ISSN 2047-3338

An Optimal Job Scheduling in Grid Using Cuckoo Algorithm

M. Prakash[1], R.Saranya[2], K. Rukmani Jothi[3] and A. Vigneshwaran[4]

***Abstract*— A computational Grid is a hardware and software infrastructure that provides dependable, consistent, pervasive, and inexpensive access to high-end computational capabilities. Grids are emerging as a new computing paradigm for solving grand challenge application in Science, Engineering and Economics through sharing and collaboration of heterogeneous resources. Scheduling is one of the research issues in grid. Grid Scheduling is defined as the process of making scheduling decisions among resources from different administrative domains. The main problem in computational grid resource allocation is to discover suitable resources and to schedule separate tasks of an application on those resource in such a way to satisfy deadline requirement of the job with minimum execution time and cost. We propose a cuckoo optimization algorithm for optimal job allocation of resources on each node. This system will allocate the job optimally by considering the deadline requirement of the users and also the minimal execution time.**

Index Terms— Grid Computing, Scheduling, Cuckoo Algorithm, Resource Discovery and Resource Selection

I. INTRODUCTION

WHEN PCs were introduced in the 1970's,they were designed for personal use and were generally considered standalone computers. Later, client server model were introduced. In client server model, the client request server for its requirement and server responds it. The drawback is if server crashes, the entire process get collapsed. Then many models like peer to peer model, distributed system were introduced. In these entire models, the people will not utilize their resources 24*7 hours. The new concept called grid computing [1] were introduced which allow the people to use the resources on demand. The people will be charged only for time they are consuming the resources. The popularity of the internet and the availability of powerful computers and high-speed networks are changing the way we use computers today [9]. These technical opportunities allow us to use distributed and multi-owner resources to solve large-scale problems in science, engineering, and commerce. Recent research on topics has led to the emergence of a new paradigm known as *Grid computing*. Grid computing [9] is a term referring to the combination of computer resources from multiple administrative domains to reach a common goal. Grid computing is effective only when we schedule the resources optimally. Here comes the concept of Grid Scheduling [9]. Scheduling the resources to the job is one of the main issues in grid. The problems in grid scheduling are difficult to find the completion time of the job, resource allocation and job scheduling, resource management and fault tolerance, load balancing, considering both user and application requirement, problems in allocating I/O devices, satisfying deadline requirement of job with minimum execution time and by using efficient resource for processing[9]. A resource is something that is required to carry out the assigned job. Resources can be computers, storage space, instruments, software applications, and data, all connected through the Internet [9]. A task is defined as the atomic unit that is to be scheduled by the scheduler and assigned to the selected resource. A job is set of tasks that will be carried out on a set of resources [9]. A Local Scheduler has the information about each and every resource in its domain. A domain consists of set of nodes. A node is an autonomous entity, which is composed of one or more multiple resources. Grid Scheduler is responsible for scheduling and managing resources at a single site or at a single cluster of resources [10]. Generally, Grid Scheduler does not own the resources as the Local Scheduler does. They do not have control over the resources and also they do not have full control over all the jobs submitted to it. They just make best decisions to submit the job to the resource selected. The role of the Grid information service (GIS) is to provide information to Grid schedulers [1], [9]. GIS is responsible for collecting and predicting the resource state information, such as CPU capacities, memory size, network bandwidth, software availabilities and load of a site in a particular period. This information is provided by the various Local Schedulers in the administrative domain. Three phases in the grid computing are: resource discovery, resource selection, Job submission [10]. In the first phase, the application and user requirement is collected from the authenticated grid members. At the end of authorization filtering step the user will have a list of machines or resources to which he or she has access. To proceed in the resource discovery, user must be able to specify some minimal

[1]M.Prakash, Assistant Professor in Department of Computer Science and Engineering, Rajalakshmi Engineering College, Chennai, India
[2]R.Saranya, final year B.E in Department of Computer Science and Engineering, Rajalakshmi Engineering College, Chennai, India
[3]K.Rukmani Jothi, final year B.E in Department of Computer Science and Engineering, Rajalakshmi Engineering College, Chennai, India
[4]A.Vigneswaran, final year B.E in Department of Computer Science and Engineering, Rajalakshmi Engineering College, Chennai, India

set of job requirement in order to filter the feasible set of resources. It may include static as well as dynamic details. In addition to the application requirement, user has to specify their requirements such as deadline, cost etc. In second phase, the resource that does not match the application requirement will be filtered out. After that, according to the user requirements the feasible resource must be selected from the set of resources. In the third phase, once the resources are chosen, application can be submitted to the resources [10]. Depending upon the application and its running time, user may monitor the progress of their application. If a job is not making sufficient progress, it may be rescheduled. After the job has been successfully completed, the resources used by the job have released and completed job will be given to the user.

