Wireless Personal Communications (2023) 130:1417–1434
https://doi.org/10.1007/s11277-023-10337-z

Reliability Based Workflow Scheduling on Cloud Computing with Deadline Constraint

Savita Khurana[1] · Gaurav Sharma[2] · Manni Kumar[3] · Nitin Goyal[4] · Bhanu Sharma[5]

Accepted: 25 February 2023 / Published online: 16 March 2023

Abstract
Distributed computing workflow is an effective paradigm to express a range of applications with cloud computing platforms for scientific research explorations. One of the most difficult application areas of cloud computing technology is task scheduling. In a cloud, heterogeneous context, job scheduling with minimal execution cost and time, as well as workflow reliability, are critical. While working in the heterogeneous cloud environment, tasks that are successfully executed are widely identified by considering the failure of the processor or any communication technologies link. It will also have an impact on the workflow's reliability as well as the user's service quality expectations. This research paper proposes a Critical Parent Reliability-based Scheduling (CPRS) method that uses the reliability parameter to plan the task while taking into account the user-defined cost and deadline metrics. The effectiveness of the algorithm is compared to current algorithms utilizing scientific workflows as a benchmark, such as Cybershake, Sipht, and Montage. The simulation results supported the assertions by efficiently allocating resources to the cloudlets and stabilizing all of the aforementioned parameters using sufficient performance metrics growth.

Keywords Cloud computing · Reliability · Workflow scheduling · Quality of service

1 Introduction

Cloud computing provides distributing computing environment to access resources over the internet [1]. Cloud computing provides an infrastructure of on-demand resources and services. The goal of cloud computing is to distribute resources in order to attain high availability, reliability, and less failure rate [2]. The services provider and the customers negotiate cost, time, deadline, reliability, etc. quality of service (QoS) parameters and maintained the service level agreement between the service provider and the customer [3]. In a large-scale heterogeneous service environment, reliability is a challenging task. In cloud computing, scientific workflows are designed in which tasks are represented as nodes and edges are the dependencies among the tasks. Several scientific

✉ Nitin Goyal
dr.nitingoyal30@gmail.com

Extended author information available on the last page of the article

workflows exist in which thousands of tasks are under a single workflow on the public cloud [4]. Resource management is one of the challenging issues for a service provider to execute such types of numerous workflows. In cloud computing, workflow procedure is carried out in two stages, first stage involves identifying and acquiring cloud resources to conduct the workflow activities. The second stage provides a timeline for assigning every task to an appropriate resource in order to meet Quality of Service (QoS) criteria including deadline and task priority considerations [5].

Prior studies on workflow scheduling in conventional parallel computing have mostly concentrated upon that task allocation phase since such distributed platforms include a monolithic set of resources that specification is immediately known. Whereas most investigators in conventional distributed systems concentrated on minimizing task scheduling, cloud research focused on certain critical characteristics such as cost, power usage, and safe performance [6]. A number of heuristics and metaheuristics scheduling algorithms have been designed to execute these scientific workflows of NP-hard problems [7]. To achieve an optimal solution to these kinds of problems, heterogeneous earliest-finish-time (HEFT) a list-based heuristic algorithm, min-min algorithms have been proposed to list the task in upward ranking and schedule them to different processors [8].

In a cloud heterogeneous service environment, many factors affect the reliability of the system. A failure can be occurred due to the processor or any kind of network failure. In a workflow, there exist task dependencies, so there is a need for communication between the task. If a communication link failure occurred then it results in an increase in its cost and execution time. So, the reliability of the node is down. So, meeting the QoS requirement like cost, deadline, and reliability is one of the challenging tasks [9–12]. To deal with the reliability of a cloud system, reliable heterogeneous earliest finish time (RHEFT) is considered the product of instruction failure rate with the execution time of the instruction to optimize the reliability and make-span [13].

Communication link failure and processor failure are the major factors that affect the quality-of-service parameters. Reliability is the probability of successful execution of the workflow schedule. Improving reliability by considering deadline and cost of execution is the main concern of the current study. Many fault-tolerant algorithms like quantitative fault-tolerant scheduling algorithm (QFEC) and resource redundancy minimizing algorithm with deadline and reliability requirements (DRR) are also explored to address the reliability of the workflow [10, 14]. In these algorithms task failure on a processor is calculated while ignoring the workflow structure. These methodologies do not consider the ranking of tasks while assigning the resources. Our proposed method considers the priority of individual tasks for resource allocation, that helps to optimize the cost value and resource scheduling in the cloud. The major contribution of this paper is mentioned below.

- Formulation of a novel approach named Critical Parent Reliability-based Scheduling (CPRS) for workflow scheduling which yields optimum reliability-based scheduling by considering cost, and deadline from users.
- An architecture for task prioritization and reliable resource allocation.
- Performance of proposed approach compared with the results of some existing algorithms.

In this paper, cost and time requirements are considered together with reliability. The initial step is to rank the resources. Following that, tasks are prioritized, and a task cluster, as well as a resource cluster, are created depending on execution time. Tasks are then mapped to appropriate reliable resources.

2 Literature Survey

Workflow scheduling is an NP-complete problem. In this paper author surveyed, the best effort-based scheduling algorithm and QoS-based scheduling algorithms in the context of grid computing [15]. The best-effort-based heuristic scheduling algorithms such as Suffrage, Minimum Completion Time, Min-min, and Max-min are discussed [16]. Workflow scheduling algorithms are categorized into two categories: QoS-based constrained and QoS-based optimization [17]. To optimize user's QoS requirements and specifications, QoS-based constrained algorithms are used. Some author's explored minimizing cost while limiting the time.

Sakellariou et al. [18] proposed a scheduling workflow to minimize the makespan. The proposed workflow ensures the schedule must be completed only when the cost factor is within the budget or tasks are remapped to low-cost VMs to meet the budget constraint. All QoS specifications are optimized by using the QoS optimization algorithm [19].

In this area, some research work focused to make a balance between time and cost [20, 21]. Further, it is assumed that at any desired time the processors are accessible. But actually, in the real scenario, network failure, and processor failure are unavoidable. Due to several reasons i.e., failure of link, variation in power, and software/hardware failures resources may become inaccessible [22]. Henceforth it is mandatory to consider the reliability of properly-organized workflow scheduling, to reduce the impact of workflow execution failure [23].

