Indian Journal of Science and Technology, *Vol 9(4), DOI: 10.17485/ijst/2016/v9i4/80553, January 2016* *ISSN (Print) : 0974-6846*
ISSN (Online) : 0974-5645

Deadline Aware Two Stage Scheduling Algorithm in Cloud Computing

Indukuri R. Krishnam Raju[1*], Penmasta Suresh Varma[1], M. V. Rama Sundari[2] and G. Jose Moses[3]

[1]Department of Computer Science and Engineering, Adikavi Nannaya University, Rajahmundry - 533296, Andhra Pradesh, India; irkbvrice@gmail.com, vermaps@yahoo.com
[2]Department of Computer Science and Engineering, ISTS College of Engineering, Rajahmundry – 533294, Andhra Pradesh, India; mvramasundari@gmail.com
[3]Department of Computer Science and Engineering, Rahgu Engineering College, Visakhapatnam - 531162, Andhra Pradesh, India; josemoses@gmail.com

Abstract

Background/Objectives: Cloud computing is a large-scale distributed computing paradigm in which a pool of abstracted, virtualized, dynamically-scalable resources such as computing power, storage, platforms and services are delivered on demand to external customers over the Internet. In cloud computing scheduling is the process of deciding how to allocate resources in the form of virtual machines for the requested jobs. **Methods**: The proposed Deadline Aware Two Stage Scheduling in cloud computing is to schedule Virtual Machines (VM) for the requested jobs received from customers. In this model each job requires two types of VM's in a sequence to complete its task. This model allocates VM's as resource to the requested jobs based on processing time and scheduling the jobs by considering deadlines with respect to response time and waiting time. **Findings and Improvements:** A simulation environment was developed and analyzed to evaluate this model by considering the evaluation metrics of average turnaround time, average waiting time and violation in deadlines when compared with First Come First Serve (FCFS) and Shortest Job First (SJF) scheduling strategies. This model reduces the evaluation metrics by constant factor when compared with other scheduling approaches.

Keywords: Cloud Computing, Resource Allocation, Two Stage Scheduling, Virtual Machine

1. Introduction

Cloud computing is a new paradigm in which computing is delivered as service rather than product, by providing shared resources, software and information to consumers as a utility over networks. One of the prime advantage of cloud computing is reduction of the capital expenditures of systems from the perspective of cloud users and service providers. The cloud computing paradigm is suitable for a wide range of applications such as hosting websites, social networking, scientific workflows, e-business, customer services and high performance computing.

Cloud computing uses virtualization and the modern web to dynamically provide resources of various kinds as services. Virtual Machine is "a representation of real machine using software that provides an operating environment which can run or host a guest operating system". Virtualization enables multiple operating systems and applications to run concurrently on a single physical host machine. Multiple Virtual Machines will share the resources of the physical host machine ensuring better utilization, optimization and resource efficiency. In cloud computing, a typical datacenter consists of computing machines connected by high speed network links. This environment is well suited for the computation of large, diverse group of tasks.

Johnson S. M.[1] presented an approach for optimal two and three stage production schedules with inclusion of set-up time that results in optimal schedule of jobs with minimum elapsed and waiting time. Dudek R. A.[2] proposed m-stage decision rule for scheduling n jobs through m machines with no passing allowed and minimization of

**Author for correspondence*

idle time accumulated on the last machine to process each job. Giglio R.[3] discussed approximate solutions to the three machine scheduling problem by applying computational methods for solving the classic three-machine scheduling model. Gorenstein S.[4] discussed an algorithm for project job sequencing with resource constraints which generates maximum flow for the given jobs in a project. The algorithm employs a maximum-flow computation as a check for feasibility with respect to available resources.

Khalid O.[5] explained deadline aware Virtual Machine scheduler for grid and cloud computing which improves utilization of resources. This approach is to optimize job deadlines when run in Virtual Machines by developing a deadline-aware algorithm that responds to job execution delays in real time and dynamically optimizes jobs to meet their deadline obligations. Mao M.[6] proposed auto scaling to minimize and meet application deadline in cloud workflows and achieved resource utilization and reduction in cost. Jain N.[7] discussed a model for near-optimal scheduling mechanisms for deadline-sensitive jobs in large computing clusters which maximize resource utilization. Indukuri R. K.[8] developed multi-stage scheduling in cloud computing by reducing total elapsed time, average awaiting time and average turnaround time. Moens H.[9] represented a cost-aware scheduling of deadline-constrained task workflows in public cloud environments to find a solution with the same cost as the best cost which could be determined by the optimal algorithm. Gao Y.[10] discussed an energy and deadline aware resource provisioning, scheduling and optimization framework for cloud systems. Indukuri R. K. R.[11] proposed two stage scheduling algorithm by minimizing total elapsed time, average waiting time and average turnaround time. Bodík P.[12] has explained deadline-aware scheduling of Big-Data Processing Jobs to maximize the throughput by proposing novel algorithm for scheduling big data jobs on large compute cluster. Shyamala K.[13] discussed different resource allocation methods in cloud computing, and concluded that efficient resource allocation can optimize cost, time and power consumption. It can also minimize the underutilization of resources, balance load, request loss and leasing cost.

