I J C T A, 9(7), 2016, pp. 3327-3339 ISSN: 0974-5572

Mixed Criticality Based Modified Preemptive PP-aware Scheduling Algorithm for Time Dependent Online Service Providers

Madhuri Bedase* and D. A. Phalke**

ABSTRACT

Various real time online services are growing day to day in computer and internet technology. Different forms of scheduling algorithms are available, for scheduling such real time applications. Scheduling is the time dependent strategy used by the system, to decide the allocation of available resources among accepted tasks to complete their execution. Throughput, turnaround time and fairness are the challenging characteristics of particular scheduler. But in practice, to achieve all these characteristics by single scheduler, is most difficult and NP-hard problem. For real time online service providers, scheduling plays a very important role, as they are time dependent applications. The main goal of these kind of application is, to complete the processing of as many as requests before deadline as early as possible and earns profit and reduce the penalty. Penalty incurs, if execution reaches to deadline or worst case execution time. To achieve this goal, in this paper, system implements the modified version of a profit-penalty aware scheduling (Modified PP-aware) for real time online service providers. The system allows users to select the high, medium and low criticality according to their urgency of receiving service from service providers. High criticality service request with expected gain processing first and then medium and low priority requests, respectively. This increases the user satisfaction. During the analysis of scheduler, System divide the interval range of deadline (D), best case (B) and worst case (W) execution time based on task size or length. The small size task require small [B, W] interval and large size task require large [B, W] interval. Because of this dividing of intervals, a particular amount of time span is allocated to particular size tasks So that system can complete as many as tasks within deadline and thereby improve the accrued profit and reducing the accrued penalty of the service provider. Our experimental results prove that, the Modified PP-aware scheduling performs better than the existing PP-aware scheduling in terms of accrued profit.

Keywords: Time dependent online services, PP-aware scheduling, Mixed criticality, Best case execution, Worst case execution, Deadline.

1. INTRODUCTION

A real-time operating system (RTOS) supporting to various real time applications, which receives the data as input user requests and in returns processing all requests to provide the services. The RTOS is applicable in work flow systems, traffic control systems, military applications and various online service providers such as online travel planning service provider. The main challenge is to process all requests withoutany buffering delay. This processing time is measured in milliseconds or shorter than milliseconds. The output of such applications depends on the consistency of amount of time taken for accepting and completing the user's tasks. There are three types of RTOS: Hard, Soft and firm real time systems. The RTOS that might cross tasks deadline is termed as soft real time OS. In some RTOS, deadline violation is strictly avoided known as hard real time OS. Because failure leads to total collapse of application. In firm RTOS, missing of deadline has no value.

* Department of Computer Engineering D. Y. Patil College of Engineering, Akurdi Pune-411035, India, *Email: madhuri.r.bedase@gmail.com*

** Department of Computer Engineering D. Y. Patil College of Engineering, Akurdi Pune-411035, India, *Email: a_dhanashree@yahoo.com*

In RTOS, various scheduling algorithms are available, such as, cooperative scheduling, preemptive scheduling, Earliest Deadline First (EDF) approach, etc. This scheduling is useful for real time online service applications. Real time online service uses real time systems to supply timely quality services to users. Scheduling is that, the strategy by which system decides resource allocation among numerous tasks. In simple words, scheduling decides, to which tasks, the processor should be allocated to complete its execution at a specific given time. Real time online service providers use this concept of scheduling. For real time online services providers, a Service Level Agreement (SLA) is signed between service providers and multiple clients. According to this SLA, service provider have to complete the processing of user requests with quality of service and in returns user has to pay a specific amount for the timely services it gets. The concept of time dependent application is depicted in Fig. 1 According to this figure, if the task executed before deadline system gets profit otherwise the penalty is applied to the system. Hence the process of scheduling is a very important strategy in such real time online service provider applications. The main aim of such scheduler is to complete all accepted tasks before the deadline, provide quality services and maximize the profit of online service providers.

Fig. 1 shows, profit function that is gained, $G(t)$ and penalty function that is loss, $L(t)$, where the deadline is 10. If task will be completed or aborted at time 10 then a service provider gains 5$ or 7.5$, respectively. If the task completed or aborted at 2 then service provider earns 9$ or 1.5$, respectively.

This paper introduce a modified version of preemptive Profit-penalty aware scheduling (Preemptive PP-aware) using a mixed criticality approach and multiple range of best and worst case execution time and deadline (D, [B, W]). The main goal of the propose system, is to maximize total accrued profit and minimize the penalty by completing most of the accepted tasks before the deadline. Existing PP-aware algorithm, performs task admission test and preemptive scheduling of all accepted tasks to maximize the profit. The three level mixed criticality and (D, [B, W]) interval partitioning schemes have proposed, which will discussed in section III.

The remaining paper includes section II, which shows the survey of related work. Section III gives the implementation details of system including architecture, problem formulation, and overview of system, mathematical model and proposed algorithm. The simulation settings, dataset and comparative results of the system is covered in Section IV. Finally this system is concluded in section V.

