DOI 10.12694/scpe.v20i2.1515
Scalable Computing: Practice and Experience ISSN 1895-1767
Volume 20, Number 2, pp. 237–258. http://www.scpe.org

HYBRID BALANCED TASK CLUSTERING ALGORITHM FOR SCIENTIFIC WORKFLOWS IN CLOUD COMPUTING

AVINASH KAUR *, POOJA GUPTA †, AND MANPREET SINGH ‡

Abstract. Scientific Workflow is a composition of both coarse-grained and fine-grained computational tasks displaying varying execution requirements. Large-scale data transfer is involved in scientific workflows, so efficient techniques are required to reduce the makespan of the workflow. Task clustering is an efficient technique used in such a scenario that involves combining multiple tasks with shorter execution time into a single cluster to be executed on a resource. This leads to a reduction of scheduling overheads in scientific workflows and thus improvement of performance. However available task clustering methods involve clustering the tasks horizontally without the consideration of the structure of tasks in a workflow. We propose hybrid balanced task clustering algorithm that uses the parameter of impact factor of workflows along with the structure of workflow. According to this technique, tasks can be considered for clustering either vertically or horizontally based on the value of the impact factor. This minimizes the system overheads and the makespan for execution of a workflow. A simulation based evaluation is performed on real workflows that shows the proposed algorithm is efficient in recommending clusters. It shows improvement of 5-10% in makespan time of workflow depending on the type of workflow used.

Key words: Metadata; Scientific Data Management; Data Sharing; Data Integration; Computer Supported Collaborative Work

AMS subject classifications. 68M20, 91C20

1. Introduction.

In past years of scientific discovery [16], the computational workflow continues to be popular among various disciplines of science, including astronomy[36], physics[11], biology [23, 33], seismology [25], chemistry [45] and others.

A workflow is a series of activities representing business processes or computational science with existing dependencies between them. These dependencies need to be satisfied for the achievement of a goal. Business workflow is a control-flow driven activity including constructs for specifying conditions, paths and also involve human interaction. It implements the company's services or products. Scientific workflow involves large scale data and/or complex computations, therefore, utilizes computing resources and storage capacities[32]. It does not involve control-flow but is data-driven, while there are exceptions such as Askalon [34].

Large amount of data processing is required by scientific workflows that consist of millions of uncommon tasks[6]. For example, the Cybershake workflow [26] containing 800,000 tasks is executed on TeraGrid [41]. These loosely coupled applications in all represent a significant amount of computation and data [11]. Existing applications such as Condo r[20] do not consider overheads in system, fault occurrence or restructuring of a workflow [40, 31].

To improve the scalability of the system and to reduce system overheads, workflow restructuring techniques such as task clustering is introduced [37, 18, 48]. It is a process where smaller tasks are merged into a larger job[37] that is the single execution unit in a workflow management system. After application of task clustering on tasks, the execution units are reduced and in turn, it leads to an increase in application computation, thus reducing system overheads.

However, various existing methods use trial-and-error approach for optimization of workflow structures. For example, Horizontal Clustering (HC) [37] merge different tasks at the same level of workflow horizontally. For a single task, the horizontal level is defined as the largest distance from start task of the Directed Acyclic Graph (DAG) to this task. The user controls the granularity of clustering that is the number of tasks within a cluster and defines either the number of jobs clustered per horizontal level or a number of tasks per clustered job. This kind of techniques ignored the dynamic properties of distributed environments [38].

Many methods are introduced for reducing the system overhead and clustering the tasks either horizontally or vertically but none of the technique employs both kinds of clustering simultaneously. The structure of the workflow plays a significant role in clustering of tasks of a workflow.

*Lovely Professional University, Phagwara, India (avinash.14557@lpu.co.in).
†Lovely Professional University, Phagwara, India (pooja.19580@lpu.co.in).
‡GNDEC, Ludhiana, India (mpreet78@gmail.com).

The work proposes hybrid balanced clustering algorithm which takes into consideration both the structure of workflow and number of jobs available for tasks to be clustered. Also, the tasks with the parent-child relationship are clustered together vertically and tasks without the relationship are considered horizontally. Hence, this helps in reducing systems overheads and faster execution of tasks and minimum wastage of resources. The important points considered in proposed work are

- Minimum tasks overheads: The overheads are reduced to the minimum while the tasks with a single parent-child relationship are clustered into one cluster. Hence, the dependency time of tasks reduces to a significant level.
- Minimum resource wastage: Algorithm ensures that the dependent tasks are provided with the data required as early as possible in order to avoid increase in waiting time and wastage of resources.

The rest of the paper is organized in the following way. The overview of related work is outlined in Section 2. Section 3 describes the workflow management system where the clustering techniques are applied. Section 4 describes the proposed algorithm. Section 5 reports the performance evaluation,results of proposed technique along with available basic clustering techniques. Section 5 ends the manuscript with conclusion and future scope.

2. Literature Review.

2.1. Load imbalance. Load balancing is a topic of significance in the area of distributed computing. To balance the computational load dynamically among different resources, transfer of some jobs is required from one resource to another in a period of time. This is called task reallocation [42]. A reallocation algorithm is proposed for tuning the parallel jobs submitted to the resource manager [42]. The batch scheduler sends submission and cancellation requests. The reallocation algorithm in a multi-cluster environment is presented. To dynamically migrate processes from overloaded computational nodes to less loaded nodes a premptive process migration method is proposed in [47]. However, it exhibits the limitation to maintain balance only with some idle nodes. In our case, we consider more tasks then available compute resources. To handle load imbalance, [15] presented techniques split tasks dynamically and consolidate them to fill idle compute nodes. Similarly, Ying et al. [46] present a load balancing algorithm based on collaborative task clustering. Level based autonomic Workflow-and-Platform Aware(WPA) task clustering technique [35] is proposed that considers the factors of workflow structure and underlying resource set size for task clustering. It aims to reduce the overhead in systems and wastage of resources.

In comparison to the techniques discussed above, our work merges the tasks based on their runtime distribution also considering the data dependencies. The overheads in our work are not introduced during runtime. Also, an approach to dynamically select the order of clustering whether vertical or horizontal is proposed.

2.2. Granularity. In scientific workflows, the techniques to control the granularity of tasks is also addressed. A label-based and level-based clustering approach is proposed by Singh et al. [37]. According to level-based clustering, considering the same horizontal level tasks are clustered. In it user specifies the number of tasks to be considered in a single cluster. In another approach of label-based clustering, the labeling of tasks is accomplished manually by the user. This method is more prone to error due to manual interaction. A task grouping and ungrouping algorithm are proposed, where information of application and resources is not known in advance [13]. Their work does not consider data dependencies but reduces queuing and scheduling overhead. An algorithm is proposed by Muthuvelu et al. [29] that group tasks based on their runtime to a resource capacity. Then, they proposed a technique [28] to determine the granularity of task based on CPU time, resource constraints and task file size. An online scheduling algorithm [27, 30] that cluster tasks based on application deadline, user's budget and resource network utilization was introduced. Ang et al.[2] and Ng et al.[21] aimed to increase the performance in the scheduling of tasks by introducing a factor of bandwidth. Further, Liu and Liao [24] proposed a technique for executing fine-grained jobs by grouping tasks considering the processing capacity of available resources and bandwidth. An approach for reusing and repurposing workflow is presented. It uses the metric of semantic similarity between layer hierarchies of a workflow. It adopted a graph skeleton based clustering technique for grouping layer hierarchies into clusters. This technique ranked the clusters. The similarity computation used is dependent on syntactic variations [49].

As there is a large number of processes involved in the execution of a scientific workflow. This may lead to

high level of failures. A general task failure model is proposed that uses maximum likelihood based estimation for improving the execution time performance of scientific workflows [7]. This framework fails to take advantage of the combination of horizontal and vertical clustering.

2.3. Structural similarity. In [22] SFLA technique is proposed for encoding of workflows through workflow representations by exploiting set descriptors. In [50] author proposes a method for reusing and repurposing of a workflow by calculating the semantic similarity between layers of different workflows.These hierarchies are grouped into clusters. The author proposed a SimiFlow architecture for supporting clustering of workflows based on similarity.

2.4. Data dependency. Although the proposed techniques significantly decrease the impact of queuing time overhead and scheduling, they did not consider the factor of data dependencies. As clustering, the tasks horizontally increases the problem of dependency imbalance and runtime imbalance. To overcome these problems, three new methods of clustering Horizontal Runtime balancing, Horizontal impact factor balancing and Horizontal Distance Balancing are proposed in[10]. In these algorithms, only horizontal clustering is performed. In a workflow, there can be tasks with single parent single child relationship. In these kinds of tasks, vertical clustering can prove to be more advantageous then horizontal clustering technique. A general task failure model is proposed that uses maximum likelihood based estimation for improving the execution time performance of scientific workflows [7]. This framework fails to take advantage of the combination of horizontal and vertical clustering considering the single parent and child relationship in the nodes of a workflow. The deciding factor is unexplainable in research so as to choose whether to cluster tasks vertically or horizontally so as to maintain parallelism.

The existing work suffers from one or the following drawbacks

- The data dependencies between tasks not considered
- The runtime imbalance and dependency imbalance not considered
- The structure of workflow not considered
- The maximum parallelism between tasks not exploited

This work proposes Hybrid balanced task clustering technique to perform clustering while maintaining the parallelism of the system considering the single parent single child relationships in the nodes of workflow. Hybridizations of horizontal and vertical clustering technique has been achieved using impact factor based clustering technique. The cluster size is dynamically set as per job runtime to maintain the parallelism. Dependency variance is calculated using distance metrics. Horizontal and vertical clustering is performed as per the available resources, so that the parallelism is not affected. Hybrid clustering improves the performance of scientific workflow in cloud computing and a further benefit to data placement.

