Potential Finish Time and Min-mean Algorithm for allocating Meta-Tasks on distributed Computational Grid

K.Sentamilselvan, G.K.Kamalam

Abstract*: Grid is widely distributed and promising technology that enables the integrated and heterogeneous resource sharing for solving computationally challenging scientific engineering problems. In a distributed grid environment, allocating the tasks to the available computing resources proves complex and it is an NP-Complete problem as resources are geo-graphically distributed. In this paper presents a new task scheduling algorithm, called Potential Finish Time Min-mean Task Scheduling Algorithm (PFTSA), to enhance the selection of the suitable resources, which is responsible for the scheduling process. The proposed algorithm (PFTSA) schedules the tasks to the suitable resources by considering the potential finish time of the tasks, average execution time and waiting time of the tasks and average completion time of the resources. The proposed algorithm (PFTSA) results in minimum makespan as well as improved resource utilization. The experimental results indicate the PFTSA is a promising algorithm than the existing Min-min algorithm.

***Keywords*: Grid, Heuristic scheduling, Job Scheduling, Task Scheduling**

I. INTRODUCTION

Grid computing provides major contributions in the research areas. It's driven in different areas like resource sharing, secure access, resource utilization, distance, standards [1]. Grid helps scientists to store and analyse the large amount of data. One machine is not sufficient to analyse the scientific and engineering applications. Hence it leads go for geographically distributed system environment with high speed network connections [2].

In the grid environment, one of the major challenging issues is that mapping of the tasks for the computing resources and that time has to be an optimal. There are so many task scheduling algorithms like. Min-Min, Max-Min, Approximation algorithms, Simulated Annealing algorithm, Minimum Completion time algorithm, Minimum Execution time algorithm, Heuristic scheduling algorithm, and Genetic algorithm available to solve real world problems. Mapping/scheduling of the tasks for the computing resources is an NP-complete problem [3]. For the NP- complete problem could give better results in polynomial time. But,

approximation algorithms and Heuristic scheduling algorithm will give acceptable solutions for the scheduling problems [13].

The paper presents a heuristic approach for mapping the meta-tasks to the grid resources.

Manuscript published on November 30, 2019.
* Correspondence Author

Mr.K.Sentamilselvan*, Assistant Professor in Kongu Engineering College Erode
Dr.G.K. Kamalam**,** Assistant Professor(SLG) in Kongu Engineering College Erode

II. LITERATURE SURVEY

In Min-Min algorithm, every task is allocated to the machine that complete the task at earliest to reduce the makespan time. More or less similar to min-min algorithm is the max-min algorithm. This will map the job at earliest completion time with maximum execution time and waiting time among all jobs [12].

The heuristic scheduling algorithms can divided into two different types. One is online mode and another one is batch mode scheduling. In online mode, scheduling the tasks are performed as soon as the task will arrives. Second one, Batch mode the tasks are scheduled with batch wise [14].

If the tasks are scheduled in random order to the next available resource is called Opportunistic Load Balancing (OLB) algorithm. The main drawback of this algorithm is not considering the makespan time. Thus, this OLB leads to poor performance [1].

In Minimum Execution Time (MET) algorithm, it will assign the task to available resource whose estimated execution time is minimum. The major disadvantage is, while scheduling it will not consider the waiting time of the task. MCT is similar to MET. In Minimum Completion Time (MCT) algorithm, it will assign the task to available resource whose estimated completion time is minimum. The major disadvantage is, while scheduling it will consider one task at a time [2].

In Min min algorithm scheduling proceeds with a set of all unmapped tasks. The algorithm estimates the execution time and waiting time of each task on each available resource. From the estimation it chooses the appropriate resource. Appropriate resource is the one which provides the overall minimum completion time among a batch of tasks. This process will repeat until the tasks are mapped [1, 4].

In DMMCWTSA algorithm works in two parts. In part1, the algorithm estimates the task worth value for each task ti in the task set.

The tasks are grouped in ascending order based on the task worth value and stored in the variable named Worth Set WS.

