Communications on Applied Nonlinear Analysis
ISSN: 1074-133X
Vol 32 No. 1 (2025)

Nonlinear Analysis of EDF Scheduling Framework for Resource Distribution in Cloud Computing

Neelima Priyanka Nutulapati[1], Sukumar Babu Bandarupalli[2], B Sunil Kumar Babu[3], Subbulakshmi R[4]

[1]Professor, Department of CSE, Potti Sriramulu Chalavadi Mallikarjunarao College Engineering and Technology, Andhra Pradesh, India. dr.priyanka.nutulapati@gmail.com

[2]Research Scholar, Department of CSE, JNTU Kakinada, Andhra Pradesh, India. sukumar316@gmail.com

[3]Research Scholar, Department of ECE, Gandhi Institute of Engineering and Technology University, Gunupur, Odisha, India. sunil478kumar@gmail.com

[4]Assistant Professor, Department of CSE, Karpagam Institute of Technology, Coimbatore, Tamilnadu, India, subbulakshmi.cse@karpagamtech.ac.in

Article History:

Received: *17-07-2024*

Revised: *30-08-2024*

Accepted: *12-09-2024*

Abstract:
In modern times, the field of information technology is significantly shifting towards cloud computing. This innovative technology involves the permanent storage of resources on servers, which clients access via the internet. Typically, a cloud comprises a collection of resources known as virtual machines, capable of managing both computational tasks and storage needs. As the demand for cloud services continues to rise, the scheduling of these resources presents a considerable challenge. Scheduling refers to the workflow management necessary to complete tasks within the system. An efficient scheduler adapts its scheduling policy dynamically in response to changing conditions and task requirements. Cloud computing represents a recently developed technology where resources, whether physical or virtual machines, are permanently stored on servers and accessed by users through the internet. The cloud offers various service models, including Platform as a Service (PaaS), Software as a Service (SaaS), and Infrastructure as a Service (IaaS). In this context, IaaS refers to a set of resources, including virtual machines, that provide computational and storage services. Given the increasing adoption of cloud technology, resource scheduling has become a critical task. Hence, optimizing the scheduling of requests to ensure the best utilization of cloud resources is essential. This paper introduces a Nonlinear Analysis of EDF Scheduling Framework for resource allocation that considers deadlines and processing times when assigning resources to specific jobs. The performance metrics, including Average Turnaround Time, Average Waiting Time, and Average Deadline Violation, show a significant improvement when compared to traditional scheduling models such as First-Come, First-Served (FCFS), the Shortest Job First (SJF), and Simple Earliest Deadline First (EDF) models.

Keywords: Resources, Scheduling, Virtual Machines, Deadlines, deadlock violations, FCFS, SJF, EDF, Average Waiting Time, Average Turnaround time, jobs, process, etc.

1. Introduction

Cloud Computing represents an innovative technology where resources are permanently stored on servers, accessible to clients via the internet. This technology encompasses both physical and virtual machines, allowing users to leverage these resources through online connectivity. This access is made possible by the internet, a huge network of interconnected public and private networks that may be accessed through a variety of devices. Cloud services offer dynamic solutions to end users by utilizing scalable and virtualized resources over the internet. Users typically engage with these services on a pay-per-use basis, ensuring that the services provided are reliable and designed to maximize the quality and utility of network storage and system resources. As a flexible service provider, the cloud operates on a consumption-based model. Many industry experts anticipate that cloud technology will continue to gain prominence within the IT sector. The cloud offers various service models, including Software-as-a-Service (SaaS), which allows users to access software through a single platform (e.g., salesforce.com); Platform-as-a-Service (PaaS), which provides a shared platform for software and data access (e.g., Azure Services, Amazon Web Services, Google App Engine); and Infrastructure-as-a-Service (IaaS), which focuses on security and backup support (e.g., Amazon EC2, VMware, IBM BlueMix). Cloud Computing finds applications across diverse fields such as scientific research, commercial enterprises, and education. Within the cloud ecosystem, job scheduling plays a crucial role in ensuring efficient workload distribution and resource management. The primary objective of scheduling algorithms in this environment is to optimize resource utilization effectively. In a cloud platform, processing user tasks necessitates the use of cloud resources[1]. Typically, in cloud computing, the resources needed for a job request are provided in the form of Virtual Machines (VMs). The scheduler in cloud computing is responsible for managing a specified number of job requests and allocating the necessary cloud resources (VMs) for each request. In this environment, a variety of resource types may be required to fulfil a job request. Scheduling involves the orderly sharing of available resources among the jobs. When a request is made, the corresponding resources are allocated as Virtual Machines (VMs). To address multiple requests, a sequence of various resources may be needed. The Scheduler organizes these resource types systematically to ensure optimal sharing among the job requests, facilitating effective resource utilization. Resources can include machine instances, data storage devices, applications, or environments. Proper allocation of these cloud resources to clients is a critical aspect of cloud computing. Therefore, an efficient scheduling model that is aware of deadlines is essential for managing the specified number of jobs.

