CS 2301-01 Operating Systems 1

Instructor: Professor Ravichandran Purushothaman

Name: Ryohei Hayashi

Learning Journal 3

Completely Fair Scheduler (CFS) is the default process scheduler in the Linux kernel, designed to provide a fair distribution of CPU time among running processes. It is an advanced scheduling algorithm that aims to maximize CPU efficiency by balancing the workload across all available CPU cores.

**How CFS is Run on Linux**

CFS operates based on a concept called "virtual runtime," which ensures that each process receives a fair share of CPU time. The scheduler maintains a red-black tree, a balanced binary search tree, where each node represents a runnable process, sorted by their virtual runtime. When a method is to be scheduled, the scheduler selects the leftmost node (the process with the smallest virtual runtime), ensuring that processes that have received less CPU time are prioritized.

The virtual runtime is calculated by accounting for the actual runtime of the process adjusted by its weight. This weight is derived from the process's nice value, which users can set to indicate the priority of the process. A lower nice value results in a higher priority, thus a smaller virtual runtime increment. This mechanism helps CFS to achieve its goal of fairness, ensuring that processes with lower priority do not starve while higher priority processes still get the CPU time they require.

CFS also implements a feature called "sleeper fairness," ensuring that interactive processes, which may spend significant time waiting for user input, are not penalized for sleeping. When such processes become runnable again, their virtual runtime is adjusted to give them an equitable share of CPU time, thus maintaining overall system responsiveness.

**Why CFS is Preferred for Running a CPU Efficiently**

1. Fairness: CFS is designed to ensure that all processes receive a fair share of CPU time based on their priority and previous usage. This prevents any single process from monopolizing the CPU, leading to a balanced distribution of resources.
2. Responsiveness: The sleeper fairness mechanism enhances the responsiveness of the system by ensuring that interactive processes, which frequently wait for user input, are promptly scheduled once they become active again. This results in a more responsive user experience, particularly in desktop environments.
3. Efficiency: CFS's red-black tree structure and the virtual runtime calculation enable it to efficiently manage process scheduling with minimal overhead. The balanced nature of the tree ensures that operations like insertion, deletion, and finding the next process to run are performed in logarithmic time, contributing to the overall efficiency of the scheduler.
4. Scalability: CFS is scalable across multiple CPU cores, making it suitable for modern multi-core processors. It can effectively distribute the workload among all available cores, ensuring that no single core is overburdened while others remain idle. This balanced distribution enhances the overall performance and efficiency of the system.

In conclusion, the Completely Fair Scheduler (CFS) in Linux is a sophisticated process scheduling algorithm that provides fairness, responsiveness, efficiency, and scalability. By using a virtual runtime and a red-black tree structure, CFS ensures that all processes receive an equitable share of CPU time, making it an ideal choice for efficiently running a CPU in a modern operating system (Kalin, 2019).

Word Count: 516

References

1. Kalin, M. (2019, February 5). *CFS: Completely fair process scheduling in Linux.* Opensource.com. https://opensource.com/article/19/2/fair-scheduling-linux