Hamid T.[14] mentioned a novel optimization algorithm in the basis of gravitation law and mass interactions. The advantages of this algorithm are velocity, implementation time and optimal scheduling of works in workflow workshop environment. Bargheri R.[15] presented a model for Scheduling Workflow Applications on the Heterogeneous Cloud Resources based upon the idea of detecting the set of tasks that could run concurrently and distribute them into different sub-workflows and then allocate each sub-workflow in resource cluster instead of allocating individual tasks. Hemamalini M.[16] discussed memory constrained load shared minimum execution time grid task scheduling algorithm in a heterogeneous environment and obtained better results with low memory utilization. Jemina R. P.[17] presented a parallel approach by considering Bee Colony Optimization (BCO) in parallel with Particle Swarm Optimization (PSO) for cloud task scheduling. The two metrics namely Makespan and resource utilization are evaluated and an optimal task to resource mapping is achieved with hybridization. Shahina A. B.[18] explained a framework for resource allocation in a hybrid cloud is proposed, by combining cuckoo with PSO algorithm the search space gets increased and it effectively obtains better solutions, thereby avoiding the local optima problem of PSO. This solution guarantees the QoS requirements and increases the IaaS providers benefit. Kewal K. N.[19] discussed Two Stage Flow Shop Scheduling under Fuzzy Environment. In this paper fuzzy flow shop scheduling is used to minimize rental cost of machines under the constraints of job-block and fuzzy transportation time.

Very little work is reported regarding scheduling in cloud computing with deadlines. Hence we considered a Deadline Aware Two Stage Scheduling in cloud computing to reduce average waiting time and average turnaround times and minimizing the deadline violation with respect to waiting time and response time.

2. Deadline Aware Two Stage Scheduling Algorithm in Cloud Computing

In cloud computing a job requires cloud resources to complete the user's task. Deadline Aware Two Stage Scheduling model is shown in the Figure 1. The resources in cloud are in the form of Virtual Machines. The scheduler schedules the given n number of job requests and assigns the required cloud resources (VM's) for each job request. In Deadline Aware Two Stage Scheduling the scheduler receives n job from various users and assigns the Virtual Machines as resources by scheduling the job requests. In this model a job requires a series of Virtual Machines of two different types in sequence to complete its task.

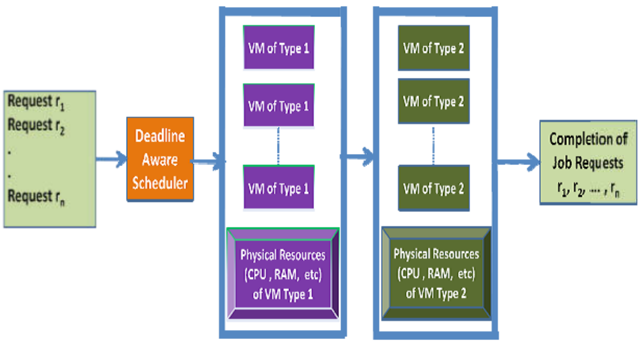

Figure 1. Model of job scheduling for resource allocation.

In the present proposed algorithm we considered the following notations. Let $r_1, r_2, \dots, r_n$ be the set of job requests in the cloud computing, where n is number of job requests and r_i is i^{th} job request. In cloud computing the resources required for a job are considered as Virtual Machine. Each Job request r_i requires t_{i1} units of time to process on Virtual Machine of type-1 (VM_1) and t_{i2} units of time to process on Virtual Machine of type-2 (VM_2) to complete its task.

Let dw_i be the deadline of r_i with respect to total waiting time and dr_i be the deadline of r_i with respect to the response time. s_i is starting time of r_i on Virtual Machine of type 1 and c_i is completion time of r_i request on Virtual Machine of type 2. We consider Average Waiting Time, Average Turnaround Time and violation in deadlines with respect to waiting time and response time are evaluation metrics of the proposed model.

The proposed algorithm works as follows: Initially the algorithm considers a solution vector (array) to store the scheduling sequence of job requests. This algorithm recursively identifies a job request r_i with shortest time among t_{i1} or t_{i2} of all unprocessed jobs. If the shortest time is t_{i1} then the job r_i is added to the solution_vector at front, otherwise (shortest time is t_{i2}) r_i is added to the solution_vector from end. The job request r_i with smallest t_{i1} will be added at front of the solution vector since it will be processed initially in the schedule to minimize waiting time of all jobs in the schedule. Waiting time of a r_i is the time taken to start its work on Virtual Machine of type-1, plus the time elapsed when it completes on Virtual Machine type-1 and starts its work on Virtual Machine of type-2. The scheduling sequence solution vecto $r = \{r_1, r_2, r_3, \dots r_n\}$ can be divided into p sub sequences since we have p number of instances for each VM type. $Seq_1 = \{r_i$ / (i mod p) = 1 where $1 \le i \le n$ }, $Seq_2 = \{ r_i$ / (i mod p) = 2 where $1 \le i \le n$ }, …, $Seq_p = \{ r_i$ / (i mod p) = 0 where $1 \le i \le n$ }. Each sub sequence Seq_i will be processed on instance of Virtual Machine type-1 and later on an instance of Virtual Machine type-2. For example the scheduling sequence which consists of 9 job request $\{r_1, r_2, r_3, r_4, r_5, r_6, r_7, r_8, r_9\}$ can be divided into 3 sub sequences where p = 3 as follows. $Seq_1 = \{r_1, r_4, r_7\}$, $Seq_2 = \{r_2, r_5, r_8\}$, $Seq_3 = \{r_3, r_6, r_9\}$.

After generating p number of Sub Scheduling Sequences Seq_i ($1 <= i <= p$), heuristic approach is applied to reduce deadline violation with respect dw_i and dr_i. Each job request r_k in sub sequence Seq_i if $t_{k1} + t_{k2} < t_{(k+1)1} + t_{(k+1)2}$ and ($c_k + t_{(k+1)1} + t_{(k+1)2} < dw_i$ and $s_k + t_{(k+1)1} < dr_i$) then positions of job requests r_k and r_{k+1} will be swapped with each other where k > size of (Seq_i)/2.