Reliability is calculated by the occurrence of failure of every processor obeying the theory of Poisson distribution having failure rate (λ), which is expected no of failure in unit time t having positive real value [24]. According to the exponential distribution function, the Reliability of the processor is represented by $e - \lambda t$. To enhance the reliability of the system within defined deadline constraints, two algorithms minimum cost match schedule and a progressive reliability maximization schedule were designed [25]. An effective fault-tolerant reliability cost-driven algorithm is proposed in which scheduling algorithms can tolerate one failure in the system [26]. The product of failure rate and execution time of independent tasks is analyzed with the extension of the heuristic-based HEFT algorithm proposed by Dongarra et al. [13]. A task is executed on a processor having the lowest value of the product at that time [27].

The reliability of resources i.e., failure of processor or any node links for communication are addressed by Nik et al. [21]. The authors propose four algorithms that minimize the cost by considering the user-defined reliability and deadline. It firstly divides the workflows into clusters by taking critical parents. Further resources are assigned as per taking the reliability of the resource and applying the time constraint-based algorithm to meet the defined constraints. Reddy and Kumar [28] presented the regressive whale optimization (RWO) method for cloud services task scheduling i.e., an optimization method that scheduled tasks based on a genetic algorithm. The fitness function is determined through the constraints of power consumption, and resource usage. The suggested technique varies from the conventional whale optimization algorithm (WOA) in the formal declaration stage, in which a linear extrapolation position update is introduced. The response is derived arbitrarily in this methodology, with no concern of such resource sort. The Resource Sort delivers the relevant data to VM, that aids in meeting the deadlines constraint. Moreover, it dismisses cloud properties like resilience, and remuneration valuation, as well as the heterogeneous nature of infinite resources, such as performance discrepancies besides VM accretion and cancellation latencies.

Chakravarthi et al. [29] proposed a cost-effective firefly-based algorithm (CEFA) for resolving workflow interruptions during a Cloud computing on infrastructure as a service (IaaS) podium. The suggested CEFA employs a unique technique for task encapsulation, population initiation, with fitness analysis that provide effective workflow execution cost and optimal completions under deadline constraint. Zhang et al. [30] suggested an efficient priority and relative distance (EPRD) methodology for reducing scheduling algorithms duration in constrained workflow applications while maintaining timeline constrain. This technique is made up of two phases. To begin, a task precedence buffer is created. Afterward, VM is assigned to a task based on the relative position. EPRD provides a technique that has the potential to significantly enhance VM usage and scheduling efficiency.

Iranmanesh et al. [31] proposed a hybrid genetic algorithm named "deadline-constraint and cost-effective hybrid genetic algorithm for task scheduling (DCHG-TS)". Due to the consideration of load balancing factor and adaptive fitness, it performs better performance. Additionally, a heuristic method is employed among the estimated sample chromosomes, as well as an efficient process used to generate the remaining fundamental population genomes. Green cloud data centers (GCDCs) proposed by Yuan et al. [32]. The authors employed spatial variation and fragmentation in such inter strategy that produces an optimal balance amongst time and expense to perform overall activities. In this system, nevertheless, the starting population is formed at random and does not accommodate for the optimum load allocation among processors. Concurrently, in addition to its great simplicity and flexibility, new and updated operators in the genetic algorithm have been employed, that considerably decreases the time required to resolve to the best solution.

3 Cloud System Model

In workflow scheduling, scientific workflows are represented as a directed acyclic graph i.e., DAG. Let G be a graph $G = (\mathrm{V}, \mathrm{E}, \mathrm{ET}, \mathrm{TC})$ where V is the set of vertices that represents tasks $\left(T_1, T_2, \ldots T_n\right)$, (E_i, E_j) is the edge or link from task T_i to T_j [33]. If more than one entry nodes and exit nodes exist then two dummy nodes T_{entry} and T_{exit} will be created having zero cost. TE is $r * s$ matrix in which TE_{i*j} the execution time of T_i on resource R_j. TC is the transmission cost with $r * r$ matrix in which $TC_{i,k}$ is the data transmission cost from T_i to T_k.

The following are the general attributes used in the scheduling of tasks [34].

1. Precede (T_i) denotes the predecessor of task T_i is given directed acyclic graph. If any task has no predecessor, then it means it is the entry task. If any DAG has multiple entry vertex, then a dummy node is created in the graph having zero weight and zero communication link weight.
2. Succ (T_i) denotes the successor of task T_i. If no successor is existed, then called it is an exit node. If a number of exit nodes exist, then a dummy exit node is created in the graph with zero weight and zero communication links from the multiple exit nodes.
3. Makespan: It is the maximum execution time of the longest task in the DAG. It is denoted as

$$\text{Makespan} = Max\left\{FT\left(T_L\right)\right\} \tag{1}$$

FT is the last finish time of the exit node. If more than one exit nodes exist then T_L is considered which has taken maximum time to complete.

4. Critical Path (*CP*): the longest path in the schedule of DAG is considered a critical path. The critical path of having minimum computational cost is taken into account for the scheduling of tasks.
5. $ST(T_i, P_j)$: it denotes the start time of a task T_i on processor P_j and computed as

$$ST(T_i, P_j) = max\left\{ T_{avail}(P_j), \max_{T_l \in prec(T_i)} \{ FT(T_l) + C_{l,i} \} \right\} \quad (2)$$

where $\{T_{avail}(P_j)\}$ is the availability of a processor P_j to start execution of Task T_i, T_l is the predecessor of task T_i and $FT(T_l)$ gives the last finish time of the predecessor of the current task.

Compare the finish time of predecessor and availability of the processor to execute, the maximum value of these becomes the starting time of the current task. The $C_{l,i}$ is the commination cost it can be zero if predecessor node T_l is assigned to processor (P_j).

6. $EFT(T_i, P_j)$: it is the Expected Finish Time of a task T_i is on a processor P_j which is the sum of starting time and computational cost of execution of the task and defined as

$$EFT(T_i, P_j) = ST(T_i, P_j) + W_{i,j} \quad (3)$$

4 Problem Formulation

Reliability of resources are addressed in this paper as any workflow execution cannot be guaranteed to be 100% accurate. However, if workflow schedules satisfy the user's reliability constraint while considering cost, makespan and deadline then that workflow execution is considered reliable.

