ISSN- 2350-0530(O), ISSN- 2394-3629(P)
DOI: https://doi.org/10.29121/granthaalayah.v5.i7.2017.2103

INTERNATIONAL JOURNAL OF RESEARCH – GRANTHAALAYAH
A knowledge Repository

Science

# IMMEDIATE/BATCH MODE SCHEDULING ALGORITHMS FOR GRID COMPUTING: A REVIEW

**J.Y Maipan-uku[1], I. Rabiu[2], Dr. Amit Mishra[3]**
[1, 2, 3] Department of Mathematics and Computer Science, Faculty of Natural Sciences, Ibrahim Badamasi Babangida University, Lapai, Niger State, Nigeria

## Abstract

Immediate/on-line and Batch mode heuristics are two methods used for scheduling in the computational grid environment. In the former, task is mapped onto a resource as soon as it arrives at the scheduler, while the later, tasks are not mapped onto resource as they arrive, instead they are collected into a set that is examined for mapping at prescheduled times called mapping events. This paper reviews the literature concerning Minimum Execution Time (MET) along with Minimum Completion Time (MCT) algorithms of online mode heuristics and more emphasis on Min-Min along with Max-Min algorithms of batch mode heuristics, while focusing on the details of their basic concepts, approaches, techniques, and open problems.

***Keywords:*** Grid Computing; Scheduling; MET; MCT; Min-Min; Max-Min.

***Cite This Article:*** J.Y Maipan-uku, I. Rabiu, and Dr. Amit Mishra. (2017). "IMMEDIATE/BATCH MODE SCHEDULING ALGORITHMS FOR GRID COMPUTING: A REVIEW." *International Journal of Research - Granthaalayah*, 5(7), 1-13. https://doi.org/10.29121/granthaalayah.v5.i7.2017.2103.

## 1. Introduction

Scheduling in (Kokilavani, 2011), is considered to be an important issue in the current grid concept. The need for efficient scheduling increases to achieve better performance computing. Typically, it is difficult to find an optimal resource allocation which minimizes the schedule length of jobs and efficiently utilizes the resources. The main phases of scheduling in a grid computing environment, according to (Amalarethinam, 2010) are; resource discovery, gathering resource information and application execution as shown in figure 1.1. The choice of the best pair of jobs and resources in the second phase proved to be NP-complete problem. In this paper, we have classified the existing research and provided a survey on the MCT, MET, Min-Min, and Max-Min grid scheduling algorithms by focussing on their concepts, techniques, weak points, and open problems. The main objectives of this paper are:

- To provide a comprehensive review of their (MCT, MET, Min-Min & Max-Min) literature.
- To discover open problems in the field of immediate/batch mode heuristics.

ISSN- 2350-0530(O), ISSN- 2394-3629(P)
DOI: 10.5281/zenodo.826213

- To summarize the existing research results for the different types of problem.

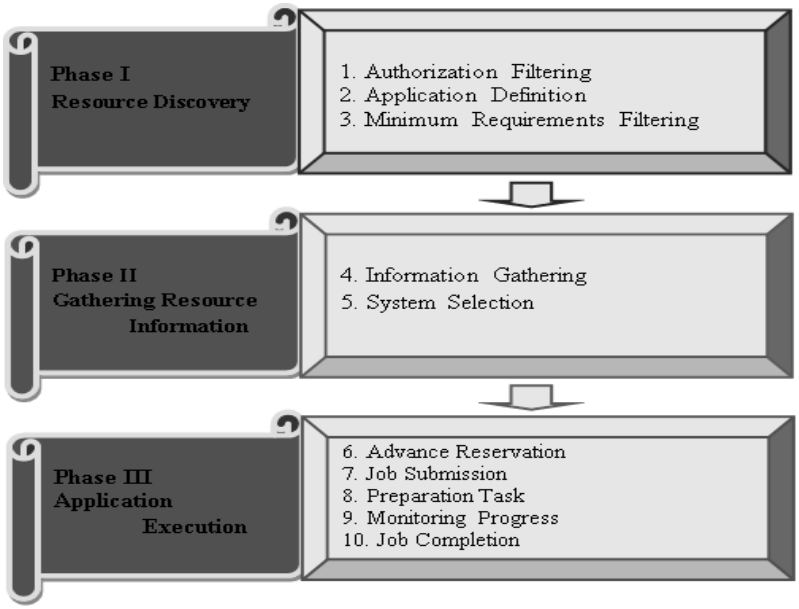


Figure 1.1: Grid Scheduling Infrastructure Organization

Immediate and batch scheduling is well-known methods, largely explored for many computing environments and different types of applications. They are also useful for Grid scheduling. In immediate mode, jobs are scheduled as soon as they arrive at the system, without waiting for the next time interval when the scheduler gets activated or the job arrival rate is small having thus available resources to execute jobs immediately. In batch mode, task's, jobs or applications are grouped in batches and scheduled as a group. Batch mode scheduling methods are simple and yet powerful heuristics that is distinguished for their efficiency. In contrast to immediate scheduling, batch scheduling could take better advantage of job and resource characteristics in deciding which job to allocate to which resource since they dispose of the time interval between two successive activations of the batch scheduler. Immediate scheduling methods include Opportunistic Load Balancing, Minimum Completion Time, Minimum Execution Time, Switching Algorithm and k-Percent best and between batch mode methods there are Min-Min, Max-Min, Sufferage, Relative Cost and Longest Job to Fastest Resource – Shortest Job to Fastest Resource (Mukherjee, 2011).