Algorithm 1: Pseudo-code for Deadline Aware Two Stage Scheduling,

Input: n job requests with processing times $t_{i1,}$ t_{i2} and deadlines dw_i , dr_i

p number of instances for VM type 1 and VM Type 2,

Output : Scheduling sequences Seq_1, Seq_2, $Seq_{3, \dots}$ $Seq_{p,}$

```
Begin.
i = 1, j = n.
solution_vector = empty.
FOR each job request r_i with shortest time period
among all unprocessed jobs do.
If t_i1 < t_i2 then.
Add the job request r_i to the solution vector at
index i.
i = i + 1;
Else.
Add the job request r_i to the solution vector at
index j.
j = j – 1.
End if.
End.
For i.=.1 to n do.
j = i % p.
Seq_j = Seq_i + solution_vector [i].
End.
For i = 1 to p do.
For each job request r_k in Seq_i with k > size_
of(Seq_i)/2 do,
If (t_k1 + t_k2) < (t_(k+1)1 + t_(k+1) 2) then,
c_k = completion time of job k in sequence S_i,
If c_k + t_(k+1)1 + t_(k+1)2 < dw_i and s_k + t_(k+1)1 < dr_i then,
Swap (r_k, r_k+1),
End if,
End if.
```

25. Calculate evaluation metrics for sub schedule S_i.
26. End.
27. Calculate aggregate evaluation metrics for entire schedule.
28. End.

The performance metrics can be computed by the following computations for a given scheduling sequence. Waiting time for request ri is difference between completion time (c_i) minus total processing time of request (t_{i1}+ t_{i2}), i.e:

$$w_i = c_i - (t_{i1} + t_{i2}) \quad (2.1)$$

Deadline violations of a job request r_i with respect to waiting time (dvw_i) is the difference between waiting time w_i and deadline violation with respect to waiting time dw_i, i.e:

$$dvw_i = w_i - dw_i \text{ where } w_i > dw_i \text{, otherwise } 0 \quad (2.2)$$

Deadline violations of a job request r_i with respect to response time (dvr_i) is the difference between starting time w_i and deadline violation with respect to response time dr_i, i.e:

$$dvr_i = s_i - dr_i \text{ where } s_i > dr_i \text{, otherwise } 0 \quad (2.3)$$

Average Waiting Time (AWT) and Average Turnaround Time (ATT) of all job requests will be computed as follows:

$$AWT = \sum_{i=1}^{n} (w_i) / n \quad (2.4)$$

$$ATT = \sum_{i=1}^{n} (c_i) / n \quad (2.5)$$

Average Deadline Violation with respect to Waiting time (ADVW) and Average Deadline Violation with respect to Response time (ADVR) can be calculated as follows:

$$ADVW = \sum_{i=1}^{p} dvw_i / n \quad (2.6)$$

$$ADVR = \sum_{i=1}^{p} dvr_i / n \quad (2.7)$$

3. Methodology and Performance Evaluation

A custom simulation environment has been developed to analyze the First Come First Serve (FCFS) Scheduling, Shortest Job First (SJF) Scheduling, Two Stage Scheduling and Deadline Aware Two Stage Scheduling with p instances for each type of Virtual Machine. Gaussian distribution is used to generate job request processing time on Virtual Machines. Initially we have applied First Come First Serve (FCFS) Scheduling algorithm for the given instance by considering the jobs in order of arriving and splitting the scheduling sequence into p sub sequences. After that SJF Scheduling is applied to the given instance by sorting the jobs in ascending order of (t_{i1} + t_{i2}) and splitting the scheduling sequence into sub scheduling sequences. Initially Two Stage Scheduling algorithm is applied for the given instance.

Deadline Aware Two Stage Scheduling algorithm is applied at the end for the given instance to generate optimal scheduling sequence and splits the scheduling sequence into p number of scheduling sub sequences. At the end we have calculated the required performance evaluation metrics for the above scheduling strategies.

First we have taken 16 job requests on two Virtual Machine types with four (p = 4) instances of each Virtual Machine type that are available. Consider an instance with n = 16 and t_{i1} and t_{i2} as shown in Table 1. Also dw_i is the deadline for total waiting time and dr_i is the deadline for response time for each job considered randomly with respect to processing time.

First Come First Server Scheduling is applied to the given instance and calculated the evaluation metrics for the instance as in Table 2. Waiting time w_i, Deadline Violations with respect to waiting time (dvw_i) and Deadline violations with respect to response time (dvr_i) are calculated using Equations (2.1), (2.2) and (2.3) respectively as shown in Table 2. The scheduling sub sequences given by this algorithm for this instance are Seq_1 = {0, 1, 2, 3} Seq_2 = {4, 5, 6, 7} Seq_3 = {8, 9, 10, 11} Seq_4 = {12, 13, 14, 15}. The 4 jobs {1, 3, 6, 7} violated deadline with respect to waiting time and the 6 jobs {1, 3, 6, 7, 11, 15} as violated deadline with respect to response time.