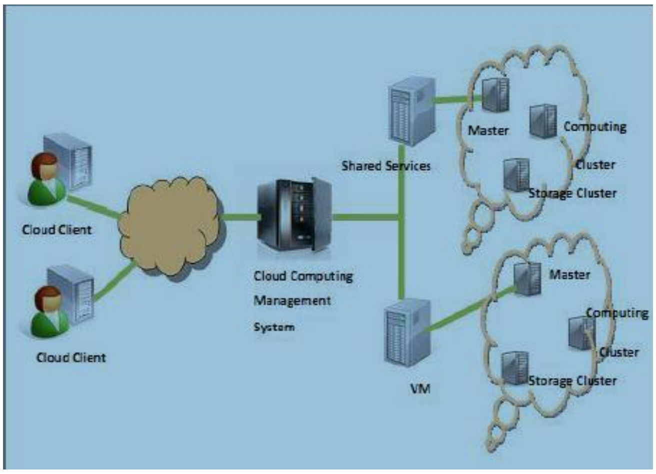

Figure 1: Cloud Computing Architecture

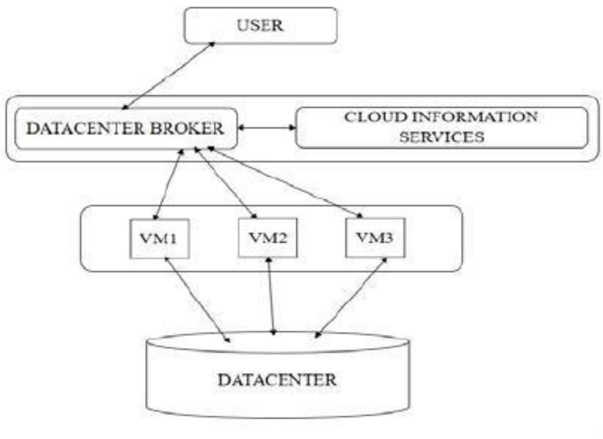

Figure 2: Scheduling Process in Cloud

In this paper, we present the design and development of a Nonlinear Analysis of EDF Scheduling model for resource allocation, focusing on performance metrics such as Average Turnaround Time, Average Waiting Time, and Average Deadline Violation.

2. Related Work

Greedy Based Job Scheduling Algorithm [2]: This algorithm emphasizes Quality of Service (QoS), recognizing cloud computing as a service driven by business needs. Its primary objective is to minimize completion time, thereby offering a more efficient solution to scheduling challenges. By categorizing tasks according to QoS, the algorithm invokes the relevant functions. When evaluated against other algorithms utilizing the Berger model and existing CloudSim scheduling strategies, it demonstrated superior performance.

Earliest Feasible Deadline First [4]: This algorithm seeks to enhance the efficiency of the Earliest Deadline First (EDF) approach by reducing its time complexity. The scheduling is primarily determined by deadlines, and the algorithm optimizes time complexity through process migration across different machines.

A Scheduling Algorithm based on Priority for VM Allocation [5]: This algorithm aims to deliver greater benefits to service vendors and providers. In scenarios where resources are insufficient to meet all requests, it introduces a priority-based approach to identify the most suitable allocations. This method significantly enhances resource utilization compared to the First-Come, First-Served (FCFS) strategy.

Improved Cost Based Algorithm [6]: This algorithm significantly enhances the efficiency of resource allocation for jobs compared to traditional cost-based scheduling methods. It organizes tasks based on the processing capabilities of the available resources.

A Priority Based Job Scheduling Algorithm [7]: In this scheduling framework, each job is assigned a priority value, guiding the allocation of resources accordingly. The author addresses challenges related to complexity, consistency, and makespan, asserting that performance can be enhanced by minimizing makespan.