The related work in Grid computing is discussed in section II. Section IV describes the proposed architecture in Grid environment. The Cuckoo Optimization algorithm which schedules the resource optimally is proposed in section V. We have concluded that this system will allocate the job optimally by considering the deadline requirement of the users and also the minimal execution time in section VI.

II. RELATED WORK

Yang Gaoa [1] proposes two models for predicting the completion time of jobs at both the system-level and application-level. The single service model predicts the completion time of the job in a grid that provides one type of service. The multiple services model predicts the completion time of a job that runs in a Grid which offers multiple types of services. They have developed two algorithms that use predictive models to schedule the jobs. In application-level scheduling, genetic algorithms are used to minimize the average completion time of jobs through optimal job allocation.

Resource allocation and scheduling is a fundamental issue in achieving high performance on enterprise grid computing. I/O is also a critical resource, hence the allocation of I/O resources must be coordinated to allow the system to operate most effectively. J.H. Abawajy [2] mainly focuses on I/O and service-demands of parallel jobs in homogeneous and heterogeneous systems with background workload. The performance of the proposed scheduling policy is studied under various system and workload parameters through simulation. They have compared the performance of the proposed policy with a static space–time sharing policy. The results show that the proposed policy performs substantially better than the static space–time sharing policy.

Effective task scheduling is essential for obtaining high performance in heterogeneous distributed computing systems (HeDCSs). However, finding an effective task schedule in HeDCSs requires the consideration of both the heterogeneity of processors and high interprocessor communication overhead, which results from non-trivial data movement between tasks scheduled on different processors. *M.I. Daoud, N. Kharma* [3] have presented a new high-performance scheduling algorithm, called the longest dynamic critical path (LDCP) algorithm, for HeDCSs with a bounded number of processors. The LDCP algorithm is a list-based scheduling algorithm that uses a new attribute to efficiently select tasks for scheduling in HeDCSs. The efficient selection of tasks enables the LDCP algorithm to generate high-quality task schedules in a heterogeneous computing environment. The performance of the LDCP algorithm was compared with two of the best existing scheduling algorithms for HeDCSs: the HEFT and DLS algorithms.

Yajun Li,Yuhang Yanga, Maode Mab, Liang Zhoua,[4] addresses the load balancing problem by presenting a hybrid approach to the load balancing of sequential tasks under grid computing environments. The main objective is to arrive at task assignments that could achieve minimum execution time, maximum node utilization and a well-balanced load across all the nodes involved in a grid. A first-come-first-served and a carefully designed genetic algorithm are selected as representatives of both classes to work together to accomplish the goal. The simulation results show that the algorithm can achieve a better load balancing performance as compared to its `pure' counterparts.

L. Mohammad Khanli et al. have presented QoS based scheduling solutions in a specific architecture called Grid-JQA [5]. This scheduling solution applies an aggregation formula that is a combination of parameters together with weighting factors to evaluate QoS. The Khanli's scheduling algorithm is not practical and seems to be an unpractical mathematical solution.

In the bidding model, the key challenge of resource selection is that there is no global information system to facilitate optimum decision-making; hence requesters can only obtain partial information revealed by resource providers. To address this problem, Chien-Min Wang [6] have proposed a set of resource selection heuristics to minimize the turnaround time in a non-reserved bidding-based Grid environment, while considering the level of information about competing jobs revealed by providers.

G.Kavitha [7], proposes an efficient method of resource selection for distributed grid environments that is based on execution trust of a resource and the Quality of Service defined for the user. The resources selected are from the trusted list of resources that satisfy the user requirements of computation power for job execution with shorter response time and budget constraints imposed by the user. The selection strategy is based on user QOS parameters which improve the performance of jobs submitted to the grid and the utilization of resources that participate in the Grid. Simulation results show that the proposed model selects the appropriate resources among the available trustworthy resources with an improved power – cost ratio and maximizes the reliability of the resource.

Zhangzian [8] proposed a dynamic job scheduling algorithm for heterogeneous computational grid of autonomous nodes.In this algorithm, the communication time between nodes and scheduler is overlapped with the computation time of the nodes.so, the communication overhead can be little. The scheduler would not allocate the job to the resources which is already fully utilized.

