International Journal of Soft Computing 11 (1): 7-17, 2016
ISSN: 1816-9503

Data and Job Aware Scheduling for Uncertainty Resource Constraints by Multi-Stage Stochastic Integer Programming

[1]G. Kalpana and [2]D.I. George Amalarethinam
[1]Department of CSE, SRM University, Kattankulathur, Chennai, India
[2]Department of Computer Science, Jamal Mohamed College (Autonomous), Tiruchirappalli, India

Abstract: The uncertainty nature of a grid makes the traditional job scheduling algorithms to work improperly in an open, heterogeneous grid resembling in the real world environment. As most of the job scheduling tasks cannot be fit commonly into a single description model considering data assumption available in grid nodes, the job scheduling becomes the challenging task. In present research, new mathematical computation method called Multi-Stage Stochastic Integer Programming (MSIP) is proposed for overcoming process uncertainty during job and data aware scheduling of a grid computing. In case of computational load as generated by grid applications, the uncertainty problem can be solved by MSIP approach. A number of jobs can be defined with job execution time and resources after completion of MSIP. This is done for expressing the uncertain expected time to compute of jobs and analyze operation properties based on the resources for computing data and job aware scheduling in grid computing environment. The proposed harmony search called Hybrid Ant Colony (HAC) offers efficient grid utilization for both resource providers and consumers by solve problems related to data and job aware scheduling. Using this method, both resource providers and consumers are allowed to take autonomous scheduling decisions. Moreover, sufficient data incentives can be derived based on Community workflow Scheduler (CSF) of their interest. In other hand, experimental results ensured the ability of the proposed MSIP algorithms in representing uncertainty in a grid computing environment.

Key words: Grid computing, resource management, job scheduling, uncertainty, Community Scheduler Framework (CSF), Harmony search with Ant Colony (HAC), Multi-Stage Stochastic Integer Programming (MSIP), grid simulation toolkit

INTRODUCTION

Grid computing belongs to a group of computer resources performing multiple administrative domains for accomplishing a common goal (Xiaoxia, 2011; Tonellotto *et al*., 2007). It is a distributed computing model which helps in easy accessing of computer resources sharing large geographical area thus providing large virtual organization (Xhafa and Abraham, 2010; Riad *et al*., 2010). It is widely defined as "the seamless integration and cooperative computing environment" in wide regions as it provides a distributed computing infrastructure. Thus, large-scale advanced scientific and engineering problems can be solved using grid computing through resource sharing over high-speed communication networks. The resource allocation process in particular, mapping of tasks to various resources is the main task of grid computing. The heterogeneous processors and network links of slower orders of magnitude than parallel computer are been included in the grids of computational resources. Hence, the poor performance is usually resulted on executing grids of applications those designed for parallel computers. This is due to the workload distribution not considering the heterogeneity into account. Few such examples of such large-scale applications include Collaborative/e-Science Computing (Paniagua *et al*., 2005) and data-intensive computing to name a few.

Scheduling is typically an important mechanism in case of most of the grid systems. In simpler manner, job scheduling can be performed by mere assignment of the incoming tasks to the available compatible resources. However, using more advanced and sophisticated schedulers becomes more profitable. In general, the schedulers expect reaction dynamics of the grid system by typical evaluation of the present load of the resources followed by notification of joining or dropping new resources from the system. The organization of schedulers can be in a hierarchical or distributed form so as to deal the large scale of the grid. There are some grid characteristics to be taken into account before scheduling the tasks in the grid environment. Such grid characteristics include:

Corresponding Author: G. Kalpana, Department of CSE, SRM University, Kattankulathur, Chennai, India

- Geographical distribution showing the location of the resources of grid at distant places
- Heterogeneity of a grid consisting both hardware software resources such as files, software components, sensor programs, scientific instruments, display devices, computers, supercomputers and networks and so on
- Resource sharing as different organizations may own the grid resources
- Multiple administrations so that each organization may establish different security and administrative policies for resource access
- Resource coordination such that for combining computing capabilities for coordinating grid resources. The distributed ownership of the grid resources makes job scheduling as highly complicated task. There are two types of load balancing algorithm such as static and dynamic. As the entire information pertaining to the tasks and resources such as execution time of the tasks, speed of the processor are available once scheduling the application, it is referred to as static scheduling

The heterogenous nature of geographical distributions of resources has been owned by different organizations having individual accesses to policies and cost models. Each of the resource owners and end-users has different goals, objectives, strategies and demand patterns. There used to be competition among grid users distributed in the grid among which competition exists for accessing the available grid resources. The highly competitive grid environment requires satisfying various demands of resource users and a provider thus making the quality of service as crucial (Chatrapati *et al*., 2010). There will be a delay in execution time and decrease in throughput of the grid system due to poor task scheduling algorithm. Both the complexity and essentiality of the task scheduling are needed to be significantly addressed in the field of the grid computing (Parsa and Entezari-Maleki, 2009). Moreover, the formal definition of grid scheduling problem is also necessary. The job scheduling efficiency of an algorithm has a direct influence on the efficiency of network communication, reliability of computing grid, makespan of the grid system (Hong *et al*., 2005). Numerous researches are growing on the domain of task scheduling (Zhang *et al*., 2006; Ji *et al*., 2006), however, those hardly concern the dynamics and uncertainty in the grid environment.

The conventional task scheduling algorithms are not suitable for an open, heterogeneous, uncertain and dynamic grid environment. The dynamics and uncertainty of grid system thus differentiates them from the traditional heterogeneous computing systems. Hence, representation of these dynamics and uncertainty is considered. Most of the current grid approaches are either task-oriented or resource-oriented. As an instance, data required for its computation belongs to tasks and whereas data assumed to be available in grid nodes belongs to resource. An integrated scheduling approach is the method in which both task and data allocation will be optimized. The combined task of scheduling both jobs and data, further increase the complexity of large-scale problems.