Credit-Based Scheduling Algorithm [8]: In this algorithm, the author emphasizes two key scheduling parameters: user priority and task length. Credits are assigned based on these factors, leading to a job order that is determined by sorting tasks according to their credit values in descending order. Consequently, the scheduler prioritizes the job with the highest credit value for processing. This approach demonstrates improved efficiency compared to earlier scheduling methods.

Efficient Round Robin CPU Scheduling Algorithm [9]: The author introduces a novel approach that integrates quantum time with the Shortest Job First (SJF) algorithm. Jobs are organized in ascending order and can be preempted based on the defined quantum time. The results indicate that the average waiting time and turnaround time achieved through this Efficient Round Robin (ERR) method surpass those of traditional CPU scheduling algorithms. ERR ensures fairness in scheduling and is particularly effective in time-sharing environments, as each process receives an equal time slice, minimizing prolonged waiting for CPU access.

Priority-Based Earliest Deadline First Scheduling Algorithm [10]: This algorithm merges elements from both the Priority-Based Scheduling Algorithm and the Earliest Deadline First algorithm. Its primary focus is on optimizing memory utilization and resource allocation. By enhancing the completion times of preempted jobs, this algorithm significantly boosts scheduling efficiency. The

Communications on Applied Nonlinear Analysis
ISSN: 1074-133X
Vol 32 No. 1 (2025)

authors address the challenge of waiting times for preempted tasks by employing a waiting queue to manage these processes effectively[11].

3. Proposed Work: Nonlinear Analysis of EDF Scheduling Framework for Resource Distribution in Cloud Computing

Resource allocation in cloud computing typically involves assigning virtual machines to fulfil job requests. Given that users have specific requirements regarding response and waiting times, new EDF Scheduling model based on deadlines has been developed. In this model, the scheduler receives 'n' job requests from various users and allocates resources in the form of virtual machines to manage these requests[3][9]. The model recognizes that processing a job may necessitate 'm' different types of virtual machines in a sequential manner to meet the task's deadline while considering both response and waiting times. An analytical model has been created to evaluate average turnaround time, average waiting time, and deadline violations, comparing it with other scheduling strategies such as First Come First Serve (FCFS), Shortest Job First (SJF), and Earliest Deadline First scheduling models. The allocation of virtual machines (VMs) for each job request in the proposed model is illustrated in Figure 1. Let $req1_1$, $req2_2$, ..., req_n represent the set of job requests in cloud computing, where 'n' indicates the total number of requests and req_i signifies the i^{th} job request. Each job request req_i requires ti1 units of time on a VM of type-1, ti2 units on a VM of type-2, and so forth, up to t_{im} units of time on a VM of type-M to complete its task. Let dli denote the deadline for req_i to finish its task, with deadlines calculated using a new approach aimed at minimizing deadline violations.

$dl_i = (t_{i1}+t_{i2}+ \dots +t_{i4}) /4 + (\text{Median}(t_{i1}) * \alpha_1 + t_{i1} * \alpha_2) + (\text{Median}(t_{i2}) * \alpha_3 + t_{i2} * \alpha_4) + \dots\dots + (\text{Median}(t_{im}) * \alpha_{2m-1} + t_{i4} * \alpha_{2m})$ Where $0 < \alpha_i < 1$

For example, when m=4

$dl_i = (t_{i1}+t_{i2}+t_{i3}+t_{i4}) /4 + (\text{Median}(t_{i1}) * \alpha_1 + t_{i1} * \alpha_2) + (\text{Median}(t_{i2}) * \alpha_3 + t_{i2} * \alpha_4) + (\text{Median}(t_{i3}) * \alpha_5 + t_{i3} * \alpha_6) + (\text{Median}(t_{i4}) * \alpha_7 + t_{i4} * \alpha_8)$