Shortest Job First Scheduling is applied to the given instance and calculated the evaluation metrics and are shown in Table 3. Waiting time w_i, Deadline violations with respect to waiting time (dvw_i) and Deadline violations with respect to response time (dvr_i) are calculated using Equations (3.1), (3.2) and (3.3) respectively are shown in Table 3. The scheduling sub sequences given by this algorithm for this instance are Seq_1 = {8, 1, 13, 9} Seq_2 = {6, 3, 5, 12} Seq_3 = {4, 2, 7, 0} Seq_4 = {11, 10, 15 , 14}. The 3 jobs {12, 7, 0} violated deadline with respect to waiting

Table 1. Scheduling instance with n = 16

r_i	t_{i1}	t_{i2}	dw_i	dr_i
0	360	238	897	630
1	5	269	411	275
2	139	383	783	409
3	219	209	642	489
4	316	192	762	586
5	165	306	706	435
6	160	245	607	430
7	53	470	784	323
8	252	20	522	522
9	233	103	504	503
10	595	151	111	865
11	196	539	1102	466
12	203	269	708	473
13	138	170	462	408
14	798	399	1795	1068
15	368	487	1282	638

Table 2. FCFS Scheduling performance evaluation metrics with n = 16 and p = 4 (number of instances for each VM type)

Seq_i	r_i	t_{i1}	t_{i2}	s_i	c_i	w_i	dw_i	dr_i	dvw_i	dvr_i
1	0	360	238	0	598	0	897	630	0	0
1	1	5	269	360	867	593	411	275	182	85
1	2	139	383	365	1250	728	783	409	0	0
1	3	219	209	504	1459	1031	642	489	389	15
2	4	316	192	0	508	0	762	586	0	0
2	5	165	306	316	814	343	706	435	0	0
2	6	160	245	481	1059	654	607	430	47	51
2	7	53	470	641	1529	1006	784	323	222	318
3	8	252	20	0	272	0	522	522	0	0
3	9	233	103	252	588	252	504	503	0	0
3	10	595	151	485	1231	485	1119	865	0	0
3	11	196	539	1080	1815	1080	1102	466	0	614
4	12	203	269	0	472	0	708	473	0	0
4	13	138	170	203	642	334	462	408	0	0
4	14	798	399	341	1538	341	1795	1068	0	0
4	15	368	487	1139	2025	1170	1282	638	0	501

Table 3. SJF Scheduling performance evaluation metrics with n = 16 and p = 4 (number of instances for each VM type)

Seq_i	r_i	t_{i1}	t_{i2}	s_i	c_i	w_i	dw_i	dr_i	dvw_i	dvr_i
1	8	252	20	0	272	0	522	522	0	0
1	1	5	269	252	541	267	411	275	0	0
1	13	138	170	257	711	403	462	408	0	0
1	9	233	103	395	814	478	504	503	0	0
2	6	160	245	0	405	0	607	430	0	0
2	3	219	209	160	614	186	642	489	0	0
2	5	165	306	379	920	449	706	435	0	0
2	12	203	269	544	1189	717	708	473	9	71
3	4	316	192	0	508	0	762	586	0	0
3	2	139	383	316	891	369	783	409	0	0
3	7	53	470	455	1361	838	784	323	54	132
3	0	360	238	508	1599	1001	897	630	104	0
4	11	196	539	0	735	0	1102	466	0	0
4	10	595	151	196	942	196	1119	865	0	0
4	15	368	487	791	1646	791	1282	638	0	153
4	14	798	399	1159	2356	1159	1795	1068	0	91

time and the 4 jobs {12, 7, 15, 14} are violated deadline with respect to response time.

Two Stage Scheduling is applied to the given instance and calculated the evaluation metrics which are shown in Table 4. Waiting time w_i, Deadline violations with respect to waiting time (dvw_i) and Deadline violations with respect to response time (dvr_i) are calculated using Equations (2.1), (2.2) and (2.3) respectively are shown in Table 4. The Scheduling sub sequences given by this algorithm for this instance are Seq_1 = {1, 6, 15, 4} Seq_2 = {7, 5, 14, 10} Seq_3 = {13, 11, 0, 9} Seq_4 = {2, 12, 3 8}. The 2 jobs {9, 8} are violated deadline with respect to waiting time and 3 jobs {10, 9, 8} violated deadline with respect to response time.

Deadline Aware Two Stage Scheduling is applied to the given instance and calculated the evaluation metrics which are shown in Table 5. Waiting time w_i, Deadline violations with respect to waiting time (dvw_i) and Deadline

Table 4. Two Stage Scheduling performance evaluation metrics with n = 16 and p = 4 (number of instances for each VM type)

Seq_i	r_i	t_{i1}	t_{i2}	s_i	c_i	w_i	dw_i	dr_i	dvw_i	dvr_i
1	1	5	269	0	274	0	411	275	0	0
1	6	160	245	5	519	114	607	430	0	0
1	15	368	487	165	1020	165	1282	638	0	0
1	4	316	192	533	1212	704	762	586	0	0
2	7	53	470	0	523	0	784	323	0	0
2	5	165	306	53	829	358	706	435	0	0
2	14	798	399	218	1415	218	1795	1068	0	0
2	10	595	151	1016	1762	1016	1119	865	0	151
3	13	138	170	0	308	0	462	408	0	0
3	11	196	539	138	873	138	1102	466	0	0
3	0	360	238	334	1111	513	897	630	0	0
3	9	233	103	694	1214	878	504	503	374	191
4	2	139	383	0	522	0	783	409	0	0
4	12	203	269	139	791	319	708	473	0	0
4	3	219	209	342	1000	572	642	489	0	0
4	8	252	20	561	1020	748	522	522	226	39

Table 5. Deadline Aware Two Stage Scheduling performance evaluation metrics with n = 16 and p = 4 (number of instances for each VM type)