For each task ti in the worth set WS, the appropriate resource with less completion time is to be identified and it will be allocated. This process is continued till the WorthSet WS is empty. In part 2, the tasks that are allocated to the resources are reallocated to the resources which provides less completion time than the makespan produced in part 1[11].

Min-mean heuristic scheduling algorithm works in two phases. In the first phase, Min-mean heuristic scheduling algorithm starts with a set of all unmapped tasks. The algorithm calculates the completion time for each task on each resource and finds the minimum completion time for each task. From that group, the algorithm selects the task with the overall minimum completion time and allocates to the appropriate resource. Removes the task from the task set. This process repeats until all the tasks get mapped. The algorithm calculates the total completion time of all the resources and the mean completion time. In phase 2, the mean of all resources completion time is taken. The resource whose completion time is greater than the mean value is selected. The tasks allocated to the selected resources are reallocated to the resources whose completion time is less than the mean value [4,5,6,7].

In Resource Fitness Task Scheduling Algorithm (RFTSA) will allocate the tasks to the available resources by the estimated fitness value of the resource [9].

Grid architecture is composed as group of clusters with several number of worker nodes WN in a cluster. For this architecture an algorithm Best Cluster Decentralized task Scheduling Algorithm is proposed and it optimizes the completion time of the task by splitting the tasks into various sub tasks. The sub tasks are allocate to the worker node WN of various clusters in a decentralized environment. This BCDTSA algorithm effectively mapped both computing-intensive tasks and data-intensive tasks based on the best cluster value [10].

The main idea of TGS Algorithm is to divide all tasks into groups based on the task attributes. Each group will have tasks with similar attributes. These groups will be ordered to schedule based on priority that is given to the task attributes. The first scheduled group will have tasks with high task attributes value or high priority compared to that of the other groups. Then in the chosen group the task with minimum completion time will be scheduled first [12].

The heuristic scheduling algorithm is attempting to minimize the makespan, i.e., finish execution of the meta task as soon as possible. The proposed research work in scheduling the task is to design a valuable heuristic scheduling algorithm to enhance the utilization of the resources and reduce the makespan time. The Potential Finish Time Min-mean Scheduling Algorithm (PFTSA) efficiently allocates the tasks depending on the task potential finish time plus resource mean completion time.

III. MATERIALS AND METHODOLOGY

Scheduling the batch of 'n' independent tasks to the available 'm' heterogeneous grid resources is an NP-Complete problem. Distributed and dynamic heterogeneous grid resources makes the scheduling a complicated problem.

A. *Previous Research:*

The previous proposed algorithm Min-mean [4] and Improved Min-mean [5] provides enhanced mapping of tasks to resources results in reduced makespan and improved resource utilization.

Algorithm Min-mean:

```
In phase 1, the task is allocated to the resources based on the Min-min algorithm.
In phase 2,
Compute
$MeanCT = TotalCT / Total\ No.of\ Resources$
group the resources $R_k$ in a set G if $CTk > MeanCT$
resources $R_k$ in the set G are placed in the decreasing order of $CT_k$.
For tasks assigned to the resources $R_k$ in the Set G
Reschedule the task to the resource $R_j$ if $CTj > MeanCT$
$CTj = ETij + RTj$
$makespan = \max(CTj), 1 \le j \le m$
```

B. *Proposed Work:* assumptions for mapping meta-tasks to heterogeneous resources:

- Independent tasks are being mapped.
- Heuristic algorithms derive static mapping.
- One task is executed at a time in a resource. The tasks are executed based on the Potential Finish Time (PFT) of the task.
- In static mapping, value of 'n' and 'm' are known a priori.
- Task T= {T_1, T_2, … ,T_n}.
- Resource set is represented as R= {R_1, .. , R_m}.
- ETC_{ij}-estimated execution time of task T_i on resource R_j.
- CTT_i -expected completion time of task T_i
- RT_j-ready time of resource R_j.
- CT_j -expected completion time of task T_i scheduled on resource R_j.
- $CTj = ETCij + RTj$
- $makespan = \max(CTT_i)$, i varies from 1 to n
- ETC (Ti, Rj) is calculated using,

$ETC(Ti, Rj) = Lengthi / Powerj$

Lengthi - length of the task T_i in MI and Powerj - processing power of the resource R_j in MIPS.

