[Egyptian Informatics Journal 18 (2017) 205–214](http://dx.doi.org/10.1016/j.eij.2017.02.001)

Contents lists available at [ScienceDirect](http://www.sciencedirect.com/science/journal/11108665)

Egyptian Informatics Journal

journal homepage: [www.sciencedirect.com](http://www.sciencedirect.com/)

## Full length article

Green cloud environment by using robust planning algorithm



Jyoti Thaman [a](#_bookmark0),[⇑](#_bookmark5), Manpreet Singh [b](#_bookmark1)

a *M.M. University, Ambala, Haryana, India*

b *M.M. University, Sadopur, Ambala, India*

a r t i c l e i n f o

*Article history:*

Received 19 January 2016

Revised 15 December 2016

Accepted 5 February 2017

Available online 20 February 2017

*Keywords:*

Planning algorithms Scheduling algorithms Ready wueue

Robust

Cloud computing

a b s t r a c t

Cloud computing provided a framework for seamless access to resources through network. Access to resources is quantified through SLA between service providers and users. Service provider tries to best exploit their resources and reduce idle times of the resources. Growing energy concerns further makes the life of service providers miserable. User’s requests are served by allocating users tasks to resources in Clouds and Grid environment through scheduling algorithms and planning algorithms. With only few Planning algorithms in existence rarely planning and scheduling algorithms are differentiated. This paper proposes a robust hybrid planning algorithm, Robust Heterogeneous-Earliest-Finish-Time (RHEFT)[1](#_bookmark2) for binding tasks to VMs. The allocation of tasks to VMs is based on a novel task matching algo- rithm called Interior Scheduling. The consistent performance of proposed RHEFT algorithm is compared with Heterogeneous-Earliest-Finish-Time (HEFT)[2](#_bookmark3) and Distributed HEFT (DHEFT)[3](#_bookmark4) for various parameters like utilization ratio, makespan, Speed-up and Energy Consumption. RHEFT’s consistent performance against HEFT and DHEFT has established the robustness of the hybrid planning algorithm through rigorous simulations.

© 2017 Production and hosting by Elsevier B.V. on behalf of Faculty of Computers and Information, Cairo

University. This is an open access article under the CC BY-NC-ND license ([http://creativecommons.org/](http://creativecommons.org/licenses/by-nc-nd/4.0/)

[licenses/by-nc-nd/4.0/](http://creativecommons.org/licenses/by-nc-nd/4.0/)).

1. Introduction

Diverse resources with varied capabilities and connected through high speed interconnecting network provides new plat- form for distributed processing. Cloud and Grid computing evolved from such aggregation of resources. These are primarily main- tained by service providers (Amazon, IBM, Microsoft, etc). Users subscribe for services from these platforms and submit their tasks for processing. Users are served by allocating their tasks to various resources and executing them. When tasks executions times, inter- task dependencies and inter-task data transfer size is known then such task model is called static model. User’s submissions are pro-

\* Corresponding author.

*E-mail addresses:* [jyoti.thaman77@gmail.com](mailto:jyoti.thaman77@gmail.com) (J. Thaman), [dr.manpreet.singh.](mailto:dr.manpreet.singh.in@gmail.com) [in@gmail.com](mailto:dr.manpreet.singh.in@gmail.com) (M. Singh).

1 Robust Heterogeneous-Earliest-Finish-Time (RHEFT).

2 Heterogeneous-Earliest-Finish-Time (HEFT).

3 Distributed HEFT (DHEFT).

Peer review under responsibility of Faculty of Computers and Information, Cairo University.

**Production and hosting by Elsevier**

cessed in clouds by subjecting tasks to resources. Resource usage in clouds depends upon the types and sequence of tasks and resources. Work flow technologies are used to deal with increasing complex data, data-intensive application, simulations and analysis. These technologies are also used to schedule computational tasks on distributed resources, to manage dependencies among tasks and to stage data sets into and out of execution sites [[8]](#_bookmark27). These workflows are used to model computations in many scientific dis- ciplines [[9]](#_bookmark28).

A number of task scheduling algorithm are proposed in litera- ture which are broadly classified into list-scheduling algorithms, level-by-level scheduling, batch scheduling, duplication based scheduling, dependency scheduling, batch dependency scheduling algorithm, Genetic Algorithm (GA) based scheduling algorithms and hybrid algorithm. List scheduling algorithm creates a list of task while respecting task dependency. Tasks in list are processed in order of their appearance in the task list. The performance of such algorithm is comparatively better than other categories of algorithms. Level-by-level scheduling algorithms consider tasks of one level in task-graph such that task considered are indepen- dent of each other. This set of tasks may not include all the tasks in ready queue. In Genetic algorithm based solution schedules are reasonably acceptable but the computational complexity of

<http://dx.doi.org/10.1016/j.eij.2017.02.001>

1110-8665/© 2017 Production and hosting by Elsevier B.V. on behalf of Faculty of Computers and Information, Cairo University. This is an open access article under the CC BY-NC-ND license (<http://creativecommons.org/licenses/by-nc-nd/4.0/>).

algorithm is relatively high. Hybrid algorithm explores various combinations of existing classes of scheduling algorithms.

Task scheduling in heterogeneous systems is considered in Het erogeneous-Earliest-Finish-Time (HEFT) [[7]](#_bookmark29), Duplication based HEFT [[21]](#_bookmark41) and Deadline–Budget Constrained Scheduling (DBCS) [[4]](#_bookmark29). In HEFT [[7]](#_bookmark29), authors proposed a ranking of tasks on the basis of bandwidth, task’s length, and parent-child relationships. Tasks are considered for execution in order of their rank in decreasing order. Duplication based HEFT used the concept of task duplication and utilized the free cycles of VMs for execution of duplicate tasks. Distributed HEFT (DHEFT) exploits the concept of distributed approach and better exploits the concept of VM level availability for better task-VM mappings [[20]](#_bookmark42). This paper proposes a variant of HEFT called Robust HEFT (RHEFT) by using a hybrid approach and a novel scheduling algorithm for set of independent tasks. Tasks are ranked as per ranking method of HEFT, which is followed by grouping of free tasks into same group. Groups are processed in order of their creation. Tasks in a group are processed such that scheduling reduces the variance in difference of task’s execution time and VM’s mean execution time. Section [2](#_bookmark6) presents the related works, Section [3](#_bookmark8) presents the preliminary. Section [4](#_bookmark7) presents Inte- rior Scheduling (IS) and RHEFT algorithm. Section [5](#_bookmark14) presents sim- ulation set-up and performance discussion. Finally, concluded in Section [6](#_bookmark43) with a future direction.

1. Related works

This section presents a brief review of several research works done in the field of scheduling. Research work in [[10–15,8,14]](#_bookmark30) proposed scheduling solutions for workflows. Work in [[15,18,17,19,5,1]](#_bookmark33) refers to solution for independent tasks. [[4,7]](#_bookmark29) presents scheduling algorithms for heterogeneous systems. [[3]](#_bookmark29) presents a taxonomy of scheduling of tasks in clouds and grids.

Research work in [[8]](#_bookmark27), provided multiple scientific applications including astronomy, bioinformatics, earthquake science, and gravitational-wave physics is based on novel workflow profiling tools that provide detailed information (includes I/O, memory and computational characteristics) about various computational tasks that are present in the workflow. In [[10]](#_bookmark30), authors described an extension to Pegasus whereby resource allocation decisions are revised and described how adaptive processing has been retro- fitted to an existing workflow management system; a scheduling algorithm that allocates resources based on runtime performance. The results were evaluated using grid middleware over clusters. In [[11]](#_bookmark31), authors proposed a dynamic critical-path-based adaptive workflow scheduling algorithm for grids, which determines effi- cient mapping of workflow tasks to grid resources dynamically by calculating the critical path in the workflow task graph at every step. In [[12]](#_bookmark34), authors designed and analyzed a two-phase schedul- ing algorithm for utility Grids, called Partial Critical Paths (PCP), that was used to minimize the cost of workflow execution while meeting a user defined deadline and also proposed two workflow scheduling algorithms one was one-phase algorithm which is called IaaS Cloud Partial Critical Paths (IC-PCP), and a two-phase algorithm which is called IaaS Cloud Partial Critical Paths with Deadline Distribution (IC-PCPD2) that have a polynomial time complexity which make them suitable options for scheduling large workflows. Work in [[13]](#_bookmark35), proposed a new dynamic task scheduling algorithm for Heterogeneous environments called Clustering Based HEFT with Duplication (CBHD). The CBHD algorithm is considered an amalgamation between the most two important task scheduling in Heterogeneous machine, The Heterogeneous Earliest Finish Time (HEFT) and the Triplet Clustering algorithms. CBHD outper- forms the HEFT and Triplet algorithm by decreasing the makespan by 2.5%. It also achieves better load balancing than the HEFT algo-

rithm by 70%, and it increases processors utilization by 10% with respect to the HEFT and Triplet algorithms. In [[14]](#_bookmark36), the authors pre- sented a Hybrid Cloud Optimized Cost scheduling algorithm that decides which resources should be leased from the public cloud and aggregated to the private cloud to reduce costs while achieving the established desired execution time. HCOC tried to optimize the monetary execution costs while maintaining the execution time lower than Deadline.