Every cloud system tries to execute tasks with a maximum success rate. The following is a formal description of the problem addressed by this research. The input of the workflow scheduling has given workflow W, deadline D, cost C and reliability R. The problem is to map the tasks to the processors by generating workflows so that cost and time are minimized while meeting the reliability of the workflow. $R_{workflow}$ is the reliability of the workflow.

$$W_{Makespan} = AFT_L - AST_{entry}$$
$$W_{Makespan} \leq W_{Deadline}$$
$$R_{workflow} \geq R$$

To attain the reliability of the workflow, the makespan of the workflow is calculated and compared with the deadline $W_{Deadline}$ of the workflow. The calculated reliability of the workflow is compared with the required reliability of the workflow [35].

This paper proposed an approach schedule of the workflow while considering the makespan, optimized cost value, and deadline of the individual task as well as its whole workflow. As shown in Fig. 1, the proposed approach firstly initializes the task

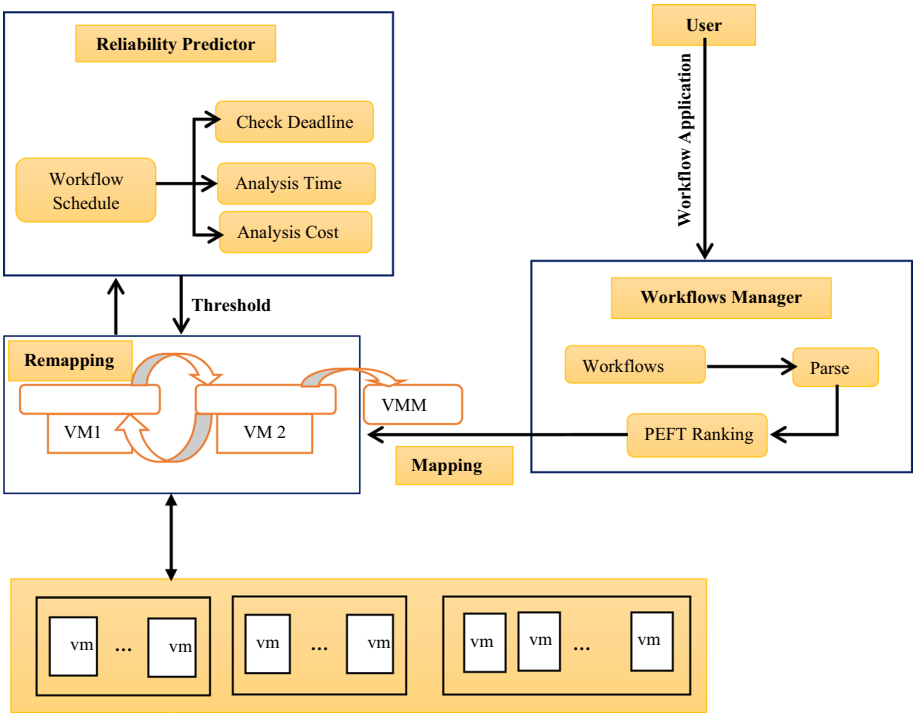


Fig. 1 Architecture of proposed model

to the processor by getting inputs from the user and computing the cost of computation and communication using Eqs. (4) and (5). Afterward, it evaluates the optimized cost of communication as well as computation, this optimization is done through Eq. (6). Based upon the optimized costs, the task is ranked to predict the optimized resource by computing the reliability of resources in the task deadline. Thus, our proposed method helps to optimize cost, reliability as well as a timeline for every task in the workflow.

5 Proposed Critical Parent Reliability-Based Scheduling (CPRS)

The motive of the research is to develop a model in which the scheduling of workflow is discussed. The reliability of the system is to calculate the failure rate of the processor. In DAG-based applications failure rate is described by the Poisson distribution (λ). Reliability is denoted as $e^{(-\lambda t)}$. Reliability depends on reliable communication. In CPRS methodology, shown in Fig. 1, inputs are taken from the user for workflow application and sent to the manager of workflow where task computed based on ranking and mapped to the virtual Machine (VM). Further, the result is evaluated using the VM and remapping process for the reliability prediction of the assigned resource. Further steps that are taken to predict the reliability are discussed below.

5.1 Rank the Tasks

The workflow is represented by DAG. In the DAG approach, V is the vertices which represent task and $E(e_i, e_j)$ is the dependency of task T_j over the task T_j. It means task T_j cannot be started until Task T_i will be executed completely.

5.1.1 Computational Cost Calculation

$V \times P$ is the computational cost matrix. Each weight $W_{i,j}$ in $V \times P$ matrix denotes the expected execution time the task T_i to execute on Processor P_j.

$$W_i = \sum_{j \in P} \frac{W_{i,j}}{P} \tag{4}$$

Here, W_i is average execution time used to calculate priority rank for the task.

5.1.2 Communicational Cost Calculation

Each edge in the DAG is associated with weight $C_{i,j}$. It is the communication cost between task T_i and T_j.

$$\overline{\mathrm{C_{i,j}}} = L + \frac{\mathrm{data}_{i,j}}{\mathrm{B}} \tag{5}$$

Here, L is the latency of the processor, B is the network bandwidth of the link, $\mathrm{data}_{i,j}$ is the amount of data transferred between T_i and T_j, $\overline{\mathrm{C_{i,j}}} = 0$; if T_i and T_j are executed on the same processor P.

5.1.3 Calculate Optimized Cost

A cost-optimized matrix is generated in which rows represent the number of tasks of a workflow and column represents the number of processors. $OC(T_i, P_j)$ indicates the maximum execution cost from T_i from its child nodes to the exit node of the workflow. The value of OC of task T_i on processor P_j is calculated recursively by using the equation below: -

$$OC(T_i, P_j) = \max_{T_j \in Succ(\mathrm{T_i})} \left\{ \left(\min_{P_w \in P} \left(OC(\mathrm{T_i}, \mathrm{P_w}) + \mathrm{w}(\mathrm{T_i}, \mathrm{P_w}) + \overline{\mathrm{C_{i,j}}} \right) \right) \parallel \left(\min_{P_w \in P} \{ (\mathrm{OC}(\mathrm{T_i}, \mathrm{P_w}) + \mathrm{w}(\mathrm{T_i}, \mathrm{P_w}), if P_w = P_k \right) \right\}$$
$$\mathrm{OC}(T_{exit}, P_k) = 0 \quad \text{if} \quad T_{exit} \text{is the last node for} \quad \forall P \tag{6}$$

where $\overline{\mathrm{C_{i,j}}}$ is the communication cost of dependent tasks in the workflow. Its value is zero if dependent tasks are allocated to the same processor otherwise it will be calculated from Eq. (1). $\mathrm{w}(T_i, \mathrm{P_w})$ is the execution cost of the task T_i on Processor $\mathrm{P_w}$. OCT value for the last exit node is zero for all $P_k \in P$.