3. System Architecture.

3.1. Workflow. The Workflow Management Coalition (WfMC) defined a workflow as the automation of a business process, in whole or part, during which documents, information or tasks are passed from one participant to another for action, according to a set of procedural rules [17]. By the definition given by WfMC, the work process is a progression of organized exercises and calculations of a business procedure which is a unique depiction of the undertakings to be executed in that business process. They are utilized to join a few diverse computational procedures into a solitary lucid procedure. The business applications can now be seen as perplexing work processes, which comprise of different changes performed on the information fancied in accomplishing the goal. Workflows offer awesome points of interest in isolating capacities from applications and in this manner offering data framework to be segmented based by first arranging them and then incorporating them.

WfMC introduces its reference model in recognizing the interfaces inside of this structure which empowers items to operate interactively at an assortment of levels. It characterizes a work management framework and the most critical interfaces of a system as in Figure 3.1.

- **Workflow Engine:** A software service that gives the run-time environment with a specific end term goal to make, oversee and execute work process cases.
- **Process Definition:** Specifies the information about the process and the workflows related to it.

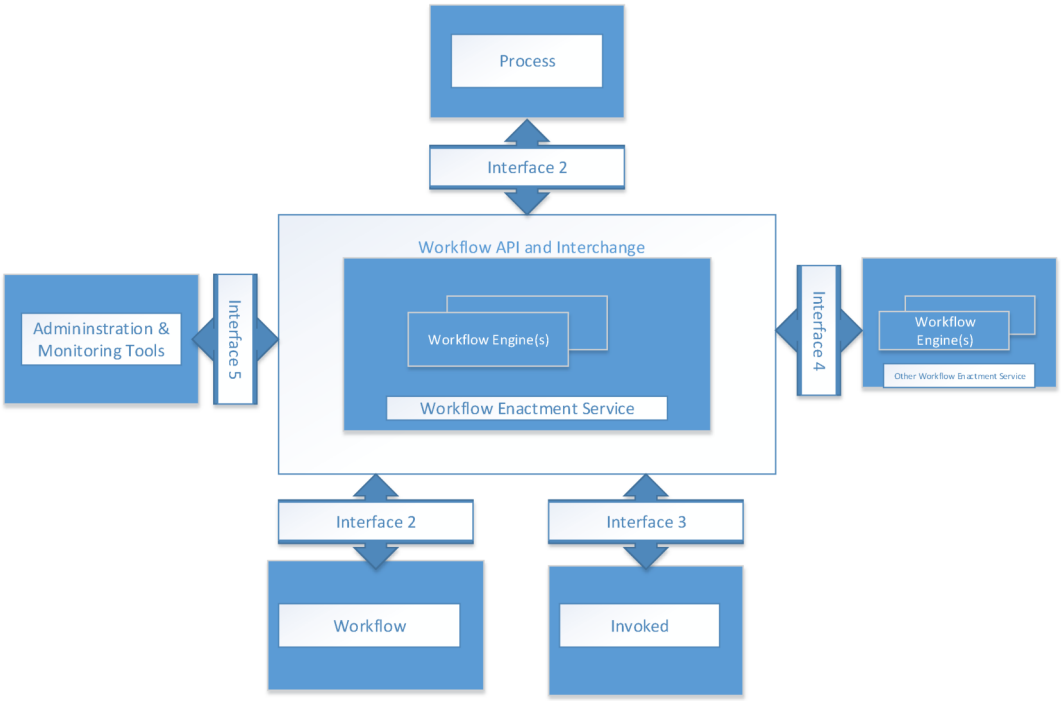

FIG. 3.1. *WfMC reference model [17]*

- **Workflow Interoperability:** This interface makes interoperability possible between different processes of a workflow.
- **Invoked Applications:** Interfaces to bolster cooperation with an assortment of IT applications.
- **Workflow Client Applications:** Interfaces to bolster connection with the client interface.
- **Administration and Monitoring:** Interfaces to give a framework observing and metric capacities to encourage the administration of composite work process application situations.

3.2. Workflow model. A workflow application $Wf=(V_i, E_i)$ is represented as a directed acyclic graph (DAG) where $V_i=v_{i1},v_{i2}..v_{in}$ is a set of vertices representing tasks and E_i represents edges control or data dependencies between them.A dependency e_{ij} is the precedence constraint of the form (v_{i1},v_{j1}) ,where $v_{i1},v_{j1} \in V_i$ and $v_{i1} \neq v_{j1}$.This refers to that the child task can only complete its execution if the parent task has completed the execution. An example workflow is shown in Figure 3.2.

3.3. Workflow execution environment. A workflow is submitted to a workflow management system for execution that resides on a submit hosts which is user interaction machine. The execution machine for a workflow can be a grid, physical cluster [12], a dedicated parallel system[26], a virtual environment such as the cloud[4] or it can also be a local machine. Figure 3.3 shows a typical execution environment for scientific workflows. The components of this environment are listed below:

Workflow Mapper responsible for the generation of an executable workflow on the basis of an abstract workflow as submitted by a composition system or a user of the workflow. It finds the appropriate computational resources and data required for execution of a workflow. For optimization of performance, Workflow Mapper also restructures the workflow and add transformation for provenance information generation and data management.

Workflow Engine is responsible for the execution of jobs as per dependencies defined by the workflow. It manages the jobs by tracking their status. The jobs whose parent jobs have completed are given to the Job Scheduler.

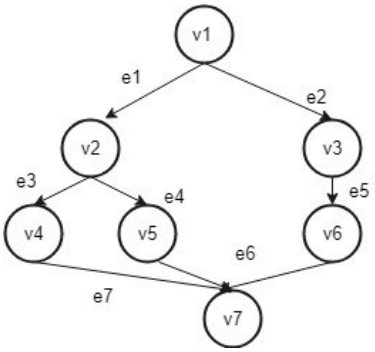

FIG. 3.2. *Workflow model*

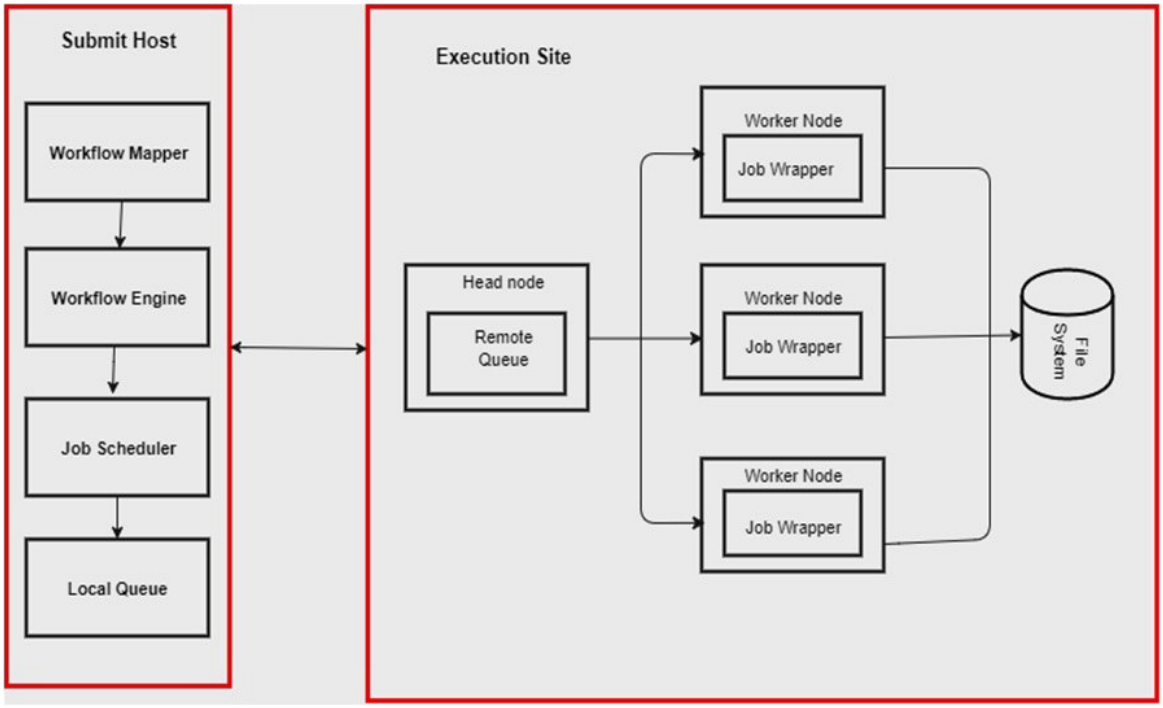

FIG. 3.3. *Workflow management system*

Local Queue and **Job Scheduler** manages workflow jobs individually and observer their execution on remote and local resources. Different scheduling algorithms such as HEFT[43] and MinMin[5] can be applied to the Job Scheduler and improve the overall runtime of workflow executions.

Job Wrapper unwraps tasks from clustered jobs. Then these are executed at the worker nodes. All of these components work cooperatively with each other to perform workflow preparation and execution.

4. Hybrid Balanced Task Clustering algorithm. In this, the proposed hybrid balanced clustering algorithm used for task clustering is discussed which is not dependent on the input of the user. It is able to cluster the tasks vertically with single parent single child relationship and the tasks horizontally as well. The system overhead is reduced while involving best utilization of resources. The flowchart of proposed technique is depicted in Figure 4.1. The symbols used in this work are explained in Table 3.1.