In [[15]](#_bookmark33), authors proposed a novel heuristic for scheduling of set of independent tasks, called Balanced Minimum Completion Time (BMCT). First phase performs initial allocation using FCFS. In next phase BMCT tries to minimize the complete execution time by swapping tasks between machines. This results in balancing of load among the machines. BMCT has shown promising results when compared with Dynamic Level Scheduling (DLS) [[16]](#_bookmark37), Heteroge- neous Earliest Finish Time (HEFT) [[7]](#_bookmark29); Critical Path On a Processor (CPOP) [[7]](#_bookmark29) etc. under consistent heterogeneous, partially consistent heterogeneous and inconsistent heterogeneous environments. In [[5]](#_bookmark29), presented multi-objective PSO based optimization algorithm for dynamic environment of clouds and optimize energy and pro- cessing time. Proposed algorithm provides an optimal balance results for multiple objectives. The experimental results illustrated that the proposed methods out-performed the Best Resource Scheduling (BRS) and Random Selection Algorithm (RSA). In [[17]](#_bookmark38), authors proposed, two task scheduling algorithm namely user- Priority Awarded Load Balance Improved Min-Min Scheduling Algorithm (PA-LBIMM) and Load Balance Improved Min-Min (LBIMM) scheduling algorithm were proposed with objectives to decrease job’s completion time, improve the load balance and sat- isfy users’ priority demands in the cloud. LBIMM performs in two phases namely first phase is min-min and second phase is preemp- tion of smaller tasks from heavenly loaded resources and migrate them to resources with fastest completion time for preempted job. In PA-LBIMM tasks are divided into two groups based high or low priority. Initially, allocation is done to tasks with higher prior- ity and then tasks of lower priority are allocated to resources. Initial allocation is realized through Min-Min scheduling algorithm. In Next phase load balancing based on preemption of tasks is per- formed. Result reported in paper proves that PALBIMM and LBIMM outperform the Min-Min algorithm in all aspects. In [[1]](#_bookmark29), authors proposed an energy efficient scheduling algorithm, (EEVS) consid- ering the deadline constraint. EEVS can support DVFS well. From the computation of total energy of a PM, authors conclude that there is an optimal frequency for it to process certain VMs. Based on the optimal frequency; authors define the optimal perfor- mance–power ratio to weight the heterogeneities of the PMs. The PM with highest optimal performance–power ratio will be used to process the VMs first unless it does not have enough computa- tion resources. Finally the cloud should be reconfigured to consol- idate the computation resources of the PMs to further reduce the energy consumption. EEVS consumes less energy and processes more VMs successfully than the existing methods. In [[4]](#_bookmark29), presented a heuristic scheduling algorithm with quadratic time complexity that considers two important constraints for QoS-based workflow scheduling, time and cost, named Deadline–Budget Constrained Scheduling (DBCS) for heterogeneous systems. DBCS has the lowest time complexity (quadratic time complexity), while other algo- rithms mostly have cubic or polynomial time complexities. In terms of the quality of results, DBCS achieves rates of successful schedules similar to higher-time complexity algorithms for both random and real application workflows on diverse platforms.

In [[18]](#_bookmark39), authors presented two novel dynamic scheduling algo- rithms for heterogeneous and federated cloud system. The objec- tive was to achieve resource optimization mechanism for preempt-able applications in autonomous heterogeneous cloud environment. Authors also proposed a dynamic procedure with

updated information. The procedure helped to achieve considerable improvement in resource utilization and energy efficiency in any given resource contentious environment. In [[19]](#_bookmark40), authors had pre- sented a thorough review of workflow scheduling algorithms under different classes. Authors proposed a paradigm to classify the exist- ing workflow scheduling algorithms and presented a useful con- cluding remark. In [[6]](#_bookmark29), authors presents a workflow schedule

Then tasks in the workflow are ordered in HEFT based on a rank function. For an exit task the rank value is:

*Rank*(*Ti*)= *ti* (3)

The rank values of other tasks are computed recursively based

on Eqs. [(1)–(3)](#_bookmark9) as shown in Eq. [(4)](#_bookmark7).

optimization algorithm (MER) that can be used with any existing workflow scheduling algorithm as a post-processing technique. It

*Rank*(*Ti*)= *ti* + max

*Tj* ∈*succ*(*Ti* )

(*cij* + *Rank*(*Tj*)) (4)

consists of three major phases to first find the trade-off points between the minimum makespan increase and the maximum resource usage reduction, and to consolidate tasks and resources leading to significant improvement in resource efficiency. Based on results from extensive experiments with five real-world scien- tific workflows confirm the claims. Finally, this work study revealed that by allowing a small degree of makespan increase, such exploitation reduces resource usage far greater than any incurred makespan increase. Based on results obtained from our extensive simulations using scientific workflow traces, we demonstrate MER is capable of reducing the amount of actual resources used by 54% with an average makespan increase of less than 10%.

In [[3]](#_bookmark29), authors Identified & explained the aspects and classifica- tions unique to workflow scheduling in the cloud environment in three categories, namely, scheduling process, task and resource. Lastly, review of several scheduling techniques are included and classified onto the proposed taxonomies. The proposed taxonomies serve as a stepping stone for those entering this research area and for further development of scheduling technique. The present tax- onomies of cloud workflow scheduling problems and techniques based on analysis of existing research literature, which classifies techniques in grid workflow scheduling, by adding new aspects unique to cloud computing and refining some existing ones. It is noticeable that almost every technique proposed so far has the assumption that resources are virtual machine instances (i.e. infrastructure-as-a-service).

Most of the works reviewed in this section refers to scheduling with objective of reducing makespan, improving resource utiliza- tion and reducing financial liabilities. Most proposals lack basic con- sideration like hybrid of scheduling techniques. With limited scope of improvement in scheduling schemes and without considering out-of-box alteration, this work presents a mathematical viable solution for improving the performance of scheduling algorithms.

1. Preliminary

In this section HEFT [[7]](#_bookmark29) algorithm is discussed as preliminary to this research. Heterogeneous-Earliest-Finish-Time (HEFT) algo- rithm was proposed by Topcuoglu et al. The algorithm is based on the computation of task’s rank. Algorithm computes average execution time for each task and average communication time between resources of two successive tasks on the basis of

parent-child relationship between concerned tasks. Let *time*(*Ti*; *r*)

be the execution time of task *Ti* on resource *r* and let *Ri* be the

set of all available resources for processing of *Ti*. The average exe- cution time of a task *Ti* is defined as

HEFT is based on global approach on scheduling without taking into consideration the complete set of tasks in ready queue. This poor approximation of ready queue tasks affects the performance of HEFT in highly resource available environment. HEFT performs allocation of tasks to VMs on the basis of ranks. HEFT is accepted widely in various projects of significant importance like ASKALON project [[22]](#_bookmark44) to provide scheduling for a quantum chemistry appli- cation, WIEN2K [[23]](#_bookmark45), and a hydrological application, Invmod [[24]](#_bookmark46) on the Austrian Grid.

1. Robust HEFT: A hybrid Planning algorithm

This section presents a hybrid planning algorithm for cloud environment which addresses the limitations of HEFT. Section [3](#_bookmark8) discussed HEFT planning algorithm which is one of the most promising planning algorithm. HEFT works on the centralized approach and utilizes the ranks of the tasks as decision parameter while subjecting next tasks to some free VM. Ranked tasks are arranged and scheduled in non-increasing order by their ranks. Next-ranked task is assigned to next free VM. This assignment/ mapping of ranked task to free/available VM is random. No suit- ability criteria were used for this mapping. As a result HEFT could poorly approximate the ready queue. The schedules obtained from HEFT were not able to utilize the available resources in best possi- ble way. These many limitations of HEFT provide motivation for some improvements in the functioning of HEFT.

* 1. *Robust HEFT (RHEFT)*

The improvement in working of planning algorithms has been presented in this section through a hybrid of HEFT and Interior Scheduling. The working of RHEFT is divided into three phases. In phase 1, HEFT is used for generation of tasks which are sorted on the basis of ranks. The working principle of HEFT is explained in Section [3](#_bookmark8). The ranks are computed using {Eqs. [(1)–(4)](#_bookmark9)}. The bene- fits of HEFT includes that a non-linear Task graph is converted to a linear list of tasks. A visible limitation of HEFT is that the task scheduled next is only a member of set of tasks in ready queue. This limitation not only reduces the utilization of resources but also increases the length of schedules. Reduced utilization and longer schedule length not only affects energy consumption but also proves to be economically inefficient.

Keeping these limitations in view, an extension of HEFT is pro-

posed by using hybrid of HEFT and IS in this work. New planning strategy is named as Robust HEFT (RHEFT).

*t* = P*r*∈*Ri time*(*Ti*; *r*)

(1)

In phase 2, RHEFT divides the resultant ranked tasks into several

*i* |*Ri*|

Let time (*eij*; *ri*; *rj*) be the data transfer time between resources *ri* and *rj* which process the task *Ti* and task *Tj* respectively. Let *Ri* and *Rj*

be the set of all available resources for processing *Ti* and *Tj* respec- tively. The average transmission time from *Ti* to *Tj* is defined by:

sets, where each set contains a set of independent ranked tasks. Phase 2 outputs are the set of tasks ready for scheduling. Each such set represents a bigger portion of set of ready tasks. Phase 2 begins at step 9 and finish at Step18, in algorithm presented in [Fig. 1](#_bookmark13). The out- put of phase 2 is directed acyclic graph where each node represents

the set of independent tasks identified in phase 2 of RHEFT {[Fig. 2](#_bookmark10)}.

