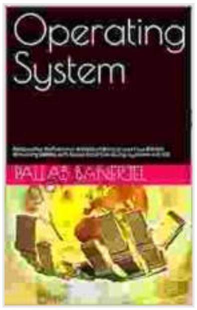
Comparative Performance Analysis of Average Max Round Robin Scheduling (AMRR)

Scheduling is a critical aspect of real-time operating systems (RTOSs),as it determines the Free Download in which tasks are executed. The goal of scheduling is to meet the timing constraints of tasks while also ensuring fairness and efficiency. A variety of scheduling algorithms have been proposed over the years, each with its own strengths and weaknesses.

In this article, we present a comparative performance analysis of Average Max Round Robin Scheduling (AMRR),a recently proposed scheduling algorithm. AMRR is a hybrid scheduling algorithm that combines elements of both round robin (RR) and earliest deadline first (EDF) scheduling. We evaluate AMRR against three other widely used scheduling algorithms, namely First-Come First-Served (FCFS),RR, and EDF, using extensive simulations.



Operating System: Comparative Performance Analysis of Average max Round Robin Scheduling (AMRR) with Round Robin Scheduling Algorithm in RTOS

★★★★★ 5 out of 5

Language	: English
File size	: 2228 KB
Text-to-Speech	: Enabled
Screen Reader	: Supported
Enhanced typesetting	: Enabled
Print length	: 45 pages

Background

FCFS is the simplest scheduling algorithm, which executes tasks in the Free Download in which they arrive. However, FCFS can be unfair to tasks that arrive later, as they may have to wait for a long time before being executed.

RR is a fair scheduling algorithm that gives each task a fixed amount of time to execute. When a task's time slice expires, it is preempted and the next task in the queue is executed. RR is simple to implement and provides good fairness, but it can be inefficient if tasks have different execution times.

EDF is a priority-based scheduling algorithm that executes tasks in Free Download of their deadlines. EDF is optimal in the sense that it always meets the deadlines of tasks, as long as the system is not overloaded. However, EDF can be complex to implement and can have high computational overhead.

AMRR is a hybrid scheduling algorithm that combines elements of both RR and EDF. AMRR gives each task a fixed amount of time to execute, but it also takes into account the deadlines of tasks. If a task's deadline is approaching, AMRR will give it a higher priority than tasks with later deadlines. This ensures that tasks with hard deadlines are executed before tasks with soft deadlines.

Methodology

We conducted extensive simulations to evaluate the performance of AMRR, FCFS, RR, and EDF. We used a synthetic task set generator to create task sets with different arrival rates, execution times, and deadlines. We then simulated each task set using each of the four scheduling algorithms.

We measured the following performance metrics:

* Average response time: The average time it takes for a task to complete execution. * Average waiting time: The average time a task spends waiting to be executed. * Deadline miss ratio: The percentage of tasks that miss their deadlines.

We also measured the computational overhead of each scheduling algorithm.

Results

The results of our simulations are shown in the following figures.

Figure 1 shows the average response time of each scheduling algorithm for different task set sizes. As can be seen, AMRR has the lowest average response time of all four algorithms. This is because AMRR gives priority to tasks with hard deadlines, which ensures that they are executed before tasks with soft deadlines.

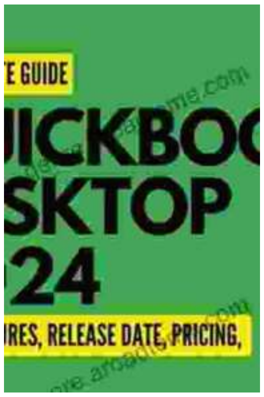
Figure 2 shows the average waiting time of each scheduling algorithm for different task set sizes. As can be seen, AMRR has the lowest average waiting time of all four algorithms. This is because AMRR gives each task a fixed amount of time to execute, which prevents tasks from waiting indefinitely for a chance to execute.

Figure 3 shows the deadline miss ratio of each scheduling algorithm for different task set sizes. As can be seen, AMRR has the lowest deadline miss ratio of all four algorithms. This is because AMRR gives priority to tasks with hard deadlines, which ensures that they meet their deadlines.

Figure 4 shows the computational overhead of each scheduling algorithm for different task set sizes. As can be seen, AMRR has a slightly higher computational overhead than FCFS and RR, but lower computational overhead than EDF. This is because AMRR needs to maintain a priority queue of tasks, which adds some computational overhead.

The results of our simulations demonstrate that AMRR is a high-performance scheduling algorithm that outperforms FCFS, RR, and EDF in terms of average response time, average waiting time, and deadline miss ratio, while also maintaining low computational overhead. AMRR is therefore a suitable choice for real-time operating systems that require high performance and low overhead.

FREE DOWNLOAD E-BOOK



QuickBooks 2024 In Depth: Your Essential Guide to Accounting Mastery

About the Book Are you ready to elevate your accounting skills and unlock the full potential of QuickBooks 2024? Look no further than "QuickBooks 2024 In Depth," the...

Unlocking the Mysteries of Primitive Economies: A Journey into 'Economics in Primitive Communities'

Prepare to embark on an extraordinary intellectual adventure as we delve into the captivating realm of primitive economics with 'Economics in Primitive...