In this research, the most important and useful computational models are presented to solve above mentioned problems of grid scheduling related to both job and data. The formal definition and proposal of the grid scheduling problem related to both data and job aware scheduling is therefore been focused. To suffice both users and providers with incentives based on their interest, autonomous scheduling decisions have also been studied.

The new computation model called Multi-Stage Stochastic Integer Programming (MSIP) is been proposed in this research to represent the uncertain execution time for job computing and to process the uncertainty in job and data aware scheduling of grid computing. Further, the design of efficient hybrid Harmony search with Ant Colony (HAC) Method for job and data aware scheduling problem referred as HACJDS in CSF has been performed. The proposed method is proven efficient in solving many computationally hard problems. The usefulness in the grid computing domain, especially for scheduling and resource allocation has also been shown in this work. Theoretical analysis and experimental results helped to illustrate and compare the proposed and existing scheduling methods thereby to represent the dynamics and uncertainty of "expected time for performing job and resources to complete job" in the grid computing environment.

Background study: Baghban and Rahmani (2008) presented a model and a job scheduling algorithm based on grid computing environments. In grid computing, numerous resources are required for executing several applications, however, often will not be available for them. Hence, resource allocation for implying the presence of a scheduling system is essential. In the proposed algorithm, the criteria for resource selection depend on the communication links, input jobs and the computational capability of resource. The assessment of the proposed algorithm usually takes place in simulated grid atmosphere with statistical models of job introduction into system in which the normal, poisson and exponential distribution has been followed individually.

The performance of job scheduling can be enhanced by exploring the use of bandwidth in scheduling framework (Keat *et al*., 2006). The scheduling framework comprise of job scheduler for forming Grid Information Service (GIS) using obtained information. Based on the information related to the resources, job scheduling was used for grouping of jobs and selecting resources. The framework has been involved in grouping and selection service where performance of job matching is based on the information gathered from collector after which the information about the network bandwidth will be made to reach each resource. Later, this information can be employed for grouping and selection service so as to gather the necessary information required for job selection. GIS also comprise all the resources information for providing to jobs to the respective resources. The fire grained job scheduling algorithm deals with grouping (Liu and Liao, 2009) as it is initiated by obtaining information about the resources. In this algorithm, light weight jobs are grouped as coarse grained jobs. The grouping based algorithm utilizes efficient resources for integrating greedy algorithm of FCFS for further improving the process of fine-grained jobs. There occurs integration of bandwidth aware job grouping based scheduling algorithm so as to implement the job grouping concept (Ang *et al*., 2009). In this algorithm, grouping of independent jobs with small processing requirement has been focused primarily and later which jobs with larger processing requirements are then scheduled based on network conditions. The bandwidth concept was also employed for performing load balancing at Stream Control Transmission Protocol (SCTP) layer. Thus, the main aim has been focused on providing the in-order delivery over multiple paths. The total job processing time has been reduced in this approach as compared to job scheduling without grouping.

Dynamic job grouping based scheduling algorithm tends to group the jobs by following MIPS of the resource (Muthuvelu *et al*., 2005). The resources are been selected by keeping priority on first come first serve order. The jobs are first selected and grouped after which they are assigned in FCFS order for comparing to resource. If group job MI is lesser than resource MIPS, the process continues until the resource MIPS becomes lesser to group job. The job scheduling involving tasks of fault-tolerant in a computational grid makes use of the Resource Fault Occurrence History (RFOH) (Khanli *et al*., 2010) algorithm. The history of occurrence of resource fault RFOH are used to be stored a Fault Occurrence History Table (FOHT) in the grid information server. In FOHT table, each row represents source and includes two columns. As one column shows the failure occurrence history for the resource, the other one shows the number of tasks executing on the resource. During task scheduling, the broker uses information in this table in the Genetic Algorithm (GA). Thus, the possibility of selecting resources with more occurrences of failures can be avoided.

In order to solve the problem of dependent task/job scheduling, Chaos-Genetic algorithm is been employed (Gharooni-Fard *et al*., 2010) in which two parameters such as time and cost has been evaluated using Quality of Service (QOS) and chaos variables instead of randomly production of the initial population. The advantages of GAs and chaos variables are combined for searching the search space by which premature convergence of the algorithm has been inhibited and hence faster solution with a faster convergence has been produced. A new deadline-scheduling algorithm, called Latest Time To Run first (LTTR) has been proposed (Lazarevic, 2008). The implementation of LTTR was propelled by extensive research on real-time system scheduling using the EDF algorithm and therefore the modified version is referred as EDF.

In LTTR, both submissions of the job deadline and estimated execution time of their job have been performed by the users. LTTR enables calculation of the difference between the deadline of the job requested deadline and the predicted job run time using additional information later which the job queue will be sorted in ascending order. The allocation of the first job in the queue will be to the first available idle resource. The traditional job scheduling algorithms cannot be applied effectively in an open, heterogeneous, uncertain and dynamic grid environment of real world. Further, they cannot handle data aware based scheduling framework and uncertain based resources. But those problems are been proposed to be overcome using MSIP Method.

MATERIALS AND METHODS

In general, grid resource management and scheduling requires multiple criteria. Dynamics and uncertainty of grid scheduling system makes the grid management as difficult and different task from the traditional heterogeneous grid scheduling computing systems. The method of representing and treating with these dynamics and uncertainty is hence, considered as one of the major challenging task in job scheduling problem. The important problem in job scheduling is that many scheduling do not fit into a common description model based on uncertain job and resource aware based management scheduling in grid computing environment. Hence, a common framework for scheduling problems cannot be defined which resulted

in implementing scheduling along with uncertain resource constrains and job aware based data intensive based scheduling process for each job as requested by user. In view to solve the uncertain problem for grid schedulers in both job and data aware resource constraint based scheduling framework, a new MSIP approach has been proposed (Vigerske, 2012) for handling uncertain resource constraints in the grid computing environment.

