I.J. Information Technology and Computer Science, **2018, 1, 68-75**
Published Online January 2018 in MECS (http://www.mecs-press.org/)
DOI: 10.5815/ijitcs.2018.01.08

Time Effective Workflow Scheduling using Genetic Algorithm in Cloud Computing

Rohit Nagar
Dr. B.R Ambedkar National Institute of Technology, Department of CSE, Jalandhar, 144011, India
E-mail: rohitnagarn92@gmail.com

Deepak K. Gupta and Raj M. Singh
Dr. B.R Ambedkar National Institute of Technology, Department of CSE, Jalandhar, 144011, India
E-mail: guptadk@nitj.ac.in, rm_singh14@yahoo.com

Received: 26 August 2017; Accepted: 07 November 2017; Published: 08 January 2018

***Abstract*—**Cloud computing is service based technology on internet which facilitates users to access plenty of resources on demand from anywhere and anytime in a metered manner i.e. pay per usage without paying much heed to the maintenance and implementation details of application. As cloud technology is evolving day by day it is being confronted by numerous challenges, such as time and cost under deadline constraints. Research work done so far mainly focused on reducing cost as well as execution time. In order to minimize cost and execution time previously existing workflow scheduling model known as predict earliest finish time is used. In this research work we have proposed a new PEFT genetic algorithm approach to further reduce the execution time on this model. A strategy is developed to let GA focus on to optimize chromosomes objective to get best suitable mutated children. After obtaining a feasible solution, the genetic algorithm focuses on optimizing the execution time. Experimental results show that our algorithm can find better solution within lesser time.

***Index Terms*—**Cloud computing, Task Scheduling, Earliest finish time, Genetic Algorithm, Makespan.

I. Introduction

Over the past few years, cloud computing has become a trending research topic for scientific research. Cloud Computing ensures reliable, scalable, pay-per-use, customized and dynamic computing environments for the end-users. Cloud computing provides many facilities for computing services, centralized servers, on-demand self-service, huge storage, databases, broad network access, rapid elasticity software over the Internet [1]. Cloud computing is nothing but the way of using a network of remotely located servers hosted all over the internet for storing, processing data and managing data, instead of using a local server or a personal computer. The companies which offers such computing services are known as cloud providers. They may charge for their cloud computing services based on usage, it is very much similar to how you pay bills for water consumption and electricity consumption at your home [2].

The cloud computing services are classified in three ways named as Software as a service (SaaS), Platform as a service (PaaS), and Infrastructure as a service (IaaS) [3]. SaaS applications are deployed over the internet for the clients in a single instance multi-tenant model and are accessed by various devices having internet capability through web browser or program interface [4]. It is one of the fastest growing services in cloud. PaaS is a development tool which provides a collaborative platform that consists of database system, operating system, programming stacks and hardware for creating business applications easily and quickly without much cost. IaaS is a way to provide computing infrastructural resources (VMs) instead of purchasing them online in multi- tenant fashion on pay per usage basis.

The problem of mapping task to their resources belongs to class of NP problem. There is no known algorithm exists which can generate the optimal solution with in feasible time period. Solutions based on the exhausted search are practically not possible. Overhead of generating scheduling is very high. PEFT algorithm is the improved version of HEFT algorithm. PEFT algorithm gives the best suitable schedule with less makespan time and less communication cost.

In this article, we discussed about scheduling in cloud computing environment. The introduction is summarized into Section 1. Related work is shown in Section 2. Problem is formulated in Section 3. The present work is explained in Section 4. Experimental analysis is included in Section 5 and Section 6 sums up the paper.

II. Related Work

For better understanding of workflow scheduling we went through several research papers. The researchers proposed many algorithms but none of them is best. Because there are various parameters which are considered to make the algorithm best among all. In 2015, T. Bridi et al presented a constraint programming technique based scheduler [5] which improves the results obtained from commercial schedulers. It was implemented to make it usable on real life high performance computing setting. The scheduler works well in both simulated and real HPC environment. This scheduling algorithm ensures robustness, flexibility and scalability [5].

In 2016, J. meena et al proposed a meta –heuristic cost effective genetic algorithm (CEGA) [6] that reduces the execution cost of the workflow while meeting the deadline in cloud computing. It also covers some big issues such as performance variation and booting time of virtual machines. The simulation experiments conducted on four scientific workflows (Montage, LIGO, CyberShake, Epigenomics) and exhibited better performance than current state of art algorithms. The proposed CEGA algorithm shows the highest hit rate for deadline constraint.

