A Smart Job Scheduling System for Cloud Computing Service Providers and Users: Modeling and Simulation

Kushal Dutta
Computer Science & Engineering Department,
Kalyani Government Engineering College,
Kalyani, Nadia- 741235, West Bengal, India
snigdhasraban@gmail.com

Ramendu Bikash Guin
Computer Science & Engineering Department,
Kalyani Government Engineering College,
Kalyani, Nadia- 741235, West Bengal, India
ramenduguin@gmail.com

Sourav Banerjee
Computer Science & Engineering Department,
Kalyani Government Engineering College,
Kalyani, Nadia- 741235, West Bengal, India
contact200683@gmail.com

Sayan Chakrabarti
Computer Science & Engineering Department,
Kalyani Government Engineering College,
Kalyani, Nadia- 741235, West Bengal, India
sayanchakrabarti7@gmail.com

Utpal Biswas
Computer Science & Engineering Department,
University of Kalyani,
Kalyani, Nadia- 741235, West Bengal,India
utpal01.in@yahoo.com

***Abstract*: Cloud computing aims on delivery of fault tolerant , scalable and reliable infrastructure to host Internet based application services. Our work presents the implementation of an efficient Quality of Service (QoS) based smart-scheduler along with Backfill strategy based light weight Virtual Machine Scheduler for dispatching jobs. The user centric smart-scheduler deals with selection of proper resources to execute high level jobs. The system centric Virtual Machine (VM) scheduler optimally dispatches the jobs to processors for better resource utilization. We also present our proposals on scheduling heuristics that can be incorporated at data center level for selecting ideal host for VM creation. Here Pollaczek-Khintchine (M/G/1) queuing model with non – preemptive priority and single server has been used to build an advanced job-scheduling system, assuming that Cloud-users' jobs come to the server following Poisson distribution while the process time to each job by the server has a general distribution. The implementation can be further extended at the host level, using load balancing in cloud environment.**

Keywords*: *Cloud Computing, Quality of Service(QoS), Non-Preemptive job scheduling, Poisson distribution, Middleware.

I. Introduction

Cloud computing aims to power the next generation data centers and enables application service providers to lease data center capabilities for deploying applications depending on user QoS (Quality of Service) requirements. Cloud applications have different composition, configuration, and deployment requirements. Quantifying the performance of resource allocation policies and application scheduling algorithms at finer details in Cloud computing environments for different application and service models under varying load, energy performance (power consumption, heat dissipation), and system size is a challenging problem to tackle.

Quantifying the performance of scheduling and allocation policies in a real Cloud environment for different application and service models under different conditions is extremely challenging because: (i) Clouds exhibit varying demand, supply patterns, and system size; and (ii) users have heterogeneous and competing QoS requirements. To use cloud-recourses efficiently and gain maximum profit simultaneously is our main focus in this paper and we are trying to achieve this goal through our simulation. Besides, Quality of Service (QoS) is another important metric regarding it. In this paper, we use the M/G/1 Queue[9] working with vacations, in which a server works at a different rate rather that completely stops during the vacation period. This feature was not found in previously released Optimistic Differentiated Scheduling and so was noticed as a major bottleneck. Server in vacation time, remains idle when there is work to be done. The server goes idle state at the end of each busy-period (when the queue becomes empty) and remains idle for some random amount of time. If the queue is still empty at the end of a vacation[2], the server immediately starts another vacation. Otherwise, the server begins a job service time. We have used Q.T.M widely recognized as powerful and realistic tools for the performance evaluation and sometimes prediction of complex mobile networks.

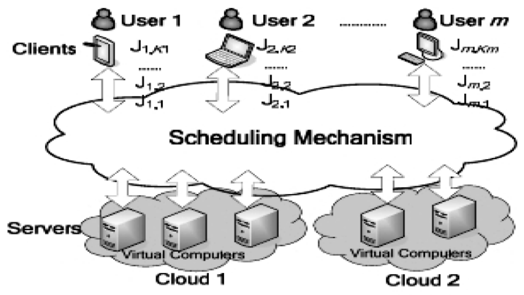


Figure 1. Scheduling mechanism

Cloud users (CU) always want how to make use of cloud resources including the due time of job finishing, computing capacity in cost- effective way whereas the cloud service providers (CP) try to gain maximum profit. Now the problem is that everyone is strict on their point and never wants to shift from their requirements. So to satisfy both ends need's, job scheduling has no other alternatives left except it.