Seq_i	r_i	t_{i1}	t_{i2}	s_i	c_i	w_i	dw_i	dr_i	dvw_i	dvr_i
1	1	5	269	0	274	0	411	275	0	0
1	6	160	245	5	519	114	607	430	0	0
1	4	316	192	165	711	203	762	586	0	0
1	15	368	487	481	1336	481	1282	638	0	0
2	7	53	470	0	523	0	784	323	0	0
2	5	165	306	53	829	358	706	435	0	0
2	14	798	399	813	2010	813	1795	1068	0	0
2	10	595	151	218	980	234	1119	865	0	0
3	13	196	539	138	873	138	1102	466	0	0
3	11	138	170	0	308	0	462	408	0	0
3	9	233	103	334	976	640	504	503	136	0
3	0	360	238	567	1214	616	897	630	0	0
4	2	139	383	0	522	0	783	409	0	0
4	12	203	269	139	791	319	708	473	0	0
4	8	252	20	342	811	539	522	522	17	0
4	3	219	209	594	1022	594	642	489	0	105

violations with respect to response time (dvr_i) are calculated using Equations (2.1), (2.2) and (2.3) respectively are shown in Table 5. The Scheduling sub sequences given by this algorithm for this instance are Seq_1 = {1, 6, 4, 15} Seq_2 = {7, 5, 14, 10} Seq_3 = {13, 11, 9, 0} Seq_4 = {2, 12, 8, 3}. The 2 jobs {9, 8} violated deadline with respect to waiting time and only one job {4} violated deadline with respect to response time.

Aggregate performance evaluation metrics for FCFS, SJF, Two Stage scheduling and Deadline Aware Two Stage scheduling like Average Waiting Time (AWT), Average Turnaround Time (ATT), Average Deadline Violation with respect to Waiting time (ADVW) and Average Deadline Violation with respect to Response time (ADVR) are calculated using Equations (2.4), (2.5), (2.6) and (2.7). Number of jobs violating deadline with respect to waiting time (v_w) and response time (v_s) are also identified for the above four scheduling strategies with n = 16 and p = 4 and shown in the Table 6.

In Deadline Aware Multi Stage Scheduling when n = 16 and p = 4 the Average Waiting Time (AWT) is reduced by 36.92% when compared with FCFS Scheduling, 26.34% when compared with SJF Scheduling, and 13.6% when compared with Multi Stage Scheduling. Similarly in Deadline Aware Multi Stage Scheduling when n = 16 and p = 4 the Average Turnaround Time (ATT) is reduced by 17.75% when compared with FCFS Scheduling, and 11.56% when compared with SJF Scheduling and 4.76% when compared with Multi Stage Scheduling. Figure 2 and Figure 3 depicts graphical representation of comparative analysis of AWT and ATT respectively.

In Deadline Aware Two Stage Scheduling when n = 16 and p = 4 the Average Deadline Violation with respect to Waiting time (ADVW) is reduced by 81.79% when compared with FCFS Scheduling, 8.34% when compared with SJF Scheduling and 74.5% when compared with Two Stage Scheduling. In Deadline Aware Two Stage Scheduling when n = 16 and p = 4 the Average Deadline Violation with respect to Response time (ADVR) is reduced by 93.37% when compared with FCFS Scheduling, 76.51% when compared with SJF Scheduling and 72.44% when compared with Two Stage Scheduling. Figure 3 depicts graphical representation of comparative analysis of ADVW and ADVR.

Number of jobs violated deadline with respect to waiting time v_w in Deadline Aware Two Stage Scheduling when n = 16 and p = 4 is 2, but 4 in FCFS Scheduling, 3 in SJF Scheduling and 2 in Two Stage Scheduling. Similarly

Table 6. Comparison of performance evaluation metrics with n = 16 and p = 4 (number of instances for each VM type)

	FCFS Scheduling	SJF Scheduling	Two Stage Scheduling	Deadline Aware Two Stage Scheduling
AWT	1305	885	668	591
ATT	3609	3189	2972	2896
ADVW	538	188	142	29
ADVR	35	35	19	16
No of Jobs violating dw_i	19	13	7	3
No of Jobs violating dr_i	10	8	5	5

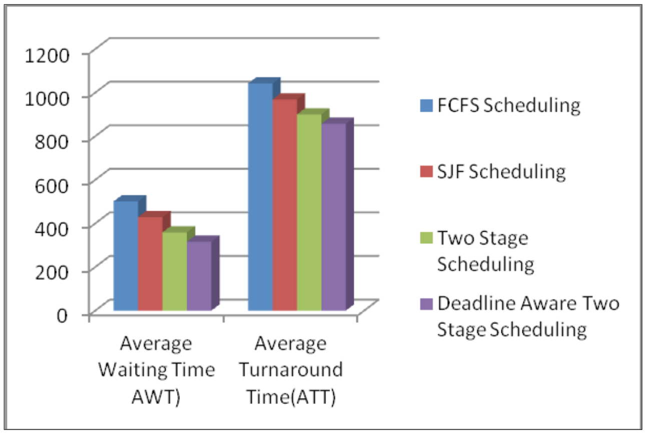

Figure 2. Comparison of AWT and ATT with n = 16 and p = 4.