In 2015, A.verma et al used Bi-criteria Priority based Particle Swarm Optimization (BPSO) [7]. He proposed scheduling algorithm for workflow tasks over the cloud processors under deadline and budget constraints. To each workflow's task a priority is given using bottom level technique. It gave reduced execution cost of schedule as compared to state of art algorithm under the same deadline and budget constraint while considering the load on resources too.

In 2014, Arabnejad et al. introduced a list scheduling algorithm [8] having less time complexity than HEFT algorithm. Authors proposed a list based scheduling technique named PEFT for heterogeneous distributed computing which gives better results than HEFT in terms of makespan. It has the same time complexity as that of HEFT. It consists of two phases, task prioritizing phase and processor selection phase. It can be assumed as improved version of HEFT algorithm. This algorithm uses a matrix called Optimistic Cost Table (OCT). OCT indicates the minimum time required for processing all the tasks which lies on the longest path from the current task to the end task. In task prioritization, task priority is calculated by cumulative OCT. Optimistic EFT is calculated to assign a processor for a task.

In 2002, H. Topcuoglu et al. presented an algorithm [9] called HEFT which provides solution for scheduling problem in DAG on heterogeneous systems. Working of HEFT algorithm takes place in two phases: task prioritizing phase and processor selection phase. In processor selection phase it minimizes the earliest finish time of the child task of each and every selected task. They has proposed two methods for scheduling a workflow task in heterogeneous environment named as HEFT and Critical path on a Processor. They work on the same line with slight differences. The latter uses the critical path and they allocate tasks on the critical path to the processor which will give minimum EFT. HEFT is better than other algorithms in same domain because of its high efficiency in terms of makespan and robust nature.

In 2011, Daoud et al. used the LDCP list based heuristic to generate the initial population [10]. Longest Dynamic critical path is a list based tri-phase heuristic. H2GS combines LDCP and GA. It uses the high quality schedule generated by LDCP as a seed for the initial population which is exploited by the customized genetic algorithm. The schedule generated by the LDCP is near an optimal schedule and when such a schedule is given as an input to the genetic algorithm, the algorithm will converge faster. It uses two-dimensional (2-D) chromosomes for representation and customized operators for searching the problem space. It has shown significant improvement in terms of speedup and normalized schedule length, over HEFT and Dynamic Level Scheduling in heterogeneous distributed system.

In 2012, Kaur et al. proposed a new Modified Genetic Algorithm for scheduling the tasks in private cloud for minimizing the makespan and cost [11]. In MGA, initial population is generating using SCFP (Smallest cloudlet to Fastest Processor), LCFP (Longest cloudlet to Fastest Processor) and 8 random schedules. Two-point crossover and simple swap are used. This gives the good performance under the heavy loads.

In 2012, Ahmad et al. proposed an effective genetic algorithm called PEGA [12], which is capable of providing the optimal results in large space with less time complexity. The direct chromosome representation is used having two parts. The right half is made by the b-level (upward rank) which give the better results in terms schedule length when compared with randomly generated population. Two fold cross over is used in which single and two-point crossover are executed one after the other in order to enhance the quality and the convergence speed of the solution.

In 2014 Shekar singh and Mala Kalra et al. proposed a genetic algorithm based approach in which initial population is generated with advance version of Max-Min by which we can get more optimized results in terms of makespan. Authors proposed Modified Genetic Algorithm. When experiments were conducted on various data sets, MGA exhibited better performance. Since scheduling of tasks is the key issue in cloud computing author used GA in the research work. In standard genetic algorithm initial population is randomly generated which doesn't produce efficient results. Hence author modified genetic algorithm. Here initial population is generated using Enhanced Max Min algorithm and then this population is given to GA to further optimize. Results show that MGA performs better than standard genetic algorithm [13].

In 2012 Saima Gulzar, Ahmad, Ehsan Ullah Munir et al. proposed an effective genetic algorithm called PEGA, which is capable of providing the optimal results in large space with less time complexity. The direct chromosome representation is used having two parts. The right half is made by the b-level (upward rank) which give the better results in terms schedule length when compared with randomly generated population. Two fold cross over is used in which single and two-point crossover are executed one after the other in order to enhance the quality and the convergence speed of the solution. The author has concluded that the PEGA provides the better schedule with smaller makespan and low time complexity [14]

In 2012 Chuan Wang, Jianhua Gu Yunlan Wang et al. presented a hybrid approach which uses a combination of successor concerned list based heuristic and a genetic algorithm. First phase is the seeding method for GA, to generate the initial population by the schedule given by the SCLS (Successor concerned list heuristic). In SCLS, Priority list of the task was formed using the upward rank. In the second phase the good quality schedule generated by the above phase is fed into the genetic algorithm. The authors had proved that HSCGS give better results than HEFT and DLS (Dynamic Level Scheduling) [15]

