International Journal for Modern Trends in Science and Technology
Volume: 03, Issue No: 02, February 2017
ISSN: 2455-3778
http://www.ijmtst.com

Double Min-Min Credit Worth Task Scheduling Algorithm for Mapping Meta-Tasks on Heterogeneous Grid Environment

Dr.G.K.Kamalam[1] | Ms.B.Anitha[2]

[1,2] Department of Information Technology, Kongu Engineering College, Erode, Tamilnadu, India.

To Cite this Article
Dr.G.K.Kamalam and Ms.B.Anitha, "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. 03, Issue 02, 2017, pp. 18-24.

ABSTRACT

Grid computing is a promising technology that enables the integrated and large-scale resource sharing for solving intensive computational applications. In a distributed heterogeneous grid environment, the mapping of scheduling the tasks to the computing resources proves difficult as resources are geo-graphically distributed. Scheduling task on the heterogeneous distributed resources is an NP-Complete problem. This paper addresses the idea of applying an efficient heuristic algorithm to the scheduling task. The mapping of tasks to resources, the selection criteria in the proposed algorithm Double Min-Min Credit Worth Task Scheduling Algorithm (DMMCWTSA) is based on execution time of the tasks and resource computational capability. The efficiency of this new proposed algorithm in terms of minimum time completion as well as resource utilization is achieved. The evaluations show that the presented heuristic scheduling algorithm is a promising algorithm that provides better minimum time completion as well as resource utilization than the Min-min heuristic scheduling algorithm.

KEYWORDS: *Grid Computing, Task Scheduling, makespan, heuristic approach*

I. INTRODUCTION

Grid Computing is a promising technology that provides consistent, pervasive, and inexpensive access to the geographically distributed heterogeneous computational resources and huge storage space and achieving high throughput in the distributed environment. According to Foster and Kesselman in [6], grid computing is hardware and software infrastructure which offer a cheap, distributable, coordinated and reliable access to powerful computational capabilities. For problem solving in the fields like earth system sciences, financial modelling and high energy physics, the approaches involving computation are widely used. The computational power of a single computer is not sufficient enough to solve computation-intensive applications. In order to meet the computational demand, geographically distributed heterogeneous and parallel systems with more number of processors are developed. The key technologies that affect the efficiency of grid are resource allocation, load balancing and task scheduling algorithms [3]. Executing a task in a grid environment involves mapping of the task to a resource which is capable of executing it. It also involves logical decision making based on the resource characteristics. Complexity of grids mainly originates from decentralized management and resource heterogeneity. These factors often lead to strong variations in the availability of the resources and an increase in the probability of

resources to fail than traditional parallel and distributed systems. Identifying appropriate resources for executing tasks remains a NP-Complete problem in heterogeneous environment. The objective of the task scheduling is to reduce the makespan and improve the resource utilisation. Makespan is defined as the total time required to complete the metatask. Over the years, many scheduling algorithms have been developed.Grid scheduling problem is a NP-Complete problem. Computing optimal solution for NP-complete problem is intractable. The reasonable assumptions are difficult to find an optimal solution, due to the dynamic grid nature. Hence, heuristic algorithms provide good solutions. Heuristic approach proves to be the efficient approach for solving the computationally hard problems. The heuristic approach includes Opportunistic Load Balancing (OLB), Minimum Completion Time (MCT), Min-min, Max-min, and Genetic Algorithm (GA). However, thes approaches do not consider the effects of QoS, which is an important criteria in grid scheduling. OLB allocates the job to the next available machine in the arbitrary order. This algorithm allocates the task without considering the expected execution time of the task on the machine [1,7]. MCT schedules the task by estimating the expected completion time for a task on all machines and allocates the task to the machine with the minimum completion time. MCT considers only one task at a time for scheduling [4,5]. Minimum Execution Time (MET) algorithm allocates the task to the machine by considering the minimum expected execution time regardless of the availability of the machine, it causes poor makespan and severe load imbalance. Min-min algorithm is the best heuristic scheduling algorithm for scheduling the meta-tasks to the machines. Min-min algorithm schedules the tasks in two phases, in first phase, the minimum expected competion time for each task in all machines is calculated. In second phase, the task with overall minimum expected completion time is selected and scheduled to the corresponding machine. The algorithm does not consider the idleness of the machine [1,7,8]. Max-min algorithm is similar to the Min-min algorithm, it differs only in the second phase. It identifies the task with the maximum expected competion time and schedules to the appropriate machine. Max-min algorithm is better only when most of the tasks arriving to the grid environment is shortest. The previous work [9,10,11] 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 the second phase, the algorithm selects the resource whose completion time is greater than mean completion time. The algorithm arranges the selected resources in the decreasing order of the completion time. The algorithm reschedules the tasks assigned on the selected resources to the resource, whose completion time is less than the mean completion time. The previous work [12] Credit Score Tasks Scheduling Algorithm is to identify the appropriate resource and to find the order in which the set of tasks to be mapped to the selected resource. The order in which the tasks to be mapped is identified based on the Credit Score of the task. The Credit Score of the task is calculated based on the execution time of the task. The previous work [14] Task Unique Credit System Scheduling Algorithm, provides the order in which the tasks are to be scheduled is determined based on the highest credit value of the task. The highest credit value of the task is calculated based on the length of the task and unique identification value of the task. The previous work [13] QoS Guided Prominent Value Tasks Scheduling Algorithm based on the task requirement of QoS classifies the tasks into high QoS tasks and low QoS tasks. The grid resources based on the task constraints are classified into high QoS provision resources and low QoS provision resources. QoS Guided Prominent Value Tasks Scheduling Algorithm performs the better mapping between the tasks and the grid resources by computing the Prominent Value for the task. The tasks are ordered into the Prominent Value Set from minimum to the highest prominent value of the task. The current research problem in task scheduling is to bring out an efficient algorithm to reduce the overall completion time of the task [2]. The scope of this work is to propose an efficient task scheduling algorithm on the basis of the highest credit value of the task.