5.2 Compute Priority of the Tasks

Firstly, calculate the optimized cost (OC) of each task and take the average of the cost computed of a task on each available processor. The calculated average value is used to rank the tasks. This process is used to calculate the rank of the tasks from the starting node to the

exit node with the longest path. In this way prepare schedules of the tasks and rank the task with minimum cost value. It is formulated as

$$Rank(T_i) = \frac{\sum_{k=1}^{P} OC(T_i, P_k)}{P}. \tag{7}$$

To execute the task, we need to select the processor which provides the guarantee that the task will be completed before its defined deadline. So, calculate the earliest finish time of the longest path which is the path from the selected node to the exit node. We calculate the optimized time of the processor by using its Earliest Finish Time, *EFT*, and its optimized cost value by using Eq. (8).

$$O_{EFT} = EFT(T_i, P_j) + OC(\mathrm{T_i}, \mathrm{P_j}) \tag{8}$$

5.3 Proposed CPRS Algorithm

In order to calculate the reliability of workflow, calculate the sub-reliability of each task in the workflow. The above algorithm computes the reliability of individual tasks. Thereafter, computes the overall reliability of the workflow by the use of Eq. 9.

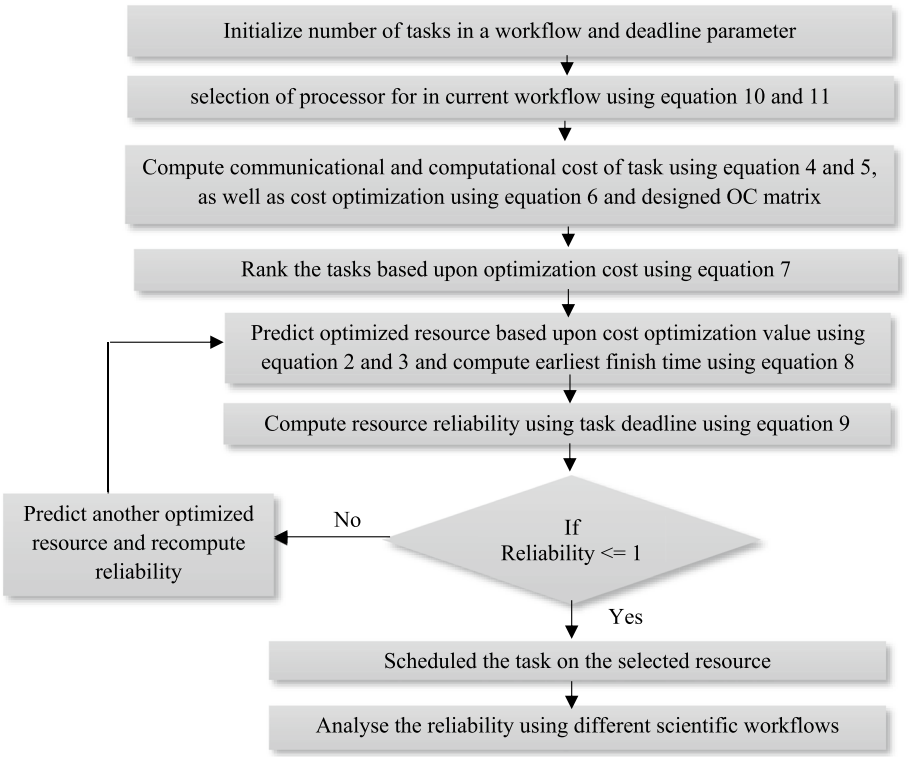


Fig. 2 CPRS working steps

$$R_{act} = \frac{R}{\sum_{i=1}^{n} R_i / n} \tag{9}$$

Here R is the reliability requirement of the workflow and R_{act} is the calculated reliability of the workflow. The proposed algorithm calculates the reliability of different types of scientific workflow applications. The working steps of proposed algorithms are shown in Fig. 2.

The following algorithm demonstrates the model CPRS:

Algorithm: **Critical Parent Reliability-Based Scheduling Algorithm**
Input: Processor (P_j), Tasks (T_i), Deadline(D_i), tasklist
Output: Optimum allocation of cloudlets onto the most reliable virtual machine

1. Initialize the tasklist [] to N, Processor[] to M
2. **Compute the priority of the task using the optimized cost function**
3. Task_List = N and Processor[P_j] = M
4. **while** Task_List != NULL
5. ***For*** each T_i task
6. ***For*** each Processor P_j
7. Compute OC matrix for T_i using equation 4,5,6.
8. ***End for***
9. ***End for***
10. ***End while***
11. For each task T_i
12. do
13. Rank the each tasks T_i using equation 7.
14. **while** Task_List!= NULL
15. **Calculate optimized cost of the resource.**
16. Initialize the Task_List as per their rank value in ascending order.
17. ***While*** Task_List not null ***do***
18. ***For*** each T_i task
19. ***For*** each P_j Processor
20. $EFT\ (T_i,P_j) = EST\ (T_i,P_j) + w_{i,j}$ (using equation 2,3)
21. ***End for***
22. ***End for***
23. Allocate T_i to resource P_j having a minimum value of O_{EFT} using equation 8.
24. ***End while***
25. **Compute the reliability of the resource**
26. ***For*** each task T_i
27. Initialize the D_i, deadline of the task T_i defined and R_i denotes the reliability of the task T_i
28. $R_i = \frac{O_{EFT}(T_i,P_j)}{D_i}$
29. ***If*** $R_i \leq 1$ then
30. Resource is reliable and executes the task to the resource.
31. ***else***
32. Reject the scheduled task.
33. ***End if***
34. ***End loop***

For selecting the appropriate resources to execute the workflow task, the computation reliability ($Rcmp_j$) is computed using Eq. (10).

$$Rcmp_j(\mathrm{t}) = e^{-\lambda_i(TET_{i,j})} \tag{10}$$

$$TET_{i,j} = \sum_{i=1}^{n} w_{i,j} \tag{11}$$

Here, $i \in nooftasksexecutedonprocessor P_j \in P$, $w_{i,j}$ is the mean execution time of T_i on processor P_j and λ is the processor failure rate that is assumed to be constant. There exists a heterogeneous computing environment in the cloud, so computation reliability may vary from resource to resource.