In 2013 Saeid Abrishami, Mahmoud Naghibzadeh et al. proposed two algorithms for workflow scheduling based on the Partial Critical path to find the optimal solution in terms of minimal cost subject to the defined deadline constraints. IC-PCP (Iaas cloud partial critical path) tries to schedule the tasks on partial critical path by allocating them to the available instances of the service before its latest finish time. ICPCP2 (Iaas cloud partial critical path with deadline distribution) uses the new method for path assigning policy and planning is done such that the remaining time of the available instance is used first to execute the task before its sub-deadline, rather than starting a new instance of the service. [16]

In 2014 A. Verma and Sakshi Kaushal et al. proposed three hybrid genetic algorithms that uses the schedule generated by the bottom level and top level as an initial population to minimize the execution cost of the schedule while following the deadline constraint. BGA (Bottom-level GA) uses the bottom level in descending order to assign the priorities, while TGA (Top-level GA) consider the top-level in increasing order. BTGA (Bottom level and top level) which uses both level has a better performance than the other two [17].

In 2012 Beibei Zhu, Hongze Qiu et al. proposed a modified genetic algorithm by improving genetic operators. Experimental studies show that the modified genetic algorithm gives optimal solution. In this paper, author presented an efficient genetic algorithm for DAG scheduling in grid system. By proposing new fitness function and applying new genetic operators, the new proposed GA can obtain an optimal solution [18].

In 2010 S. Selvarani and G. Sudha Sadhasivam et al. proposed an algorithm based on costs with user task grouping. Proposed method uses an improved cost-based scheduling algorithm for efficient mapping of tasks to the resources in cloud. This scheduling algorithm measures both resource cost and computation performance. It improves the computation /communication ratio by grouping the user's tasks according to a particular cloud resource's processing capability and sends the grouped jobs to the resources [19].

The conclusion of above research and analysis is that there is no exact algorithm can be proposed because when parameters get changed the algorithm is also has to be changed.

III. Problem Formulation

Nowadays large no. of business applications are implemented by workflows. Workflows are denoted in terms of a Directed Acyclic Graph (DAG), G= (T, E), where T is the set of tasks and E is the set of edges between the tasks. We cannot initialize any task until all of its predecessor tasks have completed. Workflow scheduling is nothing but a mapping of every task of workflow onto a best suitable resource while meeting the user's requirements considering task dependencies. Cloud is a platform for workflow execution since it has advantages like scalability, durability, on demand self-service, broad network access and pay per use model. Since it has large number of tasks and virtual machines, workflow scheduling is one of the major issues in cloud computing. We defined our problem here as: Mapping of tasks of a workflow to available resources (VMs) in cloud computing environment to minimize execution time while considering deadlines that is.

Minimize ET

Considering ET $\leq$D

where ET is the execution time (makespan) and D is the deadlines of the tasks of given workflow.

IV. Present Work

To implement our research work we need Intel core i5 machine with 1 TB HDD and 4 GB RAM on Windows 10 OS, NetBeans with Java and workflowsim simulator toolkit. Here output of PEFT algorithm for a DAG is given as input into GA as initial population. Our proposed algorithm is a combination of PEFT algorithm and GA for workflow scheduling in cloud computing environment. The algorithm reduces the execution time (makespan) while maintaining the deadline constraint.

Following are our objectives:

1. To study the existing task scheduling approaches for heterogeneous system.
2. To propose an algorithm for scheduling workflows in cloud environment aiming to minimize execution time.
3. To evaluate the proposed solution by comparing it with the existing workflow scheduling approaches.

The proposed work is divided into two steps:

1. Generating a high quality seed for inputting to GA using PEFT algorithm.
2. Obtaining an optimized schedule by GA in such a way that it will give minimal execution time in milliseconds and will finish execution of workflow before the deadline.