II. MATERIALS AND METHODS

A. Problem Definition

The task scheduling problem includes a set of 'n' meta-task and a set of 'm' heterogeneous resources. The task scheduling problem in a computational grid environment is shown to be an NP-Complete problem. The paper proposes an efficient heuristic task scheduling algorithm Double Min-Min Credit Worth Task Scheduling Algorithm (DMMCWTSA). The proposed algorithm represents an efficient order. Order represents the sequence in which the tasks are to be scheduled. The tasks are mapped to the best resources that provide the minimum completion time in the specified order. The proposed algorithm outperforms Min-min heuristic scheduling algorithm. The proposed algorithm achieves reduced makespan than the Min-min heuristic scheduling algorithm.

B. Proposed DMMCWTSA Algorithm

The mapping of task to the resource is made based on the following assumptions

- A meta-task (ie. A set of independent, non-communicating tasks) is being mapped.
- Heuristics derive a mapping statically.
- Static mapping of meta-tasks to resources is being performed.
- Each resource executes a single task at a time. The order in which the tasks are scheduled is based on the task credit.
- The number of meta-tasks to execute and the number of resources are static and known a prior.
- The accurate estimate of the expected execution time for each task on each resource is contained within an ETC matrix of size n*m, where n is the number of task and m is the number of resources.
- $ETC(t_{i,}r_j)$ represents the expected execution time of the task $t_{i,}$ on resource r_j
- Task set is represented as T={$T_{1,}, T_{2,} \dots \dots \dots T_{n,}$}
- Resource set represented as R={$R_{1,}, R_{2,}, \dots \dots . . R_{m,}$}
- ET_{ij} - expected execution time of task T_i on resource R_j
- TCT_{ij} - expected completion time of task T_i on resource R_j
- RT_j - ready time of resource R_j
- Makespan =max(TCT_{ij})
- ETC matrix is computed by the formula

$$ETC_{ij} = \frac{Tasklength_i}{Power_j}$$

where $Tasklength_i$ represents the length of the task $T_{i,}$ in MI and $Power_j$ represents the computing power of the resource R_j in MIPS

The ready time of the resource is the time at which the resource R_j completes the executing of the previously assigned tasks and is defined as

$$RT_j = \sum_{i=1}^{n} ET_{ij}$$

The proposed algorithm works in two phases. In phase 1, the algorithm calculates the task worth value for each task t_i in the task set. The order in which the tasks to be scheduled are grouped in the Worth Set WS in the ascending order of Task Worth Value (TW). For each task t_i in the worth set WS, the best resource with minimum completion time is identified and scheduled. This process is repeated until the WS is empty.