II. Related Work

Previously developed algorithms include[1,8]:

A. Hierarchical Job Scheduling

B. Iterative Scheduling algorithms on the grids

C. Stochastic algorithm for QoS-constrained workflows job scheduling in a web service-oriented grid.

D. QoS oriented Dynamic Scheduling System(DSS).

In recent years, more and more academic researchers began mean 't' and second moment t 2 .In each class with same to study the QoS of job scheduling system; we can see that references put forward the approach of QoS performance analysis for Cloud Computing services with dynamic scheduling system. However, most research papers rarely mention the differential service-oriented QoS guaranteed job scheduling system in a Cloud computing environment. Apart from this, very a few papers care about for the how to make the maximum profits for CP. For, the conditions of existence for a Cloud Computing environment are that it must make profits for the CP with the lowest system costs. To meet the CU's job QoS requirement, job scheduling system should use the Cloud computing resource as little as possible.

III. Our Proposed Scheduling Model In Cloud Computing Environment

In reality, cloud computing environment plays as a very powerful server. This server will handle the CU's jobs (See Figure.1). For each Cloud User may has different QoS requirement, usually, Cloud Users'(CU) jobs have different priorities to be processed. So we can classify the jobs priorities into several classes.

A. Queuing Structure Model

According to the CU's jobs due time of finishing which is set by CU, CU's jobs are classified into N different classes by their different priorities. Each class has a priority. The small number of i is, the higher priority of the class is. Class 1 has the highest priority in the queue.

We assumed that CU's jobs in different classes with different priority. And they come to the server with a Poisson distribution process with rate 'λ', while the process time to each job by the server has a general distribution(G). Thus, we can build a M /G /1 queuing model with non-preemptive system. '1' denotes that the number of server is one. This M/G/1 Queue provides us with a more general case when comparing it to the M/M/1 which has Poisson arrival rate and exponential service times. There is a drawback though: the M/G/1 queue does not have a general, closed form distribution for the number of transactions in the queue in steady state. However, it provides a generalization based on average values, i.e., it gives a general solution for the average number of transaction in the queue, and the application of Little's Theorem[2] provides us the corresponding result for the average time spent in the queue.

To measure the characteristics of CU's jobs and CP's computing resources, we assume that CU's jobs in the same class with priority are submitted to the Cloud according to Poisson distribution with rate λi, and job scheduling system in the Cloud will assign some resources in the Cloud to process each job with a general service time distribution of mean 't' and second moment $\overline{t^2}$ .In each class with same priority, CU's jobs are processed by the order of its arrival (FIFO).

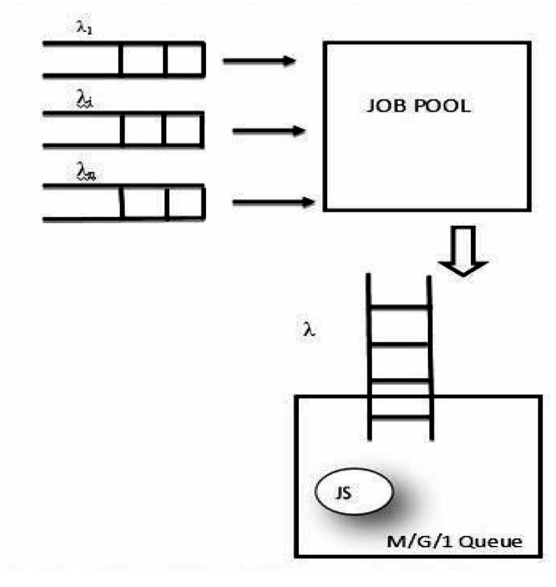


Figure 2.. Queuing Model for Cloud Computing service

For the CU's jobs in each class are in accord with Poisson distribution at the rate of λi , we can easily get the total rate

$$\lambda = \sum_{i=1}^{n} \lambda_{i.}$$

λ of all CU's jobs that submitted to the Cloud Computing environment, and

Service time 't' is represented by any probability distribution

$$L_s = \lambda E\{t\} + \frac{\lambda^2 (E^2\{t\} + \text{var}\{t\})}{2(1 - \lambda E\{t\})}$$

with mean E{t} and var{t} or second moment $\overline{t^2}$.