Steps of the Proposed Methodology:

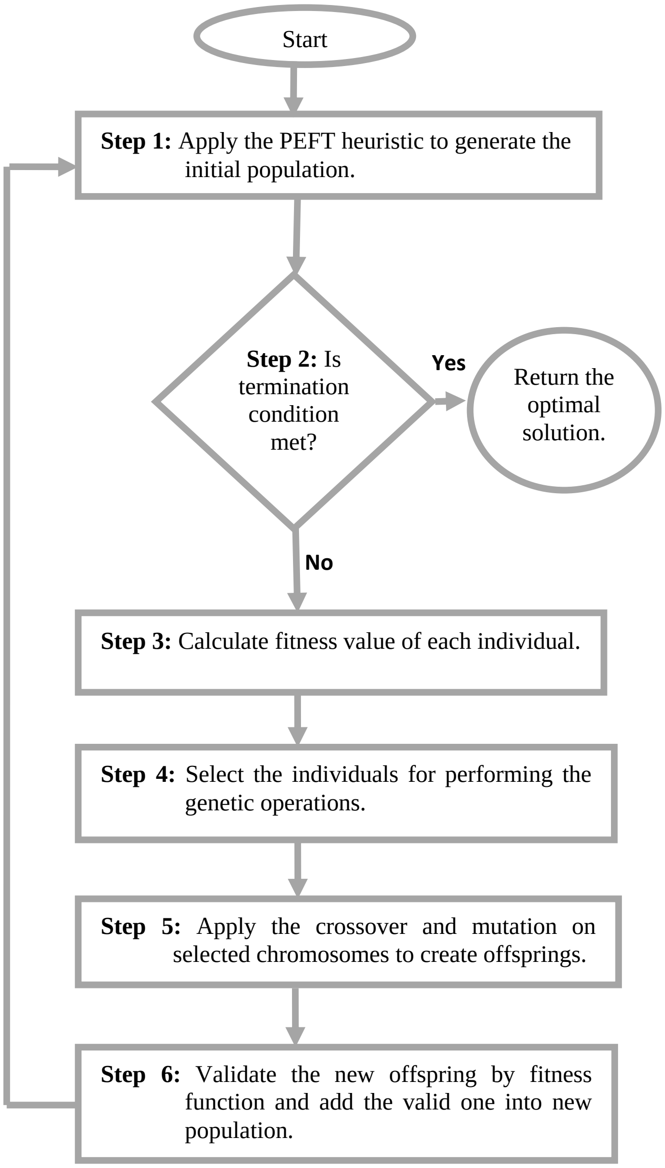
Start
Step 1: Apply the PEFT heuristic to generate the initial population.
Step 2: Is termination condition met?
Yes
Return the optimal solution.
No
Step 3: Calculate fitness value of each individual.
Step 4: Select the individuals for performing the genetic operations.
Step 5: Apply the crossover and mutation on selected chromosomes to create offsprings.
Step 6: Validate the new offspring by fitness function and add the valid one into new population.

Fig.1. Flow chart of working model of proposed scheme.

Step 1: Generate a best suitable seed using the PEFT algorithm

Step 1.1 Compute the OCT table. The OCT table is nothing but a matrix in which rows represent tasks and columns represent virtual machines. The OCT value is calculated by the equation given below. Algorithm is applied recursively in backward direction. And hence we obtain the cost for executing all the children tasks of a current task until it reaches the end task.

$$\mathrm{OCT}\ (t_i, p_k) = \max\ t_j \in \mathrm{succ}\ (t_i)\ [\min\ p_w \in P\ \{\mathrm{OCT}\ (t_j,p_w) + w(t_j\ , p_w) + \overline{ci, j})\}] \quad \text{where } \overline{ci, j} = 0 \text{ if } p_w = p_k \qquad (1)$$

Step 1.2: Calculate OCT for every node. OCT defines the rank of every node or task (rank_{oct}) as in Eq 3. Tasks are arranged in the list on the basis of descending order of the rank_{OCT}.

$$\mathrm{Rank}_{oct} \qquad (2)$$

Step 1.3: Earliest Finish Time (EFT) is calculated using the given equation to allocate a task for the resource (processor).

$$O_{EFT}\ (t_i, p_j) = \mathrm{EFT}\ (t_i, p_j)\ + \mathrm{OCT}\ (t_i, p_j) \qquad (3)$$

Step 1.4: Task is assigned to the processor (VM) which gives minimum OEFT.

Step 1.5: Repeat steps 1.4 and 1.5 if the list is not empty, otherwise return the best schedule in terms of makespan.

Step 2: If the termination condition is met than return the solution otherwise repeat steps 3 to step 6.

Step 3: And hence a best suitable schedule is generated and is given to the genetic algorithm as input. The chromosomes (individuals among the population) are encoded using direct representation. The quality of all the feasible solutions is checked by the fitness function. The fitness fu~~nction~~ will ensure that the solution has the minimum cost and is completed within the deadline.

Step 4: Select the chromosomes for applying genetic operations by using the binary tournament selection technique.