Communication Reliability of link ($Rcmm_{j,k}$) is calculated as per the execution of parent and its child task on the same resource or different. In this proposed work comparison of reliability with the different existing algorithms is done. The proposed work performed better results as compared to existing.

6 Results and Analysis

To validate the results of the CPRS algorithm, sample workflows are randomly generated with a different number of tasks from 20 to 30 and simulated on cloudSim. Three different scientific workflows Cybershake, Montage, and Sipht are studied to analysed the reliability and deadline of every workflow of the CPRS scheduling algorithm. The simulation results are validated using Cybershake, Montage, and Sipht datasets and compare the results with the existing efficient priority and relative distance (EPRD) algorithm and Cost-Effective Firefly based Algorithm (CEFA). The target reliability of the workflow is compared with the actually obtained reliability.

6.1 Experimental Setup

The virtualized environment has been realised by infusing numerous simulation parameters that have been created by randomising the workflow model established under Sects. 3 and 4. This has been done to analyse and measure the performance of the suggested approach. By varying the (a) number of virtual machines, their operating frequencies further mapped on to Million Instructions Per Second (MIPS), and computation capacities of machines considered for task allocation, a number of parameters, essentially the cloud-based expedients and resources underlying the workflow model, are considered. Furthermore, each virtual computer includes (b) existing workload; various (c) tasks may be referred to as Cloudlets, their length, and eventually, showing the (d) scientific workflow with counts and types.

Table 1 summarises the set of parameters and metrics, as well as some supplementary configuration details in terms of hardware requirements.

The experiment on Montage scientific workflow with 20 tasks and 30 tasks are simulated, shown in Fig. 3a and b. Because the proposed algorithm CPRS provides the solutions of montage workflow with less makespan, shows an average of 4% and 2% better results as compared to EPRD and CEFA respectively.

Similarly, reliability is compared with Cybershake and Sipht Workflow using 20 and 30 tasks. The results show that CPRS algorithm perform better as compared to the EPRD and CEFA algorithm. In Cybershake workflow, Fig. 4a and b illustrate that proposed algorithm results also show better outcome as compared to EPRD and CEFA algorithms. It has been observed from these graphs that EPRD performs an average of 6% and 4% better

performance for 20 and 30 tasks. But in the case of Sipht workflow, results are slightly varied from the existing algorithm.

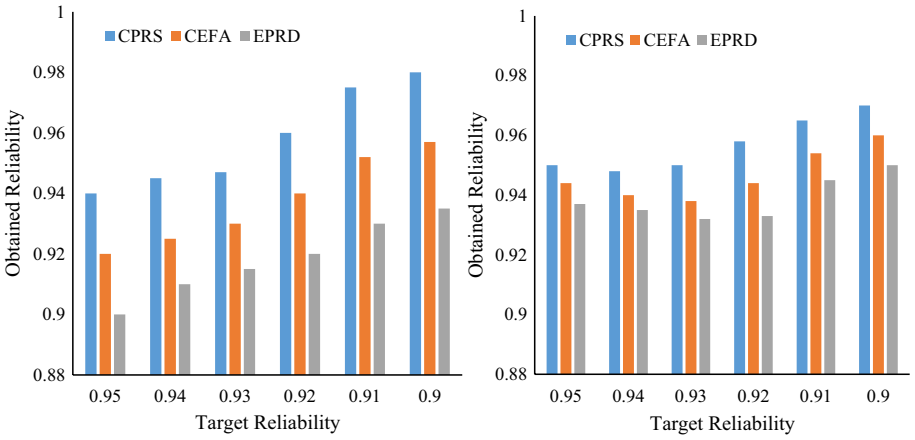
As Sipht is a complex workflow and also achieved the threshold reliability with the proposed scheduling algorithm. The results are compared on different defined threshold values. A processor is not able to perform 100% reliability every time. So, different threshold values vary from 0.95 to 0.90. Figure 5a and b illustrate the reliability of existing algorithms with comparison of proposed algorithms on 20 and 30 tasks. It has been carried out from the analysis that EPRD perform an average of 5% and 2% better than CPRS and CEFA on different tasks.

6.2 Discussion with Existing Work

In a cloud setting, the experimental study was carried out on computers with low and high heterogeneity. Based on the observations and statistical data obtained, it can be concluded that the EPRD and CEFA have lower reliability as compare to proposed CPRS algorithm. This is because existing approaches generate the initial population using the canonical dispatching rule, but as the size of the initial population grows, so does the corresponding execution time. As a result, the fundamental issue with prior methodologies is that when the initial population was produced randomly, it resulted in an exponential increase in execution time when compared to the execution time achieved utilising the proposed methodology CPRS. In a similar vein, another critical parameter that has remained the focus of the current study is incurred reliability. It was discovered that the proposed algorithm incurred the highest reliability when compared to existing algorithms because of the lowest makespan and optimised ranking, which results in task completion before the deadline and results in the highest reliability when using different scientific workflows such as Cybershake, Montage, and Sipht. All of the aforementioned concerns are addressed at the very beginning, namely, the reliability index factor, machine utilisation factor, and task prioritisation, all of which improved the overall makespan time as well as the cost factor, which was a shortcoming in traditional approaches used for comparison. Based on the results of the comparison study, it can be stated that include all of these criteria in the proposed approach at various stages greatly enhanced the results. Finally, the suggested CPRS method revealed that it surpassed existing strategies in terms of efficacy and efficiency.