- resource R_j ready time is computed by,

$$RT(R_j) \sum_{i=1,j=1}^{i=n,j=m} ETC(T_i, R_j)$$

- The completion time (queued time + execution time) of task T_i on resource R_j is calculated by

$$CTT(Ti, Rj) = ETC(Ti, Rj) + RT(Rj)$$

- The maximum CTT (T_i, R_j) is called the makespan.

- Heterogeneity-Task and resource: Variation in the execution time of the task for a given resource is termed as task heterogeneity and vice-versa is called as resource heterogeneity.

C. *Potential Finish Time Min-mean Task Scheduling (PFTSA) algorithm*

This section presents PFTSA to solve the task-scheduling problem efficiently. The algorithm enhances the overall system performance through reduced makespan and better resource utilization.

Proposed PFTSA

The main idea of the proposed algorithm is inspired from the traffic rules in the highways, where the width of the road has an important effect in increase in the number of vehicles in the traffic. Thus the proposed algorithm (PFTSA) allocate a submitted group of tasks by using this rule, where the number of tasks that are allocated to resources proportion to the processing capacity of the resources. The highest number of tasks should be scheduled to the resource which has the highest capacity. The proposed algorithm (PFTSA) schedules the tasks to the suitable resources by calculating the potential finish time of the tasks and mean completion time of the tasks and the resources. That is, the algorithm schedules each resource the best fit group of tasks to achieve approximately finish time equal to PFT.

The PFTSA consists of two stages: a grouping stage and a scheduling stage.

1. Grouping stage: Firstly, the information about resource availability and task requirements is considered. Secondly, the algorithm sorts the arrived tasks into descending order of the length of the task and the order is maintained in TSO (Task schedule order). Thirdly, the algorithm calculates the Potential Finish Time (PFT) of the scheduling model.

The total length of the tasks is calculated as,

$$TL = \sum_{i=1}^{n} \mathrm{MI_i}$$

Total Processing Capacity of the resources is calculated as,

$$TPC = \sum_{j=1}^{m} \mathrm{MIPS_j}$$

Potential Completion Time of the tasks is calculated as,

$$Mean = \frac{\mathrm{TL}}{\mathrm{TPC}}$$

$$PCT = \frac{Mean+TL}{TPC}$$

Potential Fincish Time of the tasks is calculated as,

$$PFT = \frac{PCT}{No.of\ Resources}$$

PFTSA maps the tasks based on the Tasks Scheduled Order (TSO) to the resource until the completion time of the tasks is less than the potential finish time of the tasks. Otherwise, the remaining tasks are scheduled to the next resource and so on. It is stated in the other way, each resource$_j$ should execute only a certain number of tasks, so that the total completion time TCT (resource$_j$) of the tasks submitted to the resource$_j$ is approximately equal to the PFT value, where the TCT (resource$_j$) is the summation of the completion time of all tasks that are assigned to resource$_j$. The PFT is used as a reference for all resources to detect the time that each resource should be spent in processing the tasks.

2. Scheduling Stage: In this stage, the proposed algorithm (PFTSA) assigns tasks into resources according to the PFT value. The completion time of each resource is calculated. The average of the completion time of all the resources is computed. The tasks whose completion time greater than the average completion time is selected and rescheduled to the resources whose completion time is minimum compared to that of the completion time of the previously allocated resource.