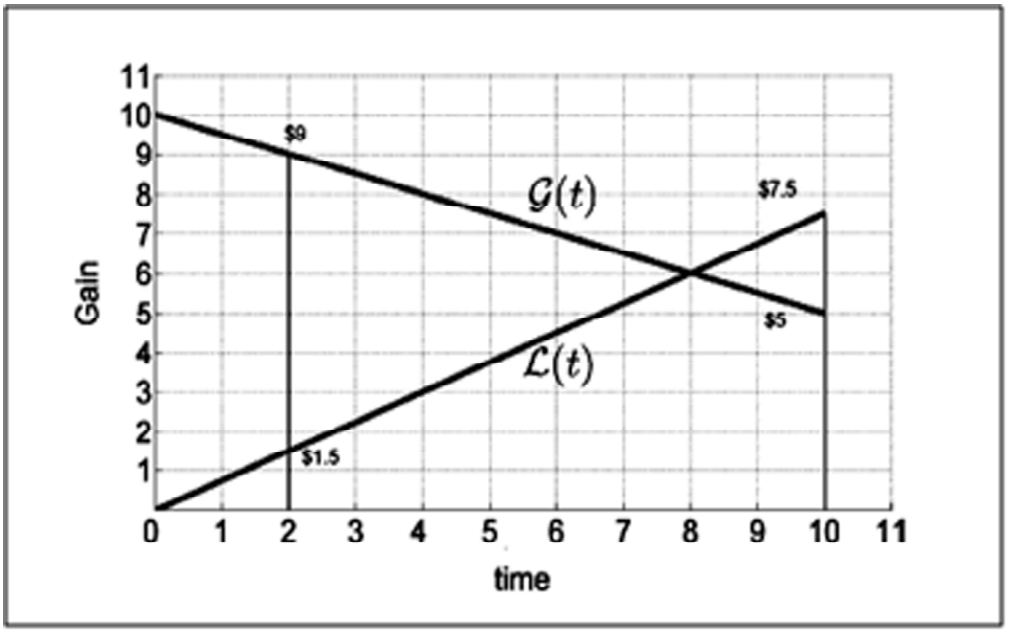

Figure 1: Gain and loss function of tasks that depend on time [15]

2. LITERATURE SURVEY

Scheduling is a very challenging and important task in real time applications. There are various types of scheduling algorithms, which are studied in [1]. These algorithms are mainly classified according to their multiprocessor or uniprocessor environment.

Most of the researchers have developed the algorithms, which are based on Earliest Deadline First (EDF) approach [2], [3], [4], [5], [6], [8], [9]. EDF is the commonly used scheduling algorithm, where the priority of the task depends on deadline.

Some of the algorithms are based on Genetic algorithm (GA), which are implemented in [10], [11]. There are other scheduling approaches are available like, Rate Monotonic (RM) [12] and Deadline Monotonic (DM) [13]. Some algorithms have developed, which are any combination of EDF, GA, RM [14], [15].

An efficient Scheduling algorithmcalled as Myopic algorithm is developed in [16] for real time multiprocessor systems. This algorithm is extended in [17]. Advantages includes, a huge number of tasks can meet the deadline, high schedulability and low overhead.

Resistance to Overload by Using Slack Time (ROBUST) algorithm is proposed, to improve the overload performance where the tightness of the task deadline is limited. According to the survey made by authors, there is no any scheduling algorithm for overloaded online systems in uniprocessor environment which having a processor utilization more than 0.25, but this paper achieves the processor utilization up to 1.0 in overloaded systems [18].

List based Particle Swarm Optimization (LPSO) algorithm is proposed in [10]. This is the novel scheduling algorithm applicable for processing real time tasks in multiprocessor environment. It is the extension to Particle Swarm Optimization (PSO) algorithm. This algorithm makes use of list scheduling technique. This algorithm processes the tasks according to its height and particles position and then to produce diverse priority value, apply PSO. It means that the scheduling criteria is based on height and particles position of task. Finally feasible solution is obtained by using list scheduling technique.

In scheduling, mainly tasks are either preemptive or non-preemptive, but the author of [19] identifying a new type of set of tasksknown as urgent priority tasks. This task set is non trivial, practical, asynchronous and preemptive and solved in polynomial time. The task is preemptive if and only if it is executed at the time of its ready state. The task is in waiting state only for one unit of time just after it is ready.

Reduction to Uniprocessor (RUN) algorithm is developed in [20]. This algorithm generates an efficient schedule of tasks. It reduces the load of multiprocessor system by converting it to the uniprocessor problem.

The real time computing systems are badly affected by the issue of preemption delay related to cache. Authors of [21] and [22], studied and proposing some advances, to minimize cache related preemption delay.

The preemptive and non-preemptive version of Profit and penalty aware scheduling algorithms are proposed in [15], applicable for all kinds of online service providers. These systems are mainly concerned with gain and loss obtained by completing the tasks before the deadline. All tasks are scheduled on the basis of its expected gain. This system has two important components: task admission test and risk factor. This PP-aware model was first designed in [23].