4.1. Background. The two major issues that are undertaken by the clustering algorithms, dependency imbalance, and runtime imbalance. Runtime imbalance refers to the unequal distribution of runtime of tasks at

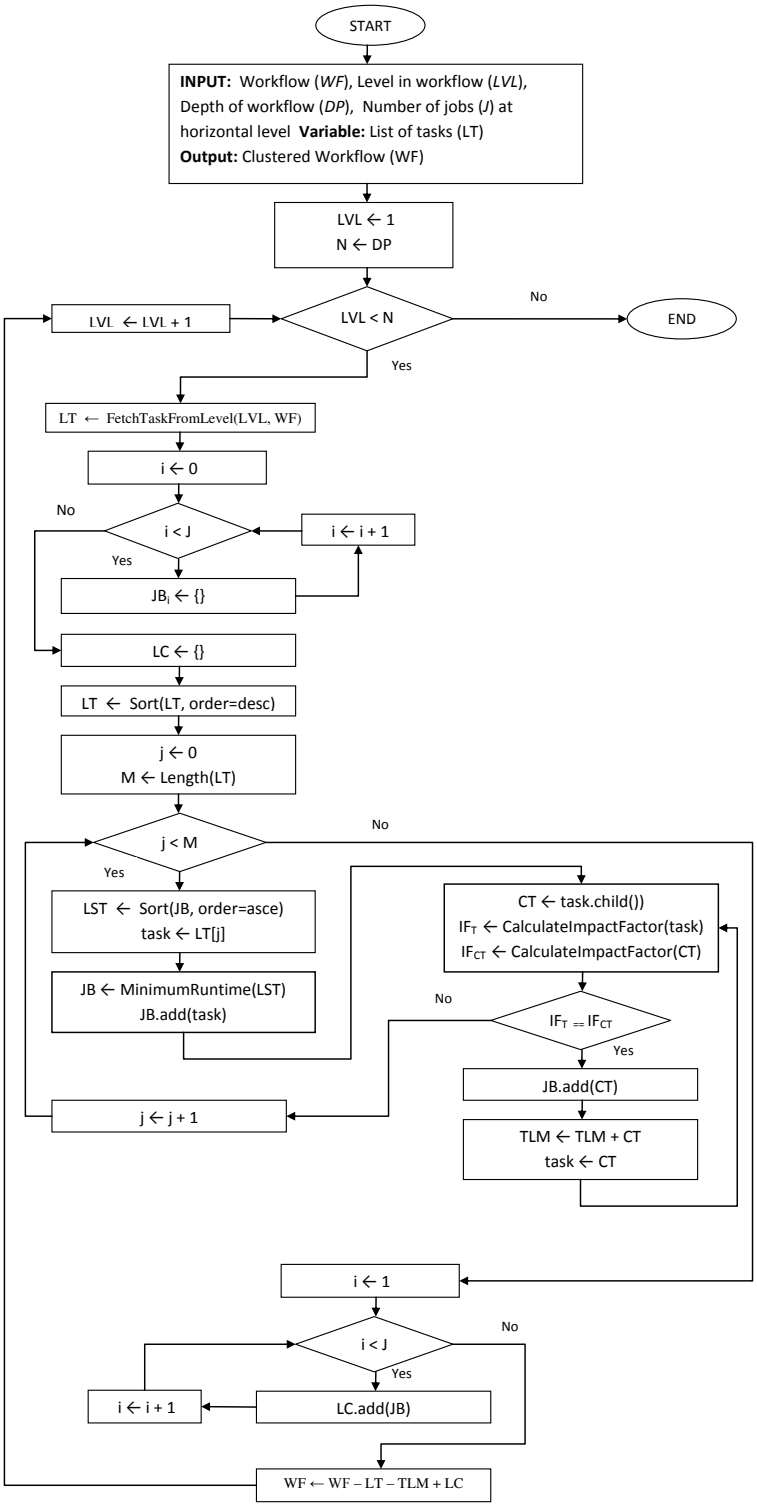

FIG. 4.1. *Flowchart of Hybrid Balanced Clustering Algorithm*

TABLE 3.1
Explanation of symbols used in this work

Abbreviation	Definition
WF	Workflow
LVL	Level in a workflow
J	Number of jobs at a horizontal level
LT	List of tasks at a level
DP	Depth of workflow
JB_i	Empty Job
LC	Empty list of clustered jobs
LST	Sorted List of jobs
CT	Child Task
IF	Impact Factor
TLM	Merged Task List

the same horizontal level. While dependency imbalance refers to clustering tasks at a level without considering the factor of dependency between tasks. This increases the waiting time for input at the next level and thus the delay in execution. The problem is also referred to as data locality. Generally, runtime imbalance leads to dependency imbalance and dependency imbalance leads to runtime imbalance. The structure of workflow is also an important factor while clustering the tasks. As horizontal clustering is always performed for tasks that may increase the problems of runtime imbalance and dependency imbalance. So instead of performing task clustering horizontally at each level of workflow, vertical clustering can also be performed of the tasks where single parent single child relationship exists between tasks besides performing clustering horizontally. This may lead to improvement in the execution of workflow while further decreasing the delays. Considering the above challenges of clustering, the introduced method aims to obtain appropriate hybrid clustering while merging the tasks vertically with the similar impact factor and remaining tasks horizontally according to number of available resources. Thus reducing the overall execution time. For the realization of the desired clustering, the proposed Hybrid balancing algorithm as follows.

Algorithm working: This technique prefers to cluster the tasks with single parent single child relationship. The problem of dependency imbalance is catered by measuring the impact factor of tasks in the workflow. The Impact Factor (IF) of task t_n is denoted by

$$IF(t_n) = \sum_{t_a \subset child(t_a)} \frac{IF(t_a)}{parent(t_a)} \tag{4.1}$$

where $child(t_a)$ denotes a set of child tasks of t_a and $parent(t_a)$ denotes set of parent tasks of t_a The impact factor is used to measure the similarity between the tasks or jobs[9].

The proposed algorithm ensures that the tasks with one parent-child relationship are clustered into one cluster. The remaining tasks are then clustered horizontally. The parent child relationship is depicted by matching the impact factor of tasks vertically. The tasks with the similar impact factor are grouped into one cluster. This help in reducing the execution time of workflow. For example figure 4.2 shows a five-level workflow composed of one task at level one, two tasks at level-two, four tasks at level-three, three tasks at level-four and one task at level-five.

Algorithm 1 shows the pseudo code of hybrid balancing algorithm that uses the combination of horizontal distance balancing [10], horizontal runtime balancing [10] and impact factor [10] of tasks. We assume the number of jobs as an input from the user. This algorithm addresses the problem of load imbalance and also considers the tasks with asymmetric structure. The algorithm begins with the first level of workflow as in Figure 4.2 by selecting tasks from each level (Lines 2-3). Tasks are clustered into a job and returned by merge procedure (Line 4).The merge procedure merges the tasks vertically and horizontally also. In merge procedure, the tasks are sorted according to decreasing order of runtime (Line 13). Considering the level three of a workflow. There are four tasks D(30s),E(10s),F(30s),G(20s). The task list LT is maintained as according to decreasing order

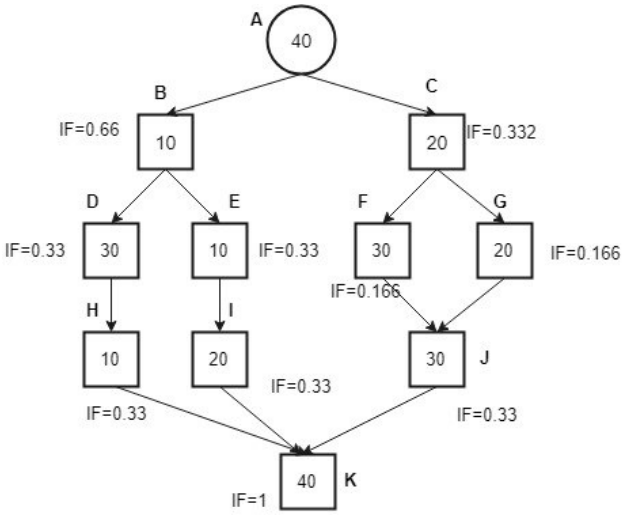

FIG. 4.2. *Example workflow*

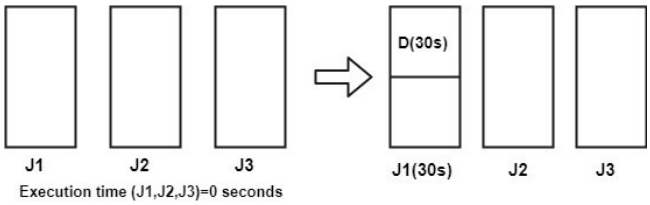

FIG. 4.3. *Merging of clusters in a job*

of runtime of tasks LT=D, F, G, E. Then horizontal runtime balancing adds the task D to the job with the shortest runtime as in Figure 4.3.

Step 2: The tasks are arranged as per the minimum distance with the task D. $Distance$=$0(D), 4(F), 4(G), 2(E)$. The impact factor of task D included in a job is matched vertically to depict parent-child relationship exists or not. If the relationship exists then the child task is added to the job where task D resides (Lines 20-22) as shown in Figure 4.4 and Figure 4.5.

Step 3: Now after the clustering of tasks D and H. The tasks in a task list(LT) are task F,G,E. The horizontal distance balancing is performed on tasks E, F and G. In this jobs J1, J2, J3 are sorted based on shortest distance between them and targeted task D. The distances are 4,4,2 respectively for tasks F, G, E. The candidate is selected with minimal distance 2 of task E. Hence impact factor of task E is matched vertically and if the impact factor matches with the below vertically then both are clustered into another cluster c2 as shown in Figure 4.6 and merged as a job J2 as shown in Figure 4.7.

Step 4: Now the tasks left in the task list LT are F and G. Similarly the process is repeated for the tasks F and G. But these tasks do not have a parent child relationship with any task. So they cannot be clustered vertically. Hence Horizontal clustering is performed for the tasks F, G forming cluster c3 as shown in Figure 4.8 and merging into job J3 as shown in Figure 4.9.