## 2. Concept of Immediate/Batch mode Scheduling Algorithms

### 2.1. Immediate Mode Scheduling Algorithms

#### *2.1.1. Minimum Execution Time (MET) also known as Limited Best Assignment (LBA)*

This algorithm finds the task which has minimum execution time and assigns to the resource that produces it less execution time. If two or more resources are taking the same execution time, then one of the resources is selected randomly (Panda et al., 2013). It served the task at first-come, first-served (FCFS or FIFO) basis. This algorithm does not consider the availability of the

ISSN- 2350-0530(O), ISSN- 2394-3629(P)
DOI: 10.5281/zenodo.826213

resource and its load. The time to find the resource with minimum execution time is 0 (nm) (Hemamalini, 2012). Table 2.1 gives a summary concept of the algorithm.

MET involved in several researches for comparison among other heuristics and hybrids (Maheswaran et al. 1999; Braun et al., 2001; Hak du & Suk 2001; Anand et al., 2011; and Hemamalini & Srinath, 2015).

### *2.1.2. Minimum Completion Time (MCT)*

This algorithm assigned task to a resource that takes its least completion time. If two resources are taking same completion time, then one of the resources is selected at random. Completion time is the sum of execution time (ET) and ready time (RT) of the resource as in equation 2.1. The time to find the resource with minimum completion time is 0 (nm). Unlike MET, MCT considers the resource ready time (Panda et al., 2013), it gives a better result than MET, and reduced load imbalance to some extent. This algorithm does not assign tasks to the overloaded resource.

Completion Time (Cij) = Execution Time (ETi) + Resource Ready Time (Rj) (2.1)

MCT takes more time to map a task to a particular resource, this implies that, the task may not be mapped to least execution time resources and it considers one task at a time. Therefore, leads to high makespan and poor resource utilization (Chaturvedi et al., 2011). This algorithm has been studied by different researchers; include (Freund et al., 1998; Xhafa et al., 2007; Kim & Kim, 2004; Braun et al., 2001; Tseng et al 2009).

Table 2.1: Summary of Immediate / Online Mode Scheduling Algorithms

| Algorithm | Studied by; | Techniques | Weak points |
|---|---|---|---|
| MET | (Braun et al., 2001; Xhafa et al., 2007; Hak du & Suk 2001) | Ensure that, each task is executed by its best resource | Does not consider the ready time of resources |
| MCT | (Freund et al., 1998; Xhafa et al., 2007; Kim & Kim, 2004; Braun et al., 2001; Tseng et al 2009) | Assigns a job to the resource with the earliest completion time | Load imbalance |

## 2.2. Batch Mode Scheduling Algorithms

### *2.2.1. Min-Min Scheduling Algorithm*

One of the most widely used batch mode algorithm for mapping independent tasks in the heterogeneous computing system is Min-Min algorithm (Pinel et al., 2012). This algorithm finds the task with minimum execution time and assigns to a resource that produces it minimum completion time. The ready time of the resource is updated. This procedure is repeated executed until all tasks are scheduled (Hemamalini, 2012).

ISSN- 2350-0530(O), ISSN- 2394-3629(P)
DOI: 10.5281/zenodo.826213

Min-Min algorithm concept implies, mapping task with minimum execution time first, to its best resource will fasten the overall execution so that, makespan is minimized (Braun et al., 2001 & Wu et al., 2000). This algorithm produces better makespan and resource utilization if the number of the heavier tasks exceeded that of smaller ones. Whereas, result to high makespan and poor resource utilization when smaller tasks are much more than heavier ones and load imbalance.

Because of it widely used for mapping independent tasks in the heterogeneous computing system among other several heuristic developed by different researchers, min-min undergone a series of change and involve in multiple comparisons among other's heuristics, we present a substantial review base on its involvement in; comparison, extension, quality of service, and load balancing.

#### *2.2.1.1. Comparison*

Earlier work on static heterogeneous computing, scheduling in (Freund et al., 1998) was introduced by Ibarra and Chul, five special algorithms were computed, Min-Min inclusive. They also studied two other strategies: i.e., when tasks need to be scheduled on just two resources, and when the resources are of the same attributes.

Due to its ability in making it likely to gain high-quality solutions in a suitable runtime, large number of researchers have worked and revealed the benefits of MinMin algorithm for heterogeneous computing scheduling. Some of these works include the following. (Maheswaran et al., 1999) review four heuristics for dynamic mapping of a Class of Independent Tasks to Heterogeneous Computing Systems include a Min-Min, (Braun et al., 2001) considered experimentally eleven algorithms for static scheduling in heterogeneous computing environments, this includes an extensive series of simple greedy constructive heuristic approaches and MinMin. Furthermore, (Fujimota et al., 2004) compared scheduling algorithms for independent coarse-grain tasks; among them include MinMin. Then, (Xhafa et al., 2007) have also evaluated several static scheduling strategies for allocations of jobs on resources using the batch mode method, including MinMin. Similarly, (Luo et al., 2007) analysed and compared a set of twenty greedy heuristics under different conditions.