Step 5: On the selected chromosomes apply the crossover and mutation genetic operators to produce the new children (generation).

Step 6: Validate the new children by the fitness function and add the good quality off-springs (valid) into the new population.

Working of Proposed Scheme using Genetic Algorithm

```
1. P ← initialize population() by PEFT // P = population
2. W ← 0                          // W = New population
3. PF ← Evaluate Fitness such that Execution Time should be less than Deadline
4. Choose two chromosomes with minimum makespan (min1 & min2)
5: Parents← Selection from (PF) on the basis of makespan
6: Offspring ← Crossover ( PFc, Parents)
                                  //PFc=crossover probability
7: Offspring ← Mutation( PFm, Offspring)
                                  //PFm= mutation probability
8: Evaluate fitness (Offspring)
9: Repeat steps 5 to 8 for remaining chromosomes in the population PF and obtain offsprings
10: Insert (Offsprings, W) End for
11:T ← PF ∪ W//merge new Offsprings with population
12: Ranking of offspring on the basis of deadline (T)
13: P ←select best individuals on the basis of minimum makespan time of (T)
14: End while
15: Return P which contains single best schedule.
```

- **Initial Population:** Initial population comes in first step, the initial population is initialized using the PEFT algorithm. The generated strings are known as chromosomes (P).
- **Evaluate Fitness of each Individual:** For an obtained solution we should be able to evaluate its quality which can be done by using fitness function. Fitness function is described using deadline concepts. The chromosomes obtained by PEFT algorithm are tested as per their deadlines. The chromosomes which meet deadline are added in to next generation population (PF).
- **Selection:** Selection plays a major role in improving the performance of any approach by selecting high quality chromosomes for the next operations. From the population (PF) the chromosomes which meets deadlines with minimum makespan two chromosomes are selected as parents.
- **Crossover:** The role of exchanging one part of other chromosomes in such a manner that the GA generates new chromosomes from previous generation (PF) by interchanging part of them. Crossover is done on the selected chromosomes to obtain crossover children.
- **Mutation:** The main purpose of mutation is to introduce a new chromosome that doesn't exist in existing population. After getting crossover mutated children are obtained.
- **Evaluate fitness of each offspring's:** The mutated children are added in the population W where offspring are evaluated as per fitness function to get their fitness. If the offspring's fitness is less in comparison of chromosomes with greatest fitness in the population PF, it will replace them with the nodes with the greatest fitness.

At last, the new offspring will be added in new generation (T). This process can repeat until we find a good result. In proposed algorithm we are repeating this process 5 times. Hence all best solutions are stored in an array W and minimum makespan chromosome is selected as final solution and tasks are allocated as per the best solution.

V. Experimental Analysis

To evaluate the performance of our proposed algorithm PEFTGA, we are performing the simulations using WorkflowSim. We consider few scientific workflows from different domains: Montage, Inspiral and Peft paper. These workflows have different structural properties and different data and computational requirements. Genetic algorithm is taken as the baseline algorithm.

Our PEFTGA algorithm is tested by:

1. Varying the number of tasks in datasets
2. Varying the inter dependencies of task in the dataset

Experimental results show that for various datasets our proposed algorithm that is Predict Earliest Finish Time with Genetic Algorithm exhibits a very good performance.

Table 1 contains values obtained from different datasets. It shows that our proposed algorithm PEFTGA gives better makespan time as compared to standard genetic algorithm. The following figures shows the study of makespan time of various scientific workflows in milliseconds.

Table 1. GA Parameters

Parameter	Value
Number of Processors	4
Number of Iterations	5
Crossover Type	Two-Point Crossover
Crossover Probability	0.3
Mutation Type	Simple Swap
Mutation Probability	0.3
Termination Condition	Number of Iterations

Table 2. Improvement in makespan comparing GA & PEFTGA

Sr. No.	Datasets	GA	PEFTGA	(GA-PEFTGA)/GA
1	Peft_Paper	18.64ms	15.69ms	15.82%
2	Montage_25	33.02ms	19.22ms	41.79%
3	Montage_50	51.04ms	33.37ms	34.61%
4	Montage_100	69.02ms	83.87ms	14.43%
5	Inspiral_30	810.16ms	414.60ms	48.82%
6	Inspiral_50	1243.73ms	785.79ms	36.81%
7	Inspiral_100	1916.08ms	1875.41ms	2.12%