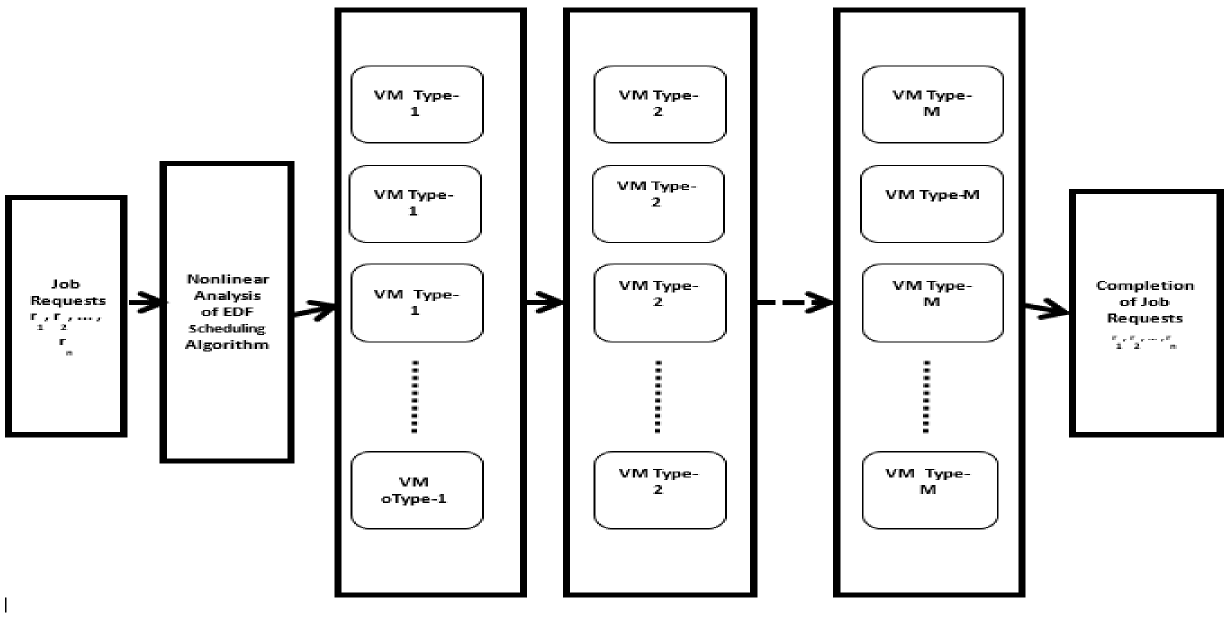

Figure 3. Nonlinear Analysis of EDF Scheduling Framework

Communications on Applied Nonlinear Analysis
ISSN: 1074-133X
Vol 32 No. 1 (2025)

The performance metrics can be determined through specific calculations based on a given scheduling sequence. The waiting time for request req_i is calculated as the difference between its completion time (ci) and the total processing time of the request (ti1 + ti2 + … + tim). The waiting time for req_i encompasses the duration taken to initiate its processing on VM of type-1, the interval between the completion of tasks on VM of type-1 and the commencement of tasks on VM of type-2, the time between finishing work on VM type-2 and starting on VM type-3, and continues in this manner up to the time elapsed between the completion of tasks on VM of type-(M-1) and the initiation of tasks on VM of type-M. We define s_i as the starting time of reqi on VM of type-1 and ci as the completion time of reqi on VM of type-M.

$w_i = c_i - (t_{i1} + t_{i2} + \ldots + t_{im})$

The difference between the task request's deadline (dl_i) and its actual turnaround time (c_i) is known as the deadline breach of the job request (dvt_i). i.e

$dvt_i = w_i - dl_i$ where $w_i > dl_i$

All job requests will have their average waiting time (AWT) and average turnaround time (ATT) calculated as follows.

$$AWT = \left(\sum_{i=1}^{n} w_i\right) / n$$

$$ATT = \left(\sum_{i=1}^{n} c_i\right) / n$$

Average Deadline Violation with respect to turn around time (ADVT) can be calculated as follows.

$$ADVT = \left(\sum_{i=1}^{n} dvt_i\right) / n$$

Algorithm: Nonlinear Analysis of EDF Scheduling Algorithm

Input : 'n' number of job requests with processing times $t_{i1}, t_{i2}, t_{i3} \ldots, t_{im}$ on
'm' types of Virtual Machines VM_1, VM_2, VM_3 and VM_m,
p number of instances are available for each Machine
$d1_i$ is deadline of i^{th} job request

Output : Optimal Scheduling sub sequences Seq_1, Seq_2, $Seq_3, \ldots Seq_p$

begin
  i=0;
solution_vector = empty;
  for k=1 to 2m do
α_k = choice(0.1,0.9);
  end for;
for each job request r_i with minimum deadline and processing time based on β and γ among all unprocessed jobs do
    add the job request r_i to the solution_vector at index i;

```
                i=i+1;
        end for;
        for i=0 to n-1 do
            j = i % p;
            supplement the scheduling subsequence Seqj with the solution vector [i]
        end for;
        for i=1 to p do
            calculate performance metrics for each scheduling sub sequence Seqi;
        end for;
      determine the overall scheduling sequence's aggregate performance metrics
    end;
```

The Nonlinear Analysis of EDF Scheduling Algorithm accepts ‘n’ job request and assign feasible values for α_i ($0 < \alpha_i < 1$) and β and γ for better performance metrics.