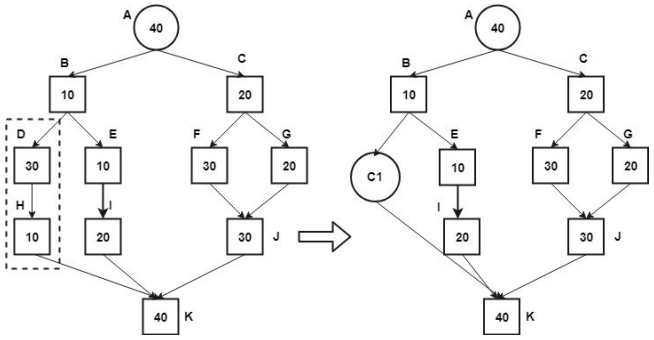

FIG. 4.4. *Clustering to avoid runtime imbalance and dependency imbalance*

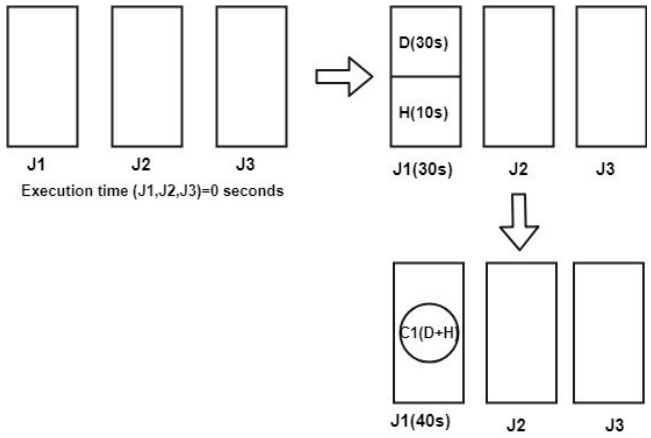

FIG. 4.5. *Merging clusters into a job*

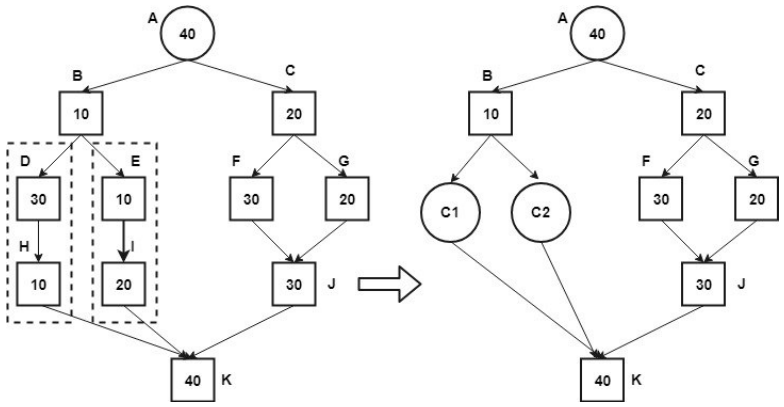

FIG. 4.6. *Clustering on runtime and dependencies*

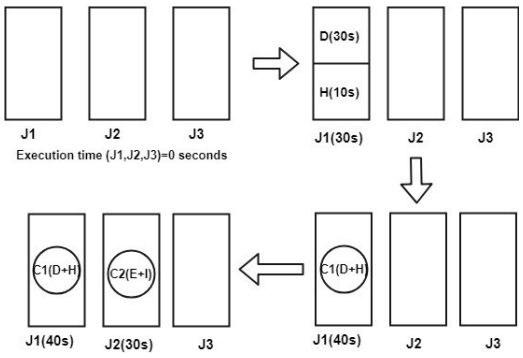

FIG. 4.7. *Merging clusters into a job*

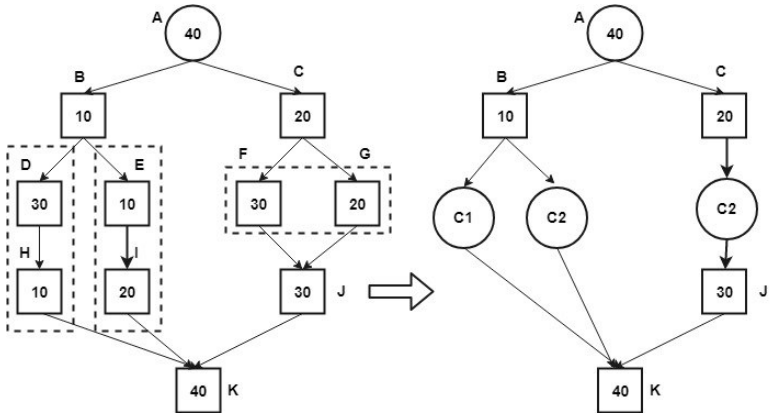

FIG. 4.8. *Clustering to avoid runtime imbalance and dependency imbalance*

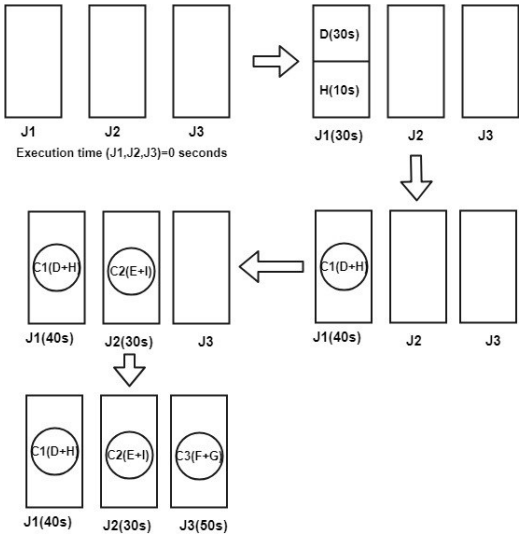

FIG. 4.9. *Merging clusters into a job*

4.2. Pseudocode for Hybrid Balanced clustering algorithm (HYB). Algorithm 1 presents it.

Algorithm 1 Hybrid Balanced Clustering algorithm.

Require: WF: workflow;J: number of jobs at horizontal level ;LVL:Level of workflow;DP:depth of workflow
procedure HYCLUSTERING(Wf)
  for $LVL < DP(WF)$ **do**
    $LT \leftarrow$ FETCHTASKSFROMLEVEL(WF, LVL) ▷ Divide WF on the basis of depth
    TLM,$LC \leftarrow$ MERGE(LT, J) ▷ Group of clustered job returned
    $WF \leftarrow WF - LT + LC - TLM$ ▷ Dependency merging
  end for
end procedure
procedure MERGE(LT, J)
  for $i < J$ **do**
    $JB_i \leftarrow\{\}$ ▷ NULL JOB
  end for
  $LC \leftarrow\{\}$ ▷ NULL JOB LIST
  Sort LT in descending of runtime
  for all $task$ in LT **do**
    $LST \leftarrow$ sort list of JB_i as per least distance with $task$
    $JB \leftarrow$ the job with minimum runtime in LST
    JB.add ($task$)
    while $IF_T = IF_{CT}$ **do** ▷ CT is child task
      JB.add(CT)
      $TLM \leftarrow TLM + CT$
      $task \leftarrow CT$
    end while
  end for
  for $i < J$ **do**
    LC.add(JB)
  end for
  return LC,TLM
end procedure

5. Performance Evaluation.

5.1. Scientific workflow applications. Montage [3] is one of the application of astronomy used for constructing large image mosaics of the sky. In it, the input images are projected onto a sphere and then the overlap for each input image is calculated. The input images are projected to the accurate orientation while maintaining the constant emission level of background for all images. At last, reprojected images are added into a final mosaic. The final resulting image provides a detailed description of the part of the sky under investigation. Figure 5.1 represents the Montage workflow. The size of the workflow is dependent upon the number of images used for constructing the desired mosaic of the sky.

Cybershake. CyberShake [14] is an application of seismology that is used for calculating Probabilistic Seismic Hazard curves for various geographic regions in Southern California. It is used in the identification of all ruptures within 200KM radius. It also changes rupture into multiple variations of rupture that differ in hypocenter locations and slip distributions. Then for each rupture variance, synthetic seismograms are calculated. After that, the peak intensity is extracted and added with the original rupture probability for production probabilistic hazards for the location. Figure 5.2 shows an illustration of the Cybershake workflow.

Epigenomics. The Epigenomics workflow [44] is a CPU-intensive application. Initially the data is obtained in the form of DNA sequence lanes from the Illumina-Solexa Genetic Analyzer. Every Solexa machine generates multiple DNA sequences. Then the mapping of DNA sequences to the accurate locations in a genome is performed by the workflow. This produces a map showing the density of sequence. A simplified structure of Epigenomics is shown in Figure 5.3.

SIPHT. The SIPHT workflow [19] is responsible for researching small untranslated RNAs (sRNAs). It is responsible for regulation of different processes such as virulence or secretion in bacteria. This predict ρ-independent transcriptional terminators. A simplified version of SIPHT workflow is depicted in Figure 5.4.