In phase 2, the tasks that are scheduled to the resources are rescheduled to the resources which provide minimum makespan compared to that of the makespan produced in phase 1.

The pseudocode for finding Credit for each task t_i is given below:

```
Alogorithm Credit
    Initialize MAX=0
    for  i=1 to n
        for j=1 to m
     if ETCij >MAX
    MAX=ETCij
            endif
        endfor
    endfor
    Compute W1= MAX/2
    Compute W2=MAX/3
    Compute W3=W1+W2
    Compute W4=W2+W3
for all submitted task ti in the metatask Mt find the maximum execution time of each task
  if MAXi<W1
        Credit_i=4
  else if W2 ≤ MAXi  ≤ W3
        Credit_i =3
  else if W3 ≤ MAXi  ≤ W4
        Credit_i =2
  else
        Credit_i =1
  endif
endfor
```

Task Worth Value

The pseudocode for finding Task Worth (TW) for each task t_i is given below:

```
Algorithm Task Worth
Fix a unique random value (URVi) for each task ti in the range 1 to n
```

for all submitted task in the metatask Mt
Compute $UVC_i=URV_i/dv$
endfor
for each task t_i in the metatask Mt
Compute $TW_i=URV_i*UVC_i$
endfor
Order the task in the Worth Set WS in the ascending order of TW_i

The value dv is determined as follows:

If the number of tasks is a two digit number dv=100, if it is a three digit number, dv=1000 and so on.

An Illustrative Example

To illustrate how the proposed algorithm Double Min-Min Credit Worth Task Scheduling Algorithm (DMMCWTSA) schedules the tasks to the resources efficiently. The following illustration considers eight tasks and eight resources. The ETC matrix for eight tasks and eight resources is given in Table 1. The sample 8×8 matrix is shown with consistent, high task, high resource heterogeneity.

Table 1 ETC Matrix-8×8 matrix for consistent, high task, high resource heterogeneity

Tasks	R1	R2	R3	R4	R5	R6	R7	R8
t1	25,137.5	52,468.0	150,206.8	289,992.5	392,348.2	399,562.1	441,485.5	518,283.1
t2	30,802.6	42,744.5	49,578.3	50,575.6	58,268.1	58,987.9	85,213.2	87,893.0
t3	242,727.1	661,498.5	796,048.1	817,745.8	915,235.9	925,875.6	978,057.6	1,017,448.1
t4	68,050.1	303,515.9	324,093.1	643,133.7	841,877.3	856,312.9	861,314.8	978,066.3
t5	6,480.2	42,396.7	98,105.4	166,346.8	240,319.5	782,658.5	871,532.6	1,203,339.8
t6	175,953.8	210,341.9	261,825.0	306,034.2	393,292.2	412,085.4	483,691.9	515,645.9
t7	116,821.4	240,577.6	241,127.9	406,791.4	1,108,758.0	1,246,430.8	1,393,067.0	1,587,743.1
t8	36,760.6	111,631.5	150,926.0	221,390.0	259,491.1	383,709.7	442,605.7	520,276.8

The maximum execution time of the task
MAXET=1587,743.1
W1=793,871.55
W2=529,247.7
W3=1323,119.25
W4=1852,366.95

The credit for each task based on execution time is computed and is given in Table 2.

Table 2 Execution Time Credit

Task	$Credit_i$
t1	4
t2	4
t3	3
t4	3
t5	3
t6	4
t7	2
t8	4

Each task is assigned a unique random value. The unique value credit for each task t_i is computed using the algorithm Task Worth and is given in Table 3.The task worth value for each task t_i is calculated using the given formula $TW_i=URV_i*UVC_i$ and the result is given in Table 3.

Table 3 Task Worth Value

Task	Credit	URV	UVC	TW
t1	4	7	0.7	2.8
t2	4	6	0.6	2.4
t3	3	1	0.1	0.3
t4	3	4	0.4	1.2
t5	3	8	0.8	2.4
t6	4	2	0.2	0.8
t7	2	3	0.3	0.6
t8	4	5	0.5	2.0

The tasks to be scheduled are ordered in the Worth Set WS in the ascending order of Tw_i

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

The algorithm works in two phases.

In phase 1, the algorithm schedules the task in the order specified in the Worth Set (WS) to the resource with the minimum completion time.