In the proposed research, job and resource aware based uncertainty problems were initially solved by using MSIP. As a hybrid HAC for data and job aware scheduling has been employed, the approach was named as HACJDS Method for CSF in which both the data and job service information for specific grid environment a has been integrated along with resource constraints for each resource provider in the grid scheduler. In CSF framework, the resource constraints based scheduling is still not efficiently performed. Hence in this research, the HAC algorithm for scheduling job and data aware in the grid computing is implemented. In the CSF framework, the job completion time has been added to each one of the job belonging to job service stage and reservation service stage. Some of the additional requirements and rank expressions pertaining to the presence of data have also been defined for performing job and data aware scheduling task. The purpose of the metascheduler is to allow interaction of end-users with underlying resource managers by means of the HAC there by to subsequently solve job and data aware scheduling problem with resource constraints.

The main objective is to minimize the turnaround time of the job with less resource to complete the job. At the same time, efficient handling of data in scheduling process in different computation sites and scheduling of the data flow between these jobs also requires importance. Let us consider N_{Rg} be the n number of resource providers in grid host environment. N_{dg} is the number of data sites in grid host environment for each number of the task. N_{jg} is the number of jobs that run in the grid host environment P_i is a set of computation sites p_i. $P = \{p_i\}, i = 1, ..., N_c$. D is the set of the data sites of $d_{ig} = i = 1, ..., N_{dg}$ in grid host environment. C is a set of computation links between computation sites C_{pipj}, data sites $C_{d_k d_l}$ and computation-data sites $C = \{C_{P_i d_k}, C_{d_k d_l}, C_{P_i d_k}\}, i, j = 1, ..., N_{gc}, k, l = 1, ..., N_{gd}$. Set of m resource consumers by $R = \{r_1, r_2, ..., r_m\}$. Every job announcement incorporate certain limit for a specific job, budget, QOS and Job smaller running jobs J_{Rt}. The purpose of the structure is to make the most of consumer objectives like makespan and smaller payment and provider objectives, like higher profit and parallel/well-organized deployment of resources. Each job consists of two phases, one of which is computation and the other is intermediate data transfer. Once a task gets executed, it will not be stopped until completing the computation work. Once an intermediate data transfer is started, it cannot stop until this transfer is finished. Every announcement incorporates certain limit for a specific job JT_i. Each job J_i needs a finite amount of time JC_{ij} to finish if it can be scheduled to computation site:

$$\text{Ext}(JC_i, p_i) = \begin{cases} JC_{ij} & JC_i \text{ is applicable on } p_j \\ \infty & \text{otherwise} \end{cases} \quad (1)$$

Then, the calculated execution time of the each task the priority level or queuing service level is calculated based on the weight value with waiting time:

$$J_{wt} = \left(\frac{\text{ext}(JC_i, p_j)}{\text{avgwt}}\right)^{\alpha} \quad (2)$$

Where:
$\text{ext}(JC_i, p_i)$ = The time that a job waits in the queue
avgwt = The average job waiting time in the system
α = The weighting factor

Job running time also can be a factor for a dynamic priority by favoring shorter running jobs over longer running jobs:

$$J_{Rt} = \left(\frac{r}{\text{avgrt}}\right)^{\beta} \quad (3)$$

Where:
r = The average runtime for a job
avgrt = The average of the average runtime for all jobs
β = The weight factor

Further, we consider the number of remaining unscheduled a job as another as:

$$J_{Js} = \left(\frac{n}{\text{avgnt}}\right)^{\gamma} \quad (4)$$

Where:
n = The number of remaining jobs
avgnt = The average number of remaining job over the entire queue
γ = The weighting factor

The combination of these above Eq. 2-4 becomes the final priority for the job in the system as:

$$\text{Priority} = \left(\frac{\text{ext}(JC_i, p_j)}{\text{avgwt}}\right)^{\alpha} \times \left(\frac{r}{\text{avgrt}}\right)^{\beta} \times \left(\frac{n}{\text{avgnt}}\right)^{\gamma} \quad (5)$$

The importance of each user in the queue may be accounted:

$$U_q = u^{\delta} \quad (6)$$

The above mentioned methods however does not handle uncertain problem for job and data aware resources in CSF framework. In order to handle uncertain problem for job execution time and resources, mathematical modeling has been employed. The proposed work initially has done the MSIP formulation for the multi-facility uncertain job execution time and resources capacity expansion problem. Let us consider J_t be the job completion time for grid resources, over which the capacity investment costs and demands are assumed to be known. The objective of this MSIP formulation is to determine the uncertain value for job execution time and resources while minimizing the job completion or execution time based on the time horizon t = 1, ..., T. Consider the JC_{ij} to denote the job completion time and M_{ij} capacity expansion of resource type and UJC_{ij} to denote the uncertain job execution time, for example, if it takes 124 sec to finish job on a computing grid under normal condition. However, if some new jobs are added to the machine concurrently or some jobs on the machine are terminated, it may delay 12 sec or may save 20 sec to finish job decision, UM_{ij} uncertain resources the problem can be stated as follows:

$$\text{CAJR: } \min \sum_{i,j=1}^{N_{gc}} (M_{ij}JC_{ij} + UM_{ij}UJC_{ij}) \tag{7}$$

satisfies that the $0 \le JC_{ij} \le JUC_{ij}$ T:

$$\sum_{t=1}^{T}\sum_{i=1}^{N_{gc}} J_{it} \ge d_t,\ t = 1,\ldots,T \tag{8}$$

d_t be the demand time for job to complete work, JUC_{ij} be the upper bound job completion time in grid host environment, respectively. Equation 9 enforces that to check the uncertain value for job and resources for this purpose assume that maximum value of job completion time and maximum resources required for job completion. Equation 7 and 8 is extended to stochastic setting on assumption that the uncertain problem parameters (Um_{ij}, UJC_{ij}, d_t) evolve as discrete time stochastic processes with a finite probability space. The resultant uncertain values of job and resources exceed or less is represented as form of the tree structure with n nodes during time period T is the root of a sub-tree T(n). The path from one job to another job in the tree is denoted as P(n). The following objective of minimizing expected job completion time and UM_{ij} uncertain resources the stochastic capacity job resource problem can be written as:

$$\text{SCAJR: } \min \sum_{n\in T(0)} P(n) \sum_{i,j=1}^{N_{gc}} (M_{ij}JC_{ij} + UM_{ij}UJC_{ij}) \tag{9}$$

$$\sum_{n\in T(0)}\sum_{i=1}^{N_{gc}} J_{in} \ge d_n,\ n \in T(0) \tag{10}$$

In above steps, the following additional parameters are also required to perform the job and data aware scheduling process in the CSF. The subsequent notations are employed in the mathematical model is considered as fitness value:

- Budget of job i which is given by user BTJ
- Unit price of the Resource to complete the number of jobs UP
- Total workload of the resource W_j
- Communication bandwidth CB_i for site p_i
- Time required to complete job J_i is defined as JC_{ij} for resource j
- m be the number of the CPUs
- Number of instructions per second ips
- Stochastic capacity job resource problem SCAJR
- Calculate the fitness for each individual, based on the UP:

$$UP = \sum_{i=1}^{n} \frac{jsrp_i}{n} \tag{11}$$

The performance of HS algorithm seems good in solving the variety of the problems. However, some drawbacks interms of the local optimum have resulted for any type of application. The memory consideration operator does not function, if New Harmony is found in memory considering as not updated in the existing HM. To overcome the limitations, a New Harmony Search algorithm combined with ant colony procedure has been proposed in which the mutation operators, elite strategy and fitness of the Genetic Algorithm (GA) have been included. The proposed HAC is used for the job and data aware scheduling for uncertain job service time and resource management. The performance of the job and data aware scheduling framework is done by following this procedure. The harmony search memory consists of the information about each job and their service rate of the job in the grid computing environment thus the Harmony Memory Considering Rate (HMCR) can be adjusted based on the job service rate and uncertain job execution time from Eq. 9-12.

In this research, the HMCR rate for job and data aware resources is updated based on the roulette wheel algorithm. The fitness index value for each job and data aware from Eq. 9-12 has been compared. If the fitness

value is high for selected job then HMCR rate assigned as high between intervals HMCR = 0.7~0.95. In the scenario of meeting condition 1 of section 2.2, the pitch adjustment produces the (i+1)th index value of a latest harmony that is the nearby value to the ith value from the probable set of values. Parameter PAR should also be appropriately set. If PAR is very close to 1 then the solution is always updating and HS is hard to converge. If it is next to 0, then little change is made and HS may be premature. So, here set PAR = 0.1~0.5 (Wang and Guo, 2013). pHMCR and pPAR were produced arbitrarily between 0 and 1, correspondingly and each operator is chosen in accordance with the following criterias:

- Condition 1: pHMCR≤HMCR and pPAR>PAR select the memory consideration
- Condition 2: pHMCR≤HMCR and pPAR≤PAR select the pitch adjustments
- Condition 3: pHMCR>HMCR select the randomization

Fitness value unit price UP of the resource, Stochastic Capacity Job Resource (SCAJR) and distance of the ants (job and resources) are inversely related. The newly generated HMCR rate is kept as same and earlier HMCR is removed from memory for each job and data with uncertain time and resources management process. The convergence speed for job and data aware scheduling of HM could not increase for all jobs, so local optimum problem occurs when it is repeated infinitely. To solve this problem for job and data aware scheduling for each harmony the inversion mutation operator is introduced from GA. Inversion mutation operator generates new job service rate and uncertain job execution time for resources and data in grid computing environment. For this generated new jobs services rate and job uncertain execution time for all resources and data aware based scheduling Ant Colony algorithm is applied to improve the scheduling results and then the pheromone is updated:

$$\tau(jsrp_i, jsrp_j) \leftarrow (1-\rho)\tau(jsrp_i, jsrp_j, ips) + \rho\Delta\tau(jsrp_i, jsrp_j, ips) \quad (12)$$

$$\tau(UJC_{ij}, UM_{ij}) \leftarrow (1-\rho)\tau(UJC_{ij}, UM_{ij}, ips) + \rho\Delta\tau(UJC_{ij}, UM_{ij}, ips) \quad (13)$$

P represents a constant in accordance with the the state transition rule as provided in Eq. 12, ants shift between one job to another job. Consider:

$$s = \begin{cases} \arg\max \left[t(jsrp_i, jsrp_j, ips)\right]^m \\ \left[\eta(jsrp_i, jsrp_j, ips)\right]^\beta \\ S \text{ othrewise} \end{cases} \quad (14)$$

$$s = \begin{cases} \arg\max \left[\tau(UJC_{ij}, UM_{ij}, ips)\right]^m \\ \left[\eta(UJC_{ij}, UM_{ij}, ips)\right]^\beta \\ S \text{ otherwise} \end{cases} \quad (15)$$

Where:

$\tau(jsrp_i, jsrp_j)$ = State transition rule for two job service rate

s = The reciprocal of distance between jobs and resources UM_{ij}

m = The number of resources

β = The parameter to determine the relative importance of jobs, respectively for each ant

The pheromone evaporation coefficient ρ is always a decimal number in range of zero to one:

$$\Delta\tau(jsrp_i, jsrp_j) = \begin{cases} L_{gb}^{-1} & \text{if } (jsrp_i, jsrp_j, ips) \in \text{global best tour} \\ 0 & \text{otherwise} \end{cases} \quad (16)$$

These procedures are repeated until satisfying the job and data aware scheduling stopping criteria. Pseudocode 1 describes the procedure of the job and data aware resource scheduling using HAC algorithm.