#### *2.2.1.2. Extension*

However, researchers have also proposed several extensions to Min-Min or new algorithms with several points of contact with this heuristic. (Wu et al., 2000) introduced Segmented Min-Min that secretly related to Min-Min. In this algorithm, tasks are sorted according to some score function of the expected time to compute in all machines (it could be the maximum, minimum or average expected time to compute among all machines). Then, the ordered sequence is segmented into groups, and finally MinMin is applied to schedule the group of larger tasks, followed by the other group. Others includes; (Yu, X., & Yu, X., 2009; Liu et al., 2009; Hesam et al., 2009; Doreen et al., 2010; Kamalam & Bhaskaran, 2010; Bansal & Hota, 2011; Soheil & Mahmoud, 2013; Kaur & Patra, 2013; Kfatheen and Banu, 2014; Cao et al., 2014; Reda et al., 2014; Anousha et al., 2014; and Vijayalakshmi & Vasudevan, 2015).

ISSN- 2350-0530(O), ISSN- 2394-3629(P)
DOI: 10.5281/zenodo.826213

Table 2.2.1: Summary of Min-Min Scheduling Algorithm base on Extension

| Title / Year | Hypothesis | Weak point |
|---|---|---|
| Segmental Min-Min, by Wu et al. 2000 | Reduce load imbalance | Same as Min-Min when tasks are of same manner |
| Segmental Min-Min for Load Balancing by Yu, X., & Yu, X 2009 | Proper use of idle resources | Same as Min-Min when tasks are of same manner |
| Average Min-Min by Liu et al. 2009 | Efficient scheduling transaction-intensive grid workflows | Time consuming due to the idleness of host node |
| Double Min-Min by Doreen et al. 2010 | Optimal Tasks scheduling & Load balance to enhance system performance in Hyper cubic P2P Grid (HPGRID). | Load imbalance when task's status favour Min-Min |
| Min-Mean by Kamalam and Bhaskaran 2010 | To increase high throughput through low makespan and proper use of idle resources | Favour Max-Min only, but behave as Min-Min when tasks are of smaller lengths |
| Efficient Min-Min Algorithm by Bansal & Hota 2011 | Efficient Load balance & Makespan reduction | Reduce the idle time of resources, but increase Makespan especially when the resource is consistent |
| An Improve Min-Min by Soheil, 2013 | Efficient resource utilization and Minimizing Completion Time | Poor resource utilization, since it only considers completion time of resource ignoring its workload |
| Resource Allocation with improved Min-Min Algorithm by Kaur & Patra, 2013 | Efficient and Effective Resource Utilization | Behave same as Min-Min when no resource produces completion task less than makespan |
| Skewness, 2014 | Minimization of Makespan & Resource Utilization | Behave same as Min-Min or Max-Min when the set of task always favours one of them |
| An Efficient Task Allocation Algorithm in Grid (TAAG) by Kfatheen and Banu (2014) | Completion time, cost, makespan and load balancing | Delay due to much process and henceforth, can cause high makespan |
| OPT-Min-Min by Cao et al (2014) | Makespan, utilization and load balancing | Need to find the heavy loaded resource each time, means more complexity |
| Sort-mid scheduling algorithm by Reda et al (2014) | Resource utilization and makespan | Behave like a max - min |
| Static Batch Mode Heuristic Algorithm for Mapping Independent | Makespan, flow time, and fitness | Insufficient resource utilization |

ISSN- 2350-0530(O), ISSN- 2394-3629(P)
DOI: 10.5281/zenodo.826213

| | | |
|---|---|---|
| Tasks in Computational Grid by Vijayalakshmi and Vasudevan (2015) | | |

#### *2.2.1.3. Quality of Service*

On the other hand, researchers presented interesting extensions to traditional Min-Min is known as Quality of Service (QoS).

He et al., 2003, being the first to propose a QoS guided Min-Min algorithm that guarantees QoS requirements by certain tasks, believe that, some tasks may require high network bandwidth to exchange a large amount of data among processors, whereas others can be satisfied with low network bandwidth. Therefore, a task that required high bandwidth will be assigned to resource that produces high network bandwidth. Similarly, (Sharma & Bansal, 2012), considered QoS in two forms: Computational based and Communication based. Subsequently, quality of service min-min was continued by the other researchers (Singh & Suri, 2008; Liu & Lu, 2010)

#### *2.2.1.4. Load Balancing*

Meanwhile, other researchers put more effort on load balancing like Kokilavani et al., 2011, that proposed Load Balanced Min-Min (LBMM) to overwhelm the limits of traditional Min-Min. It is performed in stages of two. In the first stage Min-Min algorithm is applied and in the second stage tasks is rescheduled to effectively utilize the un-used resources. LBMM algorithm reduces the makespan and increases the resource utilization. Others are (Alharbi 2012; Ghosh et al., 2012; Minal et al., 2013; Chen et al 2013; Kfatheen et al., 2014; Maipan-uku et al., 2016).