The completion time of each resource is calculated as,

$$CT_j = \sum_{i=1}^{k} \mathrm{ETC_{ij}}$$

Where k-represents the number of tasks scheduled on the resource$_j$

Sum of the completion time of 'm' grid resources is calculated as,

$$TotalCT = \sum_{j=1}^{m} \mathrm{CT_j}$$

The average completion time of the grid resources is calculated as,

$$MeanCT = \mathrm{TotalCT} / Total\ No.of\ Resources$$

PFTSA Algorithm:

```
Phase 1:
      Get the length of the tasks TL[n] as input for  n tasks
      Input for m resources, the processing power of the
   resources as PP[m]
       //sort the tasks list in the descending order of the length of the
tasks

 $for\ i = 1\ to\ n\ do$
  $for\ j = i + 1\ to\ n\ do$
    $if\ (TL[i] < TL[j])$
      $temp = TL[i]$
      $TL[i] = TL[j]$
      $TL[j] = temp$
 //sum of the length of the tasks
 $for\ i = 1\ to\ n\ do$
    $TotalLength += TL[i]$
 //sum of the processing power of the resources
 $for\ i = 1\ to\ m\ do$
   $TPC += PP[i]$
 //Potential Completion Time of the tasks is calculated as,
$Mean = \frac{TL}{Number\ of\ Tasks}$
$PCT = \frac{Mean+TL}{TPC}$
//Potential Fincish Time of the tasks is calculated as,
$PFT = \frac{PCT}{m}$
for $i = 1$ to $m$ do
    $CT_i = 0$
    $RT_i = 0$

for $i = 1$ to $n$ do
    $selected[i] = -1$

for $i = 1$ to $n$ do
    if $(selected[i] == -1)$
    for $j = 1$ to $m$ do
        $k = 0$
        if $(CT_j < PFT)$
            $CTj = ETCij + RTj$
                $selected[i] = 1$
                    $RT[j] = CT[j]$
                    $CTT[i] = CT[j]$
                    $scheduled[i] = j$
Phase 2:

for $i = 1$ to $m$

$TotalCT = CT_i$

$MeanCT = TotalCT / Total\ No.of\ Resources$

for $i = 1$ to $n$ do
  if $(CTT[i] > MeanCT)$
     $min = CT[scheduled[i]]$
     for $j = 1$ to $m$ do
        $t = CT[j] + ETC[i][j]$
        if $(t < (min)$
           $min = t$
  $CTT[i] = min$
  $CT[j] = min$
  $CT[scheduled[i]] = min$

$makespan = CTT[1]$

for $i = 2$ to $n$ do
  if $(CTT[i] > makespan)$
     $makespan = CTT[i]$
```

IV.RESULTS AND DISCUSSION

An Illustrative Example

To illustrate how the proposed algorithm PFTSA assigns n' tasks to 'm' resources efficiently. The sample data consists of eight tasks and three resources. The length of the tasks for eight tasks is shown in Table I. The processing capacity of three resources is shown in Table II.

The execution time of each task t_i on each resource r_j is calculated by

$$ETC(Ti,Rj) = Lengthi / ProcessingCapacityj$$

The ETC matrix is given in Table III.

Table-I: Tasks length with millions of instructions (MI).

Tasks	Length
T1	25100
T2	30800
T3	242700
T4	68000
T5	6400
T6	175900
T7	116800
T8	36700

. Table-II: Processing Capacity of Resources

Resources	Processing Capacity
R1	5000
R2	3000
R3	1000

Table-III: ETC Matrix-8×3 matrix for consistent, high task, high resource heterogeneity

Tasks	R1	R2	R3
T1	5.02	8.37	25.1
T2	6.16	10.27	30.8
T3	48.54	80.9	242.7
T4	13.6	22.67	68
T5	1.28	2.13	6.4
T6	35.18	58.63	175.9
T7	23.36	38.93	116.8
T8	7.34	12.23	36.7

Sum of the length of all 'n' tasks, TL=2884400
Total Processing Capacity of the resources, TPC=9000
Mean=TL/Total number of tasks=360550
Potential Completion Time of the tasks is calculated as,

$$PCT = \frac{Mean+TL}{TPC}$$

= 360.55

Potential Fincish Time of the tasks is calculated as,

$$PFT = \frac{PCT}{No.of\ Resources}$$

= 120.18

Order the tasks in the descending order of the task length. The tasks are then scheduled only in the descending order of the task length. The Tasks Scheduled Order (TSO) is

TSO = {t3, t6, t7, t4, t8, t2, t1, t5}

PFTSA algorithm, during phase 1, maps the task represented in the Tasks Scheduled Order (TSO) to the resource until the completion time of the tasks is less than the potential finish time of the tasks. Otherwise, the remaining tasks are scheduled to the next resource and so on. The task-resource selected pair at the end of the phase 1 is shown in Table IV. The completion time of all tasks is shown in Table V.