The optimal priority free real time task scheduling algorithm is proposed in [32]. This model is based on Discrete Event System (DES) with Supervisory Control Theory (SCT).

In recent most of the scheduling algorithms have implemented. These scheduling algorithms are used to manage all tasks of cloud service providers within deadline and increase the profit. Resource utilization is the most challenging task of such schedulers. Shin et al proposed an EDF based and Largest Weight First

(LWF) based scheduler to guarantee the execution of deadline and minimum resource utilization in cloud [29]. GlobeAny online scheduling algorithm is proposed to increase the profit of cloud service provider in [30].

Recent scheduling technique removes the assumption that scheduling analysis has been carried with the same level of criticality or importance. The system which executed on a platform which allows mixed criticality levels are developed in [24], [25], [26], and [31].

The problem of partitioning the resources among periodically arriving online tasks in a uniprocessor or multiprocessor environments is addressed in [27], [28]. The main objective is to derive the feasible partitioning of available resources to achieve the minimum energy consumption without violating time requirements.

3. IMPLEMENTATION DETAILS

3.1. System Architecture

In this paper, real time on-line service application is proposed that is the website for any distributor. System perform the on-line scheduling operation on website to maximize the systems profit and minimize the penalty. This system has two main parts: user interface and administrator as shown in Fig. 2.

The user interface is consists of registration and login process. When users visiting the system first time, he must be registered to the system with a valid name and password. After completing successful registration, user or client can use the services provided by the application. After user login, user can select the criticality level and upload the files on the server, if the resources are available. This uploaded file is

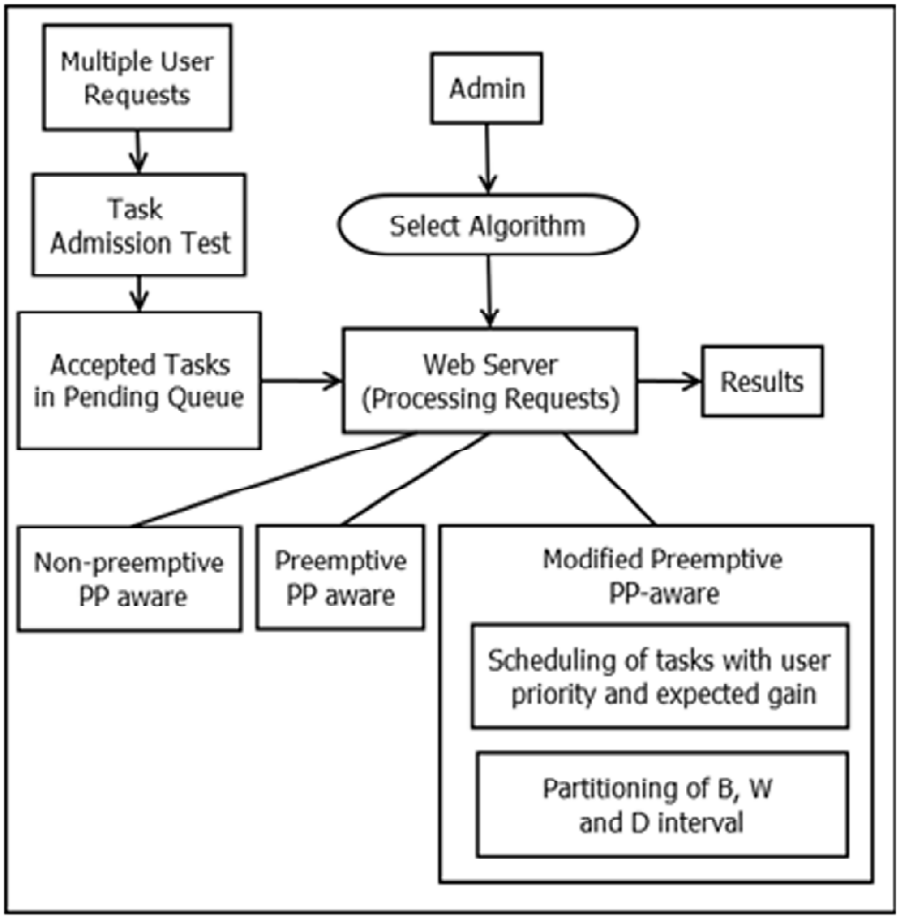

Figure 2: Architecture of Client Server Application for Online File Upload

considered as a task or user request to the system. That is input of system is user request from different users in the form of text or image file. Now, administrator controls the system. Administrator part consists of login process and select the proper algorithm to schedule all user requests, calculating the systems profit and penalty and displaying the results in terms of graphs. The administrator selects any one of the three algorithms to schedule all incoming requests to the server. These three algorithms are such as, preemptive PP-aware, non-preemptive PP-aware and modified preemptive PP-aware algorithm for scheduling. By using any one of these three algorithms, server managing and completing all incoming user requests within the specified deadline. The main aim of these scheduling algorithms is to complete all tasks within deadline, thereby increasing the systems profit and reducing the penalty. Fig. 2 depicts the architectural view of the proposed system.