Table 1 Simulation set-up

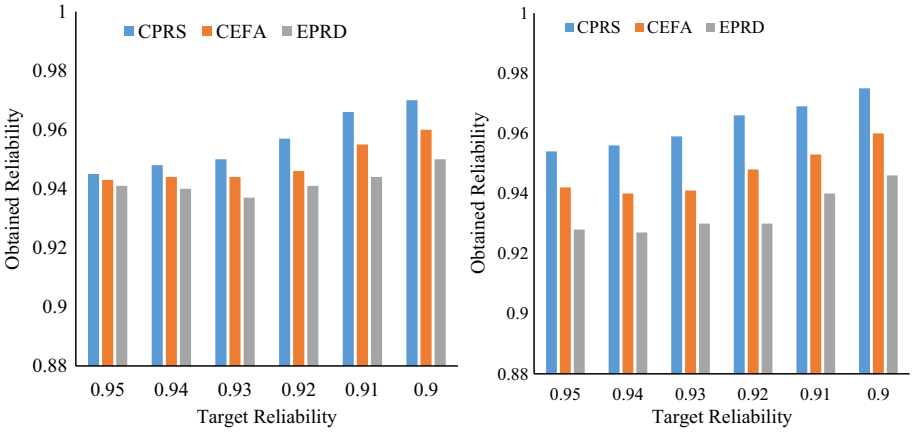
Parameters	Specifications
Workstation	DELL Inspiron 5518 Core i5 11th Gen 16 GB/512 GB SSD
Operating environment	MS Windows 10
Task length	1000–20,000
Total number of tasks	20–30
Computational capacity	1000–10,000 (MIPS)
Link bandwidth	100–200 (Mbps)
Total number of VMs	2–20
VM load	0–40(%)
Cluster size	200, 400 and 1000 hosts
Turnaround times (TRT)	40 ms, 60 ms, 80 ms and 100 ms



(a). Reliability comparison on Montage Workflow with 20 tasks

(b). Reliability comparison on Montage Workflow with 30 tasks

Fig. 3 **a** Reliability comparison on Montage Workflow with 20 tasks. **b** Reliability comparison on Montage Workflow with 30 tasks



(a). Reliability comparison on Cybershake Workflow with 20 tasks

(b). Reliability comparison on Cybershake Workflow with 30 tasks

Fig. 4 **a** Reliability comparison on Cybershake Workflow with 20 tasks. **b** Reliability comparison on Cybershake Workflow with 30 tasks

7 Conclusion

In this paper, a new reliability-aware algorithm critical parent reliability-based scheduling (CPRS) is proposed in which optimized cost, user-defined deadline, and reliability constraints are considered for scheduling workflows. Reliability is one of the important parameters of QoS. It is the successful execution of the workflow. The results of the proposed approach are compared with EPRD and CEFA approaches. For evaluation of results three

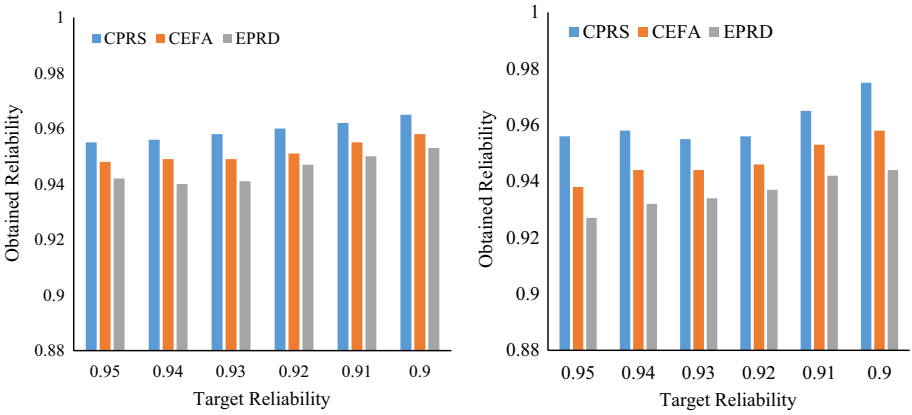


Fig. 5 **a** Reliability comparison on Sipht Workflow with 20 tasks. **b** Reliability comparison on Sipht Workflow with 30 tasks

different scientific workflows Cybershake, Sipht and Montage are taken. To evaluate the reliability of workflow, the unreliability of every individual task is computed by considering its optimized cost and deadline. The proposed algorithm results outperform as compared to the existing metaheuristic algorithm.

Authors' Contributions SK conceived the idea, designed the experiments and analysed the data; GS performed the experiments and conducted the analysis; MK and NG analysed the methods, interpreted the results and drew the conclusions; BS proofread the paper. All the authors agree with the above contribution details.

Funding Not applicable.

Availability of Data and Materials Not applicable.

Code Availability Not applicable.

Declarations

Conflict of interest No conflict of interest.

References

1. Bawa, R. K., & Sharma, G. (2012). Reliable resource selection in grid environment. arXiv preprint arXiv:1204.1516
2. Badotra, S., & Panda, S. N. (2022). Software defined networking: a crucial approach for cloud computing adoption. *International Journal of Cloud Computing, 11*(2), 123–137. https://doi.org/10.1504/IJCC.2022.122028
3. Faragardi, H. R., Shojaee, R., & Yazdani, N. (2012, June). Reliability-aware task allocation in distributed computing systems using hybrid simulated annealing and tabu search. In *2012 IEEE 14th international conference on high performance computing and communication & 2012 IEEE 9th international*