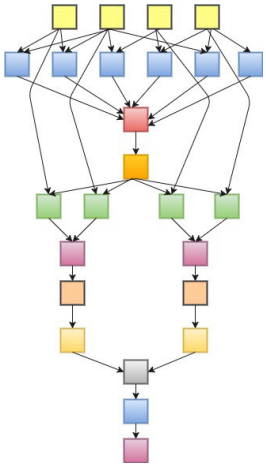

FIG. 5.1. *A simplified visualization of the Montage workflow[3].*

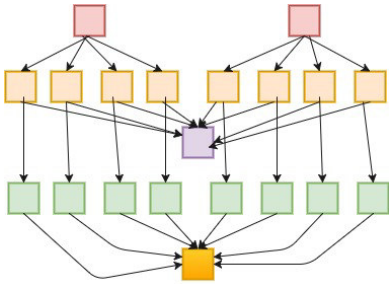

FIG. 5.2. *A simplified visualization of the CyberShake workflow [14].*

LIGO. In Laser Interferometer Gravitational Wave Observatory (LIGO)[1] workflows data is collected by large-scale interferometers for searching of gravitational wave signatures. The observers aim is to measure and detect waves as predicted by relativity. It is a data-intensive workflow. Figure 5.5 depict a version of workflow. In this workflow tasks are separated into different groups where a group consisting of interconnected tasks as shown in Figure 5.5.

5.2. Balanced Task Clustering algorithms. The task clustering techniques are classified into two different categories Horizontal clustering and vertical clustering. Further to overcome the limitations of these techniques, balanced clustering is introduced.

5.3. Description of Baseline Balanced Clustering Algorithms. In order to study the impact of the proposed clustering technique, the clustering methods considered are: Horizontal Impact Factor Balancing (HIFB) Horizontal Clustering (HC), Horizontal Distance Balancing (HDB) and Horizontal Runtime balancing (HRB) [8, 10].

Horizontal Clustering (HC): In this technique tasks at the same horizontal level of workflow is merged into a job. The horizontal level for a single task is considered as the largest distance from the first task of workflow starting from the first task. The first task is a task without a parent task.

As shown in Figure 5.6, the tasks t2, t3 and t4 are combined together into a cluster c1, thus creating just one job j1. The horizontal clustering is performed for the tasks at the same level. Thus reducing the overhead.

Vertical Clustering Clustering (VC): In this algorithm task at the same vertical level of a workflow are merged and make a single job. In this, the tasks with relationship between one parent and one child are merged.

As shown in Figure 5.7 the tasks t2, t5, t8 exhibit parent-child relationship as t2 is parent of t5 and further

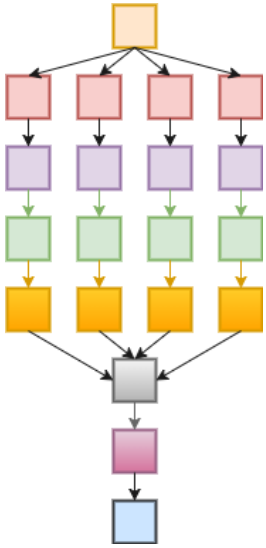

FIG. 5.3. *Epigenomics workflow with multiple branches [44].*

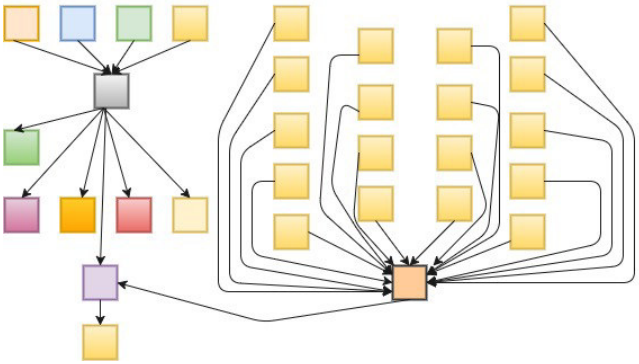

FIG. 5.4. *A simplified visualization of the SIPHT workflow [19].*

t5 is parent t8. So these tasks are clustered together into a cluster c1, thus creating a job1. Similarly t3, t6, t9 and t4, t7, t10 are clustered into clusters c2,c3 thus creating combined jobs job2, job3 respectively. The overheads are also combined into overheads o2, o3 and o4.

Horizontal Runtime Balancing (HRB): In this algorithm runtime of tasks is equally distributed between jobs. The problem of runtime variance is addressed at the same horizontal level. In this greedy method, the jobs are sorted according to the ascending order of runtime. The new task is joined to the job with a minimum runtime. This method further raises the dependency imbalance among tasks as the factor of data dependency is not considered while clustering.

As shown in Figure 5.8, there are four tasks t1, t2 and t3, t4 with runtime 20s and 30s respectively. According to this clustering method tasks t1, t3 are combined into one cluster and t2, t4 are combined into another cluster. This balances the runtime among tasks, but leads to dependency imbalance among tasks.

Horizontal Impact Factor Balancing (HIFB): It overcomes the limitation of Horizontal runtime balancing algorithm of dependency imbalance. In this algorithm, the jobs are first sorted by considering the similarity of impact factor (IF) in increasing order. Then the shortest job is selected using Horizontal Runtime Balancing. It groups the jobs sharing the same position in the workflow.

As shown in Figure 5.9 the tasks t1, t2 and t3, t4, t5 have same the impact factor. Then using HRB the tasks t1, t2 and t3, t4 are combined into clusters c1, c2.

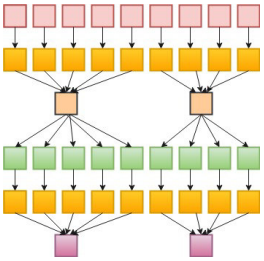

FIG. 5.5. *A simplified visualization of the LIGO Inspiral workflow [1].*

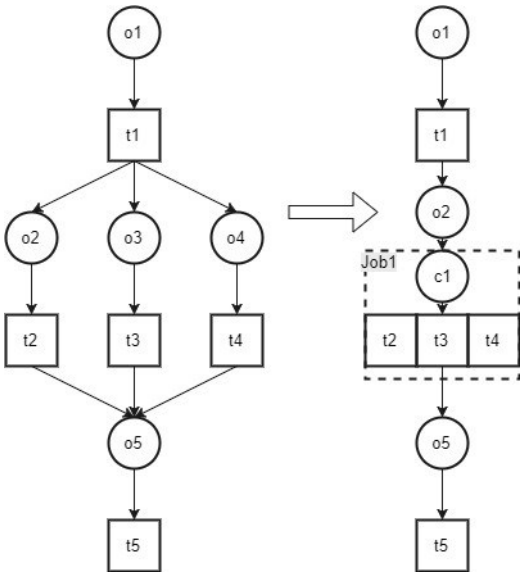

FIG. 5.6. *Horizontal Clustering.*

Horizontal Distance Balancing (HDB): In this clustering technique jobs are sorted considering the distance between them and the target job. After that Horizontal Runtime Balancing method is executed for selecting the shortest job. The tasks with minimum distance are merged thus reducing data transfer.

As in Figure 5.10 the tasks t1 and t2 are closest as compared to distance between t1, t3 and t1, t4. So t1, t2 are combined into one cluster and t3, t4 are combined into another cluster.

The performance of above discussed algorithms has been evaluated using workflowsim on various datasets e.g. Montage, LIGO.

5.4. Experiments and Results.

Experimental setup. The trace based simulation method is adopted for evaluation. The performance of the proposed technique is compared with the baseline algorithm. Real traces are used for evaluation of algorithms. In the environment, different system parameters used are a number of virtual machines, system overhead, different workflows and sizes of data.

For executing applications of workflow, open source workflow simulator, workflowsim is used. It is used for modeling an execution environment in the cloud. It is an extension of cloudsim. The DAG model of workflow is executed in it. It performs clustering and scheduling of tasks. It also performs provisioning of resources at the workflow level.

In the experiment, the simulator WorkflowSim is extended to implement the proposed hybrid balanced task clustering technique. A virtual machine cluster consisting of 20 single homogeneous core VMs is considered. This cluster is a quota for a user in some distributed environments such as Amazon EC2. Each VM detains