3.2. Problem Formulation

For given set of aperiodic task set, develop an online application that schedules all tasks preemptively using modified profit and penalty aware scheduling algorithm. Users are allowed to select their criticality level of the task according to urgency of completing tasks. Proposed system should execute maximum number of tasks and should be able to maximize total accrued profit of the system and to minimize total accrued penalty of the system.

3.3. System Overview

Three Level Mixed Criticality: From recent years, various ideas for analysis of schedulers, has been introduced by researchers. One of the interesting idea is that, consider different level of importance or criticalityof tasks for analysis of scheduler. The proposed system is design in such a way that, it allows mixed criticality tasks that shares the same platform for execution. The system sets the three levels of criticality (C). In our case, criticality is selected by users. User can select High Criticality (C_H), Medium Criticality (C_M) and Low Criticality (C_l) for uploading their files on server. High criticality requests along with highest expected gain, have highest priority to be uploaded on server. Same as, medium and low criticality tasks having second and third priority for processing.

Task Admission Test: The system incur maximum penalty once the task is accepted and later aborted. If the execution time is increases profit will decreases because the system is time dependent system. So it is very important task to take the proper decision about, whether to accept or reject the task. Therefore the task admission test is use to reject or abort the tasks earlier but not pessimistically based on inccured profit or penalty. Fig. 3 represents the process of task admission test. In this, when new tasks arrives to system parameters are initialized randomly within interval such as, release time (R), deadline (D), worst case (W) and best case (B) execution time. From this parameter calculate expected gain, loss function and probability of task that the task cannot be completedbefore deadline or at deadline. These three terms give quantity to show that how much amount of gain or loss will be obtained in future if the task is complete or aborted in future, before the actual execution starts. That is, these terms are calculated instantly, when task arrive to the system or during the execution. Using these three terms risk factors of every task is calculated. There is also a maximal risk factor P_{max} of the system, which is considered as a threshold value. This is the systems tolerable risk level. If the tasks risk factor is greater than systems maximal risk factor that is ($P(t, t0) > P_{max}$), then this task is aborted during the execution or not accepted by system for further execution. Because of this, the systemaccepts only those tasks which have probability to finish before deadline. Because of this system increases their profit. This accepted tasks are now ready to run and stored into pending queue. The administrator of this system schedules these tasks using any one scheduling algorithms, that are either preemptively or non-preemptively.

1) *Preemptive Scheduling*: All accepted tasks are stored in pending queue. In this queue, accepted tasks are sorted according to criticality of task and their associated expected gain. In preemptive scheduling,

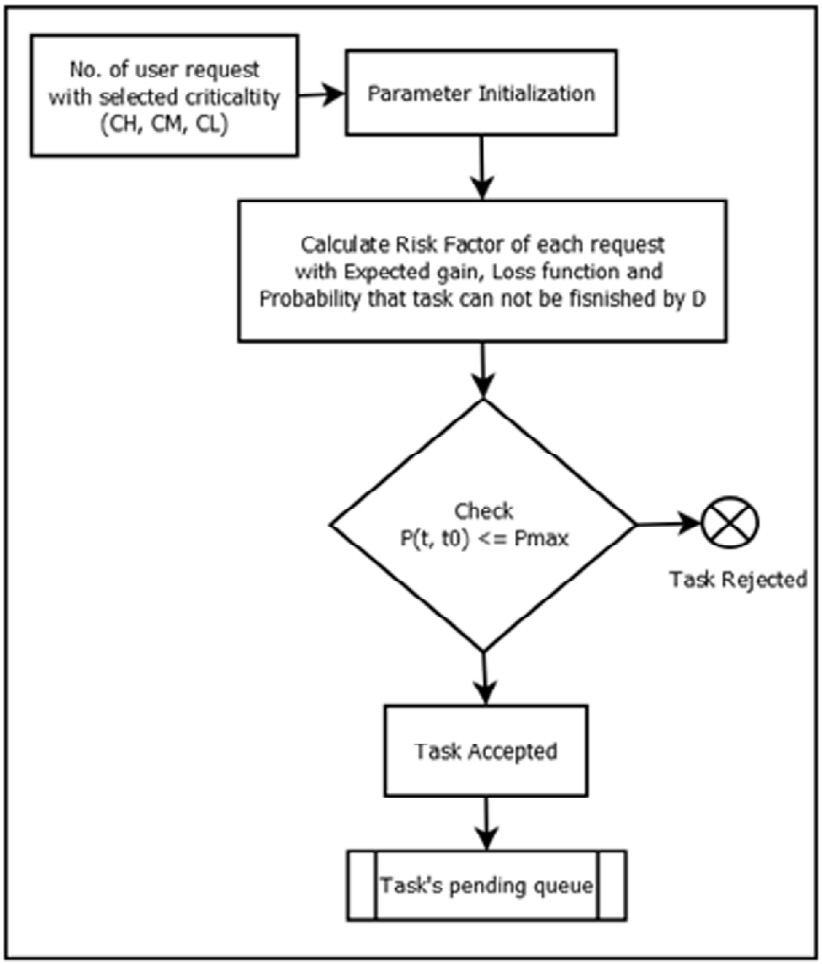

Figure 3: Flow of Task Admission Test

preemption is allowed, that is the task with high criticality and expected gain can preempt the current executing task. The remaining preempted task is treated store in queue as T_{remain}. The scheduling decisions are made at following point:

- Upon arrival of new tasks
- The time at which any task completes its execution before deadline
- When executing tasks reaches to deadline or exceeds the risk of processing.

