# OPT-Min-Min Scheduling Algorithm of Grid Resources

Lijun Cao
Hebei Normal University of Science & Technology
Email: misscao6666@163.com

Xiyin Liu, Haiming Wang, Zhongping Zhang
Hebei Normal University of Science & Technology, China
College of Information Science and Engineering Yanshan University, China
Email: liuxiyin2003@sina.com, phdwang@126.com, zpzhang@ysu.edu.cn

***Abstract*—As a newly emerged distributed computing platform, the Gird aims at the implementation of resources sharing and collaborative computing on the internet. Research on grid resource scheduling algorithm is one of the core problems of grid technology, as well as an essential part of grid resource management. Resource scheduling in grid environment refers to the application of tasks scheduling algorithm to assign the works in the grid to the right resources. During the allocation process, the computing performance of grid resource nodes, communication parameters and loading balance, etc., shall be put into consideration. Since they are all dynamic for both the grid resources and the grid resource nodes, requirements on grid resource scheduling algorithm are relatively higher if compared to those existing parallel resource scheduling algorithm.**

**Based on the Min-Min scheduling algorithm, the OPT-Min-Min scheduling algorithm is presented to overcome the drawbacks. Upon the basis of applying Min-Min pre-scheduling into stage one, by adapting the strategy of two-rounds scheduling, the assignment on heavy load resources are rescheduled to balance the load. Actual cases are used to illustrate the superiority of OPT-Min-Min scheduling algorithm to the Min-Min scheduling algorithm. Simulation experiments were carried out to the batch grid resource scheduling algorithms including Min-Min, Max-Min, Min-mean and OPT-Min-Min. According to the ETC generation technique used in experimental benchmarks, ETC matrix was generated. By the comparing and analyzing the Min-Min, Max-Min and Min-mean scheduling algorithm, the validity of OPT-Min-Min scheduling algorithm is proved.**

***Index Terms*—Grid; Scheduling Strategy; OPT-Min-Min; resource scheduling algorithm; load balance**

## I. SCHEDULING THE GRID RESOURCES

### *A Scheduling Strategy*

Scheduling strategy is an algorithm that Grid resource system employs for scheduling tasks submitted by users. All submitted tasks carry corresponding resource requirements. The grid system matches the requirements with status of available resources for the proper strategy of scheduling. By using an excellent grid resource scheduling strategy, tasks can be accomplished in minimum time while satisfying the user requirements. Both load balance and better resource utilization are achieved. In contrast, a strategy with poor performance can barely meet the minimum requirements from users. Besides, it causes unbalanced loading and lower utilization of resources. Thus the study of strategies with better performances comes in first hand when preparing for scheduling the resources, in the favor of low cost.

### *B The Process of Scheduling the Assignments*

An assignment refers to the task submitted by a user. It carries information including its ID, length, resource requirement and the status tag. There are three different status tags in total.

(1) PEND It means the assignment has been submitted but is waiting for scheduling and allocating.

(2) RUN It means the assignment has been allocated to a computation node, as in the process of scheduling.

(3) DONE It means the assignment has been successfully processed.

An assignment can be suspended due to requests e.g. applying for resources. A suspension comes in three different ways.

(1) PSUSP It means the assignment is suspended during PEND status, by administrator or user.

(2) USUSP/SSUSP It means the assignment is suspended after allocation, by the user or by the server.

A complete life cycle of an assignment starts from those stated stages, ends with a successful process (DONE) or an accidental stop (EXIT).

A complete life cycle of an assignment starts from those stated stages, ends with a successful process (DONE) or an accidental stop (EXIT). The figure 1 shows the transaction relations between all statuses within the process cycle.

The scheduling pattern employed by one grid system may vary from one to another, but the basic flow remains the same. It is consisted with accepting the assignment, scheduling the assignment via adopted grid resources scheduling algorithm, configuring the parameters of the chosen node, processing assignments on the chosen node, Etc..


doi:10.4304/jsw.9.7.1868-1875

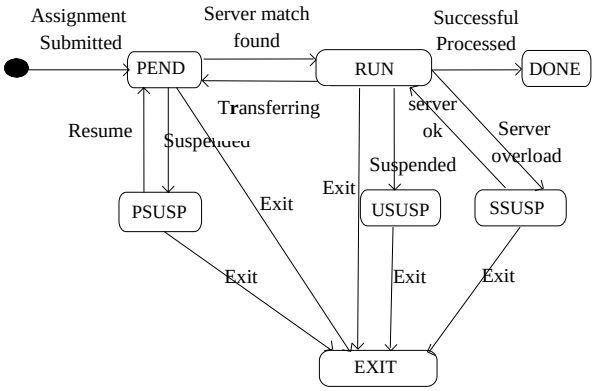


Figure 1.Execution state transition diagram

Node: Node is the smallest unit of job execution in the grid system. It carries both static and the dynamic properties. The static properties are those defined during the system initialization. They include: the system structure, operating system type, the number of processors, the processor performance (MIS), MEM memory size and SWAP space size etc.. Dynamic properties are those changing dynamically in the grid system. They include: CPU utilization rate, system page exchange rate, free time, free memory and swap space size, currently online users, and the number of operations is running, waiting or suspended.