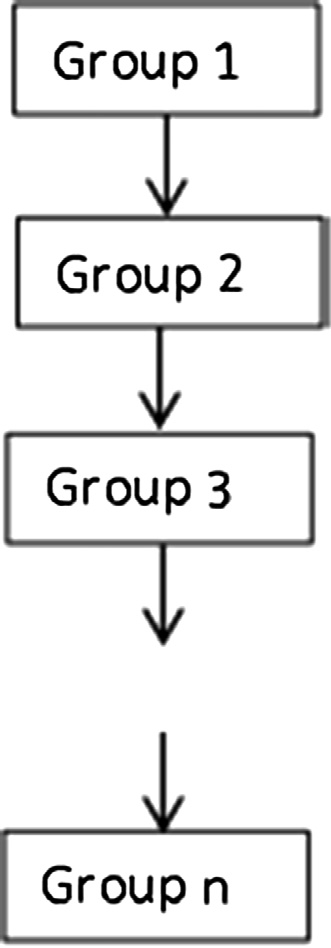
*cij* =

P*ri* ∈*Ri* ;*rj* ∈*Rj time*(*eij*; *ri*; *rj*)

|*Ri*||*Rj*|

(2)

In phase 3, IS scheduling is applied on set of independent tasks. Phase 3 begins at Step 19 in RHEFT algorithm presented in [Fig. 1](#_bookmark13). IS scheduling approach is presented ahead in the section.

* 1. *Interior scheduling (IS)*

Interior scheduling approach is a novel idea for mapping set of independent tasks to available VMs. The mapping utilizes the sta- tistical characteristics of the VMs and set of tasks at hand.

Let *Task* = {*T*1; *T*2 ... ; *Tn*} {[Table 1](#_bookmark11)} represents the set of tasks

*VM* = {*M*1*M*2; ... ; *Ml*} represents the set of available VMs with with their execution requirements. Similarly, assume that their capacity. An execution matrix may be computed using values

in set *Task* and *VM*.

2 *e*1;1 ·· · *ea*;*n* 3

.

.

*MAT* = 64 .

.

. *ea*;*n* 75

(5)

*el*;1 ·· · *ea*;*n*

where *ei*;*j* represents the execution time of task *Tj* while executing

*VM*.*MeanExecutionTimei* = (*e*1;1 + *e*1;2 + ... *e*1;*n*)/*n*. This value can be on *Mi*. Using *MAT* matrix we can compute average of each row as used in Eq. [(5)](#_bookmark12) for minimizing the variance.

(r2 = (*VM*.*MeanExecutionTime* — *Task*.*ExecutionTime* 2 (6)

) )

Using this as objective we have considered following example.

Consider an example where; *Task*

8> 78; 92; 23; 33; 55; 77; 88; 78; 102; 9>

Table 1

Fig. 2. Output directed acyclic graph of phase 2 in RHEFT.

= ><

>>:

23; 33; 55; 106; 85; 78; 91;

23; 33; 55; 79; 88; 78; 92;

26; 33; 56; 74; 88; 79; 105

>= and *VM* = {12; 7; 12; 11}.

>>;

Notation table.

S. No. Notation Meaning Values

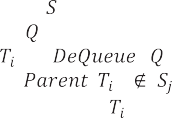
Numerical values in *Task* represents the MIPS of tasks. In total 30 tasks were considered. Numerical values in *VM* represents the MIPS of VMs and total 4 VMs were considered. A comparison in performance of Min-Max, Max-Max and IS was performed on uti- lization and makespan parameters. The comparison has been shown in [Table 2](#_bookmark15).

|  |  |  |
| --- | --- | --- |
| 1. *Ti* 2. *T*(1 × *t*) 3. *M*(*m* × 1) 4. *Q* | *ith* Task  Task Length Matrix VM Capacity Matrix FIFO Queue | –  –  –  – |
| 5. *S*(*m* × *t*) | Output Schedule Matrix | – |
| 6. *Si* | *ith* Set of independent Tasks | – |
| 7. *Mi* | *ith* Virtual Machine | – |
| 8. *L*(*m* × *t*) | Load Matrix | – |
| 9. *m* | Number of VMs | {5, 10, 20} |
| 10. *t* | Number of Tasks | {50,100} |

The results shown in [Table 2](#_bookmark15) uses min–max, max-max and IS for task-machine mappings. In IS approach any tasks *Ti* is mapped to

**Algorithm: RHEFT**

* + 1. Calculate mean execution time for each task  by using equation 1
    2. Calculate mean data transfer delay between tasks and their successors in a task graph or workflow by using equation 2
    3. Calculate Rank of each task  by using equations 3 and 4
    4. Construct a queue  by insertion of tasks in descending order by their Rank
    5. Construct a set  for addition of tasks
    6. Compute Load Matrix ( ) from and .



* + 1. Compute mean execution times ( 1) for each machine using  ()
    2. Initialize = ( , ) // Zero Matrix
    3. **While** not empty **do**

**10.** = ( )

1. **If** ( ) && ( )
2. Add task to set
3. **Else**

14.  =  + 1;

1. Construct a set  for addition of tasks
2. Add task  to set 
3. End **If**
4. End **While**
5. **For ** = 1  **do**
6. Identify the Machine id ( ) which is free and can be subjected next task



1. Identify the tasks id () in which if allotted to  results in minimum increase in variance of execution times of completed and newly submitted task 
2. Set  (, ) =  ( ,  )
3. End **For**
4. Print values of S matrix as output schedule.

**25.** End **RHEFT**

Fig. 1. Robust HEFT planning algorithm for workflows in clouds.

Table 2

Performance comparison based on Makespan and Utilization of Min-Max, Max-Max and IS

Scheme Min-Max Max-Max IS Makespan 62.92 56.83 49

Utilization 81.11 78.74 98.59

an *Mj* only if the execution time (*ej*;*i*) has smallest difference as compared mean execution time on *Mj*. This principle is expressed in Eq. [(5)](#_bookmark12) where we strive to reduce the variance of tasks assigned to each VM for a given set of independent tasks. Improved perfor- mance of IS in example above validates the strength of task-VM mapping criteria discussed above.

Each time IS approach selects a task to be scheduled next on a particular VM such that Eq. [(5)](#_bookmark12) is satisfied. The selected task is most appropriate task to be schedules next on given VM. In fact, this approach identified a task whose execution time characteris- tics for given VM exhibits correlations with execution time charac- teristics of tasks already submitted or completed on given VM. This approach can be applied to overcome the non-aligned task alloca- tion/binding issues in other task mapping or scheduling algorithms and heuristics.

1. Simulation and analysis

Simulation environment and various performance characteris- tics of different planning algorithms are presented in this section. Performance analysis is presented here is based on simulation in WorkflowSim.

* 1. *Simulation setup*

Simulation is carried out by using WorkflowSim [[20]](#_bookmark42) configured in Eclipse on an Intel Core 2 Duo, 2.0 GHz Linux based laptop. Sim- ulation is considered for task sizes equal to 50 and 100. Numbers of VMs considered for simulation purpose were equal to 5, 10 and 20 respectively. Various VM characteristics as defined in Work- flowSim are retained as-is where-is.

Each VM considered in simulation possessed 1000 MIPS, 512 MB RAM, bandwidth 1000 MB/s, Processing Elements (PEs) 1 and Image Size 10,000. VM architecture is inherited from ‘Xen’. Besides this Space Shared scheduling of tasks was considered for the simulation purpose. Maximum power consumption rate for