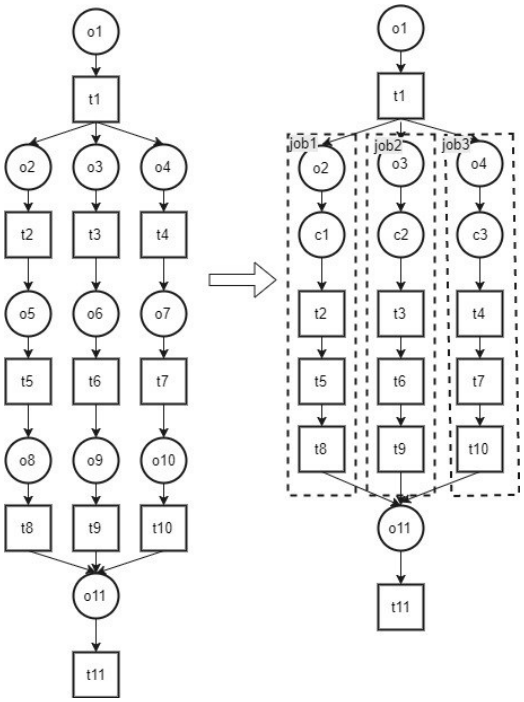

FIG. 5.7. *Vertical clustering.*

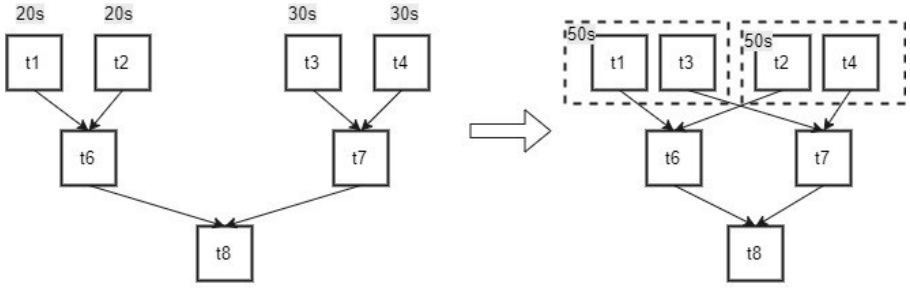

FIG. 5.8. *A sample of the HRB (Horizontal Runtime Balancing) method.*

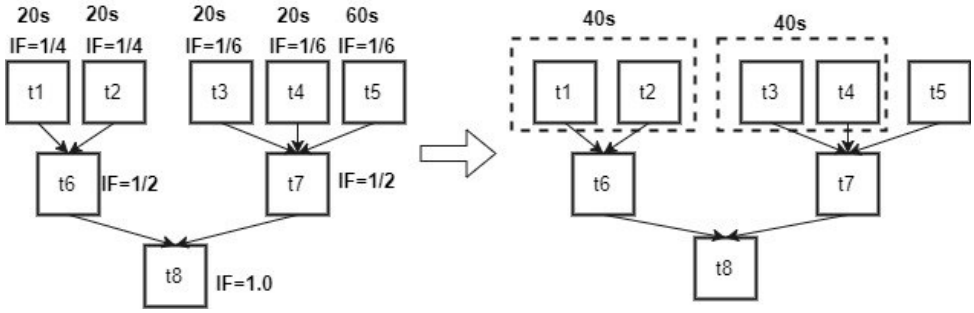

FIG. 5.9. *HIFB (Horizontal Impact Factor Balancing) method.*

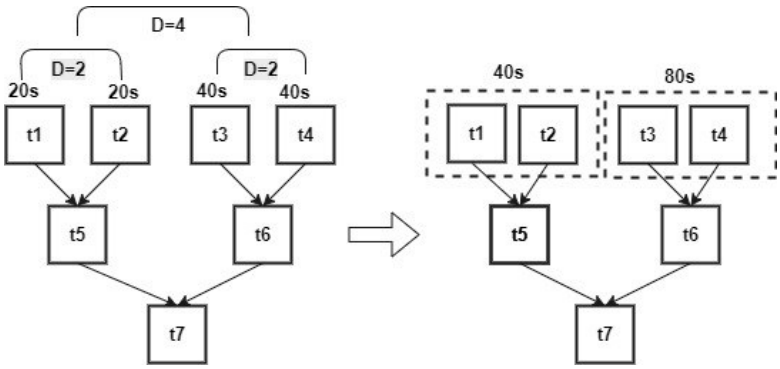

FIG. 5.10. *HDB (Horizontal Distance Balancing) method.*

TABLE 5.1
Configuration setup for the experiment

Parameter	**size**
No. of virtual machines	20
Memory Capacity	512 MB
Processing capacity	1000 MIPS
Network Bandwidth	15 MB/s

configuration of 512 MB. The processing capacity for each virtual machine is 1000 MIPS and by default, the network bandwidth was set to 15 MB/s.

The configuration setup for the experiment is described as in Table 5.1.

An evaluation of the proposed technique was performed for identification of its ability. The method was evaluated for the improvement in execution time of scientific workflow and reducing the load to minimum resources. The experiment is conducted on the following variables.

- **Execution time of workflow :** Total time taken by the workflow application to execute on the available resources.
- **Performance Gain (μ):** It is defined as an overall improvement in execution time of workflow by using the clustering algorithm over the execution of the workflow without clustering of a task. It can be evaluated using equation as follows:
  μ = (time taken without clustering-time taken with clustering)/(time taken without clustering)*100
  $\mu > 0$ for a clustering technique signifies that it leads to improvement of the execution time of a workflow . Whereas $\mu < 0$ refers to negative impact of clustering method. This negative impact lead to increase of execution time of worklfow.

Different scientific workflow applications are used in the experiments along with the fixed number of tasks for each as shown in Table 5.2.

5.5. Result and Analysis. Table 5.2 depicts the scientific workflow applications used for the experiment and the number of tasks considered for each.

Then the makespan time for each scientific application is calculated as depicted in Table 5.3.

Using the makespan time, performance gain of scientific applications cybershake, epigenomics, LIGO, Montage and SIPHT is calculated and compared as evaluated according to different baseline algorithms and proposed algorithm in Table 5.4.

Three set of experiments are conducted on the scientific workflow applications.

Experiment 1: Improvement in execution time The makespan time of balanced clustering algorithms on different scientific workflow applications cybershake, epigenomics, LIGO, Montage, SIPHT is obtained from the experiments performed. The makespan time is depicted in Table 5.3. According to the experimental evaluations the following results are depicted:

TABLE 5.2
Scientific workflow for experiment and number of tasks

Workflow	Number of tasks
Cybershake	1000
Epigenomics	997
Montage	1000
SIPHT	1000
LIGO	1000

TABLE 5.3
Makespan time of workflow with or without clustering algorithms

	HC	HRB	HIFB	HDB	HYB	without clustering
Cybershake	1511.46	1303.58	1833.68	1445.07	1352.92	2222.05
Epigenomics	238974.2	218583.8	241833.1	238754	206077.6	253462
LIGO	13188.79	12768.49	15261.21	13015.6	11635.49	17234.23
Montage	1210	924.71	936	947	923.61	1927.69
SIPHT	21154.99	9537.03	22778.28	18539.23	9499.07	23453.58

- **Cybershake Workflow Application**: In this application, there is 26% and 10% improvement in execution time by hybrid balanced clustering algorithm(HYB) then horizontal impact factor balancing(HIFB) and horizontal clustering(HC) respectively.
- **Epigenomics workflow application** : In this application, there is 14.78%, 13.68%, 13.76% improvement in execution time by Hybrid Balanced Clustering algorithm then horizontal impact factor balancing(HIFB), Horizontal Distance Balancing(HDB) and Horizontal Clustering(HC) respectively.
- **LIGO**: In this application there is 23.75%, 11.77% , 10.60% and 8.87% improvement in execution time by Hybrid Balanced Clustering algorithm then Horizontal impact factor balancing(HIFB), Horizontal clustering(HC), Horizontal Distance balancing(HDB) and Horizontal runtime balancing(HRB) respectively.
- **Montage Workflow Application** : In this application there is 23%, 2.46% improvement in execution time of workflow by Hybrid Balanced Clustering algorithm then Horizontal Clustering (HC) and Horizontal Distance Balancing respectively.
- **SIPHT** : In this application there is 58%,55% and 48% improvement in execution time by Hybrid Balanced Clustering algorithm then Horizontal Impact Factor Balancing(HIFB),Horizontal Clustering(HC) and Horizontal Distance Balancing(HDB) respectively.

Hence from the experiment 1, it is concluded that the execution time taken by Hybrid Balanced Task Clustering algorithm(HYB) is lesser as compared to other baseline algorithms. Proposed technique shows the overall improvement in execution time.

Experiment 2: Performance Gain In this experiment, the performance gain (μ) of the proposed technique is evaluated with the other clustering algorithms. In this experiment, the proposed technique is evaluated for identification of the amount to which it impacts the overall execution time of workflow.Figure 5.4 shows the performance gain μ for different clustering methods obtained by the performed experiment. From the experiments, it is depicted that all the methods of clustering retain a positive gain performance. This further improves the execution time of different workflow applications. According to obtained results Montage, CyberShake and SIPHT workflows have better performance gains with a minimum gain of 52%,41% and 59%. In comparison LIGO and Epigenomics having a minimum performance gain of 32% and 18% respectively. The variation in performance gain is due to the granularity of the average runtime of tasks in workflows. Considering the workflows the lowest value for the performance gain is for HIFB in SIPHT workflow. In all the scientific workflows, the technique Horizontal Runtime Balancing(HRB) and proposed Hybrid balanced Task clustering(HYB) method performs better than the other balancing techniques. The clustering method HDB(Horizontal Distance Balancing) and HIFB(Horizontal Impact Factor Balancing) lack in performance, because there runtime as well as

TABLE 5.4
Performance gain for various workflows

	Cybershake	**Epigenomics**	**LIGO**	**Montage**	**SIPHT**
HC	31.97903	5.715961	23.47329	37.23057	9.800593
HRB	39.11388	13.76072	25.91204	52.03015	59.33649
HIFB	17.478	4.588013	11.44826	51.44447	2.879305
HDB	34.96681	5.802842	24.4782	50.87384	20.95352
HYB	41.33435	18.69489	32.48616	52.08721	59.49842

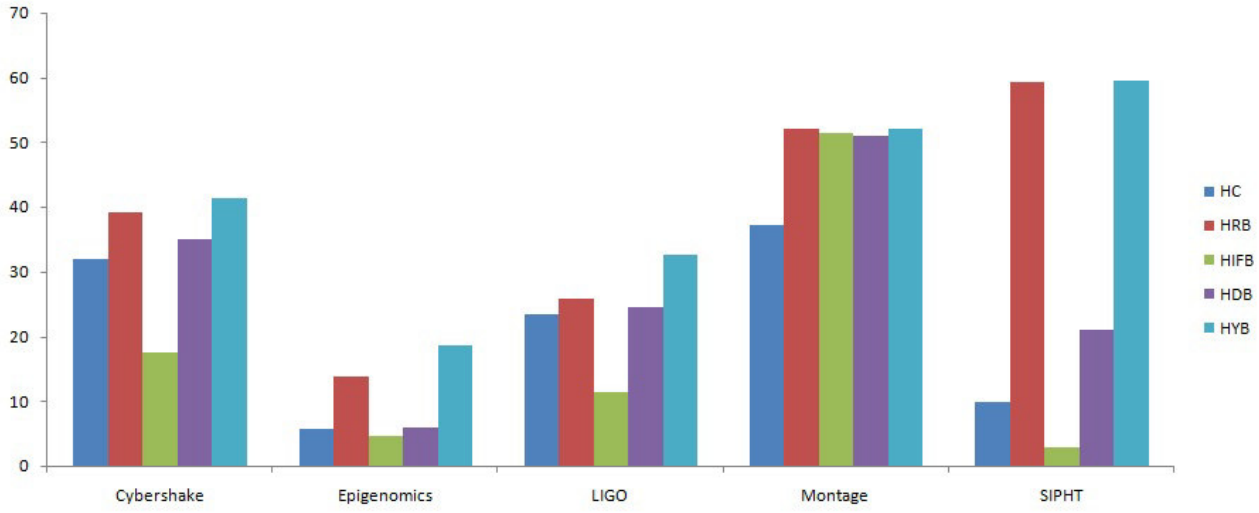

FIG. 5.11. *Comparison of performance Gain (μ) for experiment for various clustering methods.*

dependency imbalance between tasks. This groups the tasks into a cluster that should execute in parallel leading to an increase in execution time of workflow. The proposed hybrid balancing clustering method delivers a good performance since it first checks for parent-child relationship and then clusters the tasks. Thus increasing the performance gain and decreasing overall execution time of workflow as shown in Figure 5.11.