The probability that the facility is empty (idle) is computed as p_0=1- λE{t} = (1- ρ)

where Ls is the length of the queue in the system.

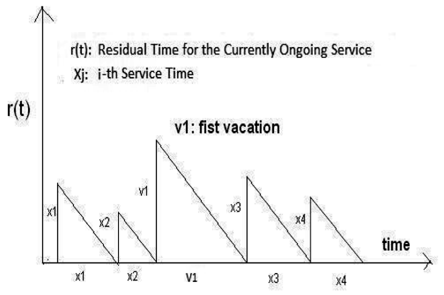


Figure 3. The Service Time for each class in a Cloud Computing Environment including Vacation Period

Here we assume the following parameters:

- the service rate of the Cloud Computing environment is μ ,
- the traffic intensity of the Cloud Computing environment is ρ ,
- the traffic intensity of jobs in each class $\rho_i = \lambda_i * t_i$.

Then we can have the average service time 't' and the traffic intensity ρ of the cloud computing environment is:

$$t = \frac{1}{\mu} = \sum_{i=1}^{n} \frac{\lambda_i t_i}{\lambda} \quad \text{and}$$

$$P = \frac{\lambda}{\mu} = \sum_{i=1}^{n} \rho_i \quad \text{where } \rho <= 1.$$

$$R(t) = \frac{1}{t}\int_0^t R(\zeta)d\zeta$$

$$= \frac{1}{t}\left[\sum_1^{M(t)} \frac{x_i^2}{2} + \sum_1^{L(t)} \frac{v_j^2}{2}\right]$$

$$= \frac{M(t)}{2t}\sum_1^{M(t)} \frac{x_i^2}{2} + \frac{L(t)}{2t}\sum_1^{L(t)} \frac{v_j^2}{2}$$

$$R = \lim_{\alpha \to \infty}\left[\frac{M(t)}{t}\frac{E[x^2]}{2} + \frac{L(t)}{t}\frac{E[v^2]}{2}\right]$$

[N.B: L(t) is the number of vacations taken upto time 't'. M(t) is the number of customers served b y time 't']

$$\text{As } t \to \infty, \ \frac{M(t)}{t} \to \lambda \ \text{ and } \ \frac{L(t)}{t} \to \lambda_v$$

(= vacation rate.)
Let I=1, if system is on vacation, I=0 if system is busy.

E[I]=P(system idle)= 1- ρ = λ_vE[v]

$$\Rightarrow \lambda_v = \frac{(1 - \rho)}{E[v]}$$

Hence

$$R = \frac{\lambda E[x^2]}{2} + \frac{(1 - \rho)E[v^2]}{2E[v]}$$

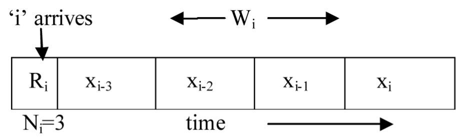
$$\text{Waiting time, } W = \frac{\lambda E[x^2]}{2(1-\rho)} + \frac{E[v^2]}{2E[v]}$$

$$\text{as } W = \frac{R}{1-\rho}$$

Let Wi=waiting time in queue of i-th arrival and Ri=Residual service time seen b y i.

[N.B: Residual time -> amount of time for current customer receiving]

service to be done, Ni= number of customers found in queue by i.



$$W_i = R_i + \sum_{i-Ni}^{i-1} x_j \quad \text{and}$$

$$E[W_i] = E[R_i] + E[X]E[N_i] = r + N_Q/\mu$$

$$W = R + \frac{\lambda W}{\mu}$$

$$\Rightarrow W = R + \rho W$$

$$\Rightarrow W = \frac{R}{1-\rho}$$

According the Little's law, the average number of jobs in each class waiting queue is N_q^i, and
N_q^i = λi*Wi

B. Cost Models

Cost models attempt to balance two conflicting costs:

- Cost of offering the service.
- Cost of delay in offering the service (customer waiting time).