### Makespan (Tasks= 100)

800.00

VM\_5 VM\_10

VM\_20

97.21

159.70

666.74

VM\_20

142.21

206.27

607.80

VM\_10

180.96

292.73

378.41

VM\_5

RHEFT

DHEFT

HEFT

700.00

600.00

**Time(Sec)**

500.00

400.00

300.00

200.00

100.00

0.00

### Makespan (Tasks = 50)

1400.00

1200.00

303.83

577.61

705.02

VM\_5

DHEFT

HEFT

243.47

567.50

866.52

VM\_10

RHEFT

151.85

361.62

1156.44

VM\_20

1000.00

**Time (Sec)**

800.00

600.00

400.00

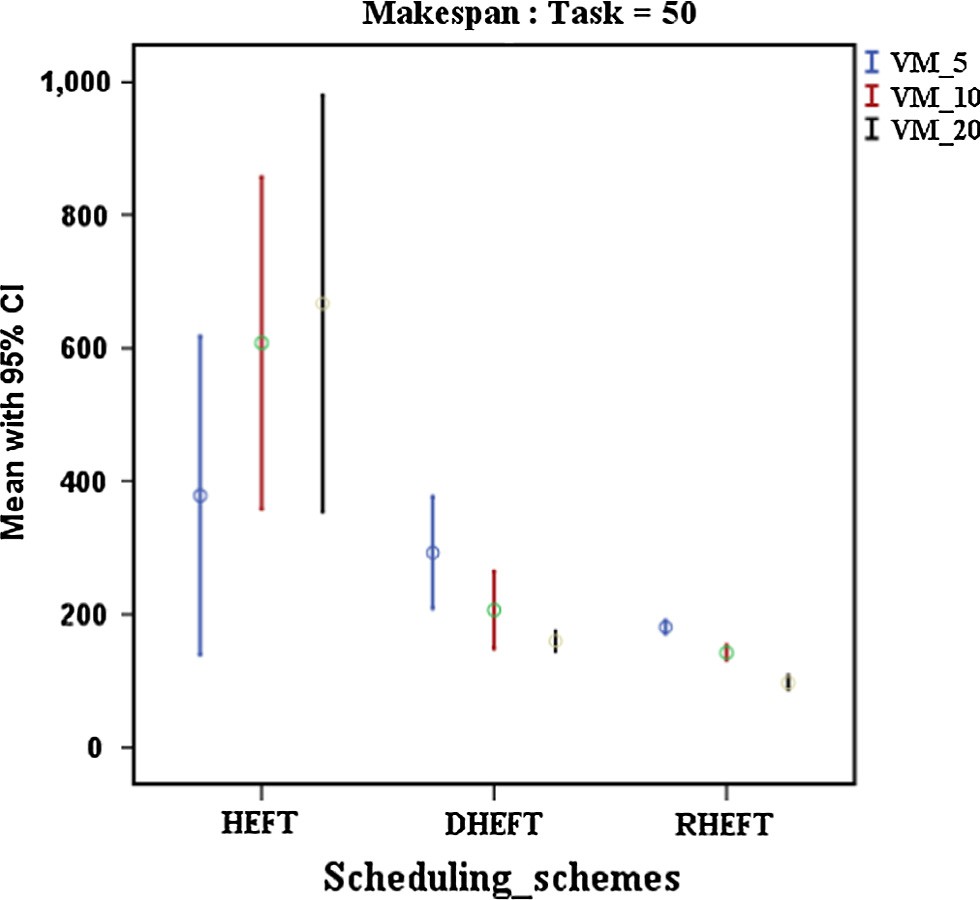
200.00

0.00

VM\_5 VM\_10 VM\_20

**Scheduling Schemes**

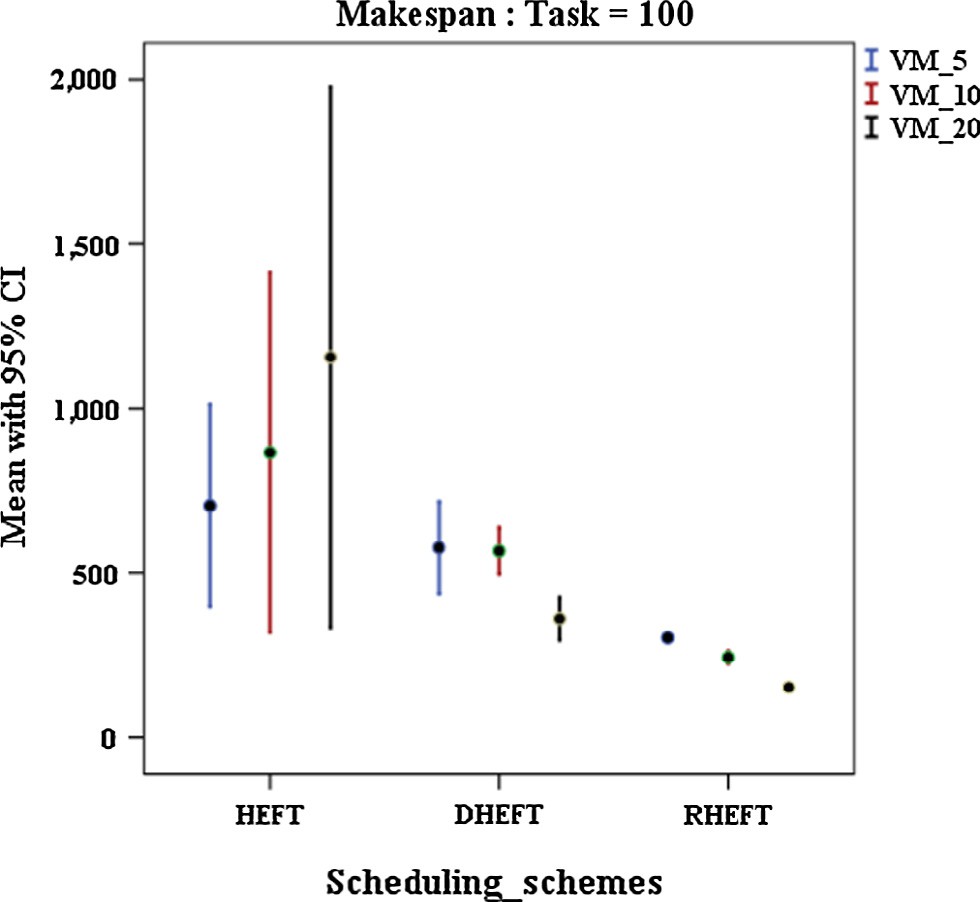
# (a)



**(b)**

**Scheduling Schemes**

# (a)



**(b)**

Fig. 3. (a): Makespan characteristics of various scheduling/planning algorithms using 50 tasks (b) makespan error graphs of various scheduling/planning algo- rithms using 50 tasks (with C.I. = 95%).

Fig. 4. (a): Makespan characteristics of various scheduling/planning algorithms using 100 tasks (b) Makespan error graphs of various scheduling/planning algorithms using 100 tasks (with C.I. = 95%).

VMs is considered fixed to 250 Watts/s. Extensive simulation of WorkflowSim supported planning algorithms like HEFT, DHEFT and new proposal RHEFT is considered. Simulation based output is drawn in [Figs. 2–9](#_bookmark10) respectively. To represent static task model, Montage workflow for 50 and 100 tasks is considered for close imitation of CPU intensive tasks. Simulation is performed to study

1. *Utilization:* Utilization is defined as the ratio of duration of actual usage and duration of actual availability. Improved utilization reduces the makespan characteristics.
2. *Energy Consumption:* Resources in clouds consumes energy from the moment when they are allocated for execution of tasks of users. Improved utilization reduces the span of usage.

*max*

the impact on Makespan, Utilization, Energy consumption and

Energy consumption is defined in Watts. Let *powern*

exploit

Speed-up. Next subsection presents a detailed discussion on vari- ous performance parameters.

* 1. *Performance discussion*

Performance plots of RHEFT and other scheduling schemes like, HEFT and DHEFT are shown in [Figs. 3–10](#_bookmark16). [Tables 3–6](#_bookmark25) provides in-depth error analysis at Confidence Interval (CI = 95%). Various significant parameters which are relevant for context in cloud computing are as follows.

a. *Makespan:* Makespan is defined as the time span between the instant when first task is scheduled and the instant when last task completed the execution. Any parallel execution of tasks reduces the makespan characteristics.

### Utilization (%) (Tasks = 50)

90.00

80.00

70.00

60.00

50.00

40.00

30.00

20.00

10.00

0.00

VM\_5 VM\_10 VM\_20

VM\_5 VM\_10

VM\_20

HEFT 66.67

22.26

13.26

DHEFT 56.38

36.14

19.85

RHEFT 79.35

55.06

37.98

**Utilization(%)**

**Utilization (%)**

maximum power consumed by *nth* server. The idle server con- sumes nearly 70% of a fully utilized server [[2]](#_bookmark29). Power con- sumption by the server *nth* at any instant of time *t* is [[2]](#_bookmark29).

*powern*(*t*)= *powern* \* 0.70 + *powern* \* 0.30 \* *Un* (7) where *Un*(*t*) represent utilization at that instant of time. Ser- vers consume a lot of power even if they are idle. It is better

*max*

*max*

if idle or lightly loaded server nodes may be vacated and switched-off.

1. *Speed-up:* Speed up is defined as the ratio of makespan of parallel execution of set of tasks to makespan of sequential execution of tasks.