Table 2.2.2: Summary of Min-Min Scheduling Algorithm, base on QoS and Load Balancing

| **Heuristics** | **Advantages** | **Disadvantages** | **References** |
|---|---|---|---|
| **QoS Min-Min** | Result to better makespan and resource utilization | Result to poor performance and load imbalance if task are of the same quality | Sharma et al 2012 And He et al. 2003 |
| **QoS Min-Min** | Improve makespan and resource utilization | Behave like Max-Min | Soheil & Mahmoud 2013 And Kfatheen et al 2014 |
| **LBMM** | Improve makespan and resource utilization | Behave like traditional min-min if task completion time is larger than the average resource completion time | Kfatheen et al 2014 And Chen et al 2013 |

#### *2.2.2. Max-Min Scheduling Algorithm*

Max-Min algorithm undergone very few extensions by researchers due to its capability of reducing idle time of resources. [Amalarethinam & Kfatheen, 2014] proposed Max-Min Average. In this algorithm, tasks are assembled like Max-Min at the first phase. For selecting a resource, mean of completion time (meanCT) is compared with resource completion time (CTj), and then if CTj is less than or equal to meanCT, task with maximum completion time is

ISSN- 2350-0530(O), ISSN- 2394-3629(P)
DOI: 10.5281/zenodo.826213

scheduled otherwise best maximum execution time is scheduled. Also (Devipriya and Ramesh, 2013) propose an improved Max-Min heuristic Model for task scheduling in Cloud Computing. In this algorithm, task with maximum execution time is assigned to resources that produce it minimum completion rather than assigning task with maximum completion time, to the resource which provides minimum execution time for the task as in Max-Min. Similarly, (Mao et al., 2014) present Max–Min Task Scheduling Algorithm for Load Balance in Cloud Computing, (Li et al., 2009) for details; others include (Kartal et al., 2014). Table 2.2 gives a summary of batch mode scheduling algorithms.

Table 2.2.3: Summary of Batch Mode Scheduling Algorithms

| Algorithm | Studied by; | Techniques | Weakpoint |
|---|---|---|---|
| Min-Min | (Braun et al., 2001; Plaszczak & Wellner, 2006; Xhafa et al., 2008; Sharma et al., 2012; Bardsiri & Hashemi, 2012) | Assign task with minimum execution time to resource that produce it minimum completion time | Gives high makespan if the smaller number of tasks exceeded the larger ones. |
| Max-Min | (Plaszczak & Wellner, 2006; Dong & Selim, 2006; Xhafa et al., 2008; Sharma et al., 2012; Hao & Liu, 2014) Elzeki et al., 2012) | Works similar to Min-Min scheduling algorithm, but schedules the larger task first | Gives high makespan if there are more heavy jobs than lighter ones |

However, some researchers have it in mind that, hybridizing Min-Min to Max-Min would yield a reasonable benefit to overcome their drawbacks. Parsa et al., 2009] introduced Resource Aware Scheduling Algorithm (RASA). In this algorithm, Min-Min is applied when the number of available machines is odd; otherwise Max-min is applied. (Etminani and Naghibzadeh, 2007) presented selective algorithm, (Gupta & Singh, 2012) has also proposed switcher algorithm that chooses between the two algorithms under a prescribed condition.

## 3. Immediate/Batch mode Scheduling Algorithms, Techniques

Let assume a grid environment, having two machines (R1 & R2) with four tasks (T1, T2, T3, and T4) to be executed. The Expected Time to Compute (ETC) matrix for the grid system is defined in the Table 3.1. Each resource produces its completion time in respect to available tasks (completion time implies the resource ready time plus task execution time). In table 3.1, resource one (R1) produces completion time of (4, 2, 5, and 3) for tasks (T1, T2, T3, and T4) while resource two (R2) produces completion time of (10, 13, 16, and 14) for tasks (T1, T2, T3, and T4) and finally, we applied the considered algorithms to map the tasks and determine their maximum finishing time (makespan) and average resource utilisation.

Table 3.1: Execution Table

| Task/Resources | $R_1$ | $R_2$ |
|---|---|---|
| $T_1$ | 4 | 10 |
| $T_2$ | 2 | 13 |
| $T_3$ | 5 | 16 |
| $T_4$ | 3 | 14 |

ISSN- 2350-0530(O), ISSN- 2394-3629(P)
DOI: 10.5281/zenodo.826213

From table 3.2 below, MET assigns all tasks to resource R1 leaving resource R2 unused (idle), with makespan of 14s. Similarly, Min-Min algorithm also assigns all tasks to resource R1 and leave resource R2 idle achieving a makespan of 14s. While, MCT assigns T1, T2, and T3 to R1 and T4 on resource R2 achieving makespan of 14s, However, Max-Min algorithm assigns tasks T3, T1 and T4 to resource R1 and task T2 to R2 with a makespan of 13s though make right use of the two resources of about 70% on R1 and 100% on resource R2 which assumed as the best algorithm of this scenario. Figure 1.2 & 1.3 demonstrates the results of their makespan and resource utilization for the scenario. We explain in the appendix the process of calculating makespan and average resource utilization

Table 3.2: Comparison of different heuristics in makespan, tasks scheduling and resource utilization

| **Tasks Scheduling** | | | | | |
|---|---|---|---|---|---|
| **Algorithms** | **$R_1$** | **$R_2$** | **Makespan** | **RU in %** | |
| | | | | **$R_1$** | **$R_2$** |
| MET | $T_1, T_2, T_3, T_4$ | *idle* | 14 | 100 | 0 |
| MCT | $T_1, T_2, T_3$ | $T_4$ | 14 | 60 | 100 |
| Min-Min | $T_2, T_4, T_1, T_3$ | *idle* | 14 | 100 | 0 |
| Max-Min | $T_3, T_1, T_4$ | $T_2$ | 13 | 70 | 100 |

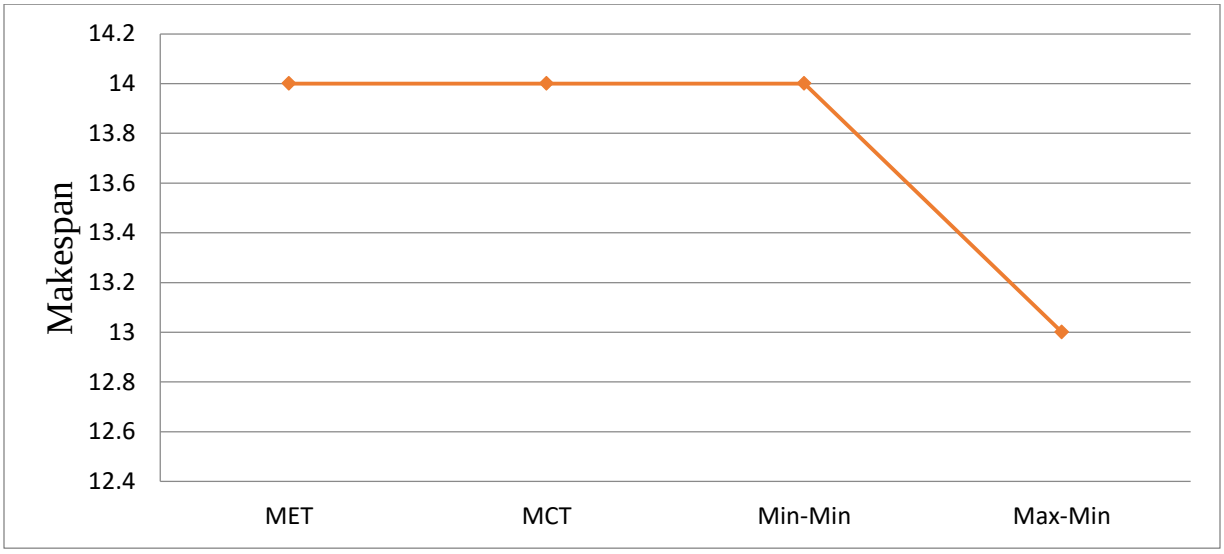


Figure 1.2: Comparison of makespan produced by different algorithms

### 3.1. Complexity

The complexity is an essential metric in theoretical analysis of algorithms that asymptotic estimates their performance. It determines the amount of time to solve the given computational problem using selected mathematical notation such as the Big O. In our case, it indicates how fast the scheduling algorithm will be in finding a feasible solution in a highly dynamic

ISSN- 2350-0530(O), ISSN- 2394-3629(P)
DOI: 10.5281/zenodo.826213

heterogeneous grid system. Table 3.3 illustrated the complexity of Immediate/Batch mode scheduling algorithms. It is worth remarking that the number of machines in a grid m is much less than the number of tasks n.

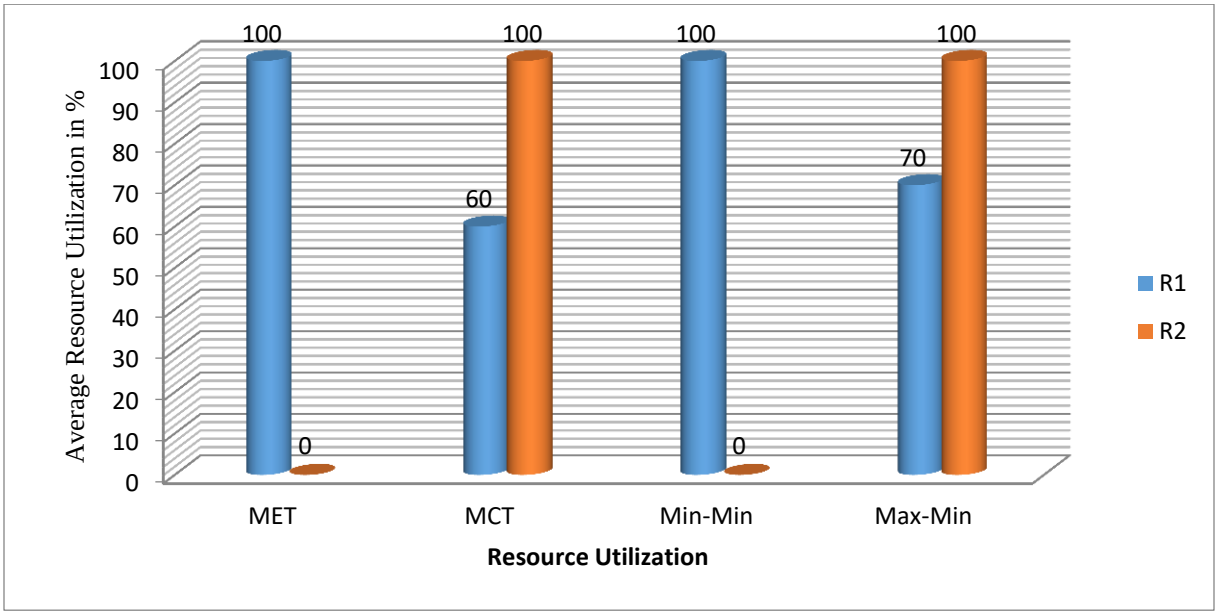


Figure 1.3: Comparison of Average resource Utilization produced by different algorithms

Table 3.3: Complexity of Immediate/Batch mode Scheduling Algorithms

| **Algorithm** | **MET** | **MCT** | **Min-Min** | **Max-Min** |
|---|---|---|---|---|
| **Complexity** | *O (nm)* | *O (nm)* | *O ($n^2m$)* | *O ($n^2m$)* |

## 4. Conclusion

Grid task scheduling had increased in throughput of available resources, as its major goal. Therefore, task scheduling is an important issue in the recent grid computing scenario, as such, an efficient task scheduling algorithm is needed to utilise the resource effectively and reduce the overall completion time. In this paper four different scheduling algorithms like Min-Min, Max-Min, MCT and MET have been reviewed. The study gave details findings about the four algorithms, summary of their techniques and weak points were given along side with a comprehensive illustrative example. This review can be used as a stepping stone for additional modification of the algorithms.

However, from all their viewed efforts, it shows that, these algorithms are commonly used by the community to solve scheduling problems. On the other hand, their solution shows that, mapping tasks to their best resources is an important challenge to this heuristic. For these reasons, a significant improvement in the computational efficiency of the algorithms could be welcome.

ISSN- 2350-0530(O), ISSN- 2394-3629(P)
DOI: 10.5281/zenodo.826213

## References

[1] Alharbi, F, (2012). Simple Scheduling Algorithm with Load Balancing for Grid Computing. Asian Transactions on Computers. 2, pp. 8-9.
[2] Amalarethinam, G.D.I. & Kfatheen V.S., (2014). Max-min Average Algorithm for Scheduling Tasks in Grid Computing Systems. International Journal of Computer Science and Information Technologies. 3, pp. 3659-62.
[3] Amalarethinam, D. G., & Mary, G. J. (2011). A new DAG based Dynamic Task Scheduling Algorithm (DYTAS) for Multiprocessor Systems. International Journal of Computer Applications, 19 (8), pp. 24-28.
[4] Anand, K.C., Rajendra, S., (2011). New Heuristic for Scheduling of Independent Tasks in Computational Grid. International Journal of Grid and Distributed Computing. 4(3), pp. 25-27.
[5] Anousha, S., Anousha, S., & Ahmadi, M. (2014). A New Heuristic Algorithm for Improving Total Completion Time in Grid Computing. In Multimedia and Ubiquitous Engineering, Springer Berlin Heidelberg, pp. 17-26.
[6] Bansal, S., & Hota, C. (2011). Efficient Algorithm on Heterogeneous Computing System. International Conference on Recent Trends in Information Systems, (78-1-4577-0792-6/11), pp. 57-61.
[7] Bardsiri, A. K., & Hashemi, S. M. (2012). A Comparative Study on Seven Static Mapping Heuristics for Grid Scheduling Problem. International Journal of Software Engineering and Its Applications, 6 (4), pp. 247-256.
[8] Braun, T. D., Siegel, H. J., and Beck, N., (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. 823 – 831.
[9] Cao, L., Liu, X., Wang, H., & Zhang, Z. (2014). OPT-Min-Min Scheduling Algorithm of Grid Resources. Journal of Software, 9 (7), pp. 1868-1875.
[10] Chaturvedi, A.K, & Sahu, R., (2011). New Heuristic for Scheduling of Independent Tasks in Computational Grid. International Journal of Grid and Distributed Computing. 4 (3), pp. 25-27.
[11] Chen, H., Wang, F., Helian, N. & Akanmu, G., (2013). User-Priority Guided Min-Min Scheduling Algorithm For Load Balancing in Cloud Computing. University of Kent, Canterbury, United Kingdom. (1 - 8), pp. 1-6.
[12] Devipriya, S., & Ramesh, C. (2013). Improved Max-Min Heuristic Model for Task Scheduling in Cloud. IEEE, pp. 883-888.
[13] Dong, F. & Selim, G. A. (2006). Scheduling Algorithms for Grid Computing: State of the Art and Open Problems. School of Computing, Queen's University, Kingston, Ontario. Technical Report No. 2006-504.
[14] Doreen, D., Miriam, H., & Easwarakumar, K. (2010). A Double Min Min Algorithm for Task Metascheduler on Hypercubic P2P Grid Systems. International Journal of Computer Science Issues, 7 (4).
[15] Etminani, K., Naghibzadeh, M., & Yanehsari, N.R., (2008). A Hybrid Min-Min Max-Min Algorithm with Improved Performance. Department of Computer Engineering, Ferdowsi University of Mashad, Iran., e.g. 32, pp.1 – 3.
[16] Freund RF, Gherrity M, Ambrosius S, Campbell M, Halderman M, Hensgen D. (1998). Scheduling resources in multi-user, heterogeneous, computing environments with SmartNet. HCW '98, Orlando, FL; pp. 184–99.
[17] Fujimoto, N., & Hagihara, K. (2004). A Comparison among Grid Scheduling Algorithms for Independent Coarse-Grained Tasks. 2(04), pp. 7-7.
[18] Ghosh, T., Goswami, R., Bera, S., & Barman, S. (2012). Load Balanced Static Grid Scheduling Using Max-Min Heuristic. 2nd IEEE International Conference on Parallel, Distributed and Grid Computing, pp. 419-423.

ISSN- 2350-0530(O), ISSN- 2394-3629(P)
DOI: 10.5281/zenodo.826213

[19] Gupta, K., & Singh, M., (2012). Heuristic Based Task Scheduling In Grid. International Journal of Engineering and Technology (IJET). 4, pp. 254 – 258.
[20] Hak Du, K., & Jin Suk, K., (2001). An On-line Scheduling Algorithm for Grid Computing Systems. Electronics and Telecommunications Research Institute, Taejon, Korea & School of Computer Science, University of Seoul, Seoul, Korea., e.g. 32, pp.2 – 6.
[21] HE. X, X-He Sun, & Laszewski. G.V., (2003). QoS Guided Min- Min Heuristic for Grid Task Scheduling. Journal of Computer Science and Technology. 18, pp. 442-451.
[22] Hemamalini, M., & Srinath, M. V. (2015). Memory Constrained Load Shared Minimum Execution Time Grid Task Scheduling Algorithm in a Heterogeneous Environment. Indian Journal of Science and Technology, 8 (15).
[23] Hemamalini, M., (2012). Review of Grid Task Scheduling in Distributed Heterogeneous Environment. International Journal of Computer Applications. 40 (24 - 30), pp. 24 – 26.
[24] Hesam, I., Ajith, A., senior member, IEEE, VS., (2009). Comparison of Heuristics for Scheduling Independent Tasks on Heterogeneous Distributed Environment. International Joint Conference on Computational Sciences and Optimization. (8 -12), pp. 8-10.
[25] Kamalam, G. K., & Bhaskaran, V. M. (2010). An improved Min-Mean heuristic scheduling algorithm for mapping independent tasks on heterogeneous computing environment. Journal of Computational cognition.
[26] Kartal, E., Tabak, B. B., C., and Cevdet, A., (2014). Improving the Performance of Independent Task Assignment Heuristics MinMin, MaxMin and Sufferage. Ieee Transactions on Parallel and Distributed Systems. 25 (1244 - 1256), pp. 1248 – 1249.
[27] Kaur, R., & Patra, P. (2013). Resource Allocation with improved MinMin Algorithm. International Journal of Computer Applications, 76 (15), pp. 61-67.
[28] Kfatheen, S.V., Banu, M.N., & Selvi, S.K., (2014). TAAG: An Efficient Task Allocation Algorithm for Grid Environment. International Journal of Engineering and Technology (IJET). 4 (1961 – 1969), pp. 1961-1968.
[29] Kfatheen, S.V., Banu, M.N., & Selvi, S.K., (2014). TLLB: Two-Level Load Balanced Algorithm for Static Meta-Task Scheduling in Grid Computing. International Journal of Computer Applications (0975 – 8887). 105 (38 - 43), pp. 38 – 41.
[30] Kim, H., & Kim, J. 2004. An online scheduling algorithm for grid computing systems, Grid and Cooperative Computing, pp. 34-39.
[31] Kokilavani, T. & Amalarethinam, D.I.G., (2011). Load Balanced Min-Min Algorithm for Static Meta-Task Scheduling in Grid Computing. International Journal of Computer Applications (0975 – 8887). 20 (2) (43 – 49), pp. 43-47.
[32] Liu, K., Chen, J., Jin, H., & Yang, Y. (2009). A Min-Min Average Algorithm for Scheduling Transaction-Intensive Grid Workflows. New Zealand. Conferences in Research and Practice in Information Technology, 99, pp. 1-8.
[33] Liu, L., & Li, G. (2010). An Improved MIN-MIN Grid Tasks Scheduling Algorithm Based on QoS Constraints. 201O International Conference on Optics, Photonics and Energy Engineering, 10 (978-1-4244-5236-1110), pp. 281-283.
[34] Luo, P., & Shi, Z. (2007). A revisit of fast greedy heuristics for mapping a class of independent tasks onto heterogeneous computing systems (K. Lü, Ed.). Journal of Parallel and Distributed Computing, 67 (0743-7315/$), pp. 695-714.
[35] Maheswaran, M., Ali, S., and Siegel, H. J., Hensgen, D., & Freund, F. R., (1999). Dynamic Mapping of a Class of Independent Tasks onto Heterogeneous Computing Systems1. Journal of Parallel and Distributed Computing. 59 (107 - 131), pp. 107 – 118.
[36] Maipan-uku, J.Y, Muhammed, A., Abdullah, A., Hussin, M. (2016). Efficient Loads Balancing for Grid Computing System Using Min-Min Scheduling Algorithm. International Journal of Applied Engineering Research (IJAER).

ISSN- 2350-0530(O), ISSN- 2394-3629(P)
DOI: 10.5281/zenodo.826213

[37] Mao, Y., Chen, X., & Li, X. (2014). Max–Min Task Scheduling Algorithm for Load Balance in Cloud Computing. Proceedings of International Conference on Computer Science and Information Technology, Advances in Intelligent Systems and Computing 255, pp. 457-465.
[38] Min-You Wu, M.U., Shu, W. & Zhang, H., (2000). A Static Mapping Algorithm for Meta-tasks on Heterogeneous Computing Systems. IEEE. 1 (1 - 11), pp. 1 – 6.
[39] Mrs. Minal, S., Prof. Rajesh, B., & Prof. Rupesh, M., (2013). Task Assignment in Heterogeneous Environment with Advanced Scheduling Algorithms. International Journal of Advanced Research in Computer Science and Software Engineering. 3 (1079 - 1083), pp. 1079 – 1082.
[40] Mukherjee, A. (2011). An Efficient Job-Grouping Based Scheduling Algorithm for Fine-Grained Jobs in Computational Grids (Doctoral dissertation).
[41] Panda, S.K., Agrawal, P., & Mohapatra, D.P., (2013). X-DualMake: Novel Immediate Mode Scheduling Heuristics in Computational Grids. Department of Computer Science and engineering Indian School of Mines, Dhanbad, India, IBM Bangalore, India, National Institute of Technology Rourkela, India. (1 – 6), pp. 1-2.
[42] Parsa, S and Reza, E. M., (2009). RASA: A New Task Scheduling Algorithm in Grid Environment. World Applied Sciences Journal 7 (Special Issue of Computer & IT). (152-160), pp. 152-155.
[43] Pinel, F., Dorronsoro, B., Pecero, J.E., Bouvry, P., & Khan, S.U. (2012). A two-phase heuristic for the energy-efficient scheduling of independent tasks on computational grids. Cluster Computer.
[44] Plaszczak, P., & Wellner, R. (2006). With Grid computing the savvy manager's guide. Amsterdam: Elsevier/Morgan Kaufmann. 1, pp. 5.
[45] Reda, N. M., Tawfik, A., Marzok, M. A., & Khamis, S. M. (2014). Sort-Mid tasks scheduling algorithm in grid computing. Journal of Advanced Research.
[46] Sharma, M.G., Bansal, P., (2012). Min-Min Approach for Scheduling in Grid Environment. International Journal of Latest Trends in Engineering and Technology (IJLTET). 1 (26 - 34), pp. 26.
[47] Singh. M and Suri. P.K, QPS., (2008). A QoS Based Predictive Max-Min, Min-Min Switcher Algorithm for Job Scheduling in a Grid. Information Technology Journal. 7 (1176 – 1181), pp. 1176-80.
[48] Soheil, A., & Mahmoud, A., (2013). An Improved Min-Min Task Scheduling Algorithm in Grid Computing. © Springer-Verlag Berlin Heidelberg., e.g. 32 (103 - 113), pp.103 – 106.
[49] Tseng, L-Y., Chin, Y-H., Wang, S-C., (2009). A minimized makespan scheduler with multiple factors for Grid computing systems. Expert Systems with Applications journal homepage: www.elsevier.com/locate/eswa. 36 (11118–11130), pp. 11120-28.
[50] Vijayalakshmi, R., & Vasudevan, V. (2015). Static Batch Mode Heuristic Algorithm for Mapping Independent Tasks in Computational Grid. Journal of Computer Science, 11 (1), pp. 224.
[51] Wu, M. Y., Shu, W., & Zhang, H. (2000, May). Segmented min-min: A static mapping algorithm for meta-tasks on heterogeneous computing systems. Inhcw. IEEE, pp. 375.
[52] Xhafa, F., Barolli, L., & Durresi, A. (2007). Batch mode scheduling in grid systems. International Journal of Web and Grid Services, 3 (1), pp. 19-19.
[53] Yu, X., & Yu, X. (2009, August). A new grid computation-based Min-Min algorithm. In Fuzzy Systems and Knowledge Discovery. FSKD'09. Sixth International Conference on. IEEE. 1, pp. 43-45.

*Corresponding author.
*E-mail address:* jymaipanuku@ibbu.edu.ng

ISSN- 2350-0530(O), ISSN- 2394-3629(P)
DOI: 10.5281/zenodo.826213

## Appendix

### 1) Makespan

Makespan is the time when finishes the latest task. This parameter shows the quality of assignment of resources from the executional time perspectives. It can be calculated as shown in equation 1.

If T = $t_1$, $t_2$, $t_3$…, $t_n$, is the set task submitted to the scheduler, and
R = $r_1$, $r_2$, $r_3$…, $r_n$, be the set of resources available at the time of task arrival,

***Makespan = max {completion [j] | j in Resource}*** **Eq. 1**

***Completion Time ($CT_{ij}$) = ($ET_{ij}$) + ($R_j$)*** **Eq. 2**

*Where $ET_{ij}$ is the expected execution time of the task $t_i$ on machine $m_j$ and rj is the ready time of mj i.e. the time when $m_j$ becomes ready to execute ti.*

### 2) Resource Utilization

Resource utilization (ru) is achieved by minimizing the idle time of a resource. This parameter shows the efficiency of an algorithm in keeping the available resources busy throughout the simulation time. Since the algorithms are mostly applicable to statics jobs, average resource utilization is considered. Equation 3 & 4 illustrates the pattern of calculating resource utilization (*ru*) and average resource utilization (*Avgru*) respectively.

$$ru = \sum \forall j, R_{ij=1} \frac{(R_{rt} - R_{it})}{makespan} * 100 \quad \textbf{Eq. 3}$$

*Where $R_{rt}$ and $R_{it}$ are resource running time and resource idle time*

$$Avgru = \frac{\sum_{i=1}^{n}(ru)}{n} \quad \textbf{Eq. 4}$$

*{n = number of resources}*