Table-IV: Task Resource Selected pair during phase 1 of the proposed algorithm (PFTSA)

R1	R2	R3
T3, T6, T7, T2,T1,T5	T4,T8	--

Table-V: Task Completion Time on the Resources (PFTSA)

Tasks	Completion Time		
	R1	R2	R3
T3	48.54	--	--
T6	83.72	--	--
T7	107.08	--	--
T4	--	22.67	--
T8	--	34.9	--
T2	113.24	--	--
T1	118.26	--	--
T5	119.54	--	--

In phase 2, the algorithm works as follows:

The mean completion time of all the resources is calculated.

MeanCT=77.22

To achieve efficient scheduling, the proposed algorithm reschedules the tasks which are scheduled on the resource whose CT > MeanCT.

The tasks selected for rescheduling and the resource to which the tasks are rescheduled are shown in the Table VI.

Table-VI: Tasks Selected for Rescheduling

Tasks	Scheduled on resource	Rescheduled on resource
T6	R1	R2
T7	R1	R2
T2	R1	R3
T1	R1	R3
T5	R1	R3

The task-resource mapped pair and makespan produced by Min-min algorithm, proposed algorithm (PFTSA) is shown in Table VII. From the result it is evident that the proposed algorithm achieves reduced makespan and also achieves better resource utilization than the existing Min-min algorithm.

Table-VII: Comparisons between algorithms in makespan

Algorithm	R1	R2	R3	makespan
Proposed Algorithm (PFTSA)	T3, T6	T4, T8, T7	T2, T1, T5	83.72
Min-min Algorithm	T5, T1, T8, T4, T6, T3	T2, T7		110.96

Benchmark Descriptions

Different instance includes 512 meta-tasks and 16 heterogeneous resources. This section brings the results based on the benchmark dataset proposed by Braun et al[2].

Benchmark dataset for twelve ETC matrix of size 512 * 16 is considered for evaluating the proposed algorithm. The row size represents the number of tasks and column size represents the number of resources, for average of 100 different random values based on the task and resource heterogeneity, and consistency metrics.

Metrics are represented as u-x-yyzz.k,

u –uniform distribution in generation of ETC matrix

x –consistency (c-consistent, i-inconsistent, s-semi-consistent or partially consistent).

An ETC matrix is consistent, whenever a resource ri executes any tasks t_i faster than resource r_j, then resource r_i executes all tasks faster than r_j. An ETC matrix is inconsistent, resource r_i may be faster than resource r_j for executing some tasks and slower for others. An ETC matrix is partially-consistent if it includes a consistent sub-matrix.

Task heterogeneity: execution time difference for a task t_i on ‘m’ resources.

yy – Task heterogeneity, hi means high, lo means low

Resource heterogeneity: execution time difference of ‘n’ tasks on a resource r_j.

zz – resource heterogeneity , hi means high, lo means low

Table VIII shows the twelve various instances of ETC model.