### Utilization (%) (Tasks = 100)

100.00

90.00

80.00

70.00

60.00

50.00

40.00

30.00

20.00

10.00

0.00

VM\_5 VM\_10 VM\_20

VM\_5 VM\_10

VM\_20

HEFT 68.93

37.69

16.66

DHEFT 62.45

33.29

22.34

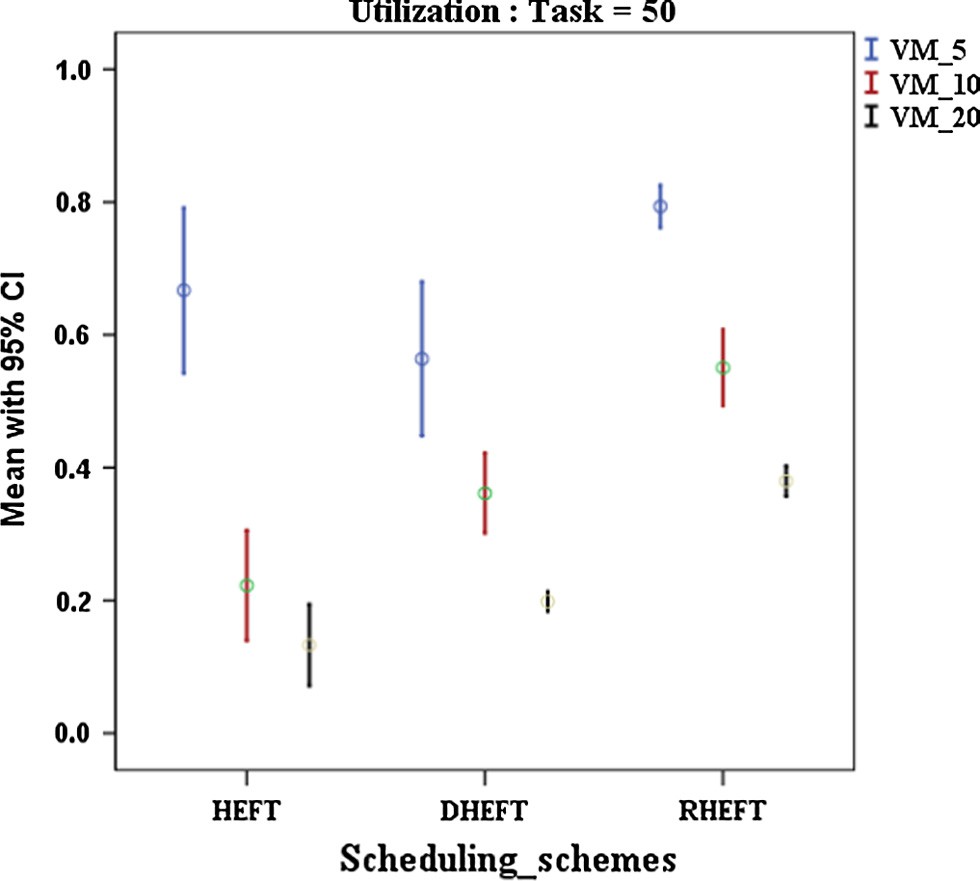
RHEFT 86.11

69.58

54.06

**Scheduling Schemes**

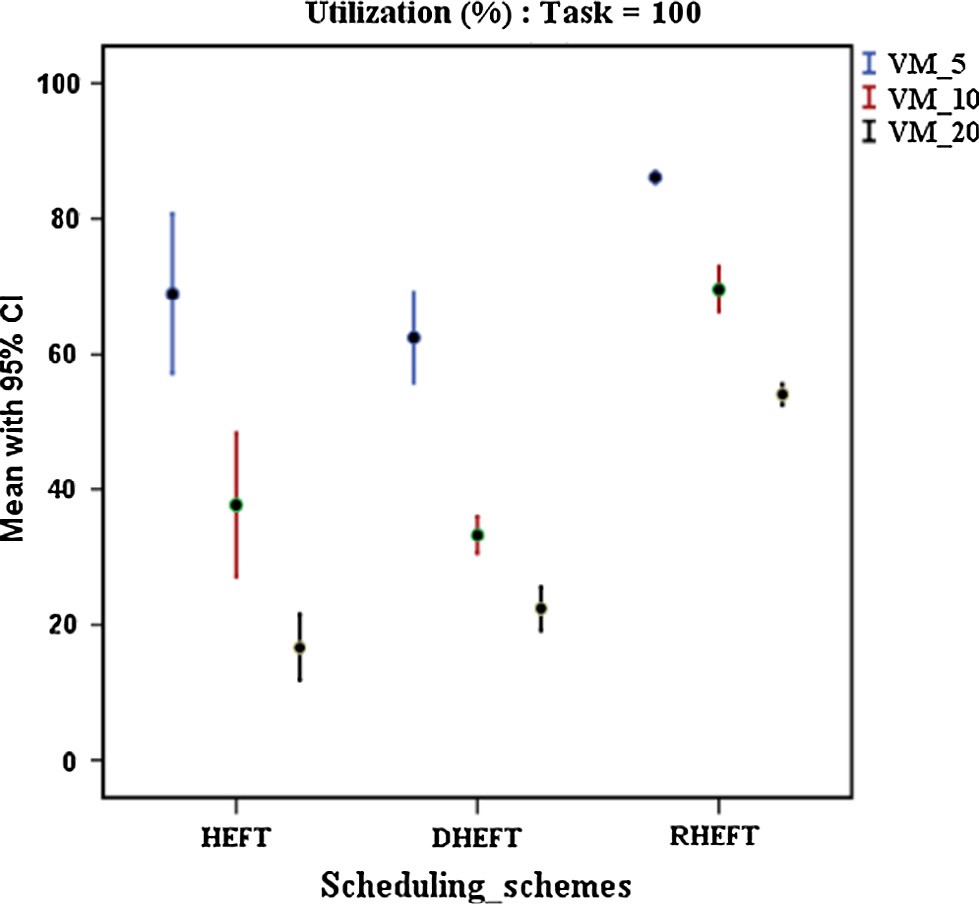
# (a)



**(b)**

**Scheduling Schemes**

(a)



(b)

Fig. 5. (a) Utilization characteristics of various scheduling/planning algorithms using 50 tasks (b) Utilization error graphs of various scheduling/planning algo- rithms using 50 tasks (with C.I. = 95%).

Fig. 6. (a) Utilization characteristics of various scheduling/planning algorithms using 100 tasks (b) Utilization error graphs of various scheduling/planning algorithms using 100 tasks (with C.I. = 95%).

[Figs. 3](#_bookmark16)(a) and [4](#_bookmark17)(a) presents makespan characteristics of HEFT, DHEFT and RHEFT for 50 tasks and 100 tasks respectively. The bars for HEFT with 50 and 100 tasks exhibit that when VM are increased from 5 to 10 and from 10 to 20 VMs respectively, makespan is on increasing spree. Rather with the increase of resources it should decrease. Another conclusion that can be drawn is that when tasks are increased from 50 to 100 tasks, slope of makespan characteris- tic for HEFT turns from positive to negative. Negative slope con- firms that with more tasks HEFT better utilizes more resources than with less number of tasks. Although more VMs are available for the execution of same set of tasks, but execution time in HEFT is increasing and is highest as compared to other schemes plotted in [Figs. 3](#_bookmark16)(a) and [4](#_bookmark17)(a). The reason for this kind of behavior is attrib- uted to fact that HEFT considers smallest set of tasks for allocation from all the tasks in ready queue.

DHEFT used the concept of distributed approach and maps the tasks without computing ranks. Distributing the decision of task- VM mapping and considering Earliest Finish Time First approach, DHEFT improves makespan characteristics in comparison to HEFT. In RHEFT, phase 2 identifies a subset of independent or free tasks which better approximates set of tasks in ready queue. In

### Speed Up (Tasks = 50)

6.00

5.00

4.00

3.00

2.00

1.00

0.00

VM\_5 VM\_10 VM\_20

VM\_5

VM\_10 VM\_20

HEFT 1.84

0.99

0.99

DHEFT 1.85

2.59

3.21

RHEFT 2.82

3.60

5.28

**Speedup(Ratio)**

**Speed up( Ratio)**

phase 3 IS, is used to schedule set of independent tasks on available resources. IS improves the makespan characteristics by generating a schedule based on {Eq. [(6)](#_bookmark12)}. Phase 3 is hybrid phase and advances the scheduling from global to sub-local level. This characteristic of RHEFT results in improvement of makespan characteristics in com- parison to HEFT and DHEFT. [Figs. 3](#_bookmark16)(a) and [4](#_bookmark17)(a) presents makespan characteristics of HEFT, DHEFT and RHEFT respectively.

Standard Error Graphs shown in [Figs. 3](#_bookmark16)(b) and [4](#_bookmark17)(b) respectively, drawn at Confidence Interval (CI = 95%) of 95, exhibits that HEFT and DHEFT has shown a lot of variations in results over repeated experimentation. RHEFT has resulted in minimum error at CI = 95%. The increases in number of tasks as well as increase in numbers of resources, both are better utilized in RHEFT. RHEFT has shown minimum error. [Tables 3a and 3b](#_bookmark25) presents lower and upper bounds of standard error w.r.t. mean makespan statistics, for 50 and 100 tasks respectively. Lower range of RHEFT dictates robust behavior of RHEFT.

[Figs. 5 and 6](#_bookmark18) plots the utilization performance and error graphs at CI = 95%, for HEFT, DHEFT and RHEFT for 50 and 100 tasks