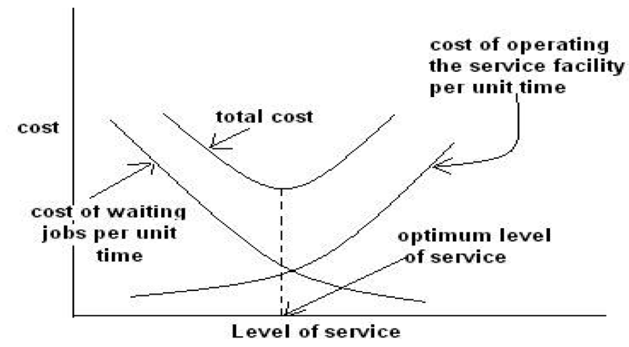


Figure 4. Cost based Queuing Decision Model

The two types of costs are in conflict because an increase in one automatically causes reduction in the other, as demonstrated below.
Letting x(=μ or c) represent the service level , the cost model can be expressed as ETC(x) = EOT(x) + EWC(x), Where

ETC = Expected total cost per unit time.

ECO = Expected cost of operating the facility per unit time.

EWC = Expected cost of waiting per unit time The simplest forms for EOC & EWC are the following linear functions:

$EOC(x) = C_1x$

$EWC(x) = C_2L_s$

Where C_1 = Marginal cost per unit of x per unit time & C_2= Cost of waiting per unit time per (waiting) jobs. L_s is the length of job queue in the system.

IV. Cost Analysis For Advanced Job Scheduling System

Let us assume that each job has its due time of finishing & T^i_{tolerant} is the maximum tolerant delay time for the Cloud Computing User's(CU) job. From the user side, to meet the QoS requirement of the Job

Cloud Computing Service Providers(CP) always want the maximum profits & it is their common ultimate aim, Apart from meeting condition (i), they must determine to what extent they should meet equation, how much is the maximum profits by assigning suitable resource service rate "μ" to each job in different priority class. We assume that :

- ∂ is the unit cost of Cloud Computing service
- ψ is the cost for the job waiting in the cloud

Now we can get the cost function for each queue:

$$\omega_i(\mu) = \partial_i \mu_i + \psi_i N^i_q$$

i

$$\omega_i(\mu_i) = \partial_i \mu_i + \psi_i * \frac{\lambda_i R}{m * n}$$

-------------(ii)

where $m = \left(1 - \sum_{k=1}^{(i-1)} \rho_k\right), \quad n = \left(1 - \sum_{k=1}^{(i)} \rho_k\right)$

$$\therefore \omega_i(\mu_i) = \partial_i \mu_i + \psi_i * \frac{\lambda_i R}{\left(1 - \sum_{k=1}^{(i-1)} \rho_k\right) * \left(1 - \sum_{k=1}^{(i)} \rho_k\right)}$$

Here N^i_q = λi* Wi (W= the waiting time of a job in class i) and

$$W_i = \frac{R}{\left(1 - \sum_{k=1}^{(i-1)} \rho_k \sum\right) * \left(1 - \rho_k\right)_{k=1}^{(1)}}$$

Where 'R' being the mean Residual service time with vacation for all jobs.

For a job in class '1', the waiting time is $W_1 = \frac{R}{1-\rho_1}$ and the waiting time of a job in class 'i' is

$$W_i = \frac{R}{\left(1 - \sum_{k=1}^{(i-1)} \rho_k \sum\right) * \left(1 - \rho_k\right)_{k=1}^{(1)}}$$

The cost function for the Cloud is:

$$\omega(\mu) = \sum_{i=1}^{n} \omega_i(\mu_i)$$

-------------(iii)

The cloud computing service provider would face a problem that is how to use job scheduling in order to maintain a suitable cloud computing resources service rate "μi" to get the whole Cloud's maximum cost value of ω (μ).

Theoretically, we can assign each $\frac{\partial(\omega_i)}{\partial(\mu_i)} = 0$, eventually, we' ll get the vector $\mu^* = \{\mu^*_1, \mu^*_2, \ldots, \mu^*_i, \ldots \mu_n^*\}$. μ* is the optimistic value for the CP to get the maximum profit. The work for the job scheduling system to do is to regulate * * *
$\{\mu_1, \mu_2, \ldots, \mu_i, \ldots \mu_n\} \rightarrow \{\mu^*_1, \mu^*_2, \ldots, \mu^*_i, \ldots \mu^*_n\}$