Table-VIII: Expected Time to Compute Model

Consistency	Heterogeneity			
	Task (High)		Task (Low)	
	Resource (High)	Resource (Low)	Resource (High)	Resource (Low)
Consistent	u-c-hihi-0	u-c-hilo-0	u-c-lohi-0	u-c-lolo-0
Inconsistent	u-i-hihi-0	u-i-hilo-0	u-i-lohi-0	u-i-lolo-0
Partially-consistent	u-pc-hihi-0	u-pc-hilo-0	u-pc-lohi-0	u-pc-lolo-0

Evaluation Parameters:Makespan

The criteria for scheduling the tasks to resources is termed as makespan. makespan is defined as,

$makespan = \max(CTT_i)$, i varies from 1 to n

Figure 1 shows that the Potential Finish Time Guided Min-mean Task Scheduling Algorithm provides better makespan than Min-min Heuristic Scheduling Algorithm. Figure 2, 3, 4 and 5 show the makespan values of Min-min and PFTSA for different instances termed high and low task heterogeneity, high and low resource heterogeneity and consistent, inconsistent and partially-consistent ETC matrix.

It is evident from the Figure 1,2,3,4 and 5 that the PFTSA algorithm results in reduced makespan than the existing algorithm Min-min.

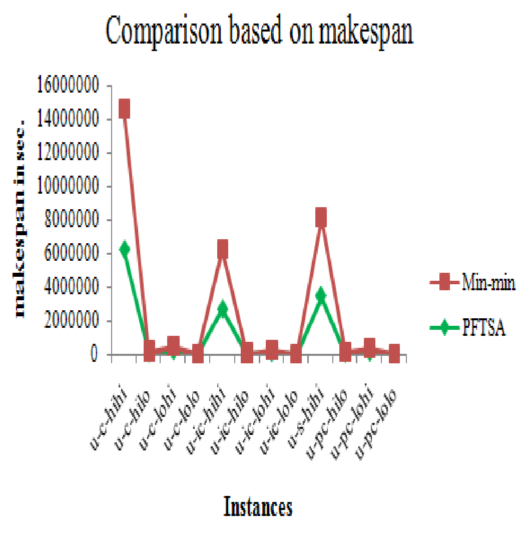

Fig.1. Makespan comparison

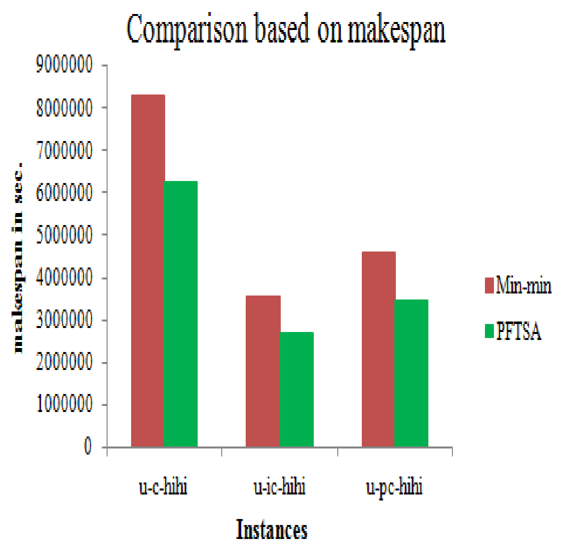

Fig. 2. Makespan for High Task High Machine Heterogeneity

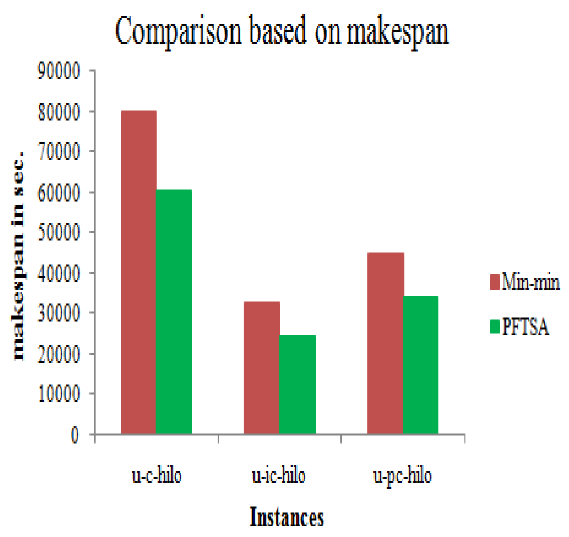

Fig. 3. Makespan for High Task Low Machine Heterogeneity

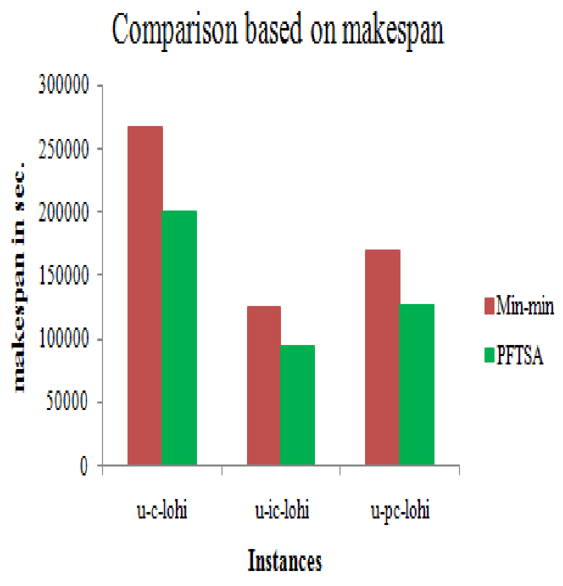

Fig. 4. Makespan for Low Task High Machine Heterogeneity