# Speedup (Tasks = 100)

8.00

7.00

6.00

5.00

4.00

3.00

2.00

1.00

0.00

VM-5 VM\_10

VM\_20

VM-5

VM\_10 VM\_20

HEFT 1.77

1.59

1.26

DHEFT 1.94

1.92

3.06

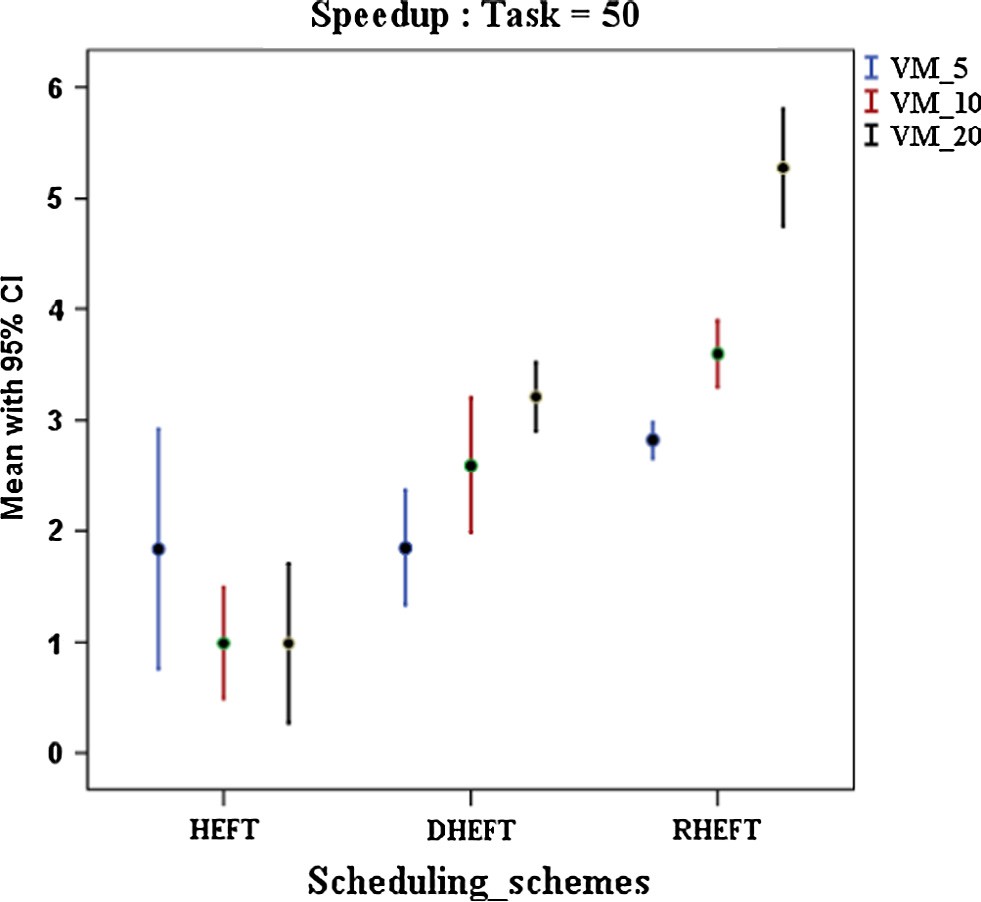
RHEFT 3.55

4.46

7.12

**Scheduling Schemes**

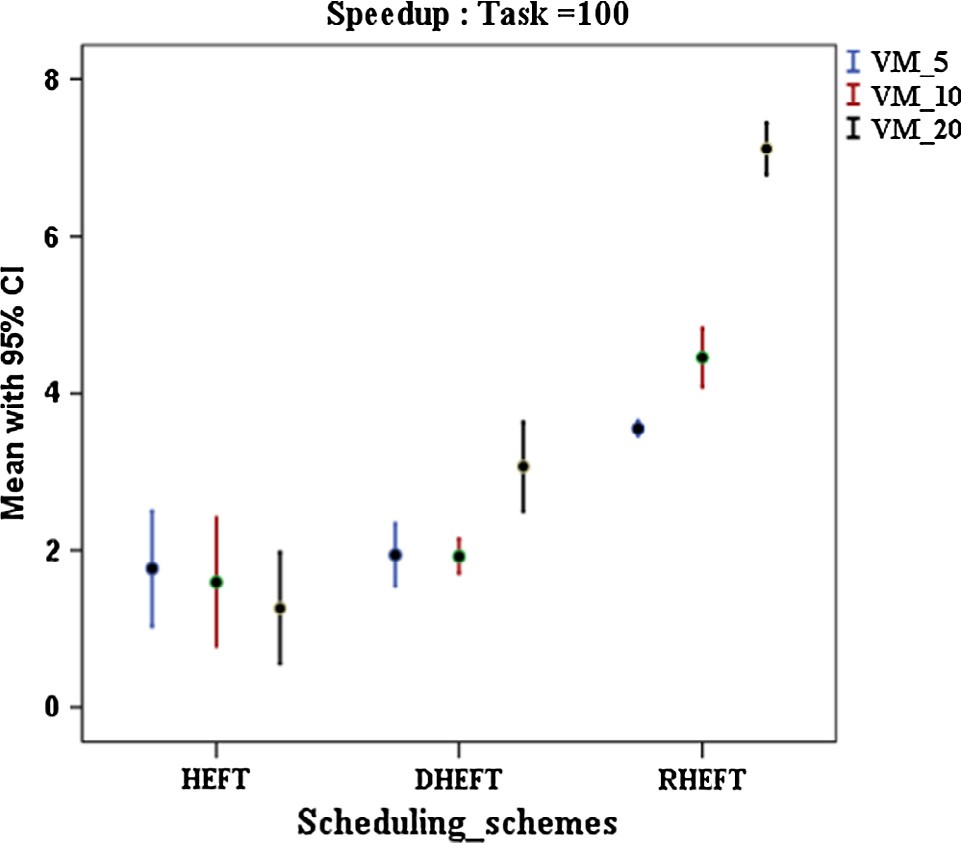
(a)



(b)

### Scheduling Schemes

(a)



## (b)

Fig. 7. (a) Speed up characteristics of various scheduling/planning algorithms using 50 tasks (b) Speed up error graphs of various scheduling/planning algorithms using 50 tasks (with C.I. = 95%)

Fig. 8. (a) Speed up characteristics of various scheduling/planning algorithms using 100 tasks (b) Speed up error graphs of various scheduling/planning algorithms using 100 tasks (with C.I. = 95%).

respectively. HEFT categorically selects tasks only on the basis of ranks of tasks. This results in under-utilized resources. This is shown in [Figs. 5](#_bookmark18)(a) and [6](#_bookmark19)(a). Even with the increase in resources, HEFT fails to use available extra resources. Increase of resources doesn’t improve the performance of HEFT rather it is sheer waste of resources. HEFT has improved in terms of utilization with increase in number of tasks, i.e., when tasks are increased from

50 to 100 utilization improved marginally at VM = 10 and VM = 20. For a given set of tasks, selection of tasks for execution is independent of available resources. Extra resources are thus waste in HEFT. In case of DHEFT, no ranks were calculated. It was based on the principle of Earliest Finish Time. In DHEFT, a better Task-VM mapping was resulted. This improves the makespan char- acteristics in DHEFT as compared to that of HEFT.

When it comes to RHEFT, the performance is much better than other schemes. Utilization in RHEFT is more than both HEFT and DHEFT, but utilization is falling with higher resource availability. The falling trend in utilization is best compensated with reduced makespan characteristics of RHEFT. RHEFT exploits the resources

to better utilization level than other schemes. That’s why this work is named as Robust HEFT (RHEFT). [Figs. 5](#_bookmark18)(b) and [6](#_bookmark19)(b) draws the standard error at CI = 95% for 50 and 100 tasks respectively. The Error data in [Tables 4a and 4b](#_bookmark23) gives better insight that utilization in RHEFT vary in smallest range among HEFT, DHEFT and RHEFT.

[Figs. 7](#_bookmark20)(a) and [8](#_bookmark21)(a), plots the speed-up achieved as a result of parallel execution of tasks as compared to sequential execution of tasks. Better performance of RHEFT is due to consistent better utilization. RHEFT performed better than HEFT and DHEFT under both scenario i.e. VM = 5, VM = 10 and VM = 20. [Figs. 7](#_bookmark20)(b) and [8](#_bookmark21)

(b) plots the standard error at CI = 95%. The range of variations in RHEFT is lowest when considered at 95% confidence Intervals. Also, considering higher values of average utilization in RHEFT, error range is acceptable. The error range of HEFT and DHEFT are poor at their low speed up levels. Error data is shown in [Tables 5a and](#_bookmark26) [5b](#_bookmark26) for 50 and 100 tasks respectively.

[Figs. 9](#_bookmark22)(a) and [10](#_bookmark24)(a) draws energy characteristics of this work. Energy consumption is based on {Eq. [(6)](#_bookmark12)}. The improved utilization and reduced makespan affects the energy consumptions. The nega-

# Energy Consumption (Tasks = 50)

140000.00

120000.00

100000.00

80000.00

60000.00

40000.00

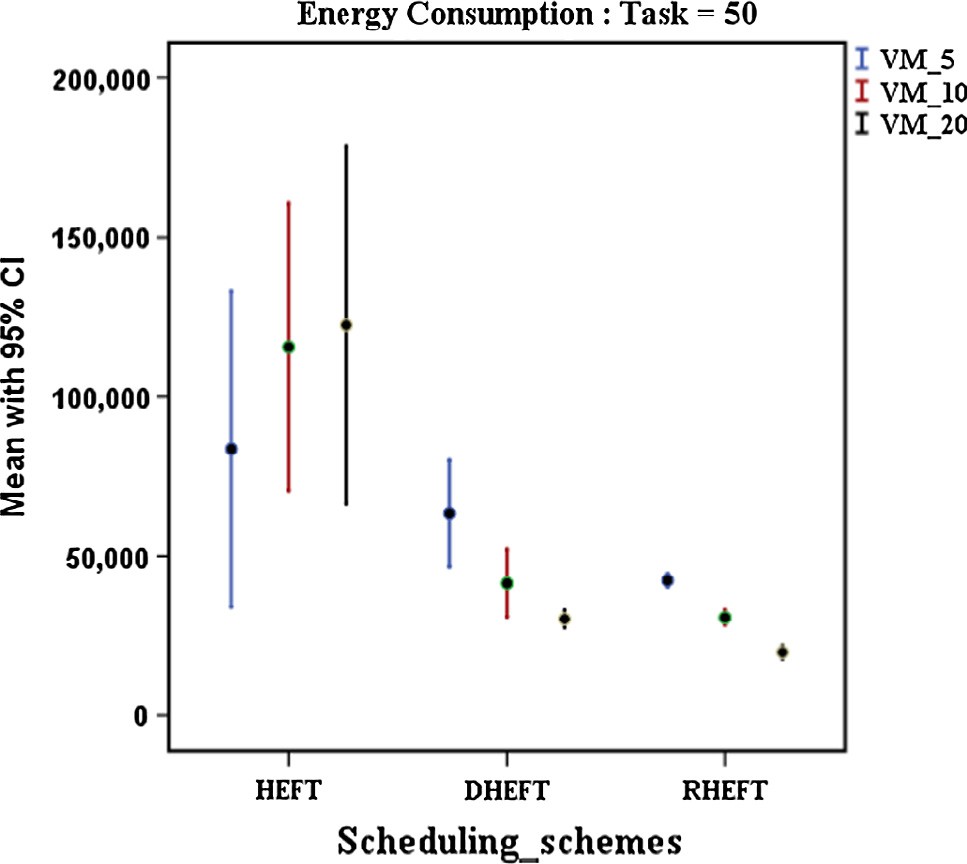
20000.00

VM\_5 VM\_10

VM\_20

**Energy Consumption (units)**

|  |  |  |  |
| --- | --- | --- | --- |
| 0.00 | HEFT | DHEFT | RHEFT |
| VM\_5 | 83482.38 | 63330.48 | 42424.81 |
| VM\_10 | 115523.88 | 41501.36 | 30748.21 |
| VM\_20 | 122428.17 | 30317.09 | 19778.68 |
|  |  | **Scheduling Schemes**  **(a)** |  |



**(b)**

Fig. 9. (a) Energy consumption characteristics of various scheduling/planning algorithms using 50 Tasks (b) Energy consumption error graphs of various scheduling/planning algorithms using 50 tasks (with C.I. = 95%)

tive slope of energy consumption in RHEFT is attributed to reduced makespan and consistently better utilization characteristics.