2) *D, B, W interval partitioning*:Existing system make some simulation setting to measure the performance of algorithm. The B, W and D are randomly generated and assign to each task. These values are randomly distributed within [1, 10], [30, 50], [10, 70]. In existing system, this same time interval is used for all tasks for completing the execution. But sometimes, smaller task gets large time span and larger tasks gets smaller time span to complete their execution. Because of this, larger tasks cannot be finished within this time interval, which leads to penalty. To overcome this drawback, our system implements the new concept, in which B, W and D intervals are divided into multiple range interval. And according to the task size, B, W and D is assign fromappropriate interval. It improves the resource utilization and completing many task's execution, thereby system increase the profit of system.

3.4. Mathematical Model

1. Let S be a system. This system is a web application. This system is applicable for any kind of real time online service providers. In this system, user can upload the text file on server, if resources are available.

This text file is input to the system. That is, the user requesting to server for uploading file on it. Here, each user request is treated as the task with Variable Execution Time and Known Probability (VEP task). System having such multiple VEP task for uploading text files on server. Now, the main aim of the server is to complete all accepted tasks within specified deadline for increasing the profit and reducing the penalty. Finally, output is calculated in terms of total accrued profit and total accrued penalty.

S = {A, AL, U, C, T, O, P}

2. Administrator.

   S = {A …}

   Administrator works at server side. Admin can select the proper algorithm for scheduling all VEP tasks and display results in terms of profit and penalty of system.

3. Set of algorithms

   S = {A, AL …}

   AL = {AL1, AL2, AL3}

   AL1 = Non-preemptive PP-aware scheduler

   AL2 = Preemptive PP-aware scheduler

   AL3 = Modified preemptive PP-aware scheduler

4. Set of Users

   S = {A, AL, U …}

   U = {u1, u2, …, un}

   U is the set of Users. These clients should be registered at server and then, they can upload files on server.

5. Mixed criticality

   S = {A, Al, U, C, …}

   C = $\{C_H, C_M, C_L\}$

   C is the set three criticality, by which user can select their tasks processing urgency.

   C_H = High criticality

   C_M = Medium Criticality

   C_L = Low Criticality

6. VEP task set, that is user requests

   S = {A, AL, U, C, T, …}

   T = { }

   T is set of VEP tasks, where every task having six parameters such a that,

   τ_i = $(Ci, Ri, [Bi, Wi], fi(C), Gi(t), Li(t), Di)$

   Where,

   C = any one of criticality selected by user.

   R = Release time of task

B = Best case execution time allocated for task

W = Worst case execution time allocated for task

D = Tasks deadline

$f(C)$ = Variable task execution time with probability density C function

$$f(t) = \frac{1}{(W - B)} \tag{1}$$

$G(t)$ = Completion TUF, represents the obtained gain of task which is completed at t time,

$$G(t) = -ag\ (t - D),\ R \le t \le D \tag{2}$$

ag = Gradient of G, randomly generated in the range [4, 10]

$L(t)$ = Abort TUF, represents the obtained loss of task which is aborted at t time,

$$L(t) = al\ (t - R),\ R \le t \le D \tag{3}$$

al = Gradient of L, randomly generated in the range [1; 5]

7. Output

S = {A, Al, U, C, T, O, …}

O = {O1, O2}

O1 = Total accrued profit,

O2 = Total accrued penalty

8. Process of scheduling all VEP tasks

S = {A, Al, U, C, T, O, P, …}

Pr = {Pr1, Pr2, Pr3, Pr4, Pr5}

Pr1 = Task admission test

Pr1 = {e1, e2, e3, e4}

This test is use to take decision regarding to execution of task, aborting a task, and when to abort a task. This test includes following steps.

e1 = Set maximal risk factor P_{max} of system, which is tolerable risk level of system.

e2 = Calculate Risk Factor (RF) of task when it arrives and at every t instant.

$$RF = \frac{L(t) * P(T > D)}{E(G(T))} \tag{4}$$

Where,

E (G(T)) = Tasks expected gain

$$E\left(G(T)\right) = \int_{ts}^{\infty} G(t).I_{[R,D]}(t).f(t).I_{[ts+B,ts+w]}(t).dt \tag{5}$$

$f(t)$ = Variable execution time of task with probability density function

$$f(t) = \left(\frac{1}{B}\right).I_{[t0+B,t0+w]}(t) \tag{6}$$

$I_{a,b}(x)$ = Indicator function

$$I_{a,b}(x) = 1,\ if\ (if\ \ x \in [a,b])\ \text{otherwise}\ 0 \tag{7}$$

$G(t)$ = *Completion TUF*

From equation (2)

$L(t)$ = Penalty function

From equation (3)

P (T > D) = Probability that task cannot be completed before D.

$$P(T > D) = 1 - P(t \le D)$$
$$= 1 - \int_0^D f(t)dt \tag{10}$$

e3 = Compare P_{max} of system and RF of task, if RF is greater than P_{max}, then task is not accepted or task is aborted during execution, otherwise task is accepted for further execution.

e4 = All accepted tasks are stores into pending queue and treated as ready to run tasks.