4. Performance Evaluation of the Framework

A specialized simulation environment has been created to evaluate the First Come First Serve (FCFS) Scheduling, Shortest Job First (SJF) Scheduling, and Earliest Deadline First Scheduling, utilizing ‘p’ Virtual Machine instances for each resource category. Job requests and their processing times are generated randomly using a Gaussian distribution. The process begins with the application of the FCFS scheduling algorithm, which organizes jobs based on their arrival order and divides the scheduling sequence into ‘p’ subsequence’s. Subsequently, the SJF Scheduling algorithm is implemented by arranging the jobs in ascending order of their total processing times ($t_{i1} + t_{i2} + t_{i3} + \ldots + t_{im}$) and further segmenting the scheduling sequence into sub-scheduling sequences. The Multi-Stage Scheduling algorithm is then applied without taking deadlines into account. Following this, the Simple Earliest Deadline First Scheduling algorithm is employed to produce an optimal scheduling sequence, which is also divided into ‘p’ scheduling sub sequences. Finally, the Nonlinear Analysis of Earliest Deadline First Scheduling algorithm is utilized to achieve an optimal scheduling sequence, again splitting it into ‘p’ scheduling sub sequences. This algorithm determines the optimal solution by assigning appropriate values to β and γ. Performance evaluation metrics are calculated at the conclusion of the process. In Table 1, we analyse n=32 jobs across m=4 types of Virtual Machines, with p=8 instances available. The parameters α1=0.2, α2=0.3, α3=0.4, α4=0.6, α5=0.2, α6=0.3, α7=0.2, and α8=0.3 are used to compute the deadlines for each job request (dli).

Table 1: Scheduling Instance n=32, m=4 and p=8

r_i	t_{i1}	t_{i2}	t_{i3}	t_{i4}	dl_i
0	124	457	77	5	687
1	402	22	104	78	525
2	322	429	35	56	777
3	150	130	158	64	500
4	405	123	202	72	663
5	31	441	33	64	631

6	172	352	160	94	719
7	204	80	75	61	440
8	248	410	181	87	818
9	448	344	142	74	843
10	90	330	0	64	551
11	160	140	180	34	510
12	330	52	18	23	434
13	237	454	236	94	883
14	71	444	41	2	625
15	204	296	18	66	595
16	322	241	30	18	594
17	328	41	272	19	561
18	277	386	210	33	799
19	297	117	177	38	566
20	376	71	115	14	523
21	54	102	201	1	413
22	162	335	71	63	633
23	54	90	178	19	400
24	180	473	195	26	808
25	232	72	210	39	511
26	267	176	178	27	595
27	496	312	148	58	837
28	20	112	171	2	387
29	51	481	66	22	671
30	36	148	238	34	481
31	376	276	122	48	720

Table 2: FCFS Scheduling for Instance n=32, m=4 and p=8

S_TYPE	**r_i**	**s_i**	**c_i**	**w_i**	**dl_i**	**dvt_i**
FCFS	0	0	663	0	687	0
FCFS	1	124	840	234	525	315
FCFS	2	526	1368	526	777	591
FCFS	3	848	1629	1127	500	1129
FCFS	4	0	802	0	663	139
FCFS	5	405	1066	497	631	435
FCFS	6	436	1575	797	719	856
FCFS	7	608	1636	1216	440	1196
FCFS	8	0	926	0	818	108
FCFS	9	248	1256	248	843	413

FCFS	10	696	1434	950	551	883
FCFS	11	786	1724	1210	510	1214
FCFS	12	0	423	0	434	0
FCFS	13	330	1351	330	883	468
FCFS	14	567	1508	950	625	883
FCFS	15	638	1845	1261	595	1250
FCFS	16	0	611	0	594	17
FCFS	17	322	982	322	561	421
FCFS	18	650	1556	650	799	757
FCFS	19	927	1738	1109	566	1172
FCFS	20	0	576	0	523	53
FCFS	21	376	764	406	413	351
FCFS	22	430	1061	430	633	428
FCFS	23	592	1214	873	400	814
FCFS	24	0	874	0	808	66
FCFS	25	180	1097	544	511	586
FCFS	26	412	1263	615	595	668
FCFS	27	679	1693	679	837	856
FCFS	28	0	305	0	387	0
FCFS	29	20	701	81	671	30
FCFS	30	71	1033	577	481	552
FCFS	31	107	1207	385	720	487