The task-resource selected pair is

t1-R2 t5-R5
t2-R6 t6-R3
t3-R1 t7-R2
t4-R1 t8-R4

The makespan produced by Min-min algorithm compared to the result of the proposed algorithm Double Min-Min Credit Worth Task Scheduling Algorithm (DMMCWTSA) in first phase is shown in Table 4.

Table 4 Comparison between algorithms in makespan and task-resource scheduled pair

	R1	R2	R3	R4	R5	R6	R7	R8	makespan
Proposed Algorithm	t3, t4	t1, t7	t6	t8	t5	t2			310,777.2
Min-min	t1, t5, t3, t8, t4	t2, t6	t7						379,155.5

From Table 4 it is evident that the proposed algorithm in the first phase achieves better makespan and improved resource utilization.

To achieve efficient scheduling, the proposed algorithm reschedules the task to the resource with the aim that the new rescheduling will result in reduced makespan than the makespan obtained in the first phase.

The makespan produced by Min-min algorithm and the makespan produced by the proposed algorithm in first phase and second phase and the task, resource selected pair in each algorithm is shown in Table 5.

Table 5 Comparisons between algorithms in makespan

	R1	R2	R3	R4	R5	R6	R7	R8	makespan
Proposed Algorithm Phase 1	t3, t4	t1, t7	t6	t8	t5	t2			310,777.2
Proposed Algorithm Phase 2	t5, t1	t4	t7	t6	t8	t2			306,034.2
Min-min	t1, t5, t3, t8, t4	t2, t6	t7						379,155.5

Double Min-Min Credit Worth Task Scheduling Algorithm (DMMCWTSA):

Phase 1:

do until all tasks in the Score Set SS are mapped
for all tasks in the Score Set SS
for all resource Rj
Compute TCTij=ETCij + RTj
end for
end for
find the task t_k with the minimum completion time
assign task t_k to the resource Rj that gives the minimum completion time
delete task from Score Set SS
Update RTj
end do
Compute makespan = max(TCT_{ij}) for all i,j

Phase 2:

for all resource Rj
for all tasks mapped to Rj
for all remaining resource Rk
TCTik=ETCik + RTk
if TCTik < TCTij
assign task t_i to the resource Rk
Update RTk
end for
end for
end for
Compute makespan = max(TCT_{ij}) for all i,j

C. Results And Discussion

This section presents the experimental results obtained for the benchmark model by Braun et al[5].

1.Benchmark Descriptions. The instances of the benchmark model are classified into 12 different types of ETC matrix for 512 tasks and 16 resources, with each of them consisting of the 100 instances based on the three metrics: task heterogeneity, resource heterogeneity, and consistency.

Instances are represented as u-x-yyzz.k where,
u – represents uniform distribution (in generating ETC matrix)
x – represents the type of consistency (c-consistent, i-inconsistent, s-semi-consistent or partially consistent).
An ETC matrix is consistent, whenever a resource r_i 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 semi-consistent if it includes a consistent sub-matrix.
Task heterogeneity: Variation in the execution time for a task t_i on all resources.
yy – heterogeneity of the tasks(hi-high, lo-low)
Resource heterogeneity: Variation in the execution time of all the tasks on a resource r_j.
zz – heterogeneity of the resources(hi-high, lo-low)
The 12 different instances are divided into three groups of four instances each and shown in Table 6.

Table 6 ETC 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
Semi-consistent	u-s-hihi-0	u-s-hilo-0	u-s-lohi-0	u-s-lolo-0

2. Evaluation Parameters.

Makespan

Makespan is the important optimization criteria for grid scheduling. Makespan is calculated as makespan=max(CT(t_i))

Figure 1 shows that the Double Min-Min Credit Worth Task Scheduling Algorithm provides better

makespan than Min-min Heuristic Scheduling Algorithm.

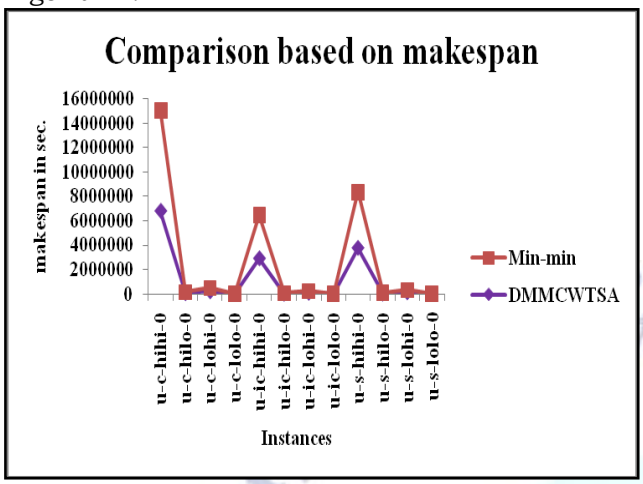

Figure 1: Comparison based on makespan

Figure 2, 3, 4, 5 show the comparison of the makespan values obtained by Min-min and DMMCWTSA in all the four instances which comprises High Task High Machine, High Task Low Machine, Low Task High Machine, Low Task Low Machine. The four instances are represented for consistent, inconsistent, semi-consistent or partially consistent heterogeneous computing systems.

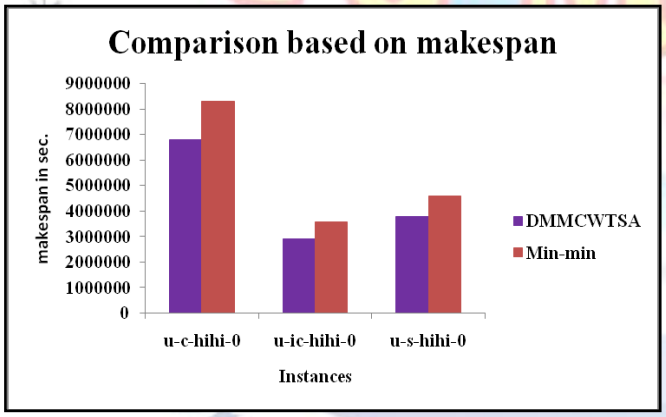

Figure 2: Comparison based on makespan for High Task High Machine Heterogeneity

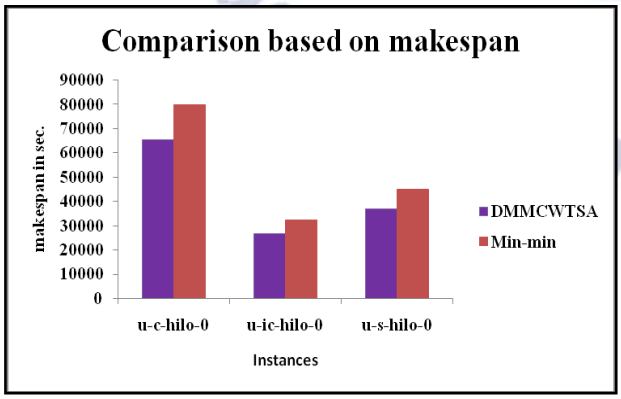

Figure 3: Comparison based on makespan for High Task Low Machine Heterogeneity

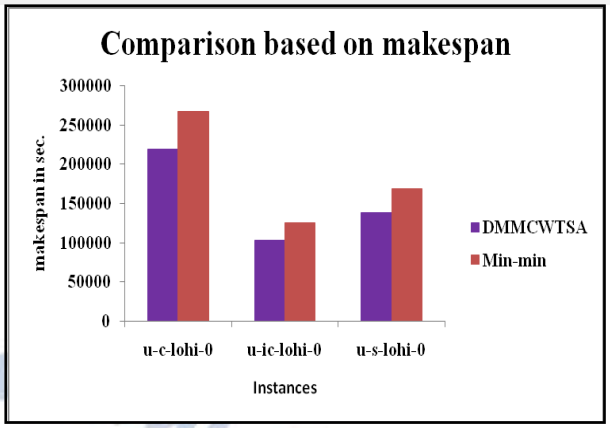

Figure 4: Comparison based on makespan for Low Task High Machine Heterogeneity

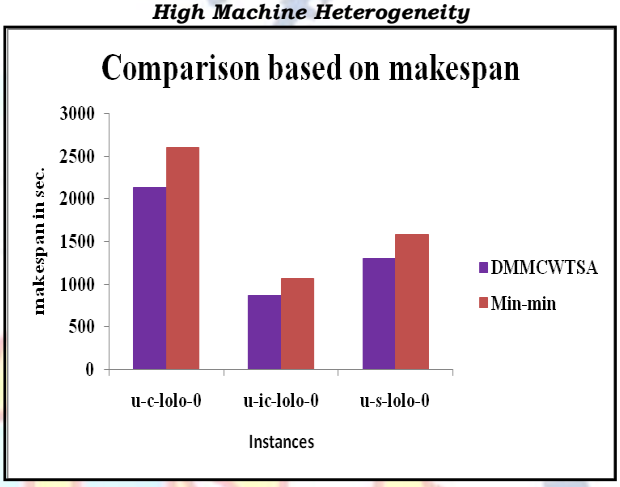

Figure 5: Comparison based on makespan for Low Task Low Machine Heterogeneity

III. Conclusion

The task scheduling is an NP-Complete problem in the computational grid environment. The paper proposes an efficient heuristic scheduling algorithm Double Min-Min Credit Worth Task Scheduling Algorithm (DMMCWTSA) by considering the task worth value and completion time in scheduling the task to the suitable resources. The proposed algorithm Double Min-Min Credit Worth Task Scheduling Algorithm (DMMCWTSA) and the Min-min heuristic scheduling algorithm are examined using the benchmark simulation model by Braun et. al.[5]. The experimental result show that the Double Min-Min Credit Worth Task Scheduling Algorithm (DMMCWTSA) outperforms the Min-min heuristic algorithm and provides reduced makespan on various instances such as task heterogeneity(high, low), resource heterogeneity(high, low) and consistency(consistent, inconsistent and semi-consistent).

References

[1] R.Armstrong, D.Hensgen, and T.Kidd, "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, 1998.

[2] H.Baghban, A.M. Rahmani, " A Heuristic on Job Scheduling in Grid Computing Environment", In Proceedings of the seventh IEEE International Conference on Grid and Cooperative Computing, pp. 141-146, 2008.

[3] TD. Braun, HJ. Siegel, N.Beck, — A Taxonomy for Descriging Matching and Scheduling Heuristics for Mixed-machine Heterogeneous Computing Systems , IEEE Workshop on Advances in Parallel and Distributed Systems, West Lafayette, pp. 330-335, 1998.

[4] T.Braun, H.Siegel, N.Beck, L.Boloni, M.Maheshwaran, A.Reuther, J.Robertson, M.Theys, B.Yao, D.Hensgen, and R.Freund, —A Comparison Study of Static Mapping Heuristics for a Class of Meta-tasks on Heterogeneous Computing Systems , In 8 th IEEE Heterogeneous Computing Workshop(HCW'99), pp. 15-29, 1999.

[5] T.D.Braun, H.J.Siegel, and N.Beck, "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, 2001.

[6] I.Foster and C. Kesselman, "The Grid: Blueprint for a Future Computing Infrastructure", Morgan Kaufmann Publishers, USA, 1998.

[7] R.F.Freund and H.J.Siegel,"Heterogeneous Processing", IEEE Computer , 26(6), pp. 13-17, 1993.

[8] R.F.Freund, and M.Gherrity, "Scheduling Resources in Multi-user Heterogeneous Computing Environment with Smart Net", In Proceedings of the 7th IEEE HCW, 1998.

[9] G.K.Kamalam, and Dr. V..Murali Bhaskaran, "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, 2010.

[10] G.K.Kamalam, and Dr. V..Murali Bhaskaran, "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, 2010.

[11] G.K.Kamalam, and Dr. V..Murali Bhaskaran, "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, 2012.

[12] Dr.G.K.Kamalam, B.Anitha and S.Mohankumar, "Credit Score Tasks Scheduling Algorithm for Mapping a Set of Independent Tasks onto Heterogeneous Distributed Computing Grid Environment", International Journal of Emerging Technology in Computer Science & Electronics (IJETCSE),Vol.20, No.2,pp.182-186, 2016.

[13] Dr.G.K.Kamalam, B.Anitha and S.Mohankumar, "QoS Guided Prominent Value Tasks Scheduling Algorithm in Computational Grid Environment", International Journal of Emerging Technology in Computer Science & Electronics (IJETCSE),Vol.20, No.2,pp.187-191, 2016.

[14] V.Manju Bargavi, Dr.G.K.Kamalam, and B.Anitha , "Task Unique Credit System Based Scheduling Algorithm for Mapping Meta-Tasks on Heterogeneous Grid Environment", International Journal of Innovative Research in Computer Science and Engineering (IJIRCSE),Vol.1, No.13,pp.13-22, 2015.