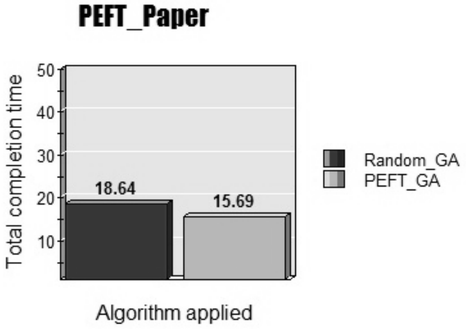

Fig.2. Peft_Paper

From figure 2, the proposed PEFTGA algorithm is applied on the dataset given in PEFT_PAPER [8] and result obtained shows that our algorithm gives 15.82% lesser makespan than original GA. The dataset in this paper have 10 inter dependent tasks with different execution time and cost as shown in PEFT_PAPER [8].

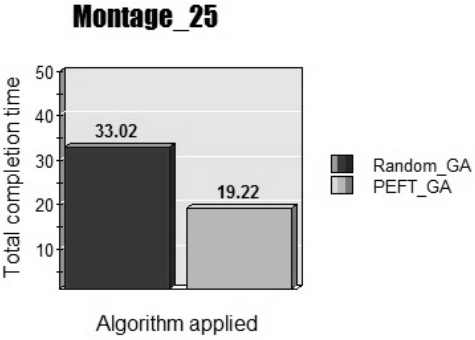

Fig.3. Montage_25

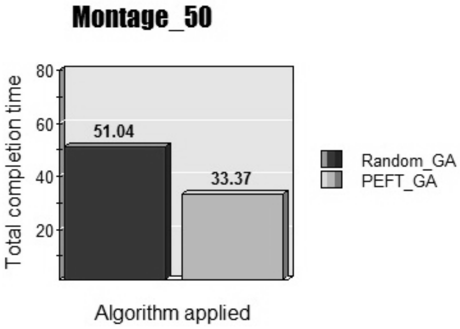

Fig.4. Montage_50

From figure 3, the proposed PEFTGA algorithm is applied on montage dataset with 25 inter dependent tasks with different execution cost and time. The result obtained gives 41.79% lesser makespan than original GA.

From figure 4, same PEFTGA algorithm is applied on Montage_50 dataset containing 50 inter dependent tasks with different execution time and cost and the result gives 34.61% lesser makespan than original GA. Here dependencies are of same type but number of tasks are more. It proves that performance of our algorithm not only depends on number of tasks but also depends on inter connectivity of tasks.

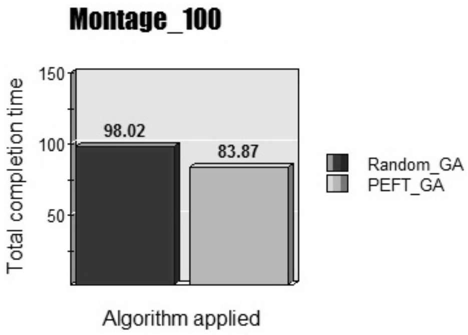

Fig.5. Montage_100

From figure 5, the proposed PEFTGA algorithm is applied on Montage_100 dataset which contains 100 tasks and is more complex in terms of inter dependencies and the result gives 14.43% lesser makespan than original GA. The proposed algorithm is tested using various datasets with different number of tasks with different inter connectivity as shown in the following figure.

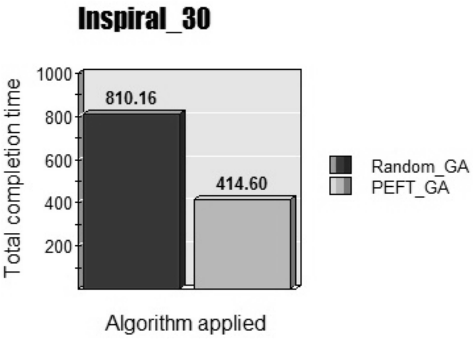

Fig.6. Inspiral_30

From figure 6, Inspiral_30 dataset containing 30 tasks with various dependencies the proposed PEFTGA algorithm is applied and the result gives 48.82% lesser makespan than original GA. As we increase the number of tasks improvement is lesser from the above results as compared to Montage_25 dataset with 25 tasks where there is more improvement. This improvement is because of variation in dependencies.