Resources: There are a lot of heterogeneous resources in Grid. Resources can be configured by both the internal and external configurations. According to the number of available resources in one node, and the demand for resources from jobs, the system runs matching configuration to the resources.

Queue: A queue is a container for tasks, users, nodes and resources in the grid, as listed previously. A queue is a commonly used linear result. It is easy to define, simple to operate, easy to implement. Due to the large quantity of users, tasks and nodes in the grid work, the queues are employed here to manage the resources in the grid. Such method has been widely used in the management of resources in a grid system.

Scheduling strategy: Scheduling strategy is an algorithm that Grid resource system employs for scheduling tasks submitted by users. All submitted tasks carry corresponding resource requirements. The grid system looks between those requirements and the status of available resources, then chooses the proper strategy of scheduling. By using an excellent grid resource scheduling strategy, tasks can be accomplished in minimum time while satisfying the user requirements. Both load balance and better resource utilization are achieved at the same time. In contrast, a strategy with poor performance can barely meet the minimum requirements from users. Besides, it causes unbalanced loading and lower utilization of resources. Thus it becomes necessary to first study strategies with better performances when preparing for scheduling the resources, in the favor of low cost.

### *C The Valuation Standards for Scheduling the Grid Resources*

The evaluation of the scheduling algorithm' performance varies, when focus shifts. When system-oriented, the resource utilization and system throughput are the major standards. However they are replaced by user satisfactory when turns to user-oriented. In general four standards are employed here to evaluate a grid assignment scheduling algorithm. They are introduced below.

(1) Make span. Make span refers to the time length counts from when the first assignment starts to be processed, all the way down to when the last assignment is fully processed. Smaller the make span, faster the grid processes tasks and better performance the algorithm has. It is the most important index during the evaluation.

(2) QoS QoS defines the quality of the service requested by user, and provided by the grid system. It has seven subtypes – Scheduling QoS, Safety QoS, Credit QoS, Cost QoS, Computing resource QoS, Data resource QoS and Net resource QoS.

(3) Load Balance. During the scheduling, Load Balance remains as one of the key factor. In a parallel computing system, it poses a direct influence on both the resource utilization and system.

(4) Cost Saving. As requested by grid resources provider, users pay for the services they obtain from the grid. Lower the payment while maintain user's satisfactory is another achievement the scheduling algorithm shall count into consideration.

## II. THE MIN-MIN SCHEDULING ALGORITHM FOR GRID RESOURCES

### *A Min-Min Scheduling Algorithm*

The Min-Min scheduling algorithm is one of the classic in grid resource scheduling. Because the Min-Min scheduling algorithm is superior in performance, it is always used as the evaluation criteria for others. The Min-Min scheduling algorithm is simple in central idea and is easy to realize. When the users submit the tasks to the grid, the system picks the one with smallest minimum completion time. It is then assigned to the resource with earliest completion time. The initial goal of Min-Min scheduling algorithm is to minimize the completion time of entire task set. Considered from both performance and complexity of implementation, it is suitable for grid scheduling.

Under the presumptions, the breakdown of the process of Min-Min scheduling will be;

While ETC is known, to solve the MCT matrix.

Same as the ETC, MCT is another n*m matrix. While ETC stands for the execution time of n tasks on m resources, MCT stands for their expected execution time. The elements in MCT matrix is marked as $CT_{ij}$, derived from equation (1);

$$CT_{ij} = ET_{ij} + r_j + TT_{ij} \quad (1)$$

Where $ET_{ij}$ stands for the execution time of task i on resource j. $r_j$ stands for the available time length of this resource. $TT_{ij}$ stands for the transfer time length for data from storage to the computing resource, during the execution of task i.

Since the transfer from storage to computing resource is a must for every scheduling process of all algorithms, $TT_{ij}$ can be ignored during comparison. Then there is

$$CT_{ij} = ET_{ij} + r_j \quad (2)$$

(2) The MCT matrix gives those earliest completion times of each task. From them the task $T_k$ with minimum earliest completion time and the resource $R_f$ are both picked out.

(3) Assign task Tk to the resource Rf, then remove Tk from the MCT matrix. After that, update the available time of resource Rf.

(4) Repeat step (2) and (3) until all tasks are assigned. Now the MCT matrix is emptied.

The Min-Min scheduling algorithm 1 is given by the previous descriptions.

Algorithm 1 – The Min-Min Scheduling Algorithm

INPUT: All tasks $T_i$(i=1,2,…,m), resources $R_j$(j=1,2,...,n) and matrix ETC (m*n);

OUTPUT: The Makespan of completion time, Assignment Table.