TABLE 1. RANDOM JOB SUBMISSION WITH I-SCHEDULING TECHNIQUE

Total Number of Job Arrived	Number of jobs served form waiting queue	Total jobs served	Total Service time (ms)	Avg service Time (ms)	Total resource allocated	Total cost ($)	Through put (jobs/sec)	Average Turn Around Time (ms)
940	836	940	1416.839	1.5072	46394	927880	662	671
820	724	820	1250.841	1.5254	40870	817400	655	636
786	678	786	1210.63	1.5402	38603	772060	648	585
664	561	664	1014.32	1.5275	33187	663740	654	466
544	432	544	848.6402	1.560	27342	546840	640	363
415	312	415	658.78	1.587	19932	221436	629	256
317	214	317	468.659	1.478	16170	323244	675	204
227	112	227	322.18	1.4192	10161	203240	895	147
168	64	168	249.40	1.4845	8294	987	673	81

V. Implementation Of Advanced Job Scheduling Algorithm

According to the aforesaid analysis, we provide the description of the core part of the algorithm. Step.1: Each class is assigned different priority with service

rate $\mu_i^0 = \frac{\overline{t_i^2}}{2\overline{t_i}}$, until $T_i \le T^i_{tolerant}$

where μ0 stands for the minimum cloud recourses that assign to class 'i'. This procedure will guarantee the QoS for the jobs in each class.

Step.2: For a given determined Cloud Computing environment, according to survey from the Cloud Computing service provider, set the value of ∂I , ψi for each class with different priority; and give a very small convergence value Є

Step.3: By equation (iii), calculate the value of $\bar{\omega}_s$

Step.4: Then increase each μ_i (assign more Cloud resources to class i), and get the value of $\bar{\omega}_e$ by equation (iii).

Step.5: Calculate the value of $\Delta = |\ \overline{\omega_s} - \overline{\omega_e}\ |$

Step.6: IF Δ <= Є then go to END
ELSE go to step. 4.

VI. Comparative Study And Analysis Of Our Cloud Simulation

After the mg1 queue or the final job queue is formed then we have created six algorithms that the middleware has to choose first to provide cloud services. Our six developed algorithms are Random Job Submission with i-Scheduling Technique, Shortest Job Priority Technique, Longest Job Priority Technique, Random Job Submission i-Scheduling Without Waiting Technique, Shortest Job Priority without Waiting Technique and Longest Job Priority without Waiting Technique respectively. In the first case ,randomly jobs are arrived and they are pushed to mg1 queue (like fig 2) in a manner three jobs from first queue, two jobs from second queue and one jobs from third queue at one go. In the second case, the jobs in the outside queues are sorted increasingly and then pushed to the mg1 queue one after another queue, i.e. first all jobs are pushed in the mg1 queue, then all jobs form second outside queue and so on. This case is the same as previous except that jobs are sorted decreasingly in the outside queues before final job queue formation. Fourth, fifth and sixth cases are similar as the first, second and third cases respectively except that in the later cases the cloud resources are too much that no jobs need to be pushed to the waiting queue before service. Among these cases, it is observed that first and fourth case always give better performance i.e. turnaround time not increasing so much as number jobs increased causing users to wait minimum amount of time to get their jobs served than the other cases. Case fourth is the best case as it has provision that cloud servers have more resources and every job always gets its demanded resource without waiting for. Fourth and fifth cases for this similar logic show good performance than second and third cases respectively. But, we cannot except a huge amount and non depleted resource initially at the cloud servers, so first case is always the best case. When the jobs occurs in a little quantity, then first , second and third cases may be raised on fourth ,fifth and sixth cases respectively. The observed tables and graphs of first and second cases are shown below The graphs of Random Job submission with i-Scheduling technique is shown below.

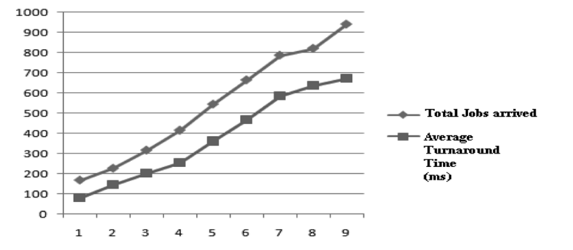


Figure 5. Random job submission with i-Scheduling Technique

TABLE 2. SHORTEST JOB PRIORITY TECHNIQUES

Total Number of Job Arrived	Number of jobs served form waiting queue	Total jobs served	Total Service time (ms)	Avg service Time (ms)	Total resource allocated	Total cost ($)	Throughput (jobs/sec)	Average Turn Around Time (ms)
912	812	912	1410.40	1.546	46415	928300	646	901
829	722	829	1271.4598	1.5337	41982	839640	651	873
781	677	781	1220.94	1.5633	38080	761600	639	744
678	566	678	1061.63	1.56	33331	666620	638	665
599	506	599	921.34	1.538	30904	618080	649	596
456	349	456	671.38	1.4723	23160	463200	678	451
363	275	363	569.399	1.5685	18166	363320	637	357
298	193	298	435.32	1.4608	15349	86244	684	290
180	80	180	260.82	1.449	8818	176360	689	174

The graphs of Shortest Job priority Techniques is shown below.