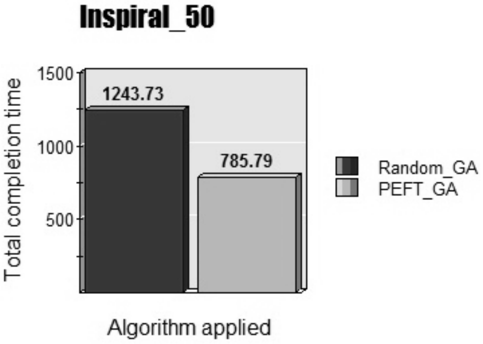

Fig.7. Inspiral_50

From figure 7, to understand that how number of tasks affects the result we used same type of Inspiral dataset with 50 inter connected tasks. And after applying the proposed PEFTGA algorithm for Inspiral_50 dataset it gives 36.81% lesser makespan than original GA which proves that our proposed algorithm is better than original genetic algorithm. It also shows less improvement as compared to improvement in inspiral_30 with 30 tasks due to varying number of tasks. In this dataset we can see great improvement because of less number of tasks. So by increasing the number of tasks we analyze the variation in the improvement in makespan.

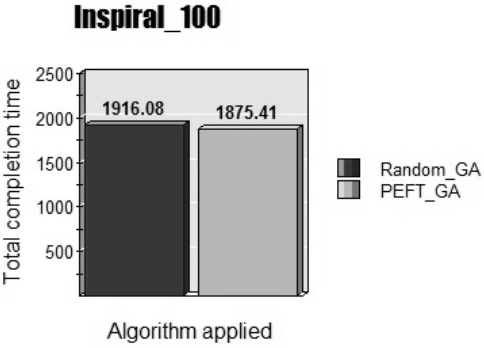

Fig.8. Inspiral_100

From figure 8, the proposed PEFTGA algorithm for Inspiral_100 dataset gives 2.12% lesser makespan than original GA. Since it is proved that the proposed algorithm gives less makespan, therefore a comparison with the baseline Genetic algorithm have been observed for completion time.

VI. Conclusion

Cloud computing has to deliver high performance in case of computing resources over the internet for workflows. Task scheduling is one of the major issues in cloud computing. To minimize this issue, we have used Predict earliest Finish Time Algorithm (PEFT) and genetic algorithm in our research work. Genetic algorithm produces inefficient results because of randomly generation of initial population. Hence we have modified it using Predict Earliest Finish Time Algorithm (PEFT) for generating its initial population. Our proposed algorithm Predict Earliest Finish Time Genetic Algorithm (PEFTGA) targets to reduce total completion time (makespan) of workflow and maximize resource utilization. We have compared our proposed algorithm with standard genetic algorithm. The results show that PEFTGA performs better scheduling of tasks on virtual machines in terms of makespan. The completion time (makespan) for the proposed PEFTGA algorithm is reduced by average 25% compared to standard GA. Since cost is proportional to the execution time hence cost of the proposed PEFTGA also gets reduced as compared to default GA. From the results, we can conclude that as compared to the original genetic algorithm PEFTGA shows the best performance for the static scheduling of directed acyclic graphs (DAGs) in heterogeneous systems.

As we can see that if number of tasks are more, improvement is less according to above figures and also if inter dependency among tasks in the datasets varies results also vary. Hence from our research work we can conclude that our proposed algorithm is best suitable for workflows with less number of tasks. So to overcome these problems in the future we would like to consider other parameters like execution costs, termination delay of virtual machines, energy consumption on data centers and data transfer costs between data centers, average makespan values and number of processors available etc. to make it more suitable for large size of datasets with complex inter dependencies of tasks.

References

[1] Mell, Peter, and Tim Grance, “The NIST definition of cloud computing”, Computer Security Division, Information Technology Laboratory, National Institute of Standards and Technology Gaithersburg, pp. 20-23, Year 2011.

[2] Kaur, P.D., I. “Unfolding the distributed computing paradigm”. International Conference on Advances in Computer Engineering, pp. 339-342 (2010).

[3] Gibson, Joel, Robin Rondeau, Darren Eveleigh, and Qing Tan.“Benefits and challenges of three cloud computing service models”, Fourth International Conference on Computational Aspects of Social Networks, IEEE, pp. 198-205, Year 2012.

[4] Silva, J.N., Veiga, L., Ferreira, P.: “Heuristics for Resource Allocation on Utility Computating Infrastructures,” 6th International Workshop on Middleware for Grid Computing, New York (2008).

[5] Bridi,T., Bartolini,A., Lombardi, M., Milano, M., and Benini, L., “A Constraint Programming Scheduler for Heterogeneous High-Performance Computing Machines,” pp. 1–14, (2016).

[6] Meena,J., Kumar, M., M., “Cost Effective Genetic Algorithm for Workflow Scheduling in Cloud Under Deadline Constraint,” vol. 4, pp. 5065–5082, (2016).