III. OUR NEW APPROACH

We propose the Cuckoo optimization algorithm for optimal job allocation. In this system, we optimally schedule the jobs with the resources satisfying the deadline requirements of the user and also executing the job with the minimal execution time and also with minimal cost, thus the user gets benefited.

IV. CUCKOO OPTIMIZATION ARCHITECTURE

In Fig 1, our System Architecture consists of GIS, Local scheduler, Grid scheduler, Job Launching and Monitoring. Grid users, who request the resources to the Grid Schedulers, should be the authenticated members in Grid. Authentication will be done by retrieving the information from the database. They can use the resource only if they are the members of Grid. Grid Schedulers will get the information from GIS.GIS has the information about all the resources in different administrative domains. Local Scheduler will provide information about the resources in its domain. By using the information available in the GIS, Grid Scheduler will map the job to the selected resource. Once the binding of the job with the resource has been done, then the job will be executed in the allocated resource. In the job launching and monitoring phase, the status of the job will be monitored and if there is any system crash or power failure, then it will search for the next suitable and best resource and allocate this resource to the job and the job will be executed. There are three phases in scheduling the jobs and executing it.

1. Resource discovery
2. Resource selection
3. Job launching and monitoring.

1. In Resource discovery, there are three sub-phases.
   - 1.1. Authorization filtering
   - 1.2. Application requirement definition
   - 1.3. User requirement definition

1.1. The first step is to determine the set of resources that the user has access to. Without authorization, job will not run on the resources. At the end of this step, user will have set of resources to which he or she has access.

1.2. To proceed in resource discovery, user must be able to specify some minimal set of job requirement in order to filter the feasible set of resources. It may include static details and the dynamic details.

1.3. In addition to the application requirement definition, user has to specify their own requirements such as deadline and cost.

2. In Resource Selection, there are two sub-phases.
   - 2.1. Minimal requirement filtering
   - 2.2. System selection

2.1. The first step is to determine the set of resources which satisfy the minimum requirements of the users. Thus in this step, it filters out the resources that do not satisfy the minimal job requirement.

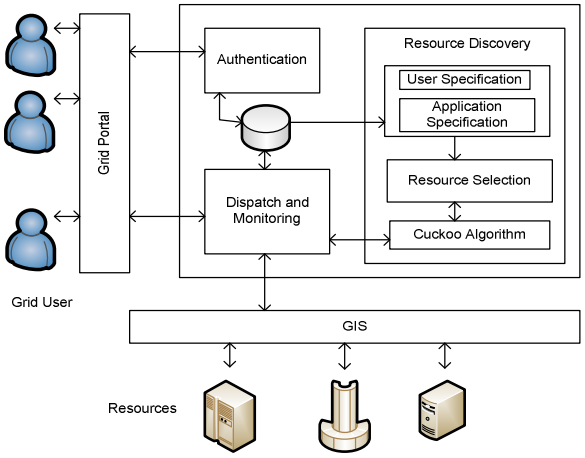

Fig. 1. Cuckoo Optimization Architecture for Job Scheduling

2.2. In the second step, our Cuckoo algorithm plays an important role. According to the user requirements, feasible resources must be selected from the filtered set of resources.

3. Job Launching and Monitoring, there are three sub-phases.
   - 3.1. Job Submission
   - 3.2. Job Monitoring
   - 3.3. Clean-up tasks

3.1. The first step is job submission. Once the resources are chosen, application can be submitted to the resources.

3.2. The next step is to job monitoring. In this phase, depending upon the running time and the application, user may monitor the progress of their application. If a job submitted is not making a sufficient progress, it has to be rescheduled. Such rescheduling is harder on a Grid system than on a single machine. For this purpose, we have developed additional primitives for interactions between local scheduler and grid scheduler.

3.3. The final step is cleanup tasks. After a job is completed, user may need to retrieve files from that resource in order to do data analysis on the results, remove temporary settings and so forth.

V. CUCKOO OPTIMISATION ALGORITHM

Cuckoo Search (CS) is an optimization algorithm developed by Xin-she Yang and Suash Deb in 2009. It was inspired by the obligate brood parasitism of some cuckoo species by laying their eggs in the nests of other host birds (of other species). Some host birds can engage direct conflict with the intruding cuckoos. For example, if a host bird discovers the eggs are not their own, it will either throw these alien eggs away or simply abandon its nest and build a new nest elsewhere. Here we map nests as the resources, cuckoo as Grid Broker, cuckoo egg as the newly arrived job and the host's eggs are considered as the jobs in the queue and the characteristics of the eggs are the constraints. If the newly

arrived job satisfies the constraints of the jobs in the resources approximately, then the job is chosen for execution. Else the job is discarded from that resource and some other optimal resource is chosen for that job.