Procedure (Job and data aware resource scheduling using HAC):

Begin

Step 1: A consumer puts forward a job declaration to the grid which is transmitted to each and every providers

Step 2: On reception of job declaration with JSR, each provider approximates whether it could be capable of satisfying the deadline. When yes, the provider transmits a bid containing the price for the job straightforwardly to the consumer and when no the provider pays no attention to the job announcement

Step 3: Once receiving the entire bids, the consumer decides the provider who charges the smallest amount and sends the job

Step 4: Objective function UP, SCAJR

Step 5: Generate initial harmonics with job service rate and job uncertain time, job uncertain resources

Step 6: Define Harmony Memory Considering Rate (pHMCR), Pitch Adjusting Rate (pPAR), mutation rate (pm)

Step 7: Generate initial harmony for job service rate and uncertain resources randomly and apply pheromone update

Step 8: while (not termination)

Step 9: for i = 1: number of jobs

Step 10: Generate random number variable (rand)

Step 11: if (rand<pHMCR)

Step 12: Generate random number variable (rand)

Step 13: if (rand<ppa), generate the nearest city to the previous harmonic

Step 14: else choose an existing harmonic the highest fitness value from Eq. 9-12

Step 15: end if

Step 16: else generate new job and data aware service rate, uncertain via randomization

Step 17: end if

Step 18: end for

Step 19: Accept the new harmonics (solutions) if it is satisfies Eq. 9-11

Step 20: Generate random number variable (rand)

Step 21: if (rand <pm) operate inversion mutation end if

Step 22: Apply the pheromone update from Eq. 12-15

Step 23: Create new best job service rate and less uncertain results for job and resources based on the ant colony procedure
Step 24: Update harmony memory and apply pheromone update from Eq. 12-15
Step 25: end while
Step 26: Find current best less job service time and uncertain measurement completion results
Step 27: End

After calculating the queue service of the each user for number of the jobs, the job service task is then performed. A job service provides an interface for placing jobs on a resource manager and interacting with the job once it has been dispatched to the resource manager. The job service provides basic matchmaking capabilities between the requirements of the job and the underlying resource manager for running the job. The job service uses information such as policies which are defined at the metascheduler level and resource information about available resource managers, queues, host and job statuses as provided by the Global Information Service (GIS). In CSF framework the reservation service allows end-users or a Job Service to reserve resources under the control of a resource manager to guarantee their availability to run a job and their data are accessed by user based on the user request and their priority. This service allows reservations for any type of resource (e.g., hosts, software licenses or network bandwidth). These types of the reservations service categories also additionally consists of the following information to perform the reservation process.

Job category information

Transfer: The transfer job implies transfer of a complete or partial file from one physical location to another one. A get or put operation or a third party transfer will perform this job.

Allocate: This job type helps in allocation of storage space at the destination site, network bandwidth or establishing a light-path on the route from source to destination. Basically, it deals with all necessary resource allocations which are pre-requisite for the data placement.

Release: This job type is used for releasing the corresponding resource that has been allocated before.

Remove: This job is used to physically remove a file from a distant or neighboring storage server, tape or disk.

Locate: Logically, this job consults a meta data catalog service:

- Close data (logical file name) (requirements expression): for each Computing Element (CE), it is evaluated as True only if specified data is held by a close Storage Element (SE) and as False otherwise
- Has close data (logical file name) (rank expression): for each CE, it is evaluated as 1 if specified data is held by a close SE and as 0 otherwise
- Size close data (logical file name) (rank expression): for each CE, evaluated as the size of the specified data if held by a close SE and as 0 otherwise

Register: This job is employed to record the file name to a meta data catalog service.

Unregister: This job unregisters a file from a meta data catalog service.

The information required to operate the metascheduler can be provided by Global Information Service (GIS) as it acts as a repository for information. Service aggregates various lower-level grid services for replicating service data between grid services and creating a dynamic data-generating and indexing node. The scheduling capabilities are provided by the queuing service to the metascheduler. The CSF implementation of the queuing service requires a plug-in scheduling framework. The detailed information on each components in the CSF are already been presented by Platform Computing Co. (2004).

RESULTS AND DISCUSSION

Several experiments were conducted for evaluating the functionality and performance of the implementation of algorithm. This can be done using resource files running under grid simulation tool. The modeling and simulation of resources can be performed using GridSim (Sulistio *et al*., 2005) simulation toolkit for application involving scheduling in large-scale parallel and distributed computing environments. The selected file project provides large-scale computing infrastructure for researches involving data-intensive and computing-intensive applications such as high energy physics, earth and life sciences. As such an extensive simulation uses huge quantity of computing power have a desire for employing parallel and cluster computing methods. In the simulation would like to model applications in the fields of biotechnology, astrophysics, network design and high-energy physics with the aim of studying effectiveness of the resource allocation techniques. The outcomes of this approach will have considerable influence on the way resource allocation carried out for solving complications on cluster and grid

computing systems. Java-based discrete-event grid simulation tool kit called GridSim. The toolkit supports modeling and simulation of heterogeneous grid resources, users and application models. It provides primitives for creation of application tasks, mapping of tasks to resources and their management. The performance of the proposed data and job aware scheduling HACJDS in CSF has been evaluated and Hybrid Encoded Cuckoo search in CSF named as HECSCSF existing methods using Gridsim simulation. To evaluate the effectiveness of schedulers, the uncertainties of applications are dealt and the grid resource availability is described and fed to the schedulers HAC to produce a schedule in CSF framework. Diverse set of the job category information with different data aware types are put forwarded at several gaps to the grid to assess the effectiveness of the algorithm under different loads. The experimental results are shown for comparing with the previous scheduling algorithm and the CSF is also shown in Table 1.