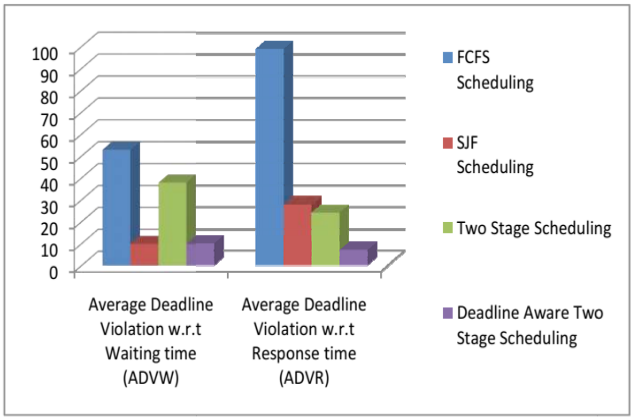

Figure 3. Comparison of ADVW and ADVR with n = 16 and p = 4.

number of jobs violated deadline with respect to response time v_r in Deadline Aware Two Stage Scheduling when n = 16 and p = 4 is 1, but 6 in FCFS Scheduling, 4 in SJF Scheduling and 3 in Two Stage Scheduling. Figure 4

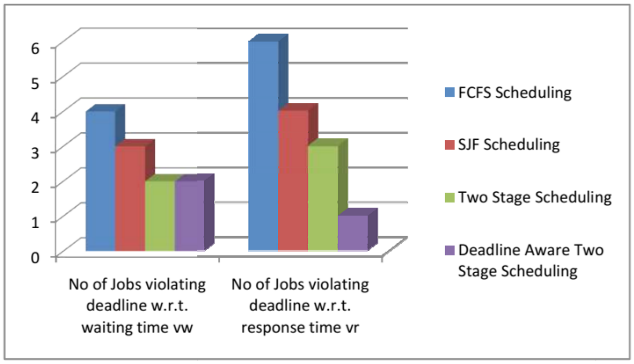

Figure 4. Comparison of no. of jobs violating deadlines with n = 16 and p = 4.

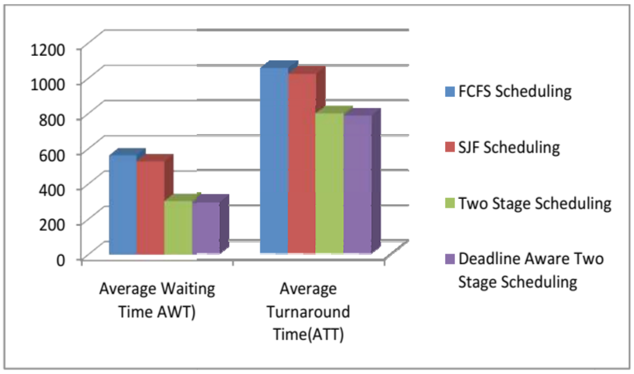

Figure 5. Comparison of AWT and ATT with n = 32 and p = 8.

depicts graphical representation of comparison analysis of v_w and v_r.

Similarly we evaluated these four algorithms with input sizes n = 32 and 128. By considering number of jobs n = 32 and number of instances p = 8 we evaluated performance evaluation metrics as shown in Table 7.

In Deadline Aware Two Stage Scheduling when n = 32 and p = 8 the Average Waiting Time (AWT) is reduced by

Table 7. Comparison of performance evaluation metrics with n = 32 and p = 8 (number of instances for each VM type)

	FCFS Scheduling	SJF Scheduling	Two Stage Scheduling	Deadline Aware Two Stage Scheduling
AWT	564	530	305	293
ATT	1048	1024	799	787
ADVW	158	90	37	15
ADVR	177	144	18	2
No of Jobs violating dw_i	12	9	4	1
No of Jobs violating dr_i	16	12	4	2

48.04% when compared with FCFS Scheduling, 44.72% when compared with SJF Scheduling, and 3.93% when compared with Two Stage Scheduling. Similarly in Deadline Aware Two Stage Scheduling when n = 32 and p = 8, the Average Turnaround Time (ATT) is reduced by 24.9% when compared with FCFS Scheduling, 23.14% when compared with SJF Scheduling and 1.5% when compared with Two Stage Scheduling. Figure 4 depicts graphical representation of comparative analysis of AWT and ATT.

In Deadline Aware Two Stage Scheduling when n = 32 and p = 8 the Average Deadline Violation with respect to Waiting time (ADVW) is reduced by 90.5% when compared with FCFS Scheduling, 83.33% when compared with SJF Scheduling and 55.88% when compared with Two Stage Scheduling. In Deadline Aware Two Stage Scheduling n = 32 and p = 8 the Average Deadline Violation with respect to Response time (ADVR) is reduced by 98.87% when compared with FCFS Scheduling, 98.6% when compared with SJF Scheduling and 88.88% when compared with Two Stage Scheduling. Figure 6 depicts graphical representation of comparison analysis ADVW and ADVR.

Number of jobs violated deadline with respect to waiting time v_w in Deadline Aware Two Stage Scheduling when n = 32 and p = 8 is 1, but 12 in FCFS Scheduling, 9 in SJF Scheduling and 4 in Two Stage Scheduling. Similarly number of jobs violated deadline with respect to response time v_r in Deadline Aware Two Stage Scheduling n = 32 and p = 8 is 2, but 16 in FCFS Scheduling, 12 in SJF Scheduling and 4 in Two Stage Scheduling. Figure 7 depicts graphical representation of comparative analysis of v_w and v_r.

By considering number of jobs n = 128 and number of instances p = 32 we evaluated performance evaluation metrics as shown in Table 8.