TABLE I: NOTATIONS USED FOR AN OPTIMAL SCHEDULING USING CUCKOO ALGORITHM

Symbol	Description
R_j	Resource in the grid j=1,2,…m
T_i	Submitted task in the grid i=1,2,…..n
D_i	Deadline of the task T_i
EC_i	Expected cost of task T_i
Mem_i	Memory requirement of task T_i
Re_k^{Ti}	Resource requirement of task T_i
C_j	CPU capability of resource R_j
OC_j	Original CPU capability of R_j
$Et_{i,j}$	Estimated execution time of T_i on resource R_j
UC_j	Utilized CPU capability of R_j
$wt_{i,j}$	Estimated waiting time of T_i on resource R_j
$ct_{i,j}$	Estimated completion time of T_i on resource R_j
w_j	Workload of resource R_j
$qet_{i,j}$	Estimated execution time of T_i in queue of resource R_j
ret_j	Remaining execution time to complete task in R_j
cet_j	Completed execution time of the running task in R_j
$cost_j$	Cost of the utilized resource
$min_{i,j}$	Amount of time task T_i utilizes R_j

Cuckoo optimization algorithm is based on three simple principles that emerge from the cuckoo's strategy:

* First, each cuckoo lays one egg (a design solution) at a time, and dumps it in a randomly chosen nest.
* Second, the best nests with a high quality egg (better solution) carry over to the next generation.
* Third, the number of available host nests is fixed, and a host and there is a finite probability of the cuckoo in the nest being discovered.

1. Grid Broker will consider one job at a time and will map the job to the optimal resource.
2. The best resource which satisfies the requirements of the users gets selected and the job is submitted to that resource.
3. Once the job is allocated to the resource, it is fixed.

The grid user specifies the job T_i, deadline D_i of the job to be completed, cost limit EC_i the user can pay, memory requirement Mem_i of the job and the application requirement Re_k^T for the job T_i. First the task T_i has to be sorted according to the deadline requirement D_i of the job. The information about the available resource in the grid has to be collected from GIS. For each resource R_i, We have to check whether memory requirement Mem_i of the task T_i is satisfied by resource memory R_i. If it satisfies, the resource R_i has to be added to the list L_i. If the list is not empty, we have to calculate the original capability OC_j, utilized capability UC_j and completion time $Ct_{i,j}$ of the job T_i . The original capability of the CPU OC_j has calculated using MIPS. The utilized capability of the CPU is calculated using eq. (1):

$$UC_j=UC_j/et_{i,j} \quad (1)$$

for all the jobs in the queue of R_j.

The difference of estimated execution time $et_{i,j}$ and the completed execution time cet_j is computed using eq. (2):

$$RET_j= et_{i,j}- cet_j \quad (2)$$

```
Initialize the population
        Ti  {T1,T2,…..,Tm}
        Di  {D1,D2,…..,Dm}
        ECi  {EC1,EC2,…..,ECm}
        Memi  {Mem1,Mem2,…..,Memm}
        Rek^T  { Re1^T , Re2^T ,….., Rem^T }
Qi  ←  Sort Qi of job Ti by Di
for ¥ Rj do
  if Memi<Rimemory
     Add Ri to the list Li
end-for
while Li≠empty do
  calculate
     Oj using MIPS
     while k<wj do
        UCj=UCj/eti,j
     end while
     Cj=OCj-UCj
     eti,j=wj/Cj
          wj-1
     wti,j=∑ eti,j+RETj
           i=1
     RETj= eti,j- cetj
     cetj=current time-entry time of job into CPU
     cti,j= wti,j+ eti,j
end while
while Li≠empty do
  Li ←  Sort Li of Ri by wti,j
  if Cj>=OCj/2 && cti,j<Di then
              n
     costj=∑ nm*minm*cost
           m=1
     if wti,j> eti,j
        tj= wti,j- eti,j
        if tj<= eti,j/4
           costj=costj*95/100
        if tj<= eti,j/4 && tj>= eti,j/2
           costj=costj*90/100
    end if
    if costj<=ECi,j
       select Ri\
    end if
  end if
 Choose the resource with minimum cost for job
execution
end while
```

The execution time of the job in execution cet_j can be calculated by finding difference between the current time and entry time of job into CPU.