Results of the experiments in terms of the cost between methods are shown in Table 2. Figure 1 shows that the performance comparison results of the proposed HACJDS and existing HECSCSF, adaptive QOS algorithm, economic adaptive QOS algorithm, proposed HACJDS have lesser computational cost when compare to existing methods. When number of jobs increased in the system usage of the system resources also amplified, the performance based on cost also increased reasonably with previous methods. The speed of computational cost of the proposed HACJDS algorithm is different from other method since, it handles uncertainty problem for both job and resources. Results of the experiments in terms of the revenue of proposed HACJDS and existing HECSCSF, adaptive QOS algorithm, economic adaptive QOS algorithm methods are shown in Table 3.

Table 1: Simulation parameters and economic attributes of the experiments

Parameters	Values
Number of grid sites	180
Number of nodes in each site	30
Bandwidth capacity (Mbps to 2 Gbps)	200
Processing capacity (Micro instruction per second)	512-1031
Total number of providers	300
Total number of users	750
Budget of users (units)	100-1000
Price of resources (units)	100-2000
Number of files	100-15000
Size of each file (MB-1 GB)	100

Table 2: Performance of the system in terms of the cost

No. of jobs	No. of resources	Adaptive QOS algorithm	Economic adaptive QOS	HECSCSF	HACJDS
50	14	502	452	356	313
100	34	771	672	564	512
150	84	1196	992	897	847
1000	104	1486	1162	956	854
2000	134	1672	1212	1013	946

Figure 2 shows that the performance comparison results of the proposed HACJDS and existing HECSCSF, adaptive QOS algorithm, economic adaptive QOS algorithm in terms of the revenue. When number of jobs increased in the system usage of the system resources also increased, it shows that the revenue of the proposed HACJDS System also increased comparatively with previous methods HECSCSF, adaptive QOS algorithm and Economic adaptive QOS System. Proposed HACJDS algorithm is different from other method since, it handles uncertainty problem for both job and resources.

The performance comparison of the proposed HACJDS and the existing methods in terms of the resource utilization with number of the jobs are shown in Table 4. It shows that results of the proposed HACJDS have high resource utilization percentage when compare to existing, HECSCSF adaptive QOS and economic based adaptive QOS algorithm (Fig. 3).

The performance comparison of the proposed HACJDS and the existing methods results are measured

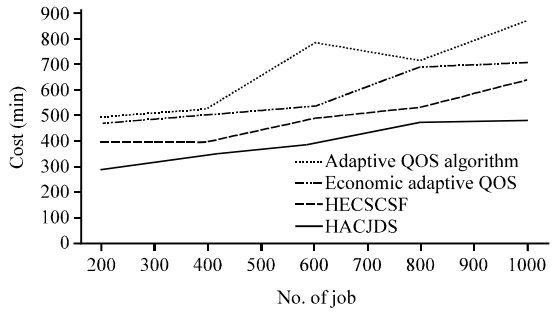

Fig. 1: Performance of the system in terms of the cost

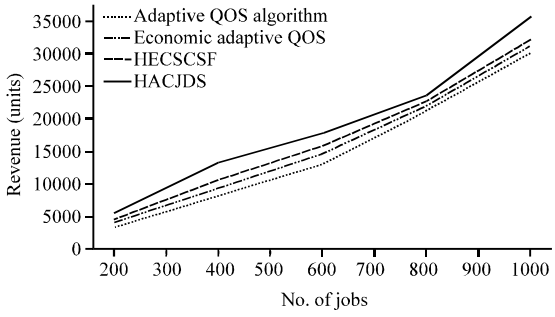

Fig. 2: Performance of the system in terms of the revenue

Table 3: Performance of the system in terms of revenue

No. of jobs	No. of resources	Adaptive QOS algorithm	Economic adaptive QOS	HECSCSF	HACJDS
50	14	16208	18123	21564	23479
100	34	21502	24024	26548	29413
150	84	25256	29802	32145	35689
1000	104	36208	31711	34589	36471
2000	134	28510	35648	39874	41378

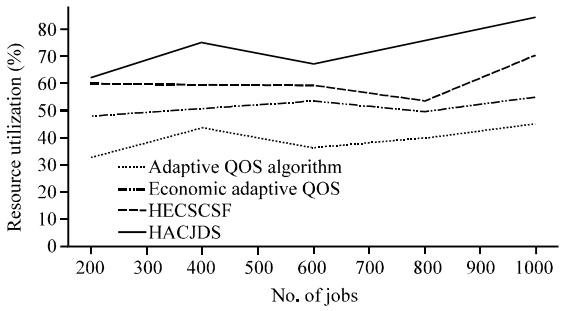

Fig. 3: Performance of the system in terms of the resource utilization

Table 4: Performance of the system in terms of resource utilization

No. of jobs	No. of resources	Adaptive QOS algorithm	Economic adaptive QOS	HECSCSF	HACJDS
50	14	67	78	86	92
100	34	71	80	88	93
150	84	74	84	88	94
1000	104	78	89	93	97
2000	134	82	93	97	99

in terms of the resource utilization with number of the jobs are shown in Table 4. It shows that results of the proposed HACJDS have high resource utilization percentage when compare to existing HECSCSF, adaptive QOS and economic adaptive QOS algorithm. When number of jobs increased in the system utilization of the system resources in terms of the percentage also increased with previous methods adaptive QOS algorithm and economic based adaptive QOS System. Proposed HACJDS algorithm is different from other method since, it handles uncertainty problem for both job and resources.

The transfer time for the data to be delivered to a computing site is the quotient of the summation of the data amount and the summation of the bandwidth from each storage site to the computing site. The transfer time for all these computing sites is the maximum among them for existing methods. Figure 4 and 5 show the experimental results for different number of data files can be compared with different data aware scheduling methods such as random and the heuristic data aware scheduling methods, it shows that the proposed data and job aware scheduling HACJDS framework have taken less time to complete the task than the existing methods for both homogenous and heterogeneous distributed system since the task are scheduled based on the utilization of the system resources in terms of the time.