Min-Min ($T_i$, $R_j$, ETC)

BEGIN

(1) FOR all task $T_i$
(2) FOR all resource $R_j$
(3) $CT_{ij}=ET_{ij}+r_j$
(4) END FOR
(5) END FOR
(6) DO UNTIL all tasks are mapped
(7) For every task, find its earliest completion time and the corresponding resource achieves that time.
(8) Find the earliest completion time for task $T_k$, assign it to the resource $R_f$ that achieves this time.
(9) Remove $T_k$ from the list.
(10) Update the preparation time of resource $R_f$.
(11) For all parameter i, update their $CT_{if}$
(12) END DO
(13) END

### B The Analysis to Min-Min Algorithm

As described there are four evaluation standards for grid scheduling algorithm; the load balancing, completion time, service quality and cost saving. The first two are most commonly used. They are as well applied to the Min-Min Scheduling Algorithm. Because the algorithm always gives priority to scheduling short assignments to resources with strong computation ability to minimize the completion time, those resources are easily overloaded. It then causes unbalance in load across over the entire grid.

## III. THE OPT-MIN-MIN SCHEDULING ALGORITHM

### A Introduction to the OPT-Min-Min Scheduling Algorithm

Min-mean algorithm is a Min-Min algorithm based on resource execution time average value meanCT. The algorithm balances system load with secondary scheduling based on a prime scheduling of Min-Min algorithm. It re-assigns the tasks on heavy load resources to the light load resources. The idea of Min-mean algorithm is mainly divided into two stages. The first phase covers a pre-scheduling by Min-Min algorithm. After the pre-scheduling, each resource carries its own execution time. The average value meanCT of all those execution time is then calculated. Any resource with execution time above the average meanCT is defined as a heavy load resource. On resources, the tasks are re-assigned to balance the load.

Although compared to the Min-Min algorithm, the Min-mean algorithm has a certain improvement on load balance; its scheduling performance is not that well as expected due to the heterogeneous nature of grid resources and tasks. It is even more noticeable in high heterogeneous situations. The OPT-Min-Min scheduling algorithm, proposed in this paper, using the same two step scheduling strategy to balance the load. But the result differs since it defines those heavy load resources in a different way.

OPT-Min-Min Algorithm is presented here as an optimized upgrade to the Min-Min Scheduling Algorithm. The OPT-Min-Min is divided into two stages when conducting. In stage one the Min-Min algorithm is employed to run pre-scheduling. In stage two, tasks on heavily loaded resources are moved to light loaded ones. In OPT-Min-Min process those resources obtained Makes pans from Min-Min are labeled as heavily loaded ones. In which Makespan=max(CTj).

(1) Extract the tasks set assigned to this heavily loaded resource, sort them ascending chronologically.

(2) Assume an reassignment of the lightest task in the set to all other resources, then update the execution time of this heavily loaded resource as well as other ones.

① If the max execution time of all resources is less than the Makespan, assign the task to the resource with minimum execution time.

② Of the time is larger than the Makespan, reconsider other tasks in the set. Reassign all available tasks to other resources.

During step ①②, update the execution time of all resources to rebating the Make span, if any task is successfully assigned.

(3) Repeat step (1) and (2), until no more tasks is available in the set with Makespan. Then terminate the process, return the final Makespan.

According to the central idea of OPT-Min-Min Scheduling Algorithm, the process is described as Algorithm 2.

Algorithm 2- the OPT-Min-Min Scheduling Algorithm.

INPUT: All tasks $T_i$(i=1,2,…,m), resources $R_j$(j=1,2,...,n) and matrix ETC(m*n);

OUTPUT: The Makespan of completion time, Assignment Table.

OPT-Min-Min ($T_i$, $R_j$, ETC)
BEGIN
// (1) – (12) The pre-scheduling with Min-Min algorithm, refer to Algorithm 1.
// Reschedule the tasks on overloaded resources.
(13)Sort all the resources ascending chronologically via execution time $CT_j$
(14) DO WHILE tasks are available in resource $R_{maxCT}$ with Makespan
(15) Sort all the tasks in $R_{maxCT}$ ascending chronologically via execution time
(16) FOR all tasks $T_{maxC\ T}$ assigned to resource $R_{maxCT}$
(17) For the remaining n-1 resources
(18) IF($CT_j$+ET<Makespan&&$CT_j$+ET is at its minimum)
(19) Reschedule T to resource $R_j$ and update the CT,
(20) END FOR
(21) IF the any task is successfully assigned, end the loop.
(22) END FOR
(23) Sort all the resources ascending chronologically via execution time $CT_j$
(24)END DO
(25)Output Makespan and Assignment table.
END

According to the OPT-Min-Min steps, stage one covers from (1) to (12), with pre-schedule by Min-Min algorithm. In this stage the load is not balanced and the resources' utilization is relatively low. The secondary scheduling, a.k.a the stage two, covers from step (13) to (24). In this stage all resources are sorted chronological ascending. Tasks set $T_{maxCT}$ is extracted from resource $R_{maxCT}$ with Make span. Referring to step (18), if Make span will decrease with assignment of task T in the set to other resources, such action will be conducted. If multiple resources meet the criteria at one time, the one with minimum execution time will catch the bid. Then redo the sorting, and repeat loop (14) – (24) until no task is available on resource $R_{maxCT}$. The final feedback is the final Make span.