conference on embedded software and systems (pp. 1088–1095). IEEE. https://doi.org/10.1109/HPCC.2012.159
4. Shojaee, R., Faragardi, H. R., Alaee, S., & Yazdani, N. (2012, November). A new cat swarm optimization based algorithm for reliability-oriented task allocation in distributed systems. In *6th international symposium on telecommunications (IST)* (pp. 861–866). IEEE. https://doi.org/10.1109/ISTEL.2012.6483106
5. Sahoo, S., Sahoo, B., Turuk, A. K., & Mishra, S. K. (2017). Real time task execution in cloud using mapreduce framework. In *Resource management and efficiency in cloud computing environments* (pp. 190–209). IGI Global. https://doi.org/10.4018/978-1-5225-1721-4.ch008
6. Olakanmi, O. O., & Dada, A. (2019). An efficient privacy-preserving approach for secure verifiable outsourced computing on untrusted platforms. *International Journal of Cloud Applications and Computing, 9*(2), 79–98. https://doi.org/10.4018/IJCAC.2019040105
7. Rani, M., Guleria, K., & Panda, S. N. (2021, September). Cloud Computing An Empowering Technology: Architecture, Applications and Challenges. 9th International Conference on Reliability, Infocom Technologies and Optimization (Trends and Future Directions) (ICRITO) (pp. 1–6). IEEE. https://ieeexplore.ieee.org/abstract/document/9596259
8. Daoud, M. I., & Kharma, N. (2008). A high performance algorithm for static task scheduling in heterogeneous distributed computing systems. *Journal of Parallel and Distributed Computing, 68*(4), 399–409. https://doi.org/10.1016/j.jpdc.2007.05.015
9. Zhou, A., Wang, S., Cheng, B., Zheng, Z., Yang, F., Chang, R. N., Lyu, M. R., & Buyya, R. (2016). Cloud service reliability enhancement via virtual machine placement optimization. *IEEE Transactions on Services Computing, 10*(6), 902–913. https://doi.org/10.1109/TSC.2016.2519898
10. Zhao, L., Ren, Y., & Sakurai, K. (2013). Reliable workflow scheduling with less resource redundancy. *Parallel Computing, 39*(10), 567–585. https://doi.org/10.1016/j.parco.2013.06.003
11. Qiu, W., Zheng, Z., Wang, X., Yang, X., & Lyu, M. R. (2013). Reliability-based design optimization for cloud migration. *IEEE Transactions on Services Computing, 7*(2), 223–236. https://doi.org/10.1109/TSC.2013.38
12. Silic, M., Delac, G., & Srbljic, S. (2014). Prediction of atomic web services reliability for QoS-aware recommendation. *IEEE Transactions on services Computing, 8*(3), 425–438. https://doi.org/10.1109/TSC.2014.2346492
13. Dongarra, J. J., Jeannot, E., Saule, E., & Shi, Z. (2007, June). Bi-objective scheduling algorithms for optimizing makespan and reliability on heterogeneous systems. In *Proceedings of the nineteenth annual ACM symposium on parallel algorithms and architectures* (pp. 280–288). https://doi.org/10.1145/1248377.1248423
14. Zheng, Q., & Veeravalli, B. (2009). On the design of communication-aware fault-tolerant scheduling algorithms for precedence constrained tasks in grid computing systems with dedicated communication devices. *Journal of Parallel and Distributed Computing, 69*(3), 282–294. https://doi.org/10.1016/j.jpdc.2008.11.007
15. Yu, J., Buyya, R., & Ramamohanarao, K. (2008). Workflow scheduling algorithms for grid computing. In *Metaheuristics for scheduling in distributed computing environments* (pp. 173–214). Springer. https://doi.org/10.1007/978-3-540-69277-5_7
16. Yu, J., & Buyya, R. (2005). A taxonomy of workflow management systems for grid computing. *Journal of Grid Computing, 3*(3), 171–200. https://doi.org/10.1007/s10723-005-9010-8
17. Arabnejad, H., & Barbosa, J. G. (2014). A budget constrained scheduling algorithm for workflow applications. *Journal of Grid Computing, 12*(4), 665–679. https://doi.org/10.1007/s10723-014-9294-7
18. Sakellariou, R., Zhao, H., Tsiakkouri, E., & Dikaiakos, M. D. (2007). Scheduling workflows with budget constraints. In *Integrated research in GRID computing* (pp. 189–202). Springer. https://doi.org/10.1007/978-0-387-47658-2_14
19. Miglani, N., & Sharma, G. (2018). An adaptive load balancing algorithm using categorization of tasks on virtual machine based upon queuing policy in cloud environment. *International Journal of Grid and Distributed Computing, 11*(11), 1–2. https://doi.org/10.14257/ijgdc.2018.11.11.01
20. Su, S., Li, J., Huang, Q., Huang, X., Shuang, K., & Wang, J. (2013). Cost-efficient task scheduling for executing large programs in the cloud. *Parallel Computing, 39*(4–5), 177–188. https://doi.org/10.1016/j.parco.2013.03.002
21. Mousavi Nik, S. S., Naghibzadeh, M., & Sedaghat, Y. (2020). Cost-driven workflow scheduling on the cloud with deadline and reliability constraints. *Computing, 102*(2), 477–500. https://doi.org/10.1007/s00607-019-00740-5

22. Kianpisheh, S., & Moghadam Charkari, N. (2014). A grid workflow Quality-of-Service estimation based on resource availability prediction. *The Journal of Supercomputing, 67*(2), 496–527. https://doi.org/10.1007/s11227-013-1014-8
23. Khurana, S., & Singh, R. K. (2018, September). Virtual machine categorization and enhance task scheduling framework in cloud environment. In *2018 international conference on computing, power and communication technologies (GUCON)* (pp. 391–394). IEEE. https://doi.org/10.1109/GUCON.2018.8675020
24. Xie, G., Zeng, G., Chen, Y., Bai, Y., Zhou, Z., Li, R., & Li, K. (2017). Minimizing redundancy to satisfy reliability requirement for a parallel application on heterogeneous service-oriented systems. *IEEE Transactions on Services Computing, 13*(5), 871–886. https://doi.org/10.1109/TSC.2017.2665552
25. Zhao, L., Ren, Y., & Sakurai, K. (2011, March). A resource minimizing scheduling algorithm with ensuring the deadline and reliability in heterogeneous systems. In *2011 IEEE international conference on advanced information networking and applications* (pp. 275–282). IEEE. https://doi.org/10.1109/AINA.2011.87
26. Qin, X., Jiang, H., & Swanson, D. R. (2002, August). An efficient fault-tolerant scheduling algorithm for real-time tasks with precedence constraints in heterogeneous systems. In *Proceedings international conference on parallel processing* (pp. 360–368). IEEE. https://doi.org/10.1109/ICPP.2002.1040892
27. Boeres, C., Sardiña, I. M., & Drummond, L. M. (2011). An efficient weighted bi-objective scheduling algorithm for heterogeneous systems. *Parallel Computing, 37*(8), 349–364. https://doi.org/10.1016/j.parco.2010.10.003
28. Narendrababu Reddy, G., & Phani Kumar, S. (2019). Regressive whale optimization for workflow scheduling in cloud computing. *International Journal of Computational Intelligence and Applications, 18*(04), 1950024. https://doi.org/10.1142/S146902681950024X
29. Chakravarthi, K. K., Shyamala, L., & Vaidehi, V. (2021). Cost-effective workflow scheduling approach on cloud under deadline constraint using firefly algorithm. *Applied Intelligence, 51*(3), 1629–1644. https://doi.org/10.1007/s10489-020-01875-1
30. Zhang, L., Zhou, L., & Salah, A. (2020). Efficient scientific workflow scheduling for deadline-constrained parallel tasks in cloud computing environments. *Information Sciences, 531*, 31–46. https://doi.org/10.1016/j.ins.2020.04.039
31. Iranmanesh, A., & Naji, H. R. (2021). DCHG-TS: A deadline-constrained and cost-effective hybrid genetic algorithm for scientific workflow scheduling in cloud computing. *Cluster Computing, 24*(2), 667–681. https://doi.org/10.1007/s10586-020-03145-8
32. Yuan, H., Liu, H., Bi, J., & Zhou, M. (2020). Revenue and energy cost-optimized biobjective task scheduling for green cloud data centers. *IEEE Transactions on Automation Science and Engineering, 18*(2), 817–830. https://doi.org/10.1109/TASE.2020.2971512
33. Pham, T. P., Durillo, J. J., & Fahringer, T. (2017). Predicting workflow task execution time in the cloud using a two-stage machine learning approach. *IEEE Transactions on Cloud Computing, 8*(1), 256–268. https://doi.org/10.1109/TCC.2017.2732344
34. Arabnejad, H., & Barbosa, J. G. (2013). List scheduling algorithm for heterogeneous systems by an optimistic cost table. *IEEE Transactions on Parallel and Distributed Systems, 25*(3), 682–694. https://doi.org/10.1109/TPDS.2013.57
35. Khurana, S., & Singh, R. (2020). Workflow scheduling and reliability improvement by hybrid intelligence optimization approach with task ranking. *EAI Endorsed Transactions on Scalable Information Systems*. https://doi.org/10.4108/eai.13-7-2018.161408