Pr2 = System made scheduling decisions at

- Arrival of new task;
- Successful completion of task execution before deadline;
- When task execution meets previously calculated preemption point, critical time or deadline.

Critical time is the time at which RF is greater than P_{max}. At this critical time the task is forcefully aborted even if the small amount of execution is remaining.

$$t_{critical} = \inf \{t + t0 : p(t, t0) > P_{max}\} \tag{11}$$

Pr3 = Task selection strategy. Highest expected gain with highest criticality task is selected for execution from pending queue. This task is executed till execution reaches to next scheduling point.

Pr4 = Apply any one of the three algorithm for scheduling all accepted VEP tasks.

Pr5 = Display output in terms of profit and penalty.

$$Total\ accrued\ profit,\ G(\Gamma) = \sum_{i=1}^{n} G(\tau_i) \tag{12}$$

$$Total\ accrued\ penalty,\ L(\Gamma) = \sum_{i=1}^{n} L(\tau_i) \tag{13}$$

$$Total\ Utility,\ U(\Gamma) = G(\Gamma) + (-G(\Gamma)) \tag{14}$$

3.5. Algorithms

3.5.1. Task Admission Test

Input: User requests with priority (High, Medium, Low) from multiple users.

Process:

1. For all requests Initialization of parameters
2. Apply Task Admission Test
3. for each request

4. Calculate Expected gain, Loss function and Probability of task shows that it cannot be completed within or at deadline.
5. Calculate risk factor $P(t, t0)$.
6. Check, $P(t, t0) \leq P_{max}$
7. If true, then task added to pending queue
8. Else task is rejected before added to system or aborted during execution.

Output:

Pending queue with sorted tasks according to user assign priority and expected gain.

3.5.2. Modified Preemptive PP-aware

Input: Sorted Tasks in Pending queue according to user priority and Expected gain.

Process:

1. Select task with high priority and highest expected gain from queue for execution
2. Calculate scheduling point t_s
3. Calculate Critical point $t_{critical}$
4. If true Continue tasks execution
5. Remove Request if following condition satisfy
6. If deadline cross or Risk factor cross
7. Execute currant task if any task is not complete or other task not arrive in same time
8. On arriving new task, check user assigned priority and gain
9. If high, Preempt current executing task
10. And remaining task store in pending queue as T_{remain}
11. Process new request as currant request
12. If (currant task complete)
13. Process task which is in waiting state

Output:

Profit = Summation of gain obtained by all tasks completed before deadline.

Penalty = Summation of loss obtained by all tasks, that crosses deadline.

4. RESULTS AND DISSCUSION

This section describes the evaluation, analysis and comparison of the modified preemptive PP-aware with preemptive and non-preemptive PP-aware algorithm in terms of system's total accrued profit.

4.1. Experimental Setup

Experimental setup includes, one server and multiple clients. To evaluate the performance of system, user randomly upload the 100 files on server. System considers tolerable risk level 3. Each file upload request is considers as VEP task.

Each task is describe as follows:

$$\tau_i = (Ci, Ri, [Bi, Wi], fi(C), Gi(t), Li(t), Di)$$

Criticality level is selected by user at the time of upload request. C is either high, medium or low critical. When the new task is arrive, B, W and D are allocated to task randomly from following interval. This allocation is depend on the file size. For example, as shown in TABLE 1, for file size ranging from 1kb to 10 mb, B, W, D are randomly generated within [1, 4], [30, 40], [10, 25] interval.

4.2. Dataset

To test the system, randomly generate 100 tasks. That is upload 100 files on server and check the performance. Format of file includes, text file, image file or any multimedia file. System apply the limit on file size. User cannot upload the file, which is greater than 100 mb.