## IV Implementation and Performance Analysis of OPT-Min-Min Algorithm

### *A The Datum of Experiments*

The presented scheduling algorithm runsthe grid simulation datum model. It wasintroduced by Braun Et al, widely applied in grid resouces scheduling algorithms. The ETC matrixis the most essential factor in this model. It contains the execution time for requested tasks on matching resources, within a certain time span.

ETC model are defined by three parameters: resource heterogeneity, task heterogeneous, and consistency. The lows in the ETC matrix stand for the resource heterogeneity. It is measured by the execution time of a given task on all the resources. For example, a grid system is low in heterogeneity if it is composed only of resources with similar properties. Or it will be high if it is composed of resources with different properties. A column in the ETC matrix stand for the task heterogeneity. It is measured by the execution time when all tasks are executed on single one resource. The high task heterogeneity may appear when the calculation requirements of tasks come in with large variation. On the contrary, when they come in with similar complexity, the task heterogeneity is low.

The presented scheduling algorithm uses the grid simulation datum model. It is introduced by Braun Et al, widely applied in grid resouces scheduling algorithms. The most essential factor in the model is the ETC matrix. The matrix contains the execution time for requested tasks on matching resources, within a certain time span.

There are four classifications in ETC matrix:

(1) High heterogeneity in tasks and high heterogeneity in resources; (HTHR)

(2) High heterogeneity in tasks and low heterogeneity in resources; (HTLR)

(3) Low heterogeneity in tasks and high heterogeneity in resources; (LTHR)

(4) Both Low heterogeneity in tasks and resources; (LTLR)

ETC matrix can be further divided into two types, consistent and inconsistent. This classification goes orthogonal with the four above. A consist ETC matrix represents the status of low heterogeneity in tasks and high heterogeneity in resources. Such matrix satisfies the criteria when task $T_i$ on resource $R_j$ is faster than $R_k$, the $R_j$ performs all tasks faster than $R_k$. A inconsistent ETC matrix represents a less obvious heterogeneity in tasks and resources, when compared to consist ones. In an inconsistent ETC matrix the tasks on resource $R_j$ are partially faster than $R_k$, while some others are slower than $R_k$. The mixture of consistent and inconsistent ETC leads to semi-consistent matrix. In details it means there is a consistent sub-matrix contained in the inconsistent matrix. The sub-matrix can be built with subsets of rows and columns.

In summary, in such datum model, according to the heterogeneity of the tasks and resources in the grid, the consistency of ETC matrix is divided into 12 different types.

Serial u-x-yyzz represents those 12 types.

(1) ETC matrix is derived under uniform distribution.

(2) xx-The type of ETC matrix. (c – consistent, s – semiconsistent, i - inconsistent)

① The ETC matrix is consistent. Such matrix satisfies the criteria when task Ti on resource $R_j$ is faster than $R_k$, the Rj performs all tasks faster than $R_k$.

② The ETC matrix is semi-consistent. A consistent sub-matrix is contained in the ETC matrix.

③ The ETC matrix in inconsistent. With in the matrix those tasks on resource R are partially faster than $R_k$, while some others are slower than $R_k$.

(3) yy-The heterogeneity of tasks. (hi – Hign heterogeneity; lo – Low heterogeneity).

The heterogeneity of tasks stands for the variation of execution time for a given resource.

(4) zz-The heterogeneity of resources. (hi – Hign heterogeneity; lo – Low heterogeneity).

The heterogeneity of resouces stands for the variation of execution time for a given task on all resources.

### *B The Generation of ETC Matrix*

The generation of a ETC matrix is defined with its heterogeneity from the view of mathematics. Commonly used methods are Vector based generation, variable coefficient generation, etc. In this paper the Vector based generation is used. The introduction to ETC matrix generation starts from the one for non-consistent ETC matrix. Both the consistent and semi-consistent ones can be produced from the non-consistent ones.

In an actual heterogeneous grid resources system, the variation of load requested by the tasks is far greater than the one of resources. Therefore, assume the high heterogeneity value of tasks is greater than the one of resources, while the value of low heterogeneity of them remains same. In the simulation datum,The typical values of heterogeneity $R_{task}$ of task and the heterogeneity $R_{mach}$ of resource are both given, as shown in table 2.

TABLE 2.,
THE HETEROGENEITY OF TASKS

| | high | low |
|---|---|---|
| task | $10^5$ | 10 |
| machine | $10^2$ | 10 |

In the generation of non-consistency matrix generation, assume n is the number of resources, m is the quantity of tasks needed to be executed in the grid of a certain time. U(a,b) is a sample grabbed from the uniform-distributed space [a,b]. Rtask and Rmach are the heterogeneities of tasks and recourses, respectively. The generation of ETC matrix is shown with Algorithm 3.

Algorithm 3, the generation of the ETC matix

Input: number of tasks m, number of resources n, the heterogeneity of tasks $R_{task}$, the heterogeneity of resouces $R_{mach}$;

Output: the ETC matrix

GenerateETC(m, n, $R_{task}$,$R_{mach}$)

For i from 0 to (m-1)

$\tau$ [i] = U(1, $R_{task}$)

For j from 0 to (n-1)

E[i,j] = $\tau$ [i]× U(1, $R_{mach}$)

End for

End for

Seen from the code, each iteration extracts a sample from the uniform-distributed space. A vector $\tau$ is generated, ranging within (1, $R_{task}$).By again apply iteration to each element in the $\tau$ [i], the row vectors of ETC matrix are generated.The elements in such vectors are formulated by multiplying the samples with $\tau$ [i].

In a generation mode based on distance, when the value ranges of $R_{task}$ and $R_{mach}$ are same, the generation paths of HTLR ETCs and LTHR ETCs will be similar to each other. But the values of Rtask and Rmach incline to be different in the generation mode based on vector. The genration of LTHR ETCs are more like to have the features of HTHR ones. Therefore, in the generation process of LTHR ETCs, applying transpose to HTLR ones is an effective way. As shown with the Algorithm 4.

Algorithm 4, the generation of the transposed ETC matix

Input: number of tasks m, number of resources n, the heterogeneity of tasks Rtask, the heterogeneity of resouces Rmach;

Output: the ETC matrix

TransGenerateETC(m, n, Rtask,Rmach)

For j from 0 to (n-1)

$\tau$ [j] = U(1, $R_{mach}$)

For i from 0 to (m-1)

E[i,j] = $\tau$ [j]× U(1, $R_{task}$)

End for

End for

The stated code generates all four class of inconsistent ETC matrix. The generation of consistent ETC can done by sorting through each row of the matrix. For the semi-consistent ETCs, since they carry sub-matrix of i*k, their generation can be done by sorting through the random sub-matrix of k resources and i tasks.

### *C The Evaluation of Algorithm's Performance*

In the heuristic algorithm for grid resource scheduling, Min-Min algorithm and Max-Min algorithm are the classical batch scheduling algorithms. OPT-Min-Min and Min-mean scheduling algorithm are improved on the basis of Min-Min scheduling algorithm. Both of them adhere to the same family tree. In this paper the experiment is carried on the datum of simulation model proposed by Braun et al. On this datum, the ETC matrix is set to be generated from the 512 tasks and 16 resources, through the method presented previously.

(1)High heterogeneity in tasks and high heterogeneity in resources

Known from Section B, the value of high heterogeneity of tasks $R_{task} = 10^5$, the value of hignheterogeneity of resources $R_{mach}=10^2$. Then the results from Batch process algorithm for grid system –the OPT-Min-Min scheduling algorithm, the Min-Min scheduling algorithm, the Max-Min scheduling algorithm, the Min-Mean scheduling algorithm; are all given in Table 3.

Figure 2 is derived from the results from Table 3.

TABLE 3.
RESULTS COMPARISON FOR OPT-MIN-MIN ALGORITHM AND OTHER ALGORITHMS U_HIHI

| Instances | OPT-Min-Min ($\times10^7$ms) | Min-Min ($\times10^7$ms)) | Max-Min ($\times10^7$ms) | Min-Mean ($\times10^7$ms) |
|---|---|---|---|---|
| u_c_hihi | 3.1348648 | 3.2693322 | 4.583936 | 3.2260172 |
| u_s_hihi | 1.2843172 | 1.3765 | 2.7531366 | 1.2944868 |
| u_i_hihi | 1.3035405 | 1.4579022 | $2.81784^7$ | 1.3793677 |

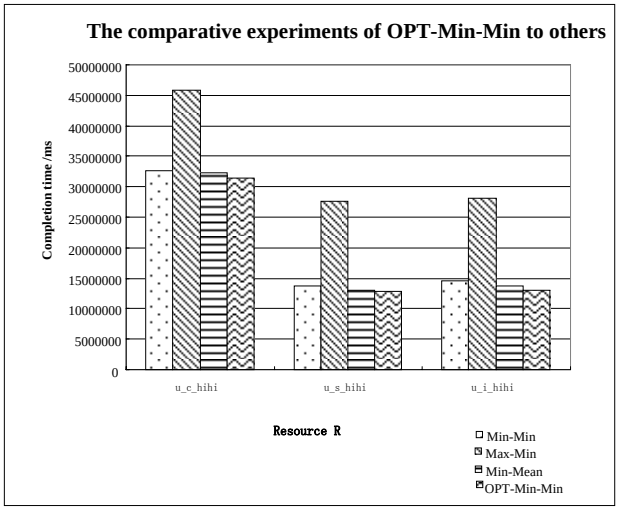


Figure 2. The experiment results of u_hihi

(2)High heterogeneity in tasks and low heterogeneity in resources

Known from Section B, the value of high heterogeneity of tasks $R_{task}$ = $10^5$ , the value of low heterogeneity of resources $R_{mach}$=10. Then the results from Batch process algorithm for grid system – the OPT-Min-Min scheduling algorithm, the Min-Min scheduling algorithm, the Max-Min scheduling algorithm, the Min-Mean scheduling algorithm; are all given in Table 4.

TABLE 4.
RESULTS COMPARISON FOR OPT-MIN-MIN ALGORITHM AND OTHER ALGORITHMS U_HILO

| Instances | OPT-Min-Min ($\times10^6$ms) | Min-Min ($\times10^6$ms) | Max-Min ($\times10^6$ms) | Min-Mean ($\times10^6$ms) |
|---|---|---|---|---|
| u_c_hilo | 4.973211 | 5.274453 | 6.540184 | 5.037062 |
| u_s_hilo | 2.842350 | 2.947291 | 5.334912 | 2.9298985 |
| u_i_hilo | 2.676013 | 2.797357 | 5.1531965 | 2.7932002 |

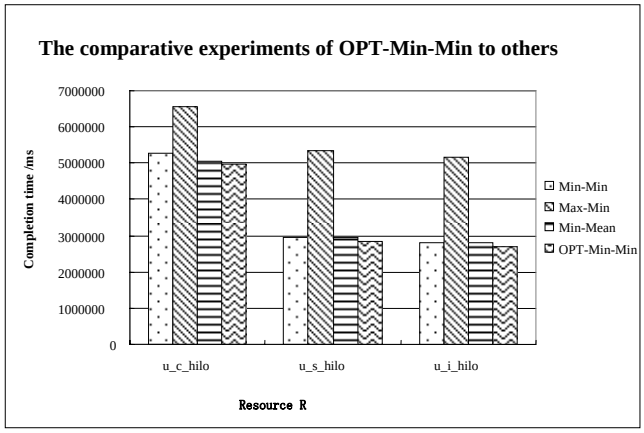


Figure 3. The experiment results of u_hilo

Figure 3 is derived from the results from Table 4.

(3)Low heterogeneity in tasks and high heterogeneity in resources

Known from Section B, the value of low heterogeneity of tasks $R_{task}$ = 10, the value of high heterogeneity of resources $R_{mach}$=102. Then the results from Batch process algorithm for grid system – the OPT-Min-Min scheduling algorithm, the Min-Min scheduling algorithm, the Max-Min scheduling algorithm, the Min-Mean scheduling algorithm; are all given in Table 5.

TABLE 5.
RESULTS COMPARISON FOR OPT-MIN-MIN ALGORITHM AND OTHER ALGORITHMS U_LOHI

| Instances | OPT-Min-Min ($\times10^3$ms) | Min-Min ($\times10^3$ms) | Max-Min ($\times10^3$ms) | Min-Mean ($\times10^3$ms) |
|---|---|---|---|---|
| u_c_lohi | 3.8217134 | 3.8765923 | 5.701304 | 3.829216 |
| u_s_lohi | 2.0907397 | 2.1972473 | 4.2852617 | 2.1578416 |
| u_i_lohi | 1.5806067 | 1.619184 | 3.196768 | 1.619184 |

Figure 4 is derived from the results from Table 5.

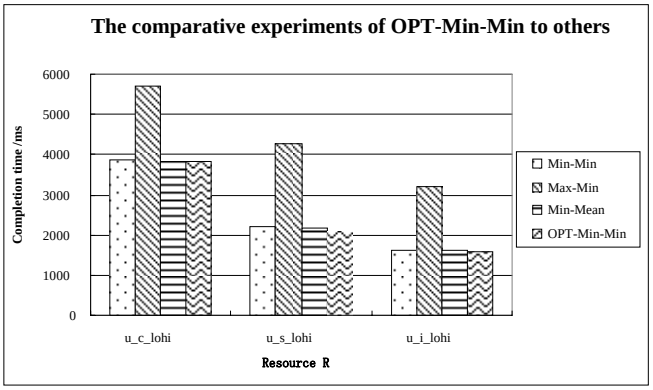


Figure 4. The experiment results of u_lohi

(4)Low heterogeneity in tasks and low heterogeneity in resources

Known from Section B, the value of high heterogeneity of tasks $R_{task}$ = 10, the value of low heterogeneity of resources $R_{mach}$=10. Then the results from Batch process algorithm for grid system – the OPT-Min-Min scheduling algorithm, the Min-Min scheduling algorithm, the Max-Min scheduling algorithm, the Min-Mean scheduling algorithm; are all given in Table 6.

TABLE 6.
RESULTS COMPARISON FOR OPT-MIN-MIN ALGORITHM AND OTHER ALGORITHMS U_LOLO

| Instances | OPT-Min-Min ($\times10^2$ms) | Min-Min ($\times10^2$ms) | Max-Min ($\times10^2$ms) | Min-Mean ($\times10^2$ms) |
|---|---|---|---|---|
| u_c_lolo | 5.6009717 | 5.825626 | 7.3758966 | 5.761336 |
| u_s_lolo | 3.0420825 | 3.2438898 | 5.8201276 | 3.172402 |
| u_i_lolo | 2.912936 | 3.0633313 | 5.786073 | 3.0356866 |

Figure 5 is derived from the results from Table 6.

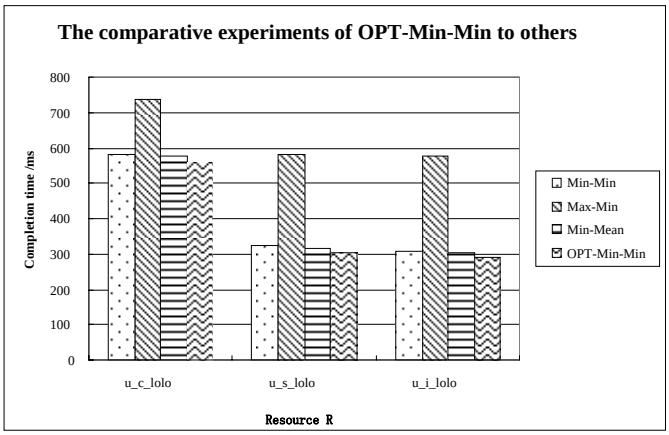


Figure 5. The experiment results of u_lolo

### *D Analysis to the Comprehensive Performance*

Sum from the experimental results, the performance of Min-Min scheduling algorithm is obviously superior to the Max-Min scheduling algorithm in batch scheduling mode, under all four heterogeneity classes. For a tasks set submitted from user to the grid system, the Makespan of Min-Min scheduling algorithm is significantly smallerthan the one of Max-Min scheduling algorithm. However due to Min-Min scheduling algorithm tends to schedule short tasks to resources with strong computing abilities, those resources get overload, while the resources with lower computing abilities stay idle. The unbalanced overall system load soon affects the throughput of the whole grid system.

Compared to the Min-Min scheduling algorithm, the Min-mean scheduling algorithm shows improvement in dealing with the load balance, as the Makespan turns smaller. However in such algorithm, the executions of heavy load resources are greater than the average execution time of all resources. Reassignment is needed for those heavy load resources. Due to the existence of heterogeneity in both tasks and resources in the grid, such definition of heavy load resources is not reasonable. In the situation where resources poss high heterogeneity, the performance of Min-mean scheduling algorithm is close to Min-Min scheduling algorithm. As for OPT-Min-Min scheduling algorithm, it defines those resources with Makespan as heavy load ones. Tasks on them are reassigned to light load resources to obtain Makespan. The reassignment repeats until the Makespan reaches its minimum. That is why the performance of OPT-Min-Min is superior to Min-Min, Max-Min and the Min-mean algorithms.

## V CONCLUSIONS

Grid resource scheduling algorithm is one of the core problems in the realmof grid technology. It keeps attracting attention from researchers. According to the characteristics of dynamic, heterogeneousityand distributionin the grid resources, the grid resource scheduling algorithm is presented. As stated, the Min-Min scheduling algorithm always tends to schedule short tasks to resources with strong computation ability, causing unbalanced load and low utilization rate of resources. The OPT-Min-Min scheduling algorithm is presented to solve this problem. It is also ranunder asimulation test, by using the datum simulation model proposed by Braun et al. The superiority of OPT-Min-Min algorithm is validated by comparing side by side with Min-Min scheduling algorithm, Max-Min scheduling algorithm and Min-mean scheduling algorithm.

## ACKNOWLEDGMENT

This work was financially supported by the Project of Major Reform to the Network Engineering Programmer in Hebei Normal University of Science and Technology, which is assigned by Hebei Education Department in 2012.

## REFERENCES

[1] Li Chunlin, Li Layuan. QoS based resource scheduling by computational economy in computational grid [J]. Information Processing Letters, 2006, 98,119-126.

[2] Yin Fei, Jiang Changiun, Deng Rong, et a1.Grid resource management policies for load-balancing and energy-saving by vacation queuing theory[J].Computers and Electrical Engineering, 2009, 35,966-979.

[3] Tracy D.Braun, Howard Jay Siegel, Noah Beck, A Comparison of Eleven Static Heuristics for Mapping a Class of Independent Tasks onto Heterogeneous Distributed Computing Systems[J], Journal of Parallel and Distributed Computing ,2001,61(6):810-837.

[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 System[C], //In 8th IEEE Heterogeneous Computing Workshop(HCW'99), 1999,15-29.

[5] Kanali Gupta, Manpreet Singh et al.Heuristic Based Task Scheduling In Grid [J]. International Journal of Engineering and Technology, 2012, 4(4):254-260.

[6] T.Kokilavani,Dr.D.l.George Amalarethinam.Load Balanced Min-Min Algorithm for Static Meta-Task Scheduling in Grid Computing[J] .International Journal of Computer Applications,2011,20(2):43-49.

[7] Kamalam.G.K, Murali Bhaskaran.V. A New Heuristic Approach: Min-mean Algorithm for Scheduling Meta-Tasks on Heterogeneous Computing Systems [J].IJCSNS International Journal of Computer Science and Network Security, 2010, 10(1):24-31.

[8] Kamalam.G.K, Murali Bhaskaran.V. An Improved Min-Mean Heuristic Scheduling Algorithm for Mapping Independent Tasks on Heterogeneous Grid Environment [J]. INTERNATIONAL JOURNAL OF COMPUTATIONAL COGNITION, 8(4) , 2010,12,81-95.

[9] Kamalam.G.K, Murali Bhaskaran.V. New Enhanced Heuristic Min-mean Scheduling Algorithm for Scheduling Meta-Tasks on Heterogeneous Grid Environment[J].European Journal of Scientific Research ISSN 1450-216X.2012,70(3):423-430.

[10] D.Doreen Hephzibah Miriam, K.S.Easwarakumar. A Double Min-Min Algorithm for Task Metascheduler on Hypercubic P2P Grid Systems[J].International Journal of Computer Science Issues, 2010,7(4), 8-18.

[11] Anand K Chaturvedi, Rajendra Sahu. New Heuristic for Scheduling of Independent Tasks in Computational Grid [J]. International Journal of Grid and Distributed Computing, 2011, 4(3), 25-36.
[12] *Ping Jin, Shichao Qu, Yu Zong, Xin Li,* CUDAP: A Novel Clustering Algorithm for Uncertain Data Based on Approximate Backbone, Journal of Software, Vol 9, No 3 (2014), 732-737, Mar 201
[13] *Lun Gao, Taifu Li, Lizhong Yao, Feng Wen, Research and* Application of Data Mining Feature Selection Based on Relief Algorithm, Journal of Software, Vol 9, No 2 (2014), 515-522, Feb 2014
[14] Le Wang, Lin Feng, Mingfei Wu, UDS-FIM: An Efficient Algorithm of Frequent Itemsets Mining over Uncertain Transaction Data Streams, Journal of Software, Vol 9, No 1 (2014), 44-56, Jan 2014
[15] *Jiechang Wen, Suxian Zhang, Junjie Yang,* A Fast Algorithm for Undedetermined Mixing Matrix Identification Based on Mixture of Guassian (MoG) Sources Model, Journal of Software, Vol 9, No 1 (2014), 184-189, Jan 2014
[16] Jun Wang, WeiRu Chen," A Reliability-oriented Evolution Method of Software Architecture Based on Contribution Degree of Component ", Journal of Software, Vol 7, No 8 (2012), pp.1744-1750, Aug 2012
[17] Qinghua Lu, Phillip John McKerrow, Zhiquan Zhou," Design and Performance of a Minimal Real-Time Operating System in a Safe Language: Experience with Java on the Sun SPOT ", Journal of Software, Vol 7, No 8 (2012), pp.1835-1844, Aug 2012
[18] Mingcheng Qu, Xiang-hu Wu, Xiao-zong Yang," A Comprehensive Optimization Model Based on Time and Cost Constraints for Resource Selection in Data Grid", Journal of Software, Vol 6, No 12 (2011), pp.2472-2478, Dec 2011
[19] Jackson Phiri, Tie-Jun Zhao, Jameson Mbale," Identity Attributes Mining, Metrics Composition and Information Fusion Implementation Using Fuzzy Inference System", Journal of Software, Vol 6, No 6 (2011), pp.1025-1033, Jun 2011
[20] Richard Lai, Sajjad Mahmood, Shaoying Liu," RAAP: A Requirements Analysis and Assessment Process Framework for Component-Based System (Invited Paper)", Journal of Software, Vol 6, No 6 (2011), pp.1050-1066, Jun 2011
[21] Jianguo Chen, Hao Chen,",A Structured Information Extraction Algorithm for Scientific Papers based on Feature Rules Learning", Journal of Software, Vol 8, No 1 (2013), pp.55-62, Jan 2013
[22] Song Chai, Yubai Li, Jian Wang, Chang Wu, A List Simulated Annealing Algorithm for Task Scheduling on Network-on-Chip, Journal of Computers, Vol 9, No 1 (2014), 176-182, Jan 2014
[23] Li-fu Chen, Hong-guang Xiao, Hai-liang Wang, Yueguan Lin, Research Development of High Precision Real-time Airborne InSAR System, Journal of Computers, Vol 9, No 1 (2014), 189-195, Jan 2014
[24] Biyun Yan, Yehua Wei, A Novel Shared-Clock Hybrid Scheduling Algorithm based on Controller Area Network, Journal of Computers, Vol 9, No 2 (2014), 373-379, Feb 2014

**Lijun Cao**, Born in 1971 , master, associate professor, college of mathematics and information science of Hebei Normal University of Science and Technology, main research directions are: data mining, grid technology.

**Xiyin Liu**, Born in 1970 , master, associate professor, College of mechanical and electrical engineering, of Hebei Normal University of Science and Technology, main research directions are: data mining, grid technology.