Error graphs in [Figs. 9](#_bookmark22)(b) and [10](#_bookmark24)(b) plots the standard error at CI = 95%. Low variation in RHEFT as compared to HEFT and DHEFT

justifies the robustness of RHEFT. This is why RHEFT is step for- ward towards Green cloud. The error tables in [Tables 6a and 6b](#_bookmark32), justifies the claims.

### Energy Consumption (Tasks = 100)

250000.00

200000.00

150000.00

VM\_5

VM\_10 VM\_20

100000.00

50000.00

0.00

Table 3b

Error Table for Makespan (95% CI) (Tasks = 100).

No. of VMs

**Energy Consumption (Units)**

Scheduling schemes

Mean 95% Confidence interval of mean Lower bound Upper bound

VM\_5 HEFT 705.020 396.970 1013.070

DHEFT 577.610 437.090 718.120

RHEFT 303.830 295.320 312.330

VM\_10 HEFT 866.520 319.750 1413.290

DHEFT 567.500 495.490 639.500

RHEFT 243.470 222.890 264.050

VM\_20 HEFT 1156.440 334.370 1978.520

DHEFT 361.620 295.830 427.4000

RHEFT 151.850 144.660 159.040

|  |  |  |  |
| --- | --- | --- | --- |
|  | HEFT | DHEFT | RHEFT |
| VM\_5 | 157869.04 | 127780.90 | 72790.62 |
| VM\_10 | 173052.90 | 113405.80 | 55339.76 |
| VM\_20 | 214792.10 | 69277.96 | 32730.26 |

#### Scheduling Schemes

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| VMs | Schemes |  | Lower bound | Upper bound |  |
| VM\_5 | HEFT | 66.6717 | 54.2043 | 79.1391 |  |
|  | DHEFT | 56.3833 | 44.7462 | 68.0205 |  |
|  | RHEFT | 79.3511 | 76.1298 | 82.5724 |  |
| VM\_10 | HEFT | 22.2612 | 13.9532 | 30.5692 |  |
|  | DHEFT | 36.1435 | 30.0653 | 42.2217 |  |
|  | RHEFT | 55.0611 | 49.2955 | 60.8268 |  |
| VM\_20 | HEFT | 13.2624 | 7.0856 | 19.4393 |  |
|  | DHEFT | 19.8467 | 18.3853 | 21.3080 |  |
|  | RHEFT | 37.9750 | 35.6656 | 40.2844 |  |

Table 4a

Error table for utilization (95% CI) (Tasks = 50)

# (a)

No. of

Scheduling

Mean 95% Confidence interval of mean

# (b)

Fig. 10. (a) Energy consumption characteristics of various scheduling/planning algorithms using 100 tasks (b) Energy consumption error graphs of various scheduling/planning algorithms using 100 tasks (with C.I. = 95%)

Table 3a

Error Table for Makespan (95% CI) (Tasks = 50).

Table 4b

Error table for utilization (95% CI) (Tasks = 100).

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| No. of | Scheduling | Mean | 95% Confidence | Interval of Mean |  |
| VMs | schemes |  | Lower bound | Upper bound |
| VM\_5 | HEFT | 68.9267 | 57.0728 | 80.7805 |  |
|  | DHEFT | 62.4500 | 55.7452 | 69.1548 |  |
|  | RHEFT | 86.1133 | 85.1471 | 87.0795 |  |
| VM\_10 | HEFT | 37.6900 | 27.1016 | 48.2784 |  |
|  | DHEFT | 33.2900 | 30.5268 | 36.0532 |  |
|  | RHEFT | 69.5767 | 66.2300 | 72.9233 |  |
| VM\_20 | HEFT | 16.6583 | 11.7654 | 21.5513 |  |
|  | DHEFT | 22.3467 | 19.0820 | 25.6114 |  |
|  | RHEFT | 54.0567 | 52.4988 | 55.6145 |  |

Table 5a

Error Table for Speedup (95% CI) (Tasks = 50).

|  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| No. of | Scheduling | Mean | 95% Confidence | interval of mean |  |  | No. of | Scheduling | Mean | 95% Confidence | interval of mean |  |
| VMs | schemes |  | Lower bound | Upper bound |  |  | VMs | schemes |  | Lower bound | Upper bound |
| VM\_5 | HEFT | 378.410 | 139.200 | 617.610 |  |  | VM\_5 | HEFT | 1.8402 | 0.758 | 2.9223 |  |
|  | DHEFT | 292.730 | 208.240 | 377.220 |  |  |  | DHEFT | 1.849 | 1.3284 | 2.3695 |  |
|  | RHEFT | 180.960 | 170.500 | 191.420 |  |  |  | RHEFT | 2.8178 | 2.6552 | 2.9804 |  |
| VM\_10 | HEFT | 607.810 | 357.620 | 857.990 |  |  | VM\_10 | HEFT | 3.5957 | 3.2953 | 3.8962 |  |
|  | DHEFT | 206.270 | 147.910 | 264.640 |  |  |  | DHEFT | 2.5922 | 1.9846 | 3.1997 |  |
|  | RHEFT | 142.200 | 130.360 | 154.050 |  |  |  | RHEFT | 0.9882 | 0.4844 | 1.492 |  |
| VM\_20 | HEFT | 666.740 | 353.700 | 979.780 |  |  | VM\_20 | HEFT | 5.2757 | 4.7438 | 5.8075 |  |
|  | DHEFT | 159.700 | 144.580 | 174.810 |  |  |  | DHEFT | 3.2074 | 2.8945 | 3.5204 |  |
|  | RHEFT | 97.2108 | 720 | 107.700 |  |  |  | RHEFT | 0.9869 | 0.2704 | 1.7033 |  |

Table 5b

Error table for speedup (95% CI) (Tasks = 100).

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| No. of | Scheduling | Mean | 95% Confidence | interval of mean |  |
| VMs | schemes |  | Lower bound | Upper bound |
| VM\_5 | HEFT | 1.7667 | 1.0291 | 2.5042 |  |
|  | DHEFT | 1.9417 | 1.5414 | 2.3420 |  |
|  | RHEFT | 3.5550 | 3.4555 | 3.6545 |  |
| VM\_10 | HEFT | 1.5950 | 0.7791 | 2.4109 |  |
|  | DHEFT | 1.9233 | 1.7055 | 2.1411 |  |
|  | RHEFT | 4.4567 | 4.0773 | 4.8360 |  |
| VM\_20 | HEFT | 1.2633 | 0.5479 | 1.9787 |  |
|  | DHEFT | 3.0633 | 2.4878 | 3.6388 |  |
|  | RHEFT | 7.1200 | 6.7878 | 7.4522 |  |

Table 6a

Error table for energy consumption (CI = 95%) (Tasks = 50).

exhibits consistent utilization across different availability levels of resources. The resultant algorithm is thus named Robust HEFT. The error analysis presented at CI = 95%, justifies the nomenclature of Robust HEFT (RHEFT). The extension of RHEFT which approximates ready queue even better than RHEFT is future scope of this work.

References

1. [Ding Y, Qin X, Liu L, Wang T. Energy efficient scheduling of virtual machines in](http://refhub.elsevier.com/S1110-8665(17)30036-1/h0005) [cloud with deadline constraint. Future Gener Comput Syst 2015](http://refhub.elsevier.com/S1110-8665(17)30036-1/h0005).
2. [Kliazovich D, Pecero JE, Tchernykh A, Bouvry P, Khan SU, Zomaya AY. CA-DAG:](http://refhub.elsevier.com/S1110-8665(17)30036-1/h0010) [communication aware directed acyclic graphs for modeling cloud computing](http://refhub.elsevier.com/S1110-8665(17)30036-1/h0010) [applications. In: IEEE 6th international conference on cloud computing. p.](http://refhub.elsevier.com/S1110-8665(17)30036-1/h0010) [277–84](http://refhub.elsevier.com/S1110-8665(17)30036-1/h0010).
3. [Smanchat S, Viriyapant K. Taxonomies of workflow scheduling problem and](http://refhub.elsevier.com/S1110-8665(17)30036-1/h0015) [techniques in the cloud. Future Gener Comput Syst 2015;52:1–12](http://refhub.elsevier.com/S1110-8665(17)30036-1/h0015).
4. [Arabnejad H, Barbosa JG, Prodan R. Low-time complexity budget–deadline](http://refhub.elsevier.com/S1110-8665(17)30036-1/h0020) [constrained workflow scheduling on heterogeneous resources. Future Gener](http://refhub.elsevier.com/S1110-8665(17)30036-1/h0020) [Comput Syst 2016;55:29–40](http://refhub.elsevier.com/S1110-8665(17)30036-1/h0020).
5. Jena RK. Multi objective task scheduling in cloud environment using nested PSO framework; 2015.

