum and A. Parveen, "Reliability Aware Cost Driven Mechanism for Scientific Workflows through Optimal Duplication Strategy," Jan. 2022, doi: 10.1109/ICONAT53423.2022.9725990.
- [17] A. Anjum and A. Parveen, "A Dynamic Approach for Live Virtual machine Migration using OU Detection Algorithm," in *Proceedings - 6th International Conference on Computing Methodologies and Communication, ICCMC 2022*, Mar. 2022, pp. 1092–1097, doi: 10.1109/ICCMC53470.2022.9753974.




- [18] A. Anjum and A. Parveen, "Optimized Load Balancing approach in cloud workflow for parallel computing," Nov. 2020, doi: 10.1109/INOCON50539.2020.9298206.
- [19] A. Anjum and R. Patil, "Load balancing for cloud ecosystem using energy aware application scaling methodologies," *Int Res J Eng Technol* 4.5, pp. 479–482, 2017.
- [20] I. Casas, J. Taheri, R. Ranjan, L. Wang, and A. Y. Zomaya, "A balanced scheduler with data reuse and replication for scientific workflows in cloud computing systems," *Future Generation Computer Systems*, vol. 74, pp. 168–178, Sep. 2017, doi: 10.1016/j.future.2015.12.005.
- [21] Z. Ahmad, A. I. Jehangiri, N. Mohamed, M. Othman, and A. I. Umar, "Fault Tolerant and Data Oriented Scientific Workflows Management and Scheduling System in Cloud Computing," *IEEE Access*, vol. 10, pp. 77614–77632, 2022, doi: 10.1109/ACCESS.2022.3193151.
- [22] Z. Ahmad, A. I. Jehangiri, M. Iftikhar, A. I. Umer, and I. Afzal, "Data-Oriented Scheduling with Dynamic-Clustering Fault-Tolerant Technique for Scientific Workflows in Clouds," *Programming and Computer Software*, vol. 45, no. 8, pp. 506–516, Dec. 2019, doi: 10.1134/S0361768819080097.
- [23] J. Schneider, "Grid Workflow Scheduling based on Incomplete Information," Jan. 2010, doi: 10.14279/depositonce-2392.
- [24] Z. Yu, C. Wang, and W. Shi, "Failure-aware workflow scheduling in cluster environments," *Cluster Computing*, vol. 13, no. 4, pp. 421–434, Mar. 2010, doi: 10.1007/s10586-010-0126-7.
- [25] Y. Tao, H. Jin, S. Wu, X. Shi, and L. Shi, "Dependable Grid Workflow Scheduling Based on Resource Availability," *Journal of Grid Computing*, vol. 11, no. 1, pp. 47–61, Sep. 2013, doi: 10.1007/s10723-012-9237-0.
- [26] M. Tao, S. Dong, and K. He, "A new replication scheduling strategy for grid workflow applications," in *Proceedings - 2011 6th Annual ChinaGrid Conference, ChinaGrid 2011*, Aug. 2011, pp. 74–80, doi: 10.1109/ChinaGrid.2011.33.
- [27] Q. Zheng, B. Veeravalli, and C. K. Tham, "On the design of fault-tolerant scheduling strategies using primary-backup approach for computational grids with low replication costs," *IEEE Transactions on Computers*, vol. 58, no. 3, pp. 380–393, Mar. 2009, doi: 10.1109/TC.2008.172.
- [28] L. Zhao, Y. Ren, Y. Xiang, and K. Sakurai, "Fault-Tolerant Scheduling with Dynamic Number of Replicas in Heterogeneous Systems," in *Proceedings - 2010 12th IEEE International Conference on High Performance Computing and Communications, HPCC 2010*, Sep. 2010, pp. 434–441, doi: 10.1109/HPCC.2010.72.
- [29] G. Xie *et al.*, "Minimizing redundancy to satisfy reliability requirement for a parallel application on heterogeneous service-oriented systems," *IEEE Transactions on Services Computing*, vol. 13, no. 5, pp. 871–886, Sep. 2020, doi: 10.1109/TSC.2017.2665552.
- [30] G. Xie, G. Zeng, R. Li, and K. Li, "Quantitative Fault-Tolerance for Reliable Workflows on Heterogeneous IaaS Clouds," *IEEE Transactions on Cloud Computing*, vol. 8, no. 4, pp. 1223–1236, Oct. 2020, doi: 10.1109/TCC.2017.2780098.
- [31] A. Benoit, M. Hakem, and Y. Robert, "Fault tolerant scheduling of precedence task graphs on heterogeneous platforms," Apr. 2008, doi: 10.1109/IPDPS.2008.4536133.
- [32] A. Girault and H. Kalla, "A novel bicriteria scheduling heuristics providing a guaranteed global system failure rate," *IEEE Transactions on Dependable and Secure Computing*, vol. 6, no. 4, pp. 241–254, Oct. 2009, doi: 10.1109/TDSC.2008.50.
- [33] G. Koslovski, W. L. Yeow, C. Westphal, T. T. Huu, J. Montagnat, and P. Vicat-Blanc, "Reliability support in virtual infrastructures," in *Proceedings - 2nd IEEE International Conference on Cloud Computing Technology and Science, CloudCom 2010*, Nov. 2010, pp. 49–58, doi: 10.1109/CloudCom.2010.23.
- [34] F. Pop and R. Prodan, *Adaptive resource management and scheduling for cloud computing*. Springer International Publishing, 2016.
- [35] B. Wu, K. Hao, X. Cai, and T. Wang, "An integrated algorithm for multi-agent fault-tolerant scheduling based on MOEA," *Future Generation Computer Systems*, vol. 94, pp. 51–61, May 2019, doi: 10.1016/j.future.2018.11.001.
- [36] T. Tamilvizhi and B. Parvathavarthini, "A novel method for adaptive fault tolerance during load balancing in cloud computing," *Cluster Computing*, vol. 22, no. S5, pp. 10425–10438, Jul. 2019, doi: 10.1007/s10586-017-1038-6.
- [37] R. Buyya, R. Ranjan, and R. N. Calheiros, "Modeling and simulation of scalable cloud computing environments and the cloudsim toolkit: Challenges and opportunities," in *Proceedings of the 2009 International Conference on High Performance Computing and Simulation, HPCS 2009*, Jun. 2009, pp. 1–11, doi: 10.1109/HPCSIM.2009.5192685.

BIOGRAPHIES OF AUTHORS



Asma Anjum    is a full-time research scholar at Visvesvaraya Technological University, Belagavi, and Karnataka. She has completed her B. E and MTech in Computer Science And Engineering in the year 2015 and 2017 respectively. She has 5 international publications amongst which three are IEEE conferences and Scopus indexed. Her research interest includes networking and cloud computing. She can be contacted at email: asmacs13@gmail.com.



Asma Parveen    got graduated in Electrical Engg., in 1993 and completed post-graduation in Computer Science and Engineering in 2004 and in 2016 she was awarded Ph.D. in Computer Science and Engineering. She has published many research papers in leading international journals and conference proceedings. She can be contacted at email: drasma.cse@gmail.com.