In Deadline Aware Two Stage Scheduling when n = 128 and p = 32 the Average Waiting Time (AWT) is reduced by 45.66% when compared with FCFS Scheduling, 31.33% when compared with SJF Scheduling and 7.87% when compared with Two Stage Scheduling. Similarly in Deadline Aware Two Stage Scheduling when n = 128 and p = 32, the Average Turnaround Time (ATT) is reduced by 20.02% when compared with FCFS Scheduling, 17.25% when compared with SJF Scheduling and 3.01% when compared with Two Stage Scheduling. Figure 8 depicts

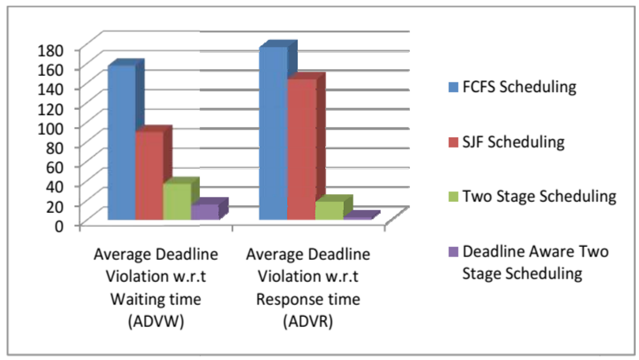

Figure 6. Comparison of ADVW and ADVR with n = 32 and p = 8.

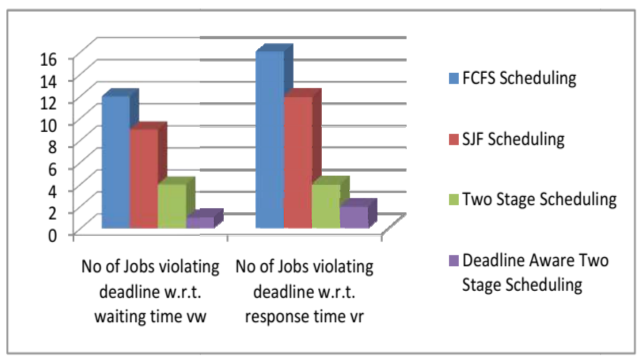

Figure 7. Comparison of no. of jobs violating deadlines with n = 32 and p = 8.

Table 8. Comparison of performance evaluation metrics with n = 128 and p = 32 (number of instances for each VM type)

	FCFS Scheduling	SJF Scheduling	Two Stage Scheduling	Deadline Aware Two Stage Scheduling
AWT	552	533	401	356
ATT	1103	1084	953	907
ADVW	153	54	104	50
ADVR	143	90	43	5
No of Jobs violating dwi	40	32	25	17
No of Jobs violating dri	50	37	23	7

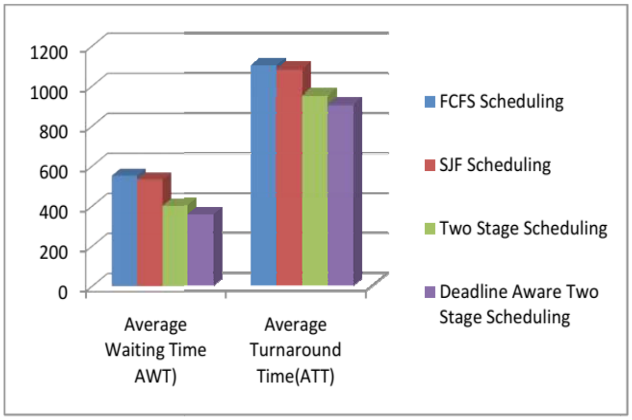

Figure 8. Comparison of AWT and ATT with n = 128 and p = 32.

graphical representation of comparative analysis of AWT and ATT.

In Deadline Aware Two Stage Scheduling when n = 128 and p = 32 the Average Deadline Violation with respect to Waiting time (ADVW) is reduced by 97.44% when compared with FCFS Scheduling, 87.11% when compared with SJF Scheduling and 90.38% when compared with Two Stage Scheduling. In Deadline Aware Two Stage Scheduling n = 128 and p = 32 the Average Deadline Violation with respect to Response time (ADVR) is reduced by 94.44% when compared with FCFS Scheduling, 90.32% when compared with SJF Scheduling and 72.72% when compared with Two Stage Scheduling. Figure 9 depicts graphical representation of comparative analysis of ADVW and ADVR.

Number of jobs violated deadline with respect to waiting time v_w in Deadline Aware Two Stage Scheduling when n = 128 and p = 32 is 3, but 23 in FCFS Scheduling, 11 in SJF Scheduling and 3 in Two Stage Scheduling.

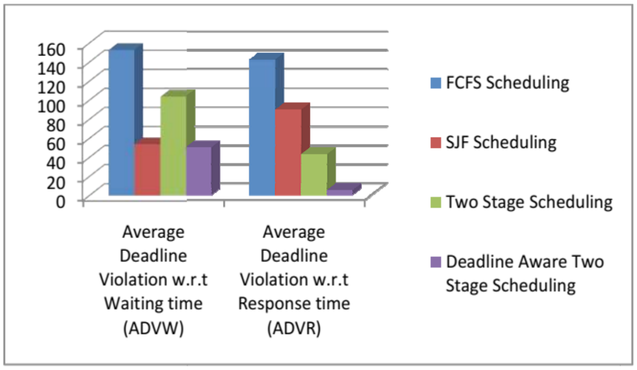

Figure 9. Comparison of ADVW and ADVR with n = 128 and p = 32.

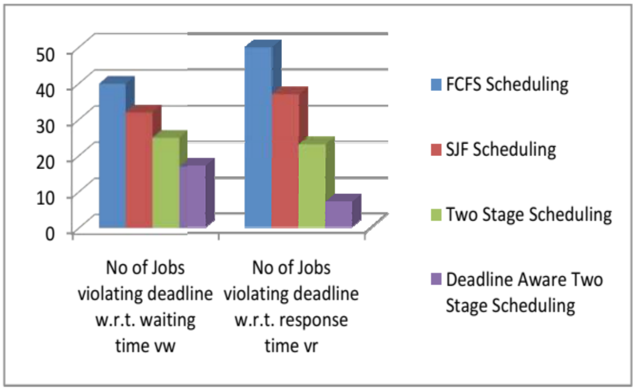

Figure 10. Comparison of no. of jobs violating deadlines with n = 128 and p = 32.