$$cet_j = \text{Current time - Entry time of job into CPU} \quad (3)$$

The completion time of the job is sum of waiting time $wt_{i,j}$ and execution time $et_{i,j}$

$$ct_{i,j}= wt_{i,j}+ et_{i,j} \quad (4)$$

After calculating the completion time and capability of CPU, we have to sort the resource R_j based on the waiting time

If the capability C_j is greater than half of the original capability and completion time $ct_{i,j}$ is less than the deadline then calculate the cost using the formula

$$cost_j = \sum_{m=1}^{n} n_m * min_m * cost_m \quad (5)$$

If the waiting time of the job $wt_{i,j}$ greater than the execution time of the job $et_{i,j}$ cost of the resource will be reduced based upon the certain condition. If the cost of the resource is less than or equal to the expected cost then select the resource which is of minimum cost for job execution.

VI. CONCLUSION

Cuckoo optimization approach selects the optimal resource from the set of available resources. The main issue in existing computational grid resource allocation is to discover suitable resources and to schedule separate tasks of an application on those resource in such a way to satisfy deadline requirement of the job with minimum execution time and cost. Thus the cuckoo optimization algorithm for optimal job allocation of resources on each node. This system will allocate the job optimally by considering the deadline requirement of the users and also the minimal execution time.

REFERENCES

[1] Yang Gaoa,, Hongqiang Rongb, Joshua Zhexue Huangc, "An Adaptive Grid Job Scheduling With Genetic Algorithms", Future Generation Computer Systems, 21,2005,pp 151–161.

[2] J.H. Abawajy, "An Adaptive Hierarchial Scheduling Policy For Enterprise Grid Computing", Journal of Network and Computer Applications ,32 ,2009,pp 770–779

[3] M.I. Daoud, N. Kharma , "A High Performance Algorithm For Static Task Scheduling In Heterogeneous Distributed Computing Systems", J. Parallel Distrib. Compu,. 68 ,2008, pp 399 – 409

[4] Yajun Li,Yuhang Yanga, Maode Mab, Liang Zhoua, "A Hybrid Load Balancing Strategy Of Sequencial Task Of Grid Computing Enviornment", Future Generation Computer Systems 25 ,2009 ,pp 819_828

[5] L.Mohamad Khalini,M.Analoui, "An approach to grid resource selection and fault management based on ECA rules", Future Generation Computer Systems 24 ,2008,pp 296–316

[6] Zhang jian and Xinda Lu,"A Dynamic Job Scheduling Algorithm for Computational Grid",Department of Computer Science and Engg. Shanghai Jiaotong University, Shanghai 200030,China

[7] G.Kavitha and V.Sankaranarayanan," Resource Selection in Computational Grid Based on User QoS and Trust", IJCSNS International Journal of Computer Science and Network Security, VOL.11 No.3, March 2011,pp 214-221

[8] Chien-Min Wanga, Hsi-Min Chenb, Chun-Chen Hsuc, Jonathan Lee b," Dynamic resource selection heuristics for a non-reserved bidding-based Grid environment", Future Generation Computer Systems, 7 August 2009, 26 (2010) 183_197.

[9] Fangpeng Dong and Selim G. Akl," Scheduling Algorithms for Grid Computing: State of the Art and Open Problems", School of Computing, Queen's University Kingston, Ontario January 2006, No. 2006-504

[10] Jennifer M. Schopf, "Ten actions when Grid Scheduling", Mathematics and Computer Science Division, Argonne National Laboratory.

M. Prakash received the B.E Degree in Computer Science and Engineering from University of Madras and M.E Degree from Sathyabama University, Chennai, India. He is pursing PhD in Faculty of Computer Science Engineering, Jawaharlal Nehru Technological University, Hyderabad, Andhra Pradesh, India. Presently he is working as a Assistant Professor in the Department of Computer Science and Engineering, Rajalakshmi Engineering College, Chennai, India. His research area includes Grid Computing and Distributed Computing. He published around 10 papers in National and International conferences and journals. He is the life member in professional society of ISTE and CSI.

R. Saranya pursuing final year B.E Computer Science and Engineering in Rajalakshmi Engineering College, Chennai. She is member of IEEE. She has done project in "The Great Mind Challenge" conducted by IBM, (saranyarajendren79@gmail.com)

K. Rukmani Jothi pursuing final year B.E Computer Science and Engineering in Rajalakshmi Engineering College, Chennai. She is IBM Tivoli certified.

A. Vigneshwaran pursuing final year B.E Computer Science and Engineering in Rajalakshmi Engineering College, Chennai. His area of interest includes Grid Computing.