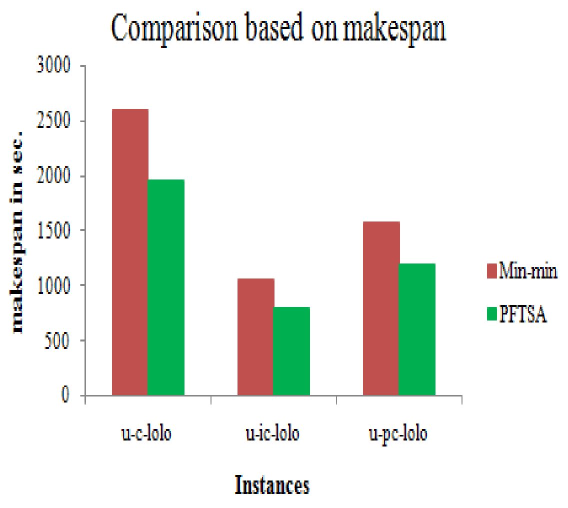

Fig. 5. Makespan for Low Task Low Machine Heterogeneity

V. CONCLUSION AND FUTURE SCOPE

An efficient PFTSA algorithm is proposed. The main idea of the proposed algorithm is inspired from the traffic rules in the highways, where the width of the way has an important effect in the vehicles speed and their arrival time. Thus the proposed algorithm (PFTSA) allocate a submitted group of tasks by using this rule, where the number of tasks that are allocated to resources proportion to the processing capacity of the resources. The proposed algorithm (PFTSA) schedules the tasks to the suitable resources by considering the potential finish time and mean completion time of the tasks and the resources. The proposed algorithm (PFTSA) and the existing Min-min is tested by the benchmark data values proposed by Braun et. al.[5]. The experimental results show that the proposed algorithm (PFTSA) results in reduced makespan than the Min-min for different instances termed high and low task heterogeneity, high and low resource heterogeneity and consistent, inconsistent and partially-consistent ETC matrix. From the results, it is evident that the proposed algorithm (PFTSA) achieves reduced makespan and better resource utilization than the existing Min-min heuristic scheduling algorithm. In future, the proposed algorithm can be designed to deal with dynamic and dependent grid tasks.