Dr. Savita Khurana received her B.Tech & M.Tech in Computer Science and Engineering from Kurukshetra University, Haryana, India. She has completed her Ph.D. from I.K. Gujral Punjab Technical University, Kapurthala, Jalandhar, Punjab, India in 2021. She is having 13 years of teaching experience. She has published various research papers/chapters in ESCI/SCOPUS/WoS Journals, Book Chapters and Conferences. She conducted/organized various technical events/workshops for students. Her research interest is Cloud Computing, Machine Learning, and Wireless Sensor Networks.

Dr. Gaurav Sharma is working as Professor in JMIT Radaur Engineering College affiliated to Kurukshetra University, India. He is having 17 years of teaching and academic experience. He obtained M.Tech from G.J.U, Hisra and Ph.D. from Punjabi University Patiala, India. He has published approximately 44 research papers/chapters in various SCI/ /ESCI/SCOPUS/WoS Journals, Book Chapters and Conferences. He has guided 12 M.Tech candidates. He organized various national level seminars/conferences and faculty development programs. He conducted/organized various technical events/workshops for diploma level students of various institutes. He is CSI member and reviewed various research papers of reputed publication houses as Springer/ Elsevier/IEEE etc. His research interests include cloud computing, Fog computing, WSN.

Mr. Manni Kumar is working as an Assistant Professor in Chandigarh University, Punjab, India. Previously, he has also worked as the Assistant Lecturer in Chitkara University, Punjab, India. He completed his Master of Engineering in the field of UWSN from Chitkara University, Punjab, India. He has filled 6 patents and has 3 journal publications, 2 conference pulications and 2 book chapters in IGI Global and Wiley publications.

Dr. Nitin Goyal is a Ph.D. in Computer Engineering from NIT Kurukshetra, India. He is specialized in underwater wireless sensor networks (UWSN), having applications of military operations. He obtained his B.Tech in 2007 and M.Tech in 2009 from Computer Science and Engineering under Kurukshetra University, India. He has around 14 years of teaching experience along with Academic work in various organisations like Chitkara Univ., JMIT Radaur, NCCE Israna. Currently, he is working with Central University of Haryana, India. He has published approximately 110 research papers in various International/National Journals, Books and Conferences out of which 55 are SCI, 3 are SCOPUS, 11 are book chapters, and 27 are conference papers. He has filed around 25 patents, out of which 7 are published and 3 are awarded. Also, he has attended 13 workshops and 3 FDP. He is the editor of 3 books from different publishers as IGI Global, Bentham Science, CRC Press. He has guided 10 M.Tech candidates and 2 Ph.D. candidates. He has also delivered 7 expert lectures on NS2, Paper Writing, Patents Filing, UWSN, Innovation and Entrepreneurship. He is a GATE, HTET and UGC-NET qualifier too. He is currently working in the area of UAV in underwater communication networks to solve some real-world problems. His research interests include MANET, FANET, UWSN and WSN. He is a Senior Member of IEEE. He has been the Guest Editor of SCI/SCIE/SCOPUS indexed reputed Journals from SAGE, Frontiers and MDPI publications. He is in the reviewer panel, editorial board member for many journals of repute, and also session chair in many international conferences.

Dr. Bhanu Sharma is currently working as Associate Professor at Chitkara University. She has received 3 Years Diploma in ECE from State Board of Technical Education Haryana and AMIETE from New Delhi. She has completed Ph.D. in ECE (Augmented Reality in Education) in Chitkara University, Rajpura, Punjab. She has more than 22 publications. She has an experience of 20 years serving industry and education. She has guided students for Analog Design Contest by Texas Instruments and for Niyantra by National Instruments. She has an expert hand on troubleshooting electronic circuits and designing.

Authors and Affiliations

Savita Khurana[1] · Gaurav Sharma[2] · Manni Kumar[3] · Nitin Goyal[4] · Bhanu Sharma[5]

Savita Khurana
savu.khurana30@gmail.com

Gaurav Sharma
gaurav13@gmail.com

Manni Kumar
mannikumar55@gmail.com

Bhanu Sharma
bhanu.sharma@chitkara.edu.in

[1] Information Technology Department, Seth Jai Parkash Mukand Lal Institute of Engineering and Technology, Radaur, Haryana, India

[2] Computer Science and Engineering Department, Seth Jai Parkash Mukand Lal Institute of Engineering and Technology, Radaur, Haryana, India

[3] Department of Computer Science Engineering, Chandigarh University, Gharuan, SAS Nagar Mohali, Punjab, India

[4] Department of Computer Science and Engineering, School of Engineering and Technology, Central University of Haryana, Mahendragarh, Haryana, India

[5] Immersive and Interactive Technology Lab, Chitkara University Institute of Engineering and Technology, Chitkara University, Rajpura, Punjab, India