No. of VMs

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | | | Lower bound | Upper bound |  |
| VM\_5 | HEFT | 83482.3750 | 33925.5424 | 133039.2076 |  |
|  | DHEFT | 63330.4750 | 46530.7385 | 80130.2115 |  |
|  | RHEFT | 42424.8083 | 40260.6029 | 44589.0137 |  |
| VM\_10 | HEFT | 115523.8750 | 70293.2319 | 160754.5181 |  |
|  | DHEFT | 41501.3583 | 30849.3341 | 52153.3825 |  |
|  | RHEFT | 30748.2125 | 28268.3684 | 33228.0566 |  |
| VM\_20 | HEFT | 122428.1688 | 66237.3683 | 178618.9692 |  |
|  | DHEFT | 30317.0854 | 27541.8024 | 33092.3684 |  |
|  | RHEFT | 19778.6750 | 17671.8589 | 21885.4911 |  |

Table 6b

Scheduling schemes

Mean 95% Confidence interval of mean

1. [Lee YC, Han H, Zomaya AY, Yousif M. Resource-efficient workflow scheduling](http://refhub.elsevier.com/S1110-8665(17)30036-1/h0030) [in clouds. Knowledge-Based Syst 2015;80:153–62](http://refhub.elsevier.com/S1110-8665(17)30036-1/h0030).
2. [Topcuoglu H, Hariri S, Wu MY. Performance-effective and low-complexity task](http://refhub.elsevier.com/S1110-8665(17)30036-1/h0035) [scheduling for heterogeneous computing. IEEE Trans Parallel Distrib Syst](http://refhub.elsevier.com/S1110-8665(17)30036-1/h0035) [2002;13(3):260–74](http://refhub.elsevier.com/S1110-8665(17)30036-1/h0035).
3. [Juve G, Chervenak A, Deelman E, Bharathi S, Mehta G, Vahi K. Characterizing](http://refhub.elsevier.com/S1110-8665(17)30036-1/h0050) [and profiling scientific workflows. Future Gener Comput Syst 2013;29](http://refhub.elsevier.com/S1110-8665(17)30036-1/h0050) [(3):682–92](http://refhub.elsevier.com/S1110-8665(17)30036-1/h0050).
4. [Juve G, Deelman E. Scientific workflows in the cloud. In: Grids, clouds and](http://refhub.elsevier.com/S1110-8665(17)30036-1/h0055) [virtualization. London: Springer; 2011. p. 71–91](http://refhub.elsevier.com/S1110-8665(17)30036-1/h0055).
5. [Lee K, Paton NW, Sakellariou R, Deelman E, Fernandes AA, Mehta G. Adaptive](http://refhub.elsevier.com/S1110-8665(17)30036-1/h0060) [workflow processing and execution in pegasus. Concurrency Comput: Pract](http://refhub.elsevier.com/S1110-8665(17)30036-1/h0060) [Exp 2009;21(16):1965–81](http://refhub.elsevier.com/S1110-8665(17)30036-1/h0060).
6. [Rahman M, Hassan R, Ranjan R, Buyya R. Adaptive workflow scheduling for](http://refhub.elsevier.com/S1110-8665(17)30036-1/h0065) [dynamic grid and cloud computing environment. Concurr Comput: Pract Exp](http://refhub.elsevier.com/S1110-8665(17)30036-1/h0065) [2013;25(13):1816–42](http://refhub.elsevier.com/S1110-8665(17)30036-1/h0065).
7. [Abrishami S, Naghibzadeh M, Epema DH. Deadline-constrained workflow](http://refhub.elsevier.com/S1110-8665(17)30036-1/h0070) [scheduling algorithms for Infrastructure as a service clouds. Future Gener](http://refhub.elsevier.com/S1110-8665(17)30036-1/h0070) [Comput Syst 2013;29(1):158–69](http://refhub.elsevier.com/S1110-8665(17)30036-1/h0070).
8. [Abdelkader DM, Omara F. Dynamic task scheduling algorithm with load](http://refhub.elsevier.com/S1110-8665(17)30036-1/h0075)

Error table for energy consumption (CI = 95%) (Tasks = 100).

[balancing for heterogeneous computing system, Egypt. Inform J 2012;13](http://refhub.elsevier.com/S1110-8665(17)30036-1/h0075)

No. of VMs

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | | | Lower bound | Upper bound |  |
| VM\_5 | HEFT | 157869.04170 | 94555.52180 | 221182.56150 |  |
|  | DHEFT | 127780.85000 | 98814.82020 | 156746.87980 |  |
|  | RHEFT | 72790.61670 | 70819.00140 | 74762.23200 |  |
| VM\_10 | HEFT | 173052.91830 | 71934.31160 | 274171.52510 |  |
|  | DHEFT | 113405.76330 | 99901.86710 | 126909.65960 |  |
|  | RHEFT | 55339.76000 | 50227.43340 | 60452.08660 |  |
| VM\_20 | HEFT | 214792.14000 | 66493.52670 | 363090.75330 |  |
|  | DHEFT | 69277.96170 | 57129.58030 | 81426.34300 |  |
|  | RHEFT | 32730.26170 | 31171.73010 | 34288.79320 |  |

Scheduling schemes

Mean 95% Confidence interval of mean

[(2):135–45](http://refhub.elsevier.com/S1110-8665(17)30036-1/h0075).

1. [Bittencourt LF, Madeira ERM. HCOC: a cost optimization algorithm for](http://refhub.elsevier.com/S1110-8665(17)30036-1/h0080) [workflow scheduling in hybrid clouds. J Internet Services Appl 2011;2](http://refhub.elsevier.com/S1110-8665(17)30036-1/h0080) [(3):207–27](http://refhub.elsevier.com/S1110-8665(17)30036-1/h0080).
2. [Sakellariou R, Henan Z. A hybrid heuristic for DAG scheduling on](http://refhub.elsevier.com/S1110-8665(17)30036-1/h0085) [heterogeneous systems. In: Parallel and distributed processing symposium,](http://refhub.elsevier.com/S1110-8665(17)30036-1/h0085) [proceedings of 18th international. IEEE; 2004](http://refhub.elsevier.com/S1110-8665(17)30036-1/h0085).
3. [Sih GC, Lee EA. A compile-time scheduling heuristic for interconnection-](http://refhub.elsevier.com/S1110-8665(17)30036-1/h0090) [constrained heterogeneous processor architecture. IEEE Trans Parallel Distrib](http://refhub.elsevier.com/S1110-8665(17)30036-1/h0090) [Syst 1993;4(2):175–87](http://refhub.elsevier.com/S1110-8665(17)30036-1/h0090).
4. [Chen H, Wang F, Helian N, Akanmu G. User-priority guided Min-Min](http://refhub.elsevier.com/S1110-8665(17)30036-1/h0095) [scheduling algorithm for load balancing in cloud computing. In: National](http://refhub.elsevier.com/S1110-8665(17)30036-1/h0095) [conference on parallel computing technologies (PARCOMPTECH). IEEE; 2013.](http://refhub.elsevier.com/S1110-8665(17)30036-1/h0095)

[p. 1–8](http://refhub.elsevier.com/S1110-8665(17)30036-1/h0095).

1. [Li Jiayin et al. Online optimization for scheduling preemptable tasks on IaaS](http://refhub.elsevier.com/S1110-8665(17)30036-1/h0100) [cloud systems. J Parallel Distrib Comput 2012;72(5):666–77](http://refhub.elsevier.com/S1110-8665(17)30036-1/h0100).
2. [Yu J, Buyya R, Ramamohanarao K. Workflow scheduling algorithms for grid](http://refhub.elsevier.com/S1110-8665(17)30036-1/h0105) [computing. In: Metaheuristics for scheduling in distributed computing](http://refhub.elsevier.com/S1110-8665(17)30036-1/h0105) [environments. Berlin, Heidelberg: Springer; 2008. p. 173–214](http://refhub.elsevier.com/S1110-8665(17)30036-1/h0105).
3. Conclusion and future discussion

A hybrid planning algorithm, RHEFT is presented in this work. RHEFT is hybrid of HEFT and a novel scheduling algorithm for inde- pendent tasks called Interior Scheduling. In RHEFT, a conversion of sequential ranked tasks into set of independent ranked tasks is per- formed and better approximates the set of tasks in the ready queue. Using IS scheduling algorithm a HEFT planning algorithm from global allocation got evolved into RHEFT planning algorithm resulted with sub-local allocation. This evolution of HEFT to RHEFT

1. [Chen W, Deelman E. Workflowsim: a toolkit for simulating scientific](http://refhub.elsevier.com/S1110-8665(17)30036-1/h0110) [workflows in distributed environments. In: E-science (E-Science), 2012 IEEE](http://refhub.elsevier.com/S1110-8665(17)30036-1/h0110) [8th international conference on. IEEE; 2012. p. 1–8](http://refhub.elsevier.com/S1110-8665(17)30036-1/h0110).
2. [Bajaj R, Agrawal DP. Improving scheduling of tasks in a heterogeneous](http://refhub.elsevier.com/S1110-8665(17)30036-1/h0120) [environment. IEEE Trans Parallel Distributed Syst 2004;15:107–18](http://refhub.elsevier.com/S1110-8665(17)30036-1/h0120).
3. [Fahringer T, Jugravu A, Pllana S, Prodan R, Seragiotto C, Truong HL. ASKALON: a](http://refhub.elsevier.com/S1110-8665(17)30036-1/h0125) [tool set for cluster and Grid computing. Concurr Comput: Pract Exp 2005;17](http://refhub.elsevier.com/S1110-8665(17)30036-1/h0125) [(2):143–69](http://refhub.elsevier.com/S1110-8665(17)30036-1/h0125).
4. Blaha P, Schwarz K, Madsen GKH, Kvasnicka D, Luitz J. wien2k. An augmented plane wave+ local orbitals program for calculating crystal properties; 2001.
5. [Rutschmann P, Theiner D. An inverse modelling approach for the estimation of](http://refhub.elsevier.com/S1110-8665(17)30036-1/h0140) [hydrological model parameters. J Hydroinform 2005](http://refhub.elsevier.com/S1110-8665(17)30036-1/h0140).