REFERENCES

1. Armstrong, R., D.Hensgen, and T.Kidd 1998. The Relative Performance of Various Mapping Algorithms is Independent of Sizable Variances in Run-time Predictions. In 7th IEEE Heterogeneous Computing Workshop (HCW'98), pp. 79-87.
2. Braun, T.D., H.J.Siegel, and N.Beck 2001. A Comparison of Eleven Static Heuristics for Mapping a Class of Independent Tasks onto Heterogeneous Distributed Computing Systems. Journal of Parallel and Distributed Computing 61, pp.810-837.
3. Foster, I., and C.Kesselman 1999. The GRID: Blueprint for a New Computing Infrastructure. Morgan, Kaufmann. Freund, R.F., and H.J.Siegel 1993. Heterogeneous Processing. IEEE Computer, 26(6), pp. 13-17.
4. Kamalam, G.K., and Dr.V.Murali Bhaskaran 2010 a. A New Heuristic Approach:Min-mean Algorithm For Scheduling Meta-Tasks On Heterogeneous Computing Systems. IJCSNS International Journal of Computer Science and Network Security, Vol.10 No.1, pp. 24-31.
5. Kamalam, G.K., and Dr.V.Murali Bhaskaran 2010 b. An Improved Min-Mean Heuristic Scheduling Algorithm for Mapping Independent Tasks on Heterogeneous Computing Environment. International Journal of Computational Cognition, Vol. 8, N0. 4, pp. 85-91.
6. Kamalam, G.K., and Dr.V.Murali Bhaskaran 2011. An Efficient Hybrid Job Scheduling for Computational Grids. International Journal of Computer Applications.
7. Kamalam, G.K., and Dr.V.Murali Bhaskaran 2012a. New Enhanced Heuristic Min-Mean Scheduling Algorithm for Scheduling Meta-Tasks on Heterogeneous Grid Environment. European Journal of Scientific Research, Vol.70 No.3, pp. 423-430.
8. Kamalam, G.K., and Dr.V.Murali Bhaskaran 2012b. Novel Adaptive Job Scheduling algorithm on Heterogeneous Grid Resources. American Journal of applied Sciences, pp. 1294-1299.
9. Kamalam, G.K., 2014a. Resource Fitness Task Scheduling Algorithm for Scheduling Tasks on Heterogeneous Grid Environment. Australian Journal of Basic and Applied Sciences, Vol.8 No.18, pp. 128-135.
10. Kamalam, G.K., 2014b. Best Cluster Decentralized Job Scheduling Algorithm for Scheduling Tasks on Heterogeneous Grid Environment. Australian Journal of Basic and Applied Sciences, Vol.8 No.18, pp. 171-177.
11. Kamalam, G.K., and Ms.B.Anitha 2017. Double Min-Min Credit Worth Task Scheduling Algorithm for Mapping Meta-Tasks on Heterogeneous Grid Environment. International Journal for Modern Trends in Science and Technology, Vol.3 No.2, pp. 18-24.
12. Kamalam, G.K., and Ms.B.Anitha 2019. Task Group Scheduling Algorithm for Mapping a Set of Independent Tasks in Each Group Based on QoS onto Heterogeneous Resources in a Distributed Grid Environment. International Journal of Applied Engineering Research, Vol.14. No.1, pp. 92-96.
13. Muhanad Tahrir Younis, Shengxiang Yang, 2018. Hybrid meta-heuristic algorithms for independent job scheduling in grid computing, Vol. 72, ,pp.498-517.
14. Paolo Bellavista, et. al, 2017. GAMESH: A grid architecture for scalable monitoring and enhanced dependable job scheduling, Vol. 71,pp.192-201.

AUTHORS PROFILE

K.Sentamilselvan received BE Degree in Computer Science and Engineering from Anna University, Chennai, TN, India in 2010. He received M.Tech degree with specialization of Information Security from Pondicherry Engineering College, Pondicherry, India in 2013. He is currently working an Assistant professor in the Department of Information Technology at Kongu Engineering College of Perundurai, Erode, TN, India. He is life member of Computer Society of India (CSI). His research interest is Hacking, Web application Security, Information Security, Cloud and Grid computing.

Dr.G.K.Kamalam is working as an Assistant Professor (SLG) in the Department of Information Technology, Kongu Engineering College, Tamil Nadu, and India. Her research area is Grid and Cloud Computing. She has presented 35 papers in national and international conferences and published 25 papers in international journals in her research and other technical areas. Her Area of Interest includes Algorithm Analysis and Optimization Techniques.