4.3. Result Set

In this paper, the implementation of Modified Preemptive PP-aware algorithm have explained. The performance of Modified Preemptive PP-aware algorithm is compared with existing preemptive PP-aware

Table 1
B, W, D Interval Partitioning

File Size	*B*	*W*	*D*
1kb to 10mb	[1,4]	[30, 40]	[10, 25]
10mb to 50 mb	[3, 6]	[35, 45]	[25, 40]
50 mb to 100 mb	[6,10]	[40, 50]	[40, 70]

Table 2
Perfromance Comparision

Algorithm Type	*Profit (%)*	*Penalty (%)*
Modified preemptive PP-aware	88	35
Preemptive PP-aware	83	50

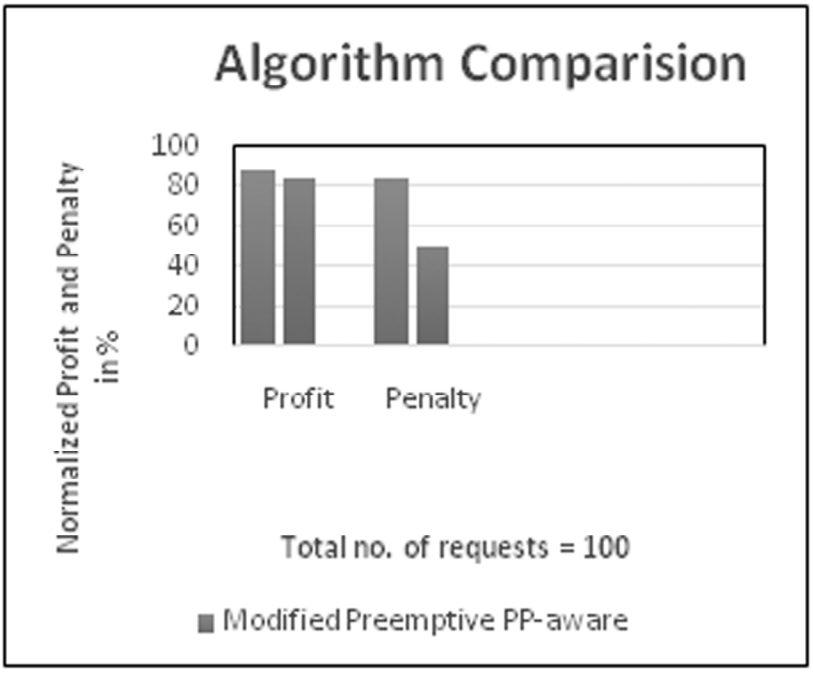

Figure 4: Comparing proposed algorithm with existing algorithm

algorithm. The comparative results are as shown in Fig. 4. If the system scheduling all tasks using Preemptive PP-aware scheduling algorithm, then Profit is 83 % and 50 % percent. But our proposed work increasing the systems profit up to 88 %. This system is better in terms of maximize profit and increasing execution speed also executing maximum number of tasks.

5. CONCLUSION AND FUTURE SCOPE

This paper presents the implementation of modified version of preemptive profit-penalty aware scheduling algorithm with mixed criticality and a dividing interval of (D, [B, W]) for analysis of system. The proposed system is applicable for any kind of real time on-line time dependent service providers, to increase the systems profit and minimize the penalty by rejecting the high risk tasks and completing as many tasks as possible before deadline. The mixed criticality approach allows users to select the urgency of processing their task. So that the user satisfaction is increasing. Also, because of this criticality, system easily identify the basic priority of tasks. Division of intervals makes a fair decision for allocating the time span for executing the tasks. So that the maximum number of tasks are completed before deadline, thereby system increases the accrued profit.

This system will be enhanced by the identification of optimal for different system with variable task load. It is a very difficult task to get the optimal threshold value of risk factor of the system. In future, this system can be implemented for real time online service providers applications and can also be applied in cloud computing.

ACKNOWLEDGMENT

The authors are thankful to the publishers, researchers for making their resources available and also appreciate the comments and suggestions obtained from the reviewers, which are useful to improve the quality of paper.

REFERENCES

[1] S. Mohammadi, “Scheduling algorithms for real time systems,” Queens University, Tech. Rep., 2005.

[2] K. Schwan and H. Zhou, “Dynamic scheduling of hard real-time tasks and real-time threads.” *IEEE Trans. Software Eng.*, vol. 18, no. 8, pp. 736–748, 1992.

[3] L. Zhang, “Scheduling algorithms for multiprocessor real-time systems,” *IEEE transactions on computers*, 1997.

[4] N. D. Thai, “Real-time scheduling in distributed systems,” in *Proceedings of the International Conference on Parallel Computing in Electrical Engineering*, ser. PARELEC ’02. Washington, DC, USA: IEEE Computer Society, 2002, pp. 165.

[5] J. Anderson, V. Bud, and U. Devi, “An edf-based scheduling algorithm for multiprocessor soft real-time systems,” in *Proceedings. 17th Euromicro Conference on Real-Time Systems*, 2005. (ECRTS 2005).

[6] A. lqbal, A. Zafar, and B. Siddique, “Dynamic queue deadline first scheduling algorithm for soft real time systems,” in *Proceedings of the IEEE Symposium on Emerging Technologies*, 2005.

[7] M. Bertogna and S. Baruah, “Limited preemption edf scheduling of sporadic task systems,” *IEEE Transactions on Industrial Informatics*, vol. 6, no. 4, pp. 579–591, 2010.

[8] A. Shah and K. Kotecha, “Efficient scheduling algorithms for realtime distributed systems,” in *1st International Conference on Parallel Distributed and Grid Computing* (PDGC), 2010, pp. 44–48.

