Multilevel Feedback Queue Scheduling (MLFQ) CPU Scheduling is like Multilevel Queue(MLQ) Scheduling but in this process can move between the queues. And thus, much more efficient than multilevel queue scheduling.

In a multilevel queue-scheduling algorithm, processes are permanently assigned to a queue on entry to the system, and processes are allowed to move between queues.

As the processes are permanently assigned to the queue, this setup has the advantage of low scheduling overhead.

Features of MLFQ:-

Multiple queues Priorities adjusted dynamically Time-slicing Feedback mechanism

Preemption

Advantages of Multilevel Feedback Queue Scheduling:

