Skewness-Based Min-Min Max-Min Heuristic for Grid Task Scheduling

Sanjaya Kumar Panda[1], Pratik Agrawal[2], Pabitra Mohan Khilar[3] and Durga Prasad Mohapatra[4]
[1,3,4] Department of Computer Science and Engineering
[1] Indian School of Mines, Dhanbad, India [2] IBM, Bangalore, India [3,4] National Institute of Technology, Rourkela, India
{sanjayauce, pratikag99}@gmail.com, {pmkhilar, durga}@nitrkl.ac.in

Abstract— **Skewness plays a very important role in task scheduling. It measures the degree of asymmetry in a data set. Moreover, the data set may be positively skewed or negatively skewed. The quartile coefficient of skewness lies in between -1 to 1. If the quartile coefficient is in between -1 to 0, then the data set is negatively skewed. Otherwise, it is positively skewed. Min-Min and Max-Min heuristic underperforms if the data set is positively skewed and negative skewed respectively. In order to overcome this problem, we have proposed a hybrid heuristic called Skewness-Based Min-Min Max-Min (SBM^2). This heuristic selects one of the heuristics (Min-Min or Max-Min) based on the skewness value. It can be calculated using three factors: first quartile, second quartile (or median) and third quartile. Simulation results show that the proposed hybrid heuristic minimizes the makespan and handle the skewed data set.**

Keywords-Skewness; Min-Min; Max-Min; Makespan; Grid Scheduling; Data Set; Machine Utilization

I. INTRODUCTION

Grid computing is a collection of high-end computers such as servers, clusters, supercomputers to solve a single problem [1]. More specifically, these computers are used for scientific and engineering applications e.g. weather forecasting, medical diagnosis etc [2]. To solve these problems, the grid resources are distributed over the world. These resources are owned by an organization, a company or an individual [3]. So, resources may join or leave at any point of time [4].

Grid resource management is responsible for resource discovery, scheduling as well as monitoring the applications / jobs submitted by the grid users. Moreover, scheduling plays a significant role to map the available resources (or machines) and the jobs. However, the mapping of the jobs to the machines in a heterogeneous environment is a well known NP-Complete problem [5]. So, an efficient scheduling heuristic is needed to map the available resources and the jobs.

Grid scheduling was extensively studied in the last decades. It was shown that Min-Min heuristic gives better makespan in comparison to other heuristics [6]. However, Max-Min heuristic gives the better machine utilization in comparison to other heuristics. But, these heuristics performances are limited to benchmark data set. This data set uses uniform distribution to generate the execution time i.e. the time taken by a meta-task in a machine. However, the skew data set (discussed in Section IV) is not well studied in the literature.

In this paper, we have defined the degree of asymmetry in a data set (referred as skewed data set). Moreover, we have shown that how this data set is really affecting the existing grid heuristics (i.e. Min-Min and Max-Min). Finally, we have proposed a Skewness-Based Min-Min Max-Min (SBM^2) heuristic to overcome the addressed problem. Note that, the proposed heuristic is applicable only for skewed data set.

II. RELATED WORKS

A huge number of heuristics have been developed to map the meta-tasks to the machines. Here, we have described very few heuristics that are closely related to our proposed heuristics. The Min-Min Max-Min selective algorithm [7] selects one of the heuristic (i.e. Min-Min or Max-Min) based on the population standard deviation. If the standard deviation is less than prescribed threshold, then Min-Min heuristic is used to assign the meta-task. Otherwise, Max-Min heuristic is used to assign the meta-task. Apart from this, if the difference between two consecutive completion times is more than a threshold and the corresponding index is present in the lower half of the data set (negative skew), then Min-Min heuristic is used to assign the meta-task. If the corresponding index is present in the upper half of the data set (positive skew), then Max-Min heuristic is used to assign the meta-task. This heuristic takes O(m^{2}n) time to assign m meta-tasks to n machines [7].

The resource aware scheduling algorithm [8] performs the Min-Min heuristic when the available machines are odd. Otherwise, Max-Min heuristic is used to assign the meta-task. This heuristic takes O(m^{2}n) time to assign m meta-tasks to n machines [8].

The semi-interquartile Min-Min Max-Min [9] chooses Min-Min or Max-Min heuristic based on semi-interquartile measure. To know whether a data set is positive skewed or negative skewed, it finds the difference between two consecutive completion times. If the difference is more than the semi-interquartile range then it stores the position where the difference is pointed out. Furthermore, it selects Max-Min heuristic if the position is present in the upper half. Otherwise, it selects Min-Min heuristic to assign the meta-task. This heuristic takes O(m^{2}n) time to assign m meta-tasks to n machines [9].

Rasooli et al. [10] presents three new dispatching rules for queuing of jobs. They are Short Job First (SJF), Long Job First (LJF) and Minimum Time To Due date (MTTD). The goal of SJF and LJF is to minimize flowtime and latency respectively.

Decai et al. [11] uses Set Pair Analysis (SPA) soft computation method to solve the task scheduling problem. Three immediate mode heuristics and three batch mode heuristics are introduced to handle the uncertainty in the computational grid.

The Round Robin Task Scheduling (RRTS) [12] uses dynamic time slice to assign a meta-task to a machine. When the time slice expires, it selects another meta-task from the task queue. However, preemption is the main overhead of this approach.

The heuristics proposed in the literature [5-12] may underperform in skewed data set. These heuristics give good results in their own data set / non-skew data set (e.g. benchmark data set). In this paper, we have introduced the SBM^2 heuristic to minimize the makespan in skewed data set. To the best of our knowledge, this is the first grid scheduling work on skewed data set.

III. Preliminaries

In this section, we have discussed the batch mode heuristics along with their merits and demerits. Before that, we have presented notations with their definition and assumptions taken in this paper.

A. *Notations*

Notation	**Definition**
TQ	Task Queue
m	Total number of meta-tasks
n	Total number of machines
T_i	Meta-task ID of meta-task i
M_j	Machine ID of machine i
$C_{i,j}$	Completion time for meta-task i on machine j
$E_{i,j}$	Execution time for meta-task i on machine j
R_j	Ready time of machine j
Q_1	First Quartile
Q_2	Second Quartile
Q_3	Third Quartile

B. *Assumptions*

In this paper, we have taken following assumptions: 1) the experiments are carried out in a heterogeneous environment. Moreover, machines are different architecture and specifications, 2) there is no priority among the meta-tasks / machines. In other words, the meta-tasks / machines are independent of each other, 3) there is no deadline for the meta-tasks, 4) we have assumed that the meta-task execution time (or data set) is known before the scheduling takes place.

C. *Scheduling Heuristics*

There are many heuristics present in grid scheduling like Minimum Execution Time (MET), Minimum Completion Time (MCT), Min-Min, Max-Min etc. But, in this subsection, we have presented two batch mode heuristics: Min-Min and Max-Min.

1) Min-Min

It is a two-phase heuristic. First, it finds the earliest completion time for each meta-task. Second, it finds one of the earliest completion time meta-task from the resultant matrix of the first phase. In other words, it chooses a short meta-task rather than a long meta-task. So, it gives good performance in the negative skewed data set and poor performance in positive skewed data set. Let us consider a two-dimensional matrix called expected execution time matrix shown in Equation 1.

$$\begin{pmatrix} E_{1,1} & E_{1,2} & \cdots & E_{1,n\text{-}1} & E_{1,n} \\ \vdots & & \vdots & \vdots & \\ E_{m,1} & E_{m,2} & \cdots & E_{m,n\text{-}1} & E_{m,n} \end{pmatrix}` \quad (1)$$

Here, $E_{1,1}$ indicates the expected execution time for meta-task 1 on machine 1. As we have considered m meta-tasks and n machines, the order of the matrix is m by n. Row and column represents meta-task and machine respectively. First, it finds the minimum value in each row of the matrix. It results in a column matrix as shown in Equation 2.

$$\begin{pmatrix} E_{1,\alpha} \\ \vdots \\ E_{m,v} \end{pmatrix} \quad (2)$$

where, $E_{1,\alpha} = \min(E_{1,1}, E_{1,2}, ..., E_{1,n-1}, E_{1,n})$

and $E_{m,v} = \min(E_{m,1}, E_{m,2}, ..., E_{m,n-1}, E_{m,n})$

Second, it finds the minimum value of the column matrix. The resultant matrix is a scalar matrix. It is shown in Equation 3. Let the element is $E_{x,y}$. So, meta-task T_x is assigned to machine M_y in the first iteration.

$$\left(E_{x,y}\right) \quad (3)$$

where, $E_{x,y} = \min(E_{1,\alpha}, ..., E_{m,v})$

The iteration continues until the TQ is empty. This heuristic takes $O(m^2n)$ time to assign m meta-tasks to n machines [7,13].

2) Max-Min

It is also a two-phase heuristic. First, it finds the earliest completion time for each meta-task. Second, it finds one of the latest completion time meta-tasks from the resultant matrix of the first phase. In other words, it chooses a long meta-task rather than a short meta-task. So, it gives good performance in positive skewed data set and poor performance in negative skewed data set.

Like Min-Min, it finds the minimum value in each row of the matrix as shown in Equation 1. The resultant matrix is a column matrix as shown in Equation 2. Second, it finds the maximum value rather than the minimum value of the column matrix. The resultant matrix is a scalar matrix. It is shown in Equation 4. Let the element is $E_{x,y}$. So, meta-task T_x is assigned in the first iteration. Note that, T_x is assigned to a machine which gives earliest completion time.

$$\left(E_{x,y}\right) \tag{4}$$

$$\text{where, } E_{x,y} = \max(E_{1,\alpha}, \ldots, E_{m,v})$$

The iteration continues until the TQ is empty. This heuristic takes $O(m^2n)$ time to assign m meta-tasks to n machines [7,13].

IV. Proposed Heuristic

A. Skewness

The Skewness value is of three types: positive, negative and zero. Positive skew shows that the range / number of meta-tasks on the left side are much more than the right side. Figure 1 illustrates the positive skew of the data set. For instance, the sequence 1, 2, 3, 4, 5, 6 and 90 is positively skewed. Negative skew shows that the range / number of meta-tasks on the right side are much more than the left side. Figure 2 illustrates the negative skew of the data set. For instance, the sequence 1, 85, 86, 87, 88, 89 and 90 is negative skewed [14]. In both the figures, x-axis denotes the meta-task and y-axis denotes the execution time. Note that, the execution time refers to the column matrix shown in Equation 2.

B. Motivation and Contribution

Min-Min heuristic gives poor performance in the positive skewed data set. Because, the data set contains too much short meta-tasks and it executes short meta-task first. However, Max-Min heuristic gives better performance in the positive skewed data set.

Max-Min heuristic gives poor performance in the negative skewed data set. Because, the data set contains too many long meta-tasks and it executes long meta-task first. However, Min-Min heuristic gives better performance in the negative skewed data set.

The above scenario gives a detailed analysis of existing heuristics. So, the main motivation is to design an efficient heuristic which takes care about positive as well as negative skewed data set.

In this paper, we use a skewness measure to choose one of the heuristic: Min-Min or Max-Min. In contrary, several mathematical measures are used in the recent papers [9, 15-16]. Skewness determines whether the data set is positive skewed or negative skewed. If the data set is positively skewed then we use Max-Min heuristic to reduce the makespan. If it is negatively skewed then we use Min-Min heuristic to reduce the makespan. The skewness is calculated in each iteration to choose one of the heuristics. This is the main contribution in this paper. Furthermore, the performances of these heuristics are evaluated using synthetic data set. Because the benchmark data set is neither positive skewed nor negative skewed.

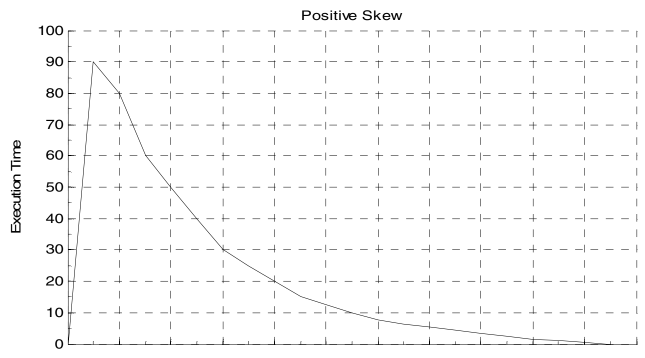

Figure 1. Positive Skewness.

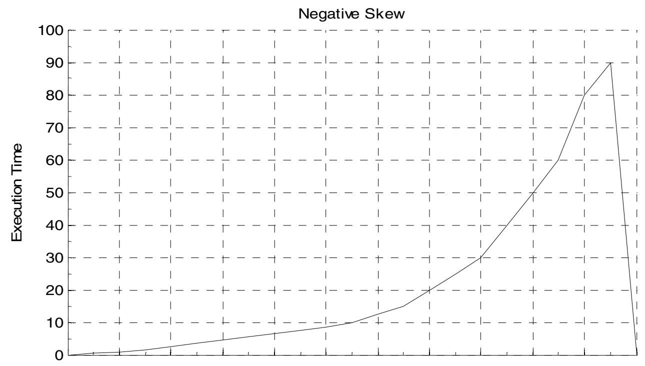

Figure 2. Negative Skewness.

C. Heuristic

The pseudocode for SBM^2 heuristic is shown below:

Skewness-Based Min-Min Max-Min Heuristic

```
1. while TQ != NULL
2.   for all meta-tasks $T_i$ in TQ
3.       for all machines $M_j$
4.           $C_{i,j} = E_{i,j} + R_j$
5.       end for
6.   end for
7.   for all meta-tasks $T_i$ in TQ
8.       Find minimum $C_{i,j}$ and machine $M_j$ that holds it
9.   end for
10.  Sort the meta-task in ascending order of $C_{i,j}$
11.  Calculate Skewness
         $Skewness = \frac{Q_3 + Q_1 - 2Q_2}{Q_3 - Q_1}$
12.  if Skewness $\geq 0$
13.  then Assign meta-task $T_i$ to machine $M_k$ that holds maximum $C_{i,k}$
14.  else Assign meta-task $T_i$ to machine $M_k$ that holds minimum $C_{i,k}$
15.  end if
16.  Delete the meta-task $T_i$ and update TQ
17. end while
```

D. Heuristic Description

Initially, the TQ contains all the meta-tasks in numeric order. These meta-tasks are assigned to the machines in random order to minimize the total processing time. When a meta-task is assigned to a machine, it is removed from the TQ. Lines 2 to 6 calculates the completion time of all meta-tasks on all machines. Moreover, the completion time is the sum of execution time and the machine ready time. Note that, the completion time matrix is a two-dimensional matrix. Next, it finds a minimum completion time machine for all meta-tasks. Here, the two-dimensional matrix is reduced to a one-dimensional column matrix (Lines 7 to 9). Next, it sorts the meta-tasks (or elements of column matrix) in ascending order of their completion time (Line 10). Then, it calculates the skewness which contain three factors: first quartile (Q_1), second quartile (Q_2) / Median and third quartile (Q_3) (Line 11) [17]. The quartile measures are also studied in our earlier paper [18].The formula of Q_1, Q_2 and Q_3 are shown in Equation 5, 6 and 7 respectively. Note that, any fractional value results the average of consecutive meta-tasks or floor function (as applicable). Here, S denotes the number of elements in the column matrix.

$$Q_1 = \left\{ \left(\frac{1}{4}(S+1)\right)^{\text{th}} \text{ item} \right\} \tag{5}$$

$$Q_2 = \left\{ \begin{array}{ll} \left(\frac{S+1}{2}\right)^{th} \textit{ item} & \textit{if } S = \textit{odd} \\ \dfrac{\left(\frac{S}{2}\right)^{th} \textit{ item} + \left(\frac{S+1}{2}\right)^{th} \textit{ item}}{2} & \textit{Otherwise} \end{array} \right\} \tag{6}$$

$$Q_3 = \left\{ \left(\frac{3}{4}(S+1)\right)^{\text{th}} \text{ item} \right\} \tag{7}$$

Skewness value ranges from -1 to 1. If the value is in between 0 to 1 then Max-Min heuristic is applied. Otherwise, Min-Min heuristic is applied (Lines 12 to 15). The while loop iterates until the TQ is empty (Lines 1 to 17).

E. Time Complexity Analysis

The time complexity analysis of first iteration is shown here. Steps 2 to 6 takes O(mn) time. Steps 7 to 9 takes O(mn) time. Steps 10 takes O(m log m) time. Step 11 to 15 takes O(1) time. However, the overall complexity is $O(m^2n)$ (Steps 1 to 17).

V. Illustration

Let us consider an example contains of four meta-tasks (T_1, T_2, T_3 and T_4) and two machines (M_1 and M_2). The execution time of the meta-tasks on the machines is shown in Table I.

TABLE I. EXECUTION TIME MATRIX (EXAMPLE I)

	M_1	M_2
T_1	5	7
T_2	8	6
T_3	13	10
T_4	60	65

We assume that the machine ready time is zero. Meta-task 1 and 4 has minimum completion time in Machine 1. On the other hand, the rest of the meta-tasks have minimum completion time in Machine 2. So, the element of the resultant column matrix is 5, 6, 10 and 60 respectively. Then, it sorts the element in ascending order of their execution time. Next, it finds the skewness. Here, S=4, Q_1 = 5.5, Q_2 = 8 and Q_3 = 35. So, skewness is 0.8305. As the value is positive, Max-Min heuristic is applied in the first iteration.

In the next iteration, S=3, Q_1 = 6, Q_2 = 7 and Q_3 = 10. So, skewness is 0.5. As the value is again positive, Max-Min heuristic is applied in the second iteration. Finally, the makespan of skewness heuristic is 60. In contrary, Min-Min and Max-Min heuristic gives 71 and 60 respectively.

Let us consider another example contains four meta-tasks (T_1, T_2, T_3 and T_4) and two machines (M_1 and M_2). The execution time of the meta-tasks on the machines is shown in Table II.

TABLE II. EXECUTION TIME MATRIX (EXAMPLE II)

	M_1	M_2
T_1	12	13
T_2	146	101
T_3	140	143
T_4	148	147

Meta-task 1 and Meta-task 3 has minimum completion time in Machine 1. On the other hand, the rest of the meta-tasks have minimum completion time in Machine 2. So, the element of the resultant column matrix is 12, 101, 140 and 147 respectively. Then, it sorts the element in ascending order of their execution time. Next, it finds the skewness. Here, S=4, Q_1 = 56.5, Q_2 = 120.5 and Q_3 = 143.5. So, skewness is -0.4713. As the value is negative, Min-Min heuristic is applied in the first iteration.

In the next iteration, S=3, Q_1 = 101, Q_2 = 143 and Q_3 = 147. So, skewness is -0.8261. As the value is again negative, Min-Min heuristic is applied in the second iteration. Finally, the makespan of skewness heuristic is 244. In contrary, Min-Min and Max-Min heuristic gives 248 and 286 respectively.

Let us consider another example contains four meta-tasks (T_1, T_2, T_3 and T_4) and two machines (M_1 and M_2). The execution time of the meta-tasks on the machines is shown in Table III.

TABLE III. EXECUTION TIME MATRIX (EXAMPLE III)

	M_1	M_2
T_1	17	19
T_2	12	11
T_3	13	15
T_4	17	18

Meta-task 2 has minimum completion time in Machine 2. On the other hand, the rest of the meta-tasks have minimum completion time in Machine 1. So, the element of the resultant column matrix is 17, 11, 13 and 17 respectively. Then, it sorts the element in ascending order of their execution time i.e. 11, 13, 17 and 17. Next, it finds the skewness. Here, S=4, Q_1 = 12, Q_2 = 15 and Q_3 = 17. So, skewness is -0.2. As the value is negative, Min-Min heuristic is applied in the first iteration.

In the next iteration, S=3, Q_1 = 13, Q_2 = 17 and Q_3 = 17. So, skewness is -1. As the value is again negative, Min-Min heuristic is applied in the second iteration. Finally, the makespan of skewness heuristic is 30. In contrary, Min-Min and Max-Min heuristic gives 30 and 30 respectively. The above example clearly shows that SBM^2 heuristic outperforms in comparing to Min-Min and Max-Min heuristic. In some situation, it gives makespan equal to Min-Min / Max-Min heuristic.

VI. Experiments

In this section, we evaluate our proposed SBM^2 heuristic by experiments and give a detailed performance comparison beside other existing heuristics e.g. Min-Min and Max-Min. We have considered both synthetic and benchmark data set to evaluate our proposed heuristic.

A. *Experimental Setup*

We have used MATLAB R2010b version 7.11.0.584 to evaluate our proposed heuristic. The heuristic is evaluated using 12 synthetic instances e.g. 50 × 10. The first value indicates the number of meta-tasks, followed by the second value indicates the number of machines. These instances are randomly generated by MATLAB. However, we have taken some skewed meta-tasks. The structure of synthetic instances is partially inherited from Sherihan [19]. However, there is no relationship between the proposed and Sherihan data set. Note that, our synthetic instances contain inconsistent element. It means a machine M_w is best for a meta-task T_x, does not indicate that machine M_w is also best for another meta-task T_z.

Moreover, the proposed heuristic is evaluated using 12 benchmark instances. These instances were proposed by Braun et al. [6, 20]. Each instance has three things: 1) uniform distribution, 2) type of matrix: consistent, inconsistent and semi-consistent, 3) heterogeneity: hihi, hilo, lohi and lolo. The first two letters of heterogeneity indicate for meta-task followed by last two letters indicates for the machine.

We have used 512 × 16 data set instances. Here, the first value shows the number of meta-tasks and the second value show the number of machines.

B. *Performance Metrics*

We compare our proposed heuristic with the existing heuristics in terms of following two metrics:

1) *Makespan:* It is the maximum time in which all the meta-tasks are successfully executed [21]. Each machine has an individual makespan. But, the maximum value of all machines is referred as system makespan. A heuristic is efficient if it has very less amount of makespan.

2) *Machine utilization:* It is the time that the machines are busy. It depends on the makespan value [22-23]. A heuristic is efficient if it has less makespan and high machine utilization.

C. *Experiments*

The numerical results of makespan value are shown in Table IV. This table contains 12 synthetic instances. The result clearly shows that SBM^2 heuristic outperforms in all instances. The semilog bar chart representation of makespan value is shown in Figure 3. In this figure, x-axis denotes the synthetic instances and y-axis denotes the makespan.

TABLE IV. Comparison of Makespan in Synthetic Data set

Instance	Min-Min	Max-Min	SBM^2
50 × 5	1.2023E+04	9.8310E+03	**9.8240E+03**
50 × 10	1.0510E+04	**9.3010E+03**	**9.3010E+03**
50 × 15	1.0201E+04	**9.3000E+03**	**9.3000E+03**
100 × 5	1.5045E+04	1.1924E+04	**1.1857E+04**
100 × 10	1.2021E+04	**9.3030E+03**	**9.3030E+03**
100 × 15	1.0331E+04	**9.1900E+03**	**9.1900E+03**
1000 × 5	8.3626E+04	8.5492E+04	**7.9871E+04**
1000 × 10	4.1502E+04	4.0530E+04	**3.9133E+04**
1000 × 15	2.9294E+04	2.6468E+04	**2.5618E+04**
10000 × 5	7.1516E+05	7.0218E+05	**6.8262E+05**
10000 × 10	4.0632E+05	3.6500E+05	**3.5279E+05**
10000 × 15	7.5693E+05	**5.5530E+05**	**5.5530E+05**

The numerical results of average machine utilization value are shown in Table V. The bar chart representation of average machine utilization value is shown in Figure 4. In this figure, x-axis denotes the synthetic instances and y-axis denotes the average machine utilization.

The numerical results of makespan and the average machine utilization value of benchmark data set are shown in Table VI. This table contains 12 benchmark instances. The result clearly shows that SBM^2 heuristic outperforms in all instances. The semilog bar chart representation of makespan value and machine utilization is shown in Figure 5 and Figure 6 respectively. In this figure, x-axis denotes the benchmark instances and y-axis denotes makespan or average machine utilization.

TABLE V. Comparison of Machine Utilization in Synthetic Data set

Instance	Min-Min	Max-Min	SBM^2
50 × 5	0.7999	0.9838	0.9813
50 × 10	0.4382	0.4977	0.4971
50 × 15	0.2355	0.2593	0.2594
100 × 5	0.7731	0.9860	0.9866
100 × 10	0.4916	0.6416	0.6412
100 × 15	0.1809	0.2068	0.2064
1000 × 5	0.9442	0.9982	0.9985
1000 × 10	0.8826	0.9976	0.9966
1000 × 15	0.8121	0.9953	0.9944
10000 × 5	0.8891	0.9998	0.9998
10000 × 10	0.7993	0.9996	0.9997
10000 × 15	0.3310	0.5014	0.4873

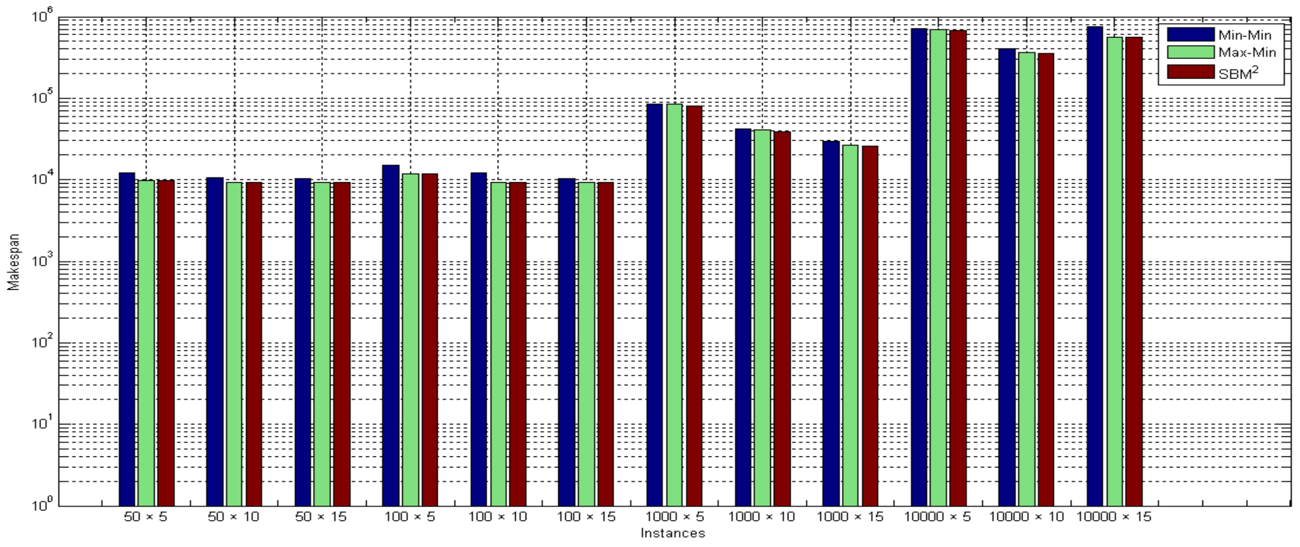

Figure 3. Graphical Representation of Makespan Values for Min-Min, Max-Min and SBM^2 Heuristics in Synthetic Data Set.

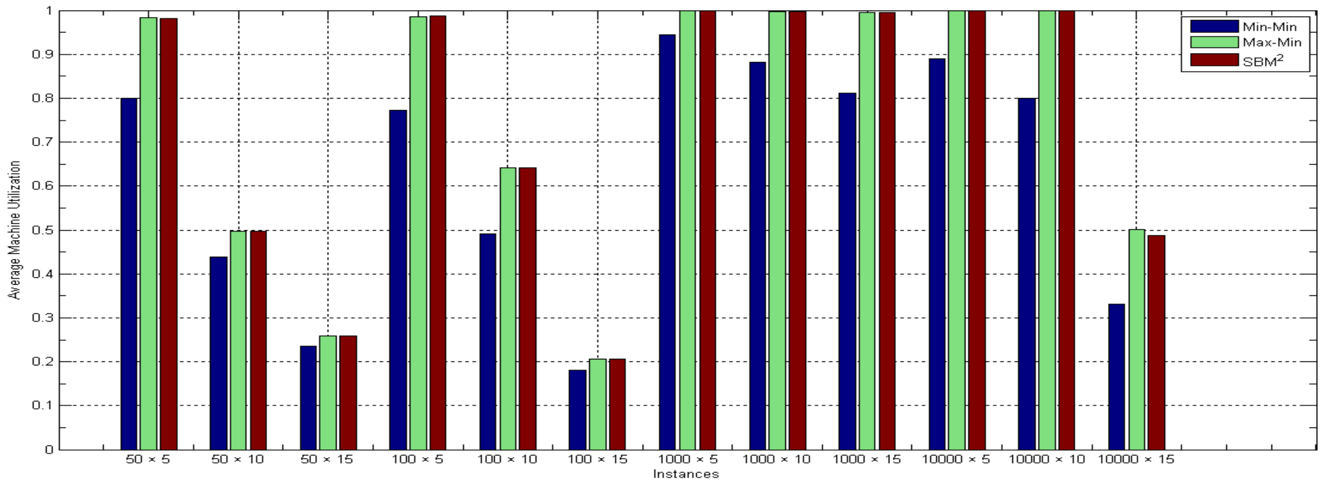

Figure 4. Graphical Representation of Average Machine Utilization Values for Min-Min, Max-Min and SBM^2 Heuristics in Synthetic Data Set.

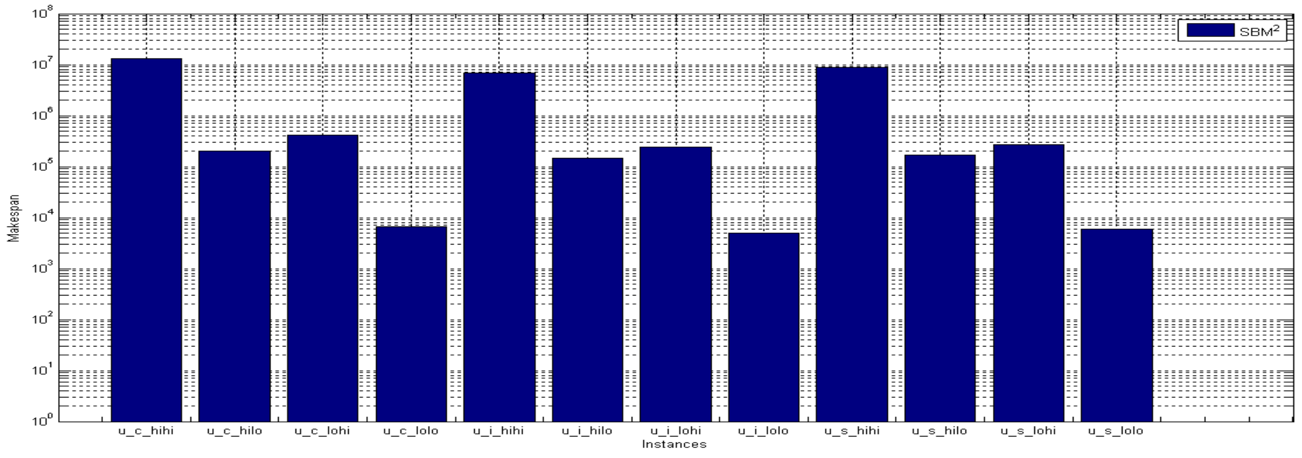

Figure 5. Graphical Representation of Makespan Values for SBM^2 Heuristic in Benchmark Data Set.

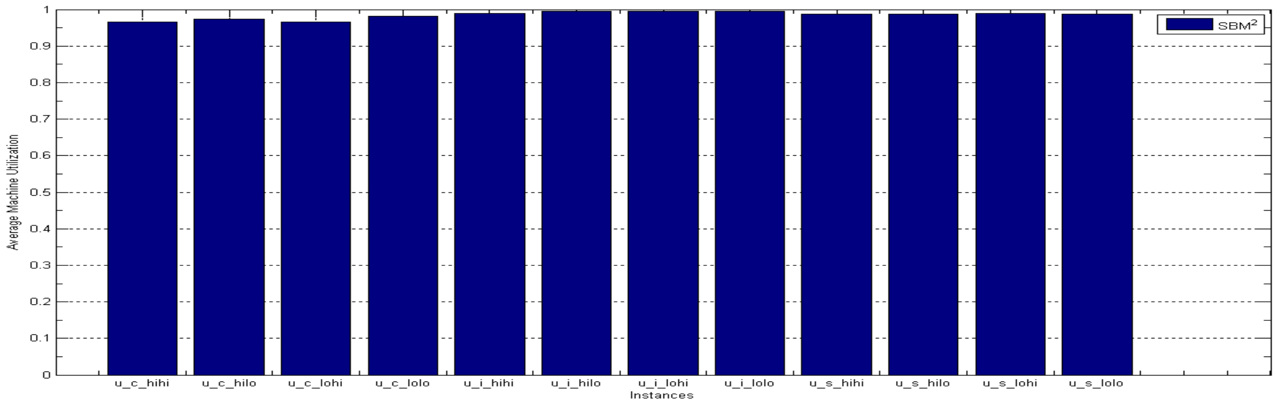

Figure 6. Graphical Representation of Average Machine Utilization Values for SBM^2 Heuristic in Benchmark Data Set.

TABLE VI. MAKESPAN AND MACHINE UTILIZATION VALUE FOR SBM^2 HEURISTIC IN BENCHMARK DATA SET

Instance	Makespan	Machine Utilization
u_c_hihi	1.3169E+07	0.9660
u_c_hilo	2.0356e+05	0.9732
u_c_lohi	4.1553e+05	0.9665
u_c_lolo	6.7253e+03	0.9807
u_i_hihi	6.7622e+06	0.9883
u_i_hilo	1.4412e+05	0.9955
u_i_lohi	2.4320e+05	0.9955
u_i_lolo	4.8814e+03	0.9953
u_s_hihi	8.7184e+06	0.9879
u_s_hilo	1.6973e+05	0.9874
u_s_lohi	2.6614e+05	0.9888
u_s_lolo	6.0238e+03	0.9882

VII. CONCLUSION

In this paper, we proposed the SBM^2 heuristic for the grid environment. The proposed heuristic uses the skewness measure to find out a near optimal solution. However, the proposed heuristic gives better result in comparison to existing Min-Min and Max-Min heuristics.

We have conducted several experiments to evaluate the two performance metrics i.e. makespan and machine utilization of the SBM^2 and other existing heuristics. The results showed that the proposed heuristic can achieve better performance in terms of makespan. The SBM^2 heuristic can also guarantee that it can easily handle skewed data set.

In the future, we will try to optimize the makespan of SBM^2 heuristic. Furthermore, we can develop new scheduling heuristics that can ensure fault tolerance without affecting the makespan.

REFERENCES

[1] R. Buyya, C. S. Yeo, S. Venugopal, J. Broberg and I. Brandic, “Cloud Computing and Emerging IT Platforms: Vision, Hype and Reality for Delivering Computing as the 5th Utility”, Future Generation Computer Systems, Elsevier, Vol. 25, pp. 599-616, 2009.

[2] I. Foster and C. Kesselman, “The Grid: Blueprint for a New Computing Infrastructure”, Morgan Kaufmann, Elsevier, 1999.

[3] W. C. Chung and R. S. Chang, “A New Mechanism for Resource Monitoring in Grid Computing”, Future Generation Computer Systems, Elsevier, Vol. 25, pp. 1-7, 2009.

[4] S. Zanikolas and R. Sakellariou, “A Taxonomy of Grid Monitoring Systems”, Future Generation Computer Systems, Elsevier, Vol. 21, pp. 163-188, 2005.

[5] T. D. Braun, H. J. Siegel, N. Beck, L. L. Boloni, M. Maheswaran, A. I. Reuther, J. P. Robertson, M. D. Theys, B. Yao, D. Hensgen and R. F. Freund, “A Comparison Study of Static Mapping Heuristics for a Class of Meta-tasks on Heterogeneous Computing Systems”, 8th Heterogeneous Computing Workshop, pp. 15-29, 1999.

[6] T. D. Braun, H. J. Siegel, N. Beck, L. L. Boloni, M. Maheswaran, A. I. Reuther, J. P. Robertson, M. D. Theys, B. Yao, D. Hensgen and R. F. Freund, “A Comparison of Eleven Static Heuristics for Mapping a Class of Independent Tasks onto Heterogeneous Distributed Computing Systems”, Journal of Parallel and Distributed Computing, Vol. 61, pp. 810-837, 2001.

[7] K. Etminani and M. Naghibzadeh, “A Min-Min Max-Min Selective Algorithm for Grid Task Scheduling”, 3rd IEEE / IFIP International Conference on Internet, 2007.

[8] S. Parsa and R. Entezari-Maleki, “RASA: A New Grid Task Scheduling Algorithm”, International Journal of Digital Content Technology and its Applications, Vol. 3, No. 2, pp. 91 – 99, 2009.

[9] S. K. Panda, S. K. Bhoi and P. M. Khilar, “A Semi-Interquartile Min-Min Max-Min (SIM^2) Approach for Grid Task Scheduling”, International Conference on Advances in Computing, Advances in Intelligent Systems and Computing, Springer, Vol. 174, pp. 415 – 421, 2012.

[10] A. Rasooli, M. Mirza-Aghatabar and S. Khorsandi, “Introduction of Novel Rule Based Algorithms for Scheduling in Grid Computing Systems”, Second Asia International Conference on Modelling & Simulation, IEEE, pp. 138 – 143, 2008.

[11] H. Decai, Y. Yuan, Z. Li-jun and Z. Ke-qin, “Research on Tasks Scheduling Algorithms for Dynamic and Uncertain Computing Grid Based on a+bi Connection Number of SPA”, Journal of Software, Vol. 4, No. 10, pp. 1102 – 1109, 2009.

[12] S. K. Panda, S. K. Bhoi and P. M. Khilar, “RRTS: A Task Scheduling Algorithm to Minimize Makespan in Grid Environment”, International Conference on Internet Computing and Information Communications, Advances in Intelligent Systems and Computing, Springer, Vol. 216, pp. 279 – 292, 2014.

[13] M. Maheswaran, S. Ali, H. J. Siegel, D. Hensgen and R. F. Freund, “Dynamic Mapping of a Class of Independent Tasks onto Heterogeneous Computing Systems”, Journal of Parallel and Distributed Computing, Vol. 59, pp. 107-131, 1999.

[14] Skewness, http://en.wikipedia.org/wiki/Skewness, Accessed on 20th December 2013.

[15] S. K. Bhoi, S. K. Panda and D. Tarai, “Enhancing CPU Performance using Subcontrary Mean Dynamic Round Robin (SMDRR) Scheduling Algorithm”, Journal of Global Research in Computer Science, Vol. 2, No. 12, pp. 17-21, 2011.

[16] S. K. Panda and S. K. Bhoi, “An Effective Round Robin Algorithm using Min-Max Dispersion Measure”, International Journal on Computer Science and Engineering, Vol. 4, No. 1, pp. 45-53, 2012.

[17] B. S. Grewal, Higher Engineering Mathematics, Khanna Publishers, Thirtynineth Edition, 2005.

[18] S. K. Panda, D. Dash and J. K. Rout, “A Group Based Time Quantum Round Robin Algorithm using Min-Max Spread Measure”, International Journal of Computer Applications, Vol. 64, No. 10, pp. 1-7, 2013.

[19] S. Abuelenin, “Trust Based Grid Batch Mode Scheduling Algorithms”, The 8th International Conference on INFOrmatics and Systems, pp. 46-54, 2012.

[20] Braun et al., https://code.google.com/p/hcsp-chc/source/browse/trunk/AE/ProblemInstances/HCSP/Braun_et_al/u_c_hihi.0?r=93, Accessed on 9th January 2014.

[21] Z. Jinquan, N. Lina and J. Changjun, “A heuristic Scheduling Strategy for Independent Tasks on Grid”, Eighth International Conference on High-Performance Computing in Asia-Pacific Region, IEEE, 2005.

[22] S. K. Panda, “Efficient Scheduling Heuristics for Independent Tasks in Computational Grids”, M. Tech. Thesis, National Institute of Technology, Rourkela, 2013.

[23] P. Agrawal, “A Novel Heuristic for a Class of Independent Tasks in Computational Grids”, M. Tech. Thesis, National Institute of Technology, Rourkela, 2013.