Experiment 3: Varying virtual machines In experiment 3, the number of virtual machines are varied while keeping the total number of tasks 1000 MIPS.

First, the experiment is performed by using a horizontal clustering technique while varying the number of virtual machines (VM) as shown in Figure 5.12.

The same experiment is conducted by using hybrid balanced clustering algorithm and the result is depicted as shown in Figure 5.13.

While comparing both the results, it is found that horizontal clustering technique performs in a similar manner inspite of more number virtual machines available. While in the proposed algorithm, the execution time decreases as the number of virtual machines increases. Hence leading to an increase in performance of execution of workflow applications. But at an instance the proposed technique performs as similar to baseline algorithms. This depicts the proposed technique also reaches a stagnation point but after more of the resource utilization in comparison to baseline algorithms.

As in Figure 5.12 the graph depicts the number of virtual machines increases the execution time decreases but at an instant the performance is constant and it does not matter on adding more virtual machines. The proposed technique is performing in a similar manner and adding up the virtual machines decreases the execution time. But at a point the performance becomes constant.

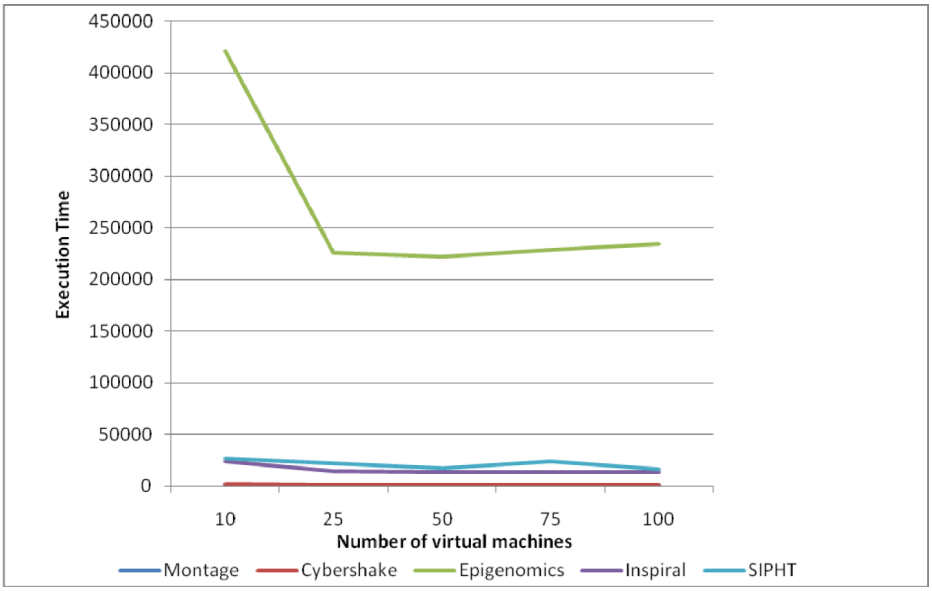

FIG. 5.12. *Execution time while varying virtual machines in Horizontal Clustering technique.*

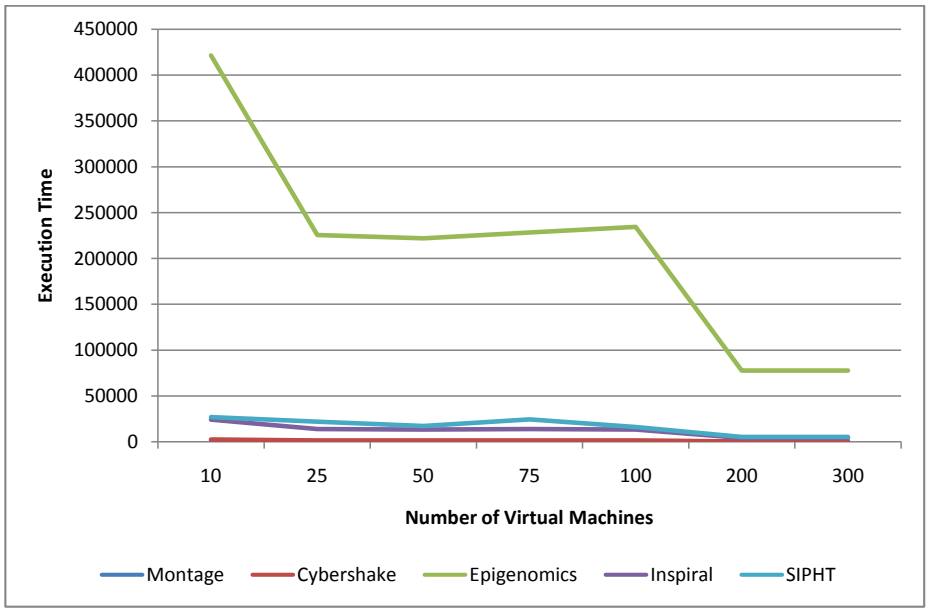

FIG. 5.13. *Execution time while varying virtual machines in Hybrid Balanced Clustering technique.*

6. Conclusion and Future Scope. The work proposes a hybrid balancing (HYB) task clustering method that reduces the full time of workflow execution and thus avoids waste of resources. The proposed approach considers distance variance and merges the tasks with similar impact factor in pipeline. A simulation experiment is conducted for evaluation of the proposed algorithm in comparison to four clustering methods: Horizontal Runtime Balancing (HRB), Horizontal Clustering (HC), Horizontal Distance Balancing (HDB) ,and Horizontal Impact Factor Balancing (HIFB). The experiment aimed to evaluate the execution time improvement. The results depict that the proposed clustering method is able to perform better than the other methods. Hence it can be used in different workflow structures.

In near future, it is intended that the size of the resource set can be identified for optimized execution of a workflow. Also, the proposed approach can be implemented in a workflow engine. So that this approach can be used for clustering tasks in a real environment.

REFERENCES

[1] Abramovici, A., Althouse, W. E., Drever, R. W., Gürsel, Y., Kawamura, S., Raab, F. J., Shoemaker, D., Sievers, L., Spero, R. E., Thorne, K. S., et al.(1992). Ligo: The laser interferometer gravitational-wave observatory. *Science*, 256(5055):325–333.

[2] Ang, T., Ng, W., Ling, T., Por, L., and Liew, C.(2009). A bandwidth-aware job grouping-based scheduling on grid environment. *Information Technology Journal*, 8(3):372–377.

[3] Berriman, G. B., Deelman, E., Good, J. C., Jacob, J. C., Katz, D. S., Kesselman, C., Laity, A. C., Prince, T. A., Singh, G., and Su, M.(2004). Montage: a grid-enabled engine for delivering custom science-grade mosaics on demand. In *SPIE Conference on Astronomical Telescopes and Instrumentation*.

[4] Berriman, G. B., Juve, G., Deelman, E., Regelson, M., and Plavchan, P.(2010). The application of cloud computing to astronomy: A study of cost and performance. In *Workshop on e-Science challenges in Astronomy and Astrophysics*.

[5] Blythe, J., Jain, S., Deelman, E., Gil, Y., Vahi, K., Mandal, A., and Kennedy, K.(2005). Task scheduling strategies for workflow-based applications in grids. In *5th IEEE International Symposium on Cluster Computing and the Grid (CCGrid '05)*.

[6] Callaghan, S., Maechling, P., Small, P., Milner, K., Juve, G., Jordan, T., Deelman, E., Mehta, G., Vahi, K., Gunter, D., Beattie, K., and Brooks, C. X.(2011). Metrics for heterogeneous scientific workflows: A case study of an earthquake science application. *International Journal of High Performance Computing Applications*, 25(3):274–285.

[7] Chen, W., Da Silva, R. F., Deelman, E., and Fahringer, T.(2016). Dynamic and fault-tolerant clustering for scientific workflows. *IEEE Transactions on Cloud Computing*, 4(1):49–62.

[8] Chen, W., Da Silva, R. F., Deelman, E., and Sakellariou, R.(2013a). Balanced task clustering in scientific workflows. In *2013 IEEE 9th International Conference on e-Science*, pages 188–195. IEEE.

[9] Chen, W., Deelman, E., and Sakellariou, R.(2013b). Imbalance optimization in scientific workflows. In *Proceedings of the 27th international ACM conference on International conference on supercomputing*, ICS '13, pages 461–462.

[10] Chen, W., Ferreira Da Silva, R., Deelman, E., and Sakellariou, R.(2015). Using imbalance metrics to optimize task clustering in scientific workflow executions. *Future Generation Computer Systems*, 46:69–84.

[11] Deelman, E., Kesselman, C., Mehta, G., Meshkat, L., Pearlman, L., Blackburn, K., Ehrens, P., Lazzarini, A., Williams, R., and Koranda, S.(2002). GriPhyN and LIGO: building a virtual data grid for gravitational wave scientists. In *11th IEEE International Symposium on High Performance Distributed Computing (HPDC '02)*.