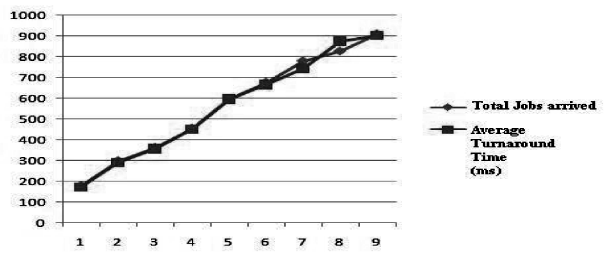


Figure 6. Shortest Job Priority technique

The fourth case graph where initial cloud resource is more is shown below, generally when the number jobs jobs arrived in a few quantity then the first case is converted to this case and it has the highest performance, i.e. low turnaround time, causing users to wait minimum time to get their jobs served.

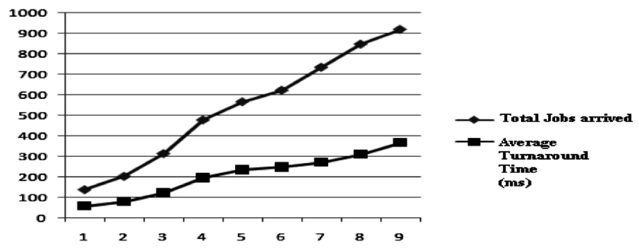


Figure 7. Random Job Submission i-Scheduling Without Waiting Technique

We see that when number of jobs varies between 900 and 100, the average turnaround time varies between 600 and 700 for first case, 900 to 1000 in second case and 300 to 400 in fourth case respectively, and this behavior is observed after many simulations.

VII. Conclusion And Future Work

In this paper, we have proposed a job scheduling algorithm of our own for efficient cloud computing resource management for both cloud computing service providers and users. Our paper is based on non-preemptive M/G/1 queuing model. We have done vacation time measurement in our algorithm, and can infer that this would suffice to reality. The realization of our algorithm, based on comparative studies clearly shows its efficiency.

Reference

[1] Weidong, H., Y. Yang, and L. Chuang. Qos Performance Analysis for Grid Services Dynamic Scheduling System. in Wireless Communications, Networking and Mobile Computing 2007. WiCom 2007. International Conference on. 2007.

[2] Emmanuel Arzuaga and David R. Kaeli, "An M/G/1 Queue Model for Multiple Applications on Storage Area Networks", Northeastern University Computer Architecture Research

[3] Keqin Li , "Experimental Performance Evaluation of Job Scheduling and Processor Allocation Algorithms for Grid Computing on Metacomputers" , Proceedings of the 18th International Parallel and Distributed Processing Symposium (IPDPS'04)

[4] Kaiqi X iong and Harry Perros , "Service Performance and Analysis in Cloud Computing" , 2009 Congress on Services – I

[5] T.M.Chen and B.R.Wilkins ," A Set of New and Efficient Formulae for Buffer Size Analysis of Real- Time Systems Using M/G/P Models", page 186-190

[6] Ling-Feng Chiang, and Jiang-Whai Dai," The Performance Analysis in MPLS Network Recovery Using M/G/1Queueing Model", 2006 IEEE International Conference onSystems, Man, and Cybernetics October 8-11, 2006, Taipei, Taiwan, page3938-3942

[7] Luqun Li, "An Optimistic Differentiated Service Job Scheduling System for Cloud Computing Service Users and Providers",2009 Third International Conference on Multimedia and Ubiquitous Engineering, page 295-299

[8] Kiran, M., et al. A prediction module to optimize scheduling in a grid computing environment. in Computer and Communication Engineering, 2008. ICCCE 2008. International Conference on. 2008

[9] Hock, N.C., Queueing Modelling Fundamentals. JOHN WILEY&SONS, 1997.