Table 3: SJF Scheduling for Instance n=32, m=4 and p=8

S_TYPE	**r_i**	**s_i**	**c_i**	**w_i**	**dl_i**	**dvt_i**
SJF	0	240	1075	412	687	388
SJF	1	90	696	90	525	171
SJF	2	583	1425	583	777	648
SJF	3	0	502	0	500	2
SJF	4	472	1274	472	663	611
SJF	5	204	822	253	631	191
SJF	6	492	1270	492	719	551
SJF	7	0	420	0	440	0
SJF	8	1034	1960	1034	818	1142
SJF	9	364	1553	545	843	710
SJF	10	0	484	0	551	0
SJF	11	20	534	20	510	24

SJF	12	0	423	0	434	0
SJF	13	877	1898	877	883	1015
SJF	14	54	643	85	625	18
SJF	15	36	620	36	595	25
SJF	16	150	761	150	594	167
SJF	17	706	1366	706	561	805
SJF	18	502	1530	624	799	731
SJF	19	286	915	286	566	349
SJF	20	330	906	330	523	383
SJF	21	0	358	0	413	0
SJF	22	125	1069	438	633	436
SJF	23	0	341	0	400	0
SJF	24	287	1629	755	808	821
SJF	25	54	607	54	511	96
SJF	26	235	1106	458	595	511
SJF	27	664	1678	664	837	841
SJF	28	0	305	0	387	0
SJF	29	180	889	269	671	218
SJF	30	0	456	0	481	0
SJF	31	231	1247	425	720	527

Table 4: Simple - EDF Scheduling for Instance n=32, m=4 and p=8

S_TYPE	r_i	s_i	c_i	w_i	dl_i	dvt_i
S-EDF	0	472	1252	589	687	565
S-EDF	1	54	660	54	525	135
S-EDF	2	501	1465	623	777	688
S-EDF	3	0	502	0	500	2
S-EDF	4	532	1334	532	663	671
S-EDF	5	456	1025	456	631	394
S-EDF	6	427	1209	431	719	490
S-EDF	7	0	420	0	440	0
S-EDF	8	937	1863	937	818	1045
S-EDF	9	596	1730	722	843	887
S-EDF	10	330	814	330	551	263
S-EDF	11	0	514	0	510	4
S-EDF	12	0	423	0	434	0
S-EDF	13	599	1739	718	883	856
S-EDF	14	430	988	430	625	363
S-EDF	15	252	836	252	595	241
S-EDF	16	150	761	150	594	167

S-EDF	17	204	864	204	561	303
S-EDF	18	487	1557	651	799	758
S-EDF	19	36	665	36	566	99
S-EDF	20	54	630	54	523	107
S-EDF	21	0	358	0	413	0
S-EDF	22	420	1219	588	633	586
S-EDF	23	0	341	0	400	0
S-EDF	24	582	1779	905	808	971
S-EDF	25	20	573	20	511	62
S-EDF	26	160	808	160	595	213
S-EDF	27	384	1449	435	837	612
S-EDF	28	0	305	0	387	0
S-EDF	29	333	1019	399	671	348
S-EDF	30	0	456	0	481	0
S-EDF	31	456	1278	456	720	558

Table 5: Nonlinear Analysis of EDF Scheduling for Instance n=32, m=4 and p=8 when β=0.1 and γ=0.9

S_TYPE	r_i	s_i	c_i	w_i	dl_i	dvt_i
N-EDF	0	240	1075	412	687	388
N-EDF	1	90	696	90	525	171
N-EDF	2	337	1359	517	777	582
N-EDF	3	0	502	0	500	2
N-EDF	4	472	1274	472	663	611
N-EDF	5	330	920	351	631	289
N-EDF	6	492	1270	492	719	551
N-EDF	7	0	420	0	440	0
N-EDF	8	689	1615	689	818	797
N-EDF	9	364	1553	545	843	710
N-EDF	10	0	484	0	551	0
N-EDF	11	20	534	20	510	24
N-EDF	12	0	423	0	434	0
N-EDF	13	877	1898	877	883	1015
N-EDF	14	54	643	85	625	18
N-EDF	15	36	620	36	595	25
N-EDF	16	150	761	150	594	167
N-EDF	17	361	1155	495	561	594
N-EDF	18	847	1753	847	799	954
N-EDF	19	180	809	180	566	243
N-EDF	20	204	780	204	523	257