[12] Fahringer, T., Prodan, R., Duan, R., Hofer, J., Nadeem, F., Nerieri, F., Podlipnig, S., Qin, J., Siddiqui, M., Truong, H.-L., et al.(2007). Askalon: A development and grid computing environment for scientific workflows. In *Workflows for e-Science*, pages 450–471. Springer.

[13] Ferreira da Silva, R., Glatard, T., and Desprez, F.(2013). On-line, non-clairvoyant optimization of workflow activity granularity on grids. In *Euro-Par 2013 Parallel Processing*, volume 8097 of *Lecture Notes in Computer Science*, pages 255–266. Springer Berlin Heidelberg.

[14] Graves, R., Jordan, T., Callaghan, S., Deelman, E., Field, E., Juve, G., Kesselman, C., Maechling, P., Mehta, G., Milner, K., Okaya, D., Small, P., and Vahi, K.(2010). CyberShake: A Physics-Based Seismic Hazard Model for Southern California. *Pure and Applied Geophysics*, 168(3-4):367–381.

[15] Guo, Z., Pierce, M., Fox, G., and Zhou, M.(2011). Automatic task re-organization in mapreduce. In *Cluster Computing (CLUSTER), 2011 IEEE International Conference on*, pages 335–343.

[16] Hey, T., Tansley, S., Tolle, K. M., et al.(2009). *The fourth paradigm: data-intensive scientific discovery*, volume 1. Microsoft research Redmond, WA.

[17] Hollingsworth, D. and Hampshire, U.(1995). Workflow management coalition: The workflow reference model. *Document Number TC00-1003*, 19.

[18] Hussin, M., Lee, Y. C., and Zomaya, A. Y.(2010). Dynamic job-clustering with different computing priorities for computational resource allocation. In *The 10th IEEE/ACM International Conference on Cluster, Cloud and Grid Computing*.

[19] Juve, G., Chervenak, A., Deelman, E., Bharathi, S., Mehta, G., and Vahi, K.(2013). Characterizing and profiling scientific workflows. *Future Generation Computer Systems*, 29(3):682–692.

[20] Kalayci, S., Dasgupta, G., Fong, L., Ezenwoye, O., , and Sadjadi, S.(2010). Distributed and adaptive execution of

condor dagman workflows. In *Proceedings of the 22nd International Conference on Software Engineering and Knowledge Engineering (SEKE'2010)*.

[21] Keat, N. W., Fong, A. T., Chaw, L. T., and Sun, L. C.(2006). Scheduling framework for bandwidth-aware job grouping-based scheduling in grid computing. *Malaysian Journal of Computer Science*, 19(2):117–126.

[22] Koohi-Var, T. and Zahedi, M.(2017). Scientific workflow clustering based on motif discovery. *International Journal of Computer Science, Engineering and Information Technology (IJCSEIT)*, 7.

[23] Lathers, A., Su, M., Kulungowski, A., Lin, A., Mehta, G., Peltier, S., Deelman, E., and Ellisman, M.(2006). Enabling parallel scientific applications with workflow tools. In *Challenges of Large Applications in Distributed Environments (CLADE 2006)*.

[24] Liu, Q. and Liao, Y.(2009). Grouping-based fine-grained job scheduling in grid computing. In *First International Workshop on Education Technology and Computer Science*.

[25] Maechling, P., Deelman, E., Zhao, L., Graves, R., Mehta, G., Gupta, N., Mehringer, J., Kesselman, C., Callaghan, S., Okaya, D., Francoeur, H., Gupta, V., Cui, Y., Vahi, K., Jordan, T., and Field, E.(2007). SCEC CyberShake WorkflowsAutomating probabilistic seismic hazard analysis calculations. In Taylor, I., Deelman, E., Gannon, D., and Shields, M., editors, *Workflows for e-Science*, pages 143–163. Springer.

[26] Mats, R., Juve, G., Vahi, K., Callaghan, S., Mehta, G., and andEwa Deelman, P. J. M.(2012). Enabling large-scale scientific workflows on petascale resources using mpi master/worker. In *Proceedings of the 1st conference of the Extreme Science and Engineering Discovery Environment*.

[27] Muthuvelu, N., Chai, I., Chikkannan, E., and Buyya, R.(2010). On-line task granularity adaptation for dynamic grid applications. In *International Conference on Algorithms and Architectures for Parallel Processing*, pages 266–277. Springer.

[28] Muthuvelu, N., Chai, I., and Eswaran, C.(2008). An adaptive and parameterized job grouping algorithm for scheduling grid jobs. In *10th International Conference on Advanced Communication Technology (ICACT 2008)*, volume 2, pages 975 –980.

[29] Muthuvelu, N., Liu, J., Soe, N. L., Venugopal, S., Sulistio, A., and Buyya, R.(2005). A dynamic job grouping-based scheduling for deploying applications with fine-grained tasks on global grids. In *Proceedings of the 2005 Australasian workshop on Grid computing and e-research*.

[30] Muthuvelu, N., Vecchiola, C., Chai, I., Chikkannan, E., and Buyya, R.(2013). Task granularity policies for deploying bag-of-task applications on global grids. *Future Generation Computer Systems*, 29(1):170 – 181. ¡ce:title¿Including Special section: AIRCC-NetCoM 2009 and Special section: Clouds and Service-Oriented Architectures¡/ce:title¿.

[31] Nayyar, A.(2011a). Interoperability of cloud computing with web services. *International Journal of ElectroComputational World & Knowledge Interface*, 1(1).

[32] Nayyar, A.(2011b). Private virtual infrastructure (pvi) model for cloud computing. *International Journal of Software Engineering Research and Practices*, 1(1):10–14.

[33] Oinn, T., Addis, M., Ferris, J., Marvin, D., Senger, M., Greenwood, M., Carver, T., Glover, K., Pocock, M. R., Wipat, A., and Li, P.(2004). Taverna: a tool for the composition and enactment of bioinformatics workflows. *Bioinformatics*, 20(17):3045–3054.

[34] Ostermann, S., Plankensteiner, K., Prodan, R., Fahringer, T., and Iosup, A.(2009). Workflow monitoring and analysis tool for askalon. In *Grid and Services Evolution*, pages 1–14. Springer.

[35] Sahni, J. and Vidyarthi, D. P.(2016). Workflow-and-Platform Aware task clustering for scientific workflow execution in Cloud environment. *Future Generation Computer Systems*.

[36] Sakellariou, R., Zhao, H., and Deelman, E.(2010). Mapping Workflows on Grid Resources: Experiments with the Montage Workflow. In Desprez, F., Getov, V., Priol, T., and Yahyapour, R., editors, *Grids P2P and Services Computing*, pages 119–132. Springer.

[37] Singh, G., Su, M., Vahi, K., Deelman, E., Berriman, B., Good, J., Katz, D. S., and Mehta, G.(2008). Workflow task clustering for best effort systems with pegasus. In *15th ACM Mardi Gras Conference*.

[38] Singh, S. P., Nayyar, A., Kumar, R., and Sharma, A.(2018). Fog computing: from architecture to edge computing and big data processing. *The Journal of Supercomputing*, pages 1–36.

[39] SIPHT (2018). SIPHT. `http://pegasus.isi.edu/applications/sipht`.

[40] Solanki, A. and Nayyar, A.(2019). Green internet of things (g-iot): Ict technologies, principles, applications, projects, and challenges. In *Handbook of Research on Big Data and the IoT*, pages 379–405. IGI Global.

[41] TeraGrid (2018). The TeraGrid Project. `http://www.teragrid.org`.

[42] Tomas, L., Caminero, B., and Carrioón, C.(2012). Improving grid resource usage: Metrics for measuring fragmentation. In *12th IEEE/ACM International Symposium on Cluster, Cloud and Grid Computing (CCGrid '12)*, pages 352–359.

[43] Topcuoglu, H., Hariri, S., and Min-You, W.(2002). Performance-effective and low-complexity task scheduling for heterogeneous computing. *IEEE Transactions on Parallel and Distributed Systems*, 13(3):260–274.

[44] USC Epigenome Center (2018). USC Epigenome Center. `http://epigenome.usc.edu`.

[45] Wieczorek, M., Prodan, R., and Fahringer, T.(2005). Scheduling of scientific workflows in the askalon grid environment. In *ACM SIGMOD Record*, volume 34, pages 56–62.

[46] Ying, H., Mingqiang, G., Xiangang, L., and Yong, L.(2009). A webgis load-balancing algorithm based on collaborative task clustering. *Environmental Science and Information Application Technology, International Conference on*, 3:736–739.

[47] Zhang, X., Qu, Y., and Xiao, L.(2000). Improving distributed workload performance by sharing both cpu and memory resources. In *Proceedings of 20th International Conference on Distributed Computing Systems, (ICDCS'2000*, pages 233–241.

[48] Zhao, H. and Li, X.(2009). Efficient grid task-bundle allocation using bargaining based self-adaptive auction. In *The 9th IEEE/ACM International Conference on Cluster, Cloud and Grid Computing*.

[49] Zhou, Z., Cheng, Z., Zhang, L.-J., Gaaloul, W., and Ning, K.(2018a). Scientific workflow clustering and recommendation leveraging layer hierarchical analysis. *IEEE Transactions on Services Computing*, 11(1):169–183.
[50] Zhou, Z., Cheng, Z., Zhang, L.-j., Gaaloul, W., and Ning, K.(2018b). Scientific Workflow Clustering and Recommendation Leveraging Layer Hierarchical Analysis. *IEEE Transactions on Services Computing*, 11(1):169–183.

Edited by: Anand Nayyar
Received: Feb 28, 2019
Accepted: Mar 16, 2019