[9] B. Khalib, “High performance nom preemptive dynamic scheduling algorithm for soft real time systems,” *International symposium on computer applications and industrial electronics*, 2012.

[10] Y. ping Chen, L. xiong Wang, and S. tan Huang, “A novel task scheduling algorithm for real-time multiprocessor systems,” in *IEEE International Conference on Control and Automation*, (ICCA), 2007, pp. 271–275.

[11] G. Zarinzad, A.M. Rahmani, and N. Dayhim, “A novel intelligent algorithm for fault-tolerant task scheduling in real-time multiprocessor systems,” in *Third International Conference on Convergence and Hybrid Information Technology*, ICCIT ’08., vol. 2, 2008, pp. 816–821.

[12] C. C. Han, "A fault-tolerant scheduling algorithm for real-time periodic tasks with possible software faults," *IEEE Transactions on Computers*, vol. 52, no. 3, pp. 362–372, 2003.

[13] V. P. Apurva shah, "Design of new scheduling algorithm llf-dm and its comparison with existing def and dm algorithms for periodic tasks," *IEEE international conference on intelligent systems and signal processing*, 2013.

[14] A. Shah and K. Kotecha, "Adaptive scheduling algorithm for realtime multiprocessor systems," in *IEEE International Advance Computing Conference*, IACC 2009, pp. 35–39.

[15] S. Li, S. Ren, Y. Yu, X. Wang, L. Wang, and G. Quan, "Profit and penalty aware scheduling for real-time online services," *IEEE Transactions on Industrial Informatics*, vol. 8, no. 1, pp. 78–89, 2012.

[16] K. Ramamritham, J. Stankovic, and P.-F. Shiah, "Efficient scheduling algorithms for real-time multiprocessor systems," *IEEE Transactions on Parallel and Distributed Systems*, vol. 1, no. 2, pp. 184–194, 1990.

[17] K. Kotecha and A. Shah, "Efficient dynamic scheduling algorithms for real-time multiprocessor systems." in *ISRST*, 2008, pp. 21–25.

[18] J. R. Haritsa, "Scheduling for overload in real-time systems," *IEEE Proceedings of International Conference on Information, Communications and Signal Processing*, 1997.

[19] S. Andrei, A. Cheng, M. Rinard, and L. Osborne, "Optimal scheduling of urgent preemptive tasks," 2012 *IEEE International Conference on Embedded and Real-Time Computing Systems and Applications*, vol. 0, pp. 377–386, 2010.

[20] P. Regnier, G. Lima, E. Massa, G. Levin, and S. A. Brandt, "Run: Optimal multiprocessor real-time scheduling via reduction to uniprocessor." in *RTSS. IEEE Computer Society*, 2011, pp. 104–115.

[21] C. G. Rhan Ha, "Analysis of cache related prremptiom delay in fixed priority preemptive scheduling," *IEEE transactions on computers*, 1998.

[22] C. G. Lee, "Bounding cache-related preemption delay for real-time systems," 1997.

[23] Y. Yu, S. Ren, N. Chen, and X. Wang, "Profit and penalty aware (pp-aware) scheduling for tasks with variable task execution time," in *Proceedings of the 2010 ACM Symposium on Applied Computing, ser.* SAC '10. New York, NY, USA: ACM, 2010, pp. 334–339.

[24] urns, Alan, and Robert Davis. "Mixed criticality systems-a review." Department of Computer Science, University of York, Tech. Rep (2013).

[25] Baruah, Sanjoy K., Alan Burns, and Robert I. Davis. "Response-time analysis for mixed criticality systems." *Real-Time Systems Symposium* (RTSS), 2011 IEEE 32nd. IEEE, 2011.

[26] Li, Haohan. Scheduling mixed-criticality real-time systems. Diss. University of North Carolina at Chapel Hill, 2013.

[27] Writer's Handbook. Mill Valley, CA: University Science, 1989.

[28] Aydin, Hakan, and Qi Yang. "Energy-aware partitioning for multiprocessor real-time systems." *Proceedings InternationalIEEEParallel and Distributed Processing Symposium*, 2003.

[29] Shin, SaeMi, Yena Kim, and Sukyoung Lee. "Deadline-guaranteed scheduling algorithm with improved resource utilization for cloud computing,"*12th Annual Consumer Communications and Networking Conference (CCNC)*, IEEE, 2015.

[30] Lu, Ping, et al. "Toward online profit-driven scheduling of inter-DC data-transfers for cloud applications,"*International Conference on Communications (ICC)*, 2015.

[31] Socci, Dario, et al. "Multiprocessor scheduling of precedence-constrained mixed-critical jobs," 1*8th International Symposium on Real-Time Distributed Computing (ISORC)*,IEEE, 2015.

[32] X. Wang, Z. Li, "Optimal Priority-Free Conditionally-Preemptive Real-Time Scheduling of Periodic Tasks Based on DES Supervisory Control," *IEEE Transactions on Systems, MAN and Cybernetics: Systems, 2016.*