N-EDF	21	0	358	0	413	0
N-EDF	22	125	1069	438	633	436
N-EDF	23	0	341	0	400	0
N-EDF	24	287	1629	755	808	821
N-EDF	25	54	607	54	511	96
N-EDF	26	580	1228	580	595	633
N-EDF	27	664	1678	664	837	841
N-EDF	28	0	305	0	387	0
N-EDF	29	286	927	307	671	256
N-EDF	30	0	456	0	481	0
N-EDF	31	477	1299	477	720	579

Table 6: Nonlinear - EDF Scheduling for Instance n=32, m=4 and p=8 when β=0.2 and γ=0.8

S_TYPE	**r_i**	**s_i**	**c_i**	**w_i**	**dl_i**	**dvt_i**
N-EDF	0	240	1075	412	687	388
N-EDF	1	90	696	90	525	171
N-EDF	2	337	1359	517	777	582
N-EDF	3	0	502	0	500	2
N-EDF	4	472	1274	472	663	611
N-EDF	5	330	920	351	631	289
N-EDF	6	492	1270	492	719	551
N-EDF	7	0	420	0	440	0
N-EDF	8	689	1615	689	818	797
N-EDF	9	364	1553	545	843	710
N-EDF	10	0	484	0	551	0
N-EDF	11	20	534	20	510	24
N-EDF	12	0	423	0	434	0
N-EDF	13	877	1898	877	883	1015
N-EDF	14	204	771	213	625	146
N-EDF	15	36	620	36	595	25
N-EDF	16	150	761	150	594	167
N-EDF	17	361	1155	495	561	594
N-EDF	18	542	1533	627	799	734
N-EDF	19	180	809	180	566	243
N-EDF	20	54	630	54	523	107
N-EDF	21	0	358	0	413	0
N-EDF	22	430	1061	430	633	428
N-EDF	23	0	341	0	400	0
N-EDF	24	592	1621	747	808	813
N-EDF	25	54	607	54	511	96

N-EDF	26	275	1109	461	595	514
N-EDF	27	664	1678	664	837	841
N-EDF	28	0	305	0	387	0
N-EDF	29	286	927	307	671	256
N-EDF	30	0	456	0	481	0
N-EDF	31	477	1299	477	720	579

The metrics for First Come First Serve (FCFS) scheduling pertaining to the specified problem instance are presented in Table 2. In a similar manner, Table 3 illustrates the scheduling metrics for Shortest Job First (SJF) scheduling, while Table 4 provides the scheduling metrics for the Simple Earliest Deadline First scheduling models. The metrics for Enhanced-EDF Scheduling are detailed in Table 5, where Beta is set to 0.1 and Gamma to 0.9 in the job selection order. Additionally, Table 6 displays the Enhanced-EDF Scheduling metrics with Beta at 0.2 and Gamma at 0.8 in the job selection order.

Table 7: Comparison of Scheduling Metrics when n=32, m=4 and p=8

S_TYPE	**Average Turn Around Time (ATR)**	**Average Waiting Time (AWT)**	**Average Deadline Violation (ADVT)**
FCFS	1148	500.5	536
SJF	961	314.3	355.6
Simple-EDF	964	316.6	355.8
Nonlinear-EDF (**β=0.1 and γ=0.9**)	951	304.3	345.6
Nonlinear-EDF (**β=0.2 and γ=0.8**)	940	292.5	334

The comparative analysis of scheduling metrics, namely ATR, AWT, and ADVT, across four scheduling models is presented in Table 7. In the FCFS Scheduling model, the ATR is recorded at 1148, whereas it is 961 in the SJF Scheduling model, 964 in the Simple EDF Scheduling model, 951 in the Nonlinear-EDF with parameters (β=0.1 and γ=0.9), and 940 in the Enhanced-EDF with parameters (β=0.2 and γ=0.8). Regarding the AWT, the FCFS Scheduling model shows a value of 500.5, in contrast to 314.3 in the SJF Scheduling model, 316.6 in the Simple EDF Scheduling model, 304.3 in the Nonlinear-EDF with (β=0.1 and γ=0.9), and 292 in the Enhanced-EDF with (β=0.2 and γ=0.8). Similarly, the ADVT in the FCFS Scheduling model is 536, while it is 355.6 in the SJF Scheduling model, 355.8 in the Simple EDF Scheduling model, 345.6 in the Nonlinear-EDF with (β=0.1 and γ=0.9), and 334 in the Nonlinear-EDF with (β=0.2 and γ=0.8). The graphical depiction of these performance metrics is illustrated in Figure 4.