[7] Verma, A., “Cost Minimized PSO based Workflow Scheduling Plan for Cloud Computing,” I.J. Information Technology and Computer Science, 08, pp. 37–43, (2015).

[8] Arabnejad., H., and Barbosa G.J., “List Scheduling Algorithm for Heterogeneous Systems by an Optimistic Cost Table,” IEEE Transactions on Parallel and

Distributed Systems, Vol: 25(3) March (2014).

[9] Topcuoglu,H., Hariri, S., and Wu, M., “Performance-effective and low-complexity task scheduling for heterogeneous computing,” IEEE transaction on Parallel and Distributed System, vol. 13, no. 3, pp. 260–274, (2002).

[10] Daoud, I.M., and Kharma, N., “A Hybrid Heuristics-Genetic Algorithm for Task Scheduling in Heterogeneous Processor Networks” Journal of Parallel and Distributed Computing, Vol. 71(11), pp. 1518-1531, (2011).

[11] Kaur,S., and Verma, “An Efficient Approach to Genetic Algorithm for Task Scheduling in Cloud Computing Environment,” I.J. of Information Technology and Computer Science, pp. 74–79, (2012).

[12] Ahmad,G.S., Munir,U.E., Nisar, W., Avenue, Q., and Cantt, W., “PEGA: A Performance Effective Genetic Algorithm for Task Scheduling in Heterogeneous Systems,” IEEE 14th International Conference on High Performance Computing and Communications, pp. 1082–1087, (2012).

[13] Shekhar Singh, and Mala kalra, “Scheduling of Independent tasks in cloud computing using modified genetic algorithm”, IEEE, Pages: 565 - 569, DOI: 10.1109/CICN.2014.128, Year 2014.

[14] A Saima Gulzar Ahmad, Ehsan Ullah Munir, and Wasif Nisar, “A Performance Effective Genetic Algorithm for Task Scheduling in Heterogeneous Systems (PEGA)”, IEEE, Year: 2012, Pages: 1082 - 1087, DOI: 10.1109/HPCC.2012.158, Year 2012.

[15] Chuan Wang, Jianhua Gu,Yunlan Wang, and Tianhai Zhao, “Hybrid Heuristic-Genetic Algorithm for Task Scheduling in Heterogeneous Multi-Core System (HSCGS)” springer, DOI: 10.1007/978-3-642-33078-0_12, Year 2012.

[16] Saeid Abrishami, Mahmoud Naghibzadeh, “Deadline constrained Workflow Scheduling Algorithms for Infrastructure as a Service Clouds” IEEE, Volume 29, Issue 1, Pages 158-169, Year 2013.

[17] Amandeep Verma,and Sakshi Kaushal, “Deadline constraint heuristic-based Genetic Algorithm for Workflow Scheduling in Cloud” IEEE, Volume 5, Issue 2, Pages 96-106, Year 2014.

[18] Beibei Zhu, Hongze, “Modified genetic algorithm for DAG scheduling in grid systems” IEEE, Pages: 465 - 468, DOI: 10.1109/ICSESS.2012.6269505, Year 2012.

[19] S. Selvarani, and G. Sudha Sadhasivam, “Improved cost-based algorithm for task scheduling in cloud computing” IEEE, Pages: 1 - 5, DOI: 10.1109/ICCIC.2010.5705847, Year 2010.

Authors’ Profiles

Rohit Nagar lives in Delhi, India. He was born on November 9, 1988. He pursued his M. Tech at the Department of Computer Science & Engineering, Dr. B.R. Ambedkar National Institute of Technology, Jalandhar. Currently he is working in Infozech Software Pvt. Limited, New Delhi as Data Analyst. In research, his current interests includes Cloud Computing and Data Analytics.

Deepak Kumar Gupta is Associate Professor at the Department of Computer Science & Engineering, Dr. B. R. Ambedkar National Institute of Technology, Jalandhar. He has total 29 years of experience including teaching and industries. In research, his current interests includes Social Media, Data Analytics and Operating System.

Raj Mohan Singh received his M Tech degree in Computer Science and Engineering from Dr. B. R. Ambedkar National Institute of Technology, Jalandhar. His research interests includes Cloud Computing, Operating System, Data Analytics and Distributed Computing.

How to cite this paper: Rohit Nagar, Deepak K. Gupta, Raj M. Singh, "Time Effective Workflow Scheduling using Genetic Algorithm in Cloud Computing", International Journal of Information Technology and Computer Science(IJITCS), Vol.10, No.1, pp.68-75, 2018. DOI: 10.5815/ijitcs.2018.01.08