Similarly number of jobs violated deadline with respect to response time v_r in Deadline Aware Two Stage Scheduling n = 128 and p = 32 is 5, but 30 in FCFS Scheduling, 21 in SJF Scheduling and 10 in Two Stage Scheduling. Figure 10 depicts graphical representation of comparative analysis of v_w and v_r.

4. Conclusion

Scheduling n jobs on two types of Virtual Machines using deadline aware Two Stage scheduling algorithm gives better performance evaluation metrics when compared with other scheduling algorithms. Experimental results have shown that Deadline Aware Two Stage Scheduling Algorithm Average Waiting Time (AWT), Average Turnaround Time (ATT), Average Deadline Violation with respect to Waiting time (ADVW), Average Deadline Violation with respect to Response time (ADVR) is reduced reasonably when compared with FCFS, SJF and Two Stage Scheduling Algorithms. Number of jobs violating deadline with respect to waiting time (v_w) and response time (v_s) is also decreased by constant factor in Deadline Aware Two Stage Algorithm when compared with FCFS, SJF and Two Stage Scheduling Algorithms.

5. References

1. Johnson SM. Optimal 2 and 3 stage production schedules with set up time included. Journal of Novel Research Logistics. 1954 Mar; 1(1):61–8.
2. Dudek RA, Teuton OF. Development of m-stage decision rule for scheduling n jobs through m machines. Journal of Operations Research. 1964 Jun; 12(3):471–97.
3. Giglio RJ, Wagner HM. Approximate solutions to the three machine scheduling problem. Journal of Operations Research. 1964 Apr; 12(2):305–24.
4. Gorenstein S. An algorithm for project job sequencing with resource constraints. Operations Research. 1972 Jul; 20(4):835–50.
5. Khalid O, Maljevic I, Anthony R, Petridis M, Parrott K, Schulz M. Deadline aware Virtual Machine scheduler for grid and cloud computing. IEEE 24th International Conference on Advanced Information Networking and Applications Workshops; 2014. p. 85–90.
6. Mao M, Humphrey M. Auto-scaling to minimize cost and meet application deadlines in cloud workflows. ACM International Conference for High Performance Computing, Networking, Storage and Analysis; Seattle, WA. 2011 Nov 12-18. 1–12.
7. Jain N. Near-optimal scheduling mechanisms for deadline-sensitive jobs in large computing clusters. 24th ACM Symposium on Parallelism in Algorithms and Architectures. 2015 May; 2(1):255–66.
8. Indukuri RKR, Varma PS, Moses GJ. Performance measure of multi stage scheduling algorithm in cloud computing. IEEE Proceedings of the International Conference on Cloud Computing Technologies, Applications and Management; Dubai. 2012 Dec 8-10. p. 8–11.
9. Moens H, Handekyn K, De Turck F. Cost-aware scheduling of deadline-constrained task workflows in public cloud environments. IEEE International Symposium on Integrated Network Management; Ghent. 2013 May 27-31. p. 68–75.
10. Gao Y, Wang Y, Gupta SK, Pedram M. An energy and deadline aware resource provisioning, scheduling and optimization framework for cloud systems. Proceedings of the Ninth IEEE Symposium International Conference on Hardware/Software Codesign and Systems Synthesis; Montreal QC. 2013 Sep 29-Oct4. p. 1–10.
11. Indukuri RKR, Varma PS, Rama Sundari MV. Performance evaluation of two stage scheduling algorithm in cloud computing. British Journal of Mathematics and Computer Science. 2015 Jan; 6(3):247–56.
12. Bodík P, Menache I, Naor J. Brief announcement: Deadline-aware scheduling of big-data processing jobs. 26th ACM Symposium on Parallelism in Algorithms and Architectures; 2014. p. 211–3.
13. Shyamala K, Sunitha Rani K. An analysis on efficient resource allocation mechanisms in cloud computing. Indian Journal of Science and Technology. 2015 May; 8(9):814–21.
14. Tavakkolai H, Hosseinabadi AAR, Yadollahi M, Mohammadpour T. Using gravitational search algorithm for in advance reservation of resources in solving the scheduling problem of works in workflow workshop environment. Indian Journal of Science and Technology. 2015 Jun; 8(11):1–16.
15. Bagheri R, Jahanshahi M. Scheduling workflow applications on the heterogeneous cloud resources. Indian Journal of Science and Technology. 2015 Jun; 8(12):1–8.
16. Hemamalini M, Srinath MV. Memory constrained load shared minimum execution time grid task scheduling algorithm in a heterogeneous environment. Indian Journal of Science and Technology. 2015 Jul; 8(15):1–8.
17. Priyadarsini RJ, Arockiam L. PBCOPSO: A Parallel Optimization Algorithm for Task Scheduling in Cloud Environment. Indian Journal of Science and Technology. 2015 Jul; 8(16):1–5.
18. Banu AS, Helen WR. Scheduling deadline constrained task in hybrid IaaS cloud using cuckoo driven particle swarm optimization. Indian Journal of Science and Technology. 2015 Jul; 8(16):1–6.
19. Nailwal KK, Gupta D, Sharma S. Two stage flow shop scheduling under fuzzy environment. Indian Journal of Science and Technology. 2015 Jul; 8(16):1–8.