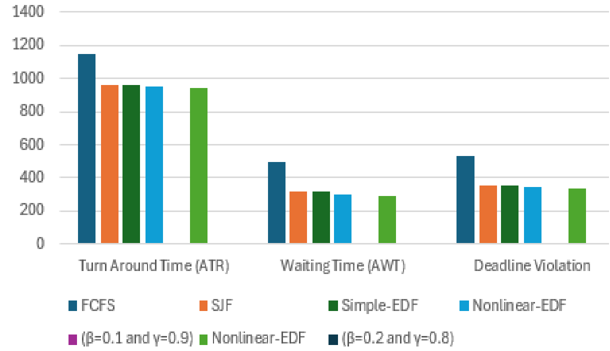

Figure 4: Comparison of Scheduling Metrics when n=32, m=4 and p=8

5. Conclusion

The Nonlinear Analysis of EDF Scheduling Framework for Resource Distribution in Cloud Computing optimizes job scheduling for resource allocation by taking into the account both deadlines and processing times. Experimental results indicate that the proposed EDF Model outperforms traditional scheduling models such as FCFS, SJF, and Simple EDF in terms of key performance metrics, including Average Turnaround Time, Average Waiting Time (AWT), and Average Deadline Violation with respect to Turnaround Time (ADVT).

References

[1] Barrie Sosinsky, “Defining Cloud Computing” in Cloud Computing Bible, Indianapolis, Indiana: Wiley Publishing, Inc. 2011.

[2] Li, Ji, LonghuaFeng, and Shenglong Fang. "A greedy-based job scheduling algorithm in cloud computing." Journal of Software 9.4 (2014): 921-925.

[3] N. Neelima Priyanka, P Suresh Varma, R Krishnam Raju, B. Sukumar Babu “Earliest Deadline First Scheduling in Cloud Computing” Jour of Adv Research in Dynamical & Control Systems, 11-Special Issue, July 2017.

[4] Barrie Sosinsky, “Defining Cloud Computing” in Cloud Computing Bible, Indianapolis, Indiana: Wiley Publishing, Inc. 2011.

[5] Sukumar Babu Bandarupalli, Neelima Priyanka Nutulapati, Prof P Suresh Varma, “A Novel CPU Scheduling Algorithm –Preemptive & Non-Preemptive” (IJMER) International Journal of Modern Engineering Research, Vol 2, Issue 6, Nov-Dec 2012.

[6] Jagbeer Singh, Bichitrananda Patra, Satyendra Prasad Singh, " An Algorithm to Reduce the Time Complexity of Earliest Deadline First Scheduling Algorithm in Real-Time System" (IJACSA) International Journal of Advanced Computer Science and Applications, February 2011.

[7] Xiao, Jing, and Zhiyuan Wang. "A Priority Based Scheduling Strategy for Virtual Machine Allocations in Cloud Computing Environment." Cloud and Service Computing (CSC), 2012 International Conference on. IEEE, 2012.

[8] Selvarani, S., and G. Sudha Sadhasivam. "Improved Cost-Based Algorithm for Task Scheduling in Cloud Computing." Computational Intelligence and Computing Research (ICCIC), 2010 IEEE International Conference on. IEEE, 2010.

[9] Ghanbari, Shamsollah, and Mohamed Othman. "A priority-based job scheduling algorithm in cloud computing." Procedia Engineering 50 (2012): 778-785.

[10] Thomas, Antony, G. Krishnalal, and VP Jagathy Raj. "Credit Based Scheduling Algorithm in Cloud Computing Environment." Procedia Computer Science 46 (2015): 913-920.

[11] Sukumar Babu B, Neelima Priyanka N, Sunil Kumar B. “Efficient Round Robin CPU Scheduling Algorithm”. International Journal of Engineering Research and Development (IJERD), Volume 4, Issue 9, November 2012, PP 36-42.