Figure 6 shows the job completion time of the proposed HACJDS Scheduling Method with existing HECSCSF, data and task aware CSF, heuristic data aware scheduling methods and lower bound scheduling methods it shows that the job completion time of the

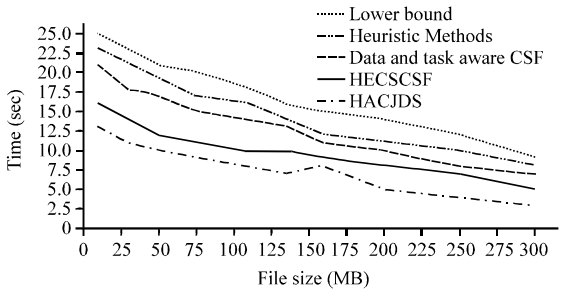

Fig. 4: Data and job aware scheduling for homogeneous distributed systems

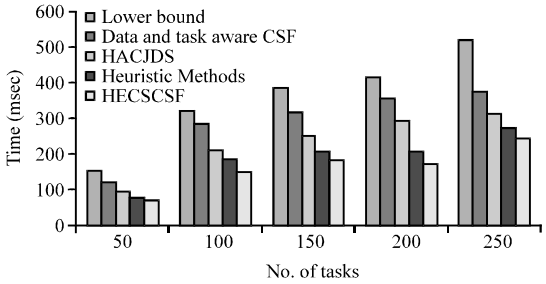

Fig. 5: Data and job aware scheduling for heterogeneous distributed systems

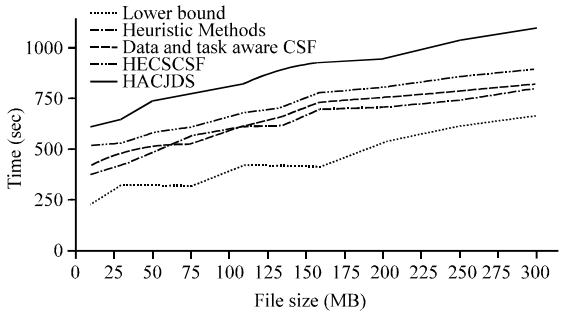

Fig. 6: Job completion time for methods

proposed system is less. Proposed HACJDS algorithm is different from other method since it handles uncertainty problem for both job and resources in efficient manner but all of the existing methods doesn't handle uncertain problem for job and resources.

Figure 7 shows the job transfer time for proposed HACJD Sscheduling algorithm and existing HECSCSF, job and data aware CSF, heuristic data aware scheduling methods and lower bound scheduling. It shows that the data transfer time for proposed HACJDS Scheduling System is less, since uncertainty problem of job and resources are managed by using MSIP with best job and data aware scheduled results in the CSF. In the proposed research, job and data aware uncertainty is measured based on the exceed job execution time and demand resources.

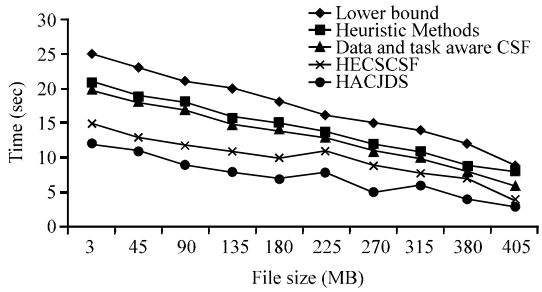

Fig. 7: Data transfer time for methods

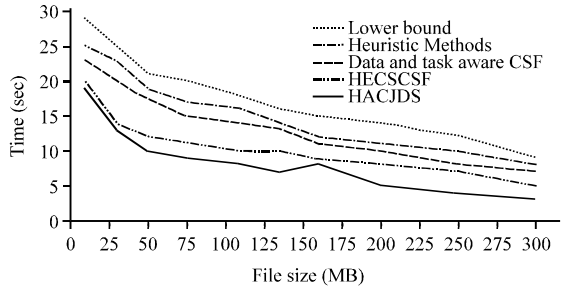

Fig. 8: Job turnaround time for methods

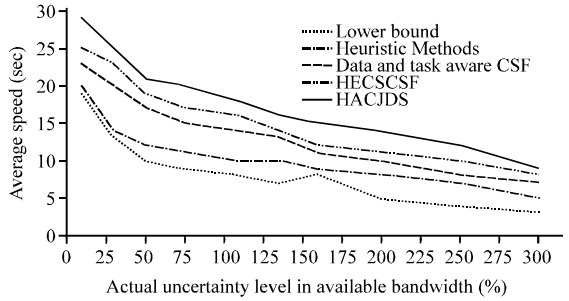

Fig. 9: Speedup produced by schedulers with uncertainty level of bandwidth

Figure 8 shows the job turnaround time of the proposed HACJDS Scheduling algorithm and existing HECSCSF, job and data aware CSF, heuristic data aware scheduling methods and lower bound scheduling. It shows that the job turnaround time of the proposed HACJDS Scheduling System is less since uncertainty problem of job and resources are managed by using MSIP with best job and data aware scheduled results in the CSF.

Figure 9 displays the average speedup for the different schedulers under the different uncertainty level of bandwidth for grid environment and compared with job and data aware scheduling methods such as lower bound, heuristic method, data and job aware CSF, HECSCSF. The design of the proposed HACJDS Method for CSF framework is suitable for uncertainty level in available bandwidth (%).

CONCLUSION

By implementing grid technology, the use of idle resources as part of a single integrated system has been made possible. The main purpose of grid computing is to make common resources such as bandwidth and databases available to a central computer. There are some challenges presented by dynamic states of the grid space in resource and job management thus requiring an efficient scheduler. The assigning of jobs to resources in such a way to execute it in the shortest possible time is the task of efficient scheduler. In the present study, measurement of the uncertainties problem of job and resources aware are based on Multi-Stage Stochastic Integer Programming (MSIP) for justifying the use of a Community Scheduler Framework (CSF) scheduler in scheduling grid jobs. There used to be uncertainty in both demands and resource availability based on Hybrid harmony search with Ant Colony (HAC). The modified generic HS algorithm has been provided with a new HS algorithm comprising of the elite policy and mutation operators in the GA and integration of the ACO algorithm inside the HS algorithm. This is performed in a view to surpass the inadequacies of the HS algorithm for solving job and resource aware scheduling problem in grid environment. The identification of different types of job categories has been done in the proposed system with data aware scheduling based on different criteria so that to considerably diminish the makespan and cost of grid user even under unpredictable network circumstances. The results showed that the effectiveness of theproposed HACJDS Scheduling algorithm relies mainly on its ability to withstand even on a high level of uncertainty. Future, scope on this research will be planned to further increase the fundamental understanding of quality of service of grid computing. This might be accomplished by identification of new QOS policies, understanding of in-depth existing policies, automatic deployment policy for choosing an appropriate policy under a given scenario and adjusting dynamic policy during adoption of different policies at different time.

REFERENCES

Ang, T.F., W.K. Ng, T.C. Ling, L.Y. Por and C.S. Liew, 2009. A bandwidth-aware job grouping-based scheduling on grid environment. Inform. Technol. J., 8: 372-377.

Baghban, H. and A.M. Rahmani, 2008. A heuristic on job scheduling in grid computing environment. Proceedings of the 7th International Conference on Grid and Cooperative Computing, October 24-26, 2008, Shenzhen, pp: 141-146.

Chatrapati, K.S., J.U. Rekha and A.V. Babu, 2010. Competitive equilibrium approach for load balancing a computational grid with communication delays. J. Theoret. Applied Inform. Technol., 19: 126-133.

Gharooni-Fard, G., F. Moein-Darbari, H. Deldari and A. Morvaridi, 2010. Scheduling of scientific workflows using a chaos-genetic algorithm. Procedia Comput. Sci., 1: 1445-1454.

Hong, L., D.J. Mu and Z.Q. Deng, 2005. A review of task scheduling for grid computing. Applic. Res. Comput., 22: 16-19.

Ji, L., J. Zhong and Z.F. Wu, 2006. An artificial immune algorithm for task scheduling in grid environment with fuzzy processing time. Comput. Sci., 33: 35-38.

Keat, N.W., A.T. Fong, L.T. Chaw and L.C. Sun, 2006. Scheduling framework for bandwidth-aware job grouping-based scheduling in grid computing. Malaysian J. Comput. Sci., 19: 117-126.

Khanli, L.M., M.E. Far and A. Ghaffari, 2010. Reliable job scheduler using RFOH in grid computing. J. Emerg. Trends Comput. Inform. Sci., 1: 43-47.

Lazarevic, A., 2008. Autonomous grid scheduling using probabilistic job runtime scheduling. Ph.D. Thesis, University of London, London, UK.

Liu, Q. and Y. Liao, 2009. Grouping-based fine-grained job scheduling in grid computing. Proceedings of the 1st IEEE International Workshop on Education Technology and Computer Science, Volume 1, March 7-8, 2009, Wuhan, Hubei, China, pp: 556-559.

Muthuvelu, N., J. Liu, N.L. Soe, S. Venugopal, A. Sulistio and R. Buyya, 2005. A dynamic job grouping-based scheduling for deploying applications with fine-grained tasks on global grids. Proceedings of the Australasian Workshop on Grid Computing and e-Research, Volume 44, January 30-February 4, 2005, Newcastle, Australia, pp: 41-48.

Paniagua, C., F. Xhafa, S. Caballe and T. Daradoumis, 2005. A parallel grid-based implementation for real-time processing of event log data of collaborative applications. Proceedings of the International Conference on Parallel and Distributed Processing Techniques, June 26-30, 2005, Las Vegas, USA., pp: 1177-1183.

Parsa, S. and R. Entezari-Maleki, 2009. RASA: A new grid task scheduling algorithm. Int. J. Digital Content Technol. Applic., 3: 91-99.

Platform Computing Co., 2004. Open source metascheduling for virtual organizations with the Community Scheduler Framework (CSF). Technical Report, Platform Computing Co.

Riad, A.M., A.E. Hassan, Q.F. Hassan, 2010. Design of SOA-based grid computing with enterprise service bus. Adv. Inform. Sci. Serv. Sci., 2: 71-82.

Sulistio, A., U. Cibej, B. Robic and R. Buyya, 2005. A toolkit for modelling and simulation of data grids with integration of data storage, replication and analysis. Technical Report No. GRIDS-TR-2005-13, Grid Computing and Distributed Systems Laboratory, University of Melbourne, Australia.

Tonellotto, N., R. Yahyapour and P. Wieder, 2007. A proposal for a generic grid scheduling architecture. Proceedings of the CoreGRID Workshop Integrated Research in GRID Computing, November 28-30, 2005, Pisa, Italy, pp: 227-239.

Vigerske, D.M.S., 2012. Decomposition in multi-stage stochastic programming and a constraint integer programming approach to mixed-integer nonlinear programming. Ph.D. Thesis, Humboldt State University, University in Arcata, California.

Wang, G. and L. Guo, 2013. A novel hybrid bat algorithm with harmony search for global numerical optimization. J. Applied Math. 10.1155/2013/696491.

Xhafa, F. and A. Abraham, 2010. Computational models and heuristic methods for Grid scheduling problems. Future Generation Comput. Syst., 26: 608-621.

Xiaoxia, L., 2011. A grid scheduling model with competitive negotiation. Int. J. Adv. Comput. Technol., 3: 173-180.

Zhang, W.Z., X.R. Liu, X.C. Yun, H.L. Zhang, M.Z. Hu and K.P. Liu, 2006. Trust-driven job scheduling heuristics for computing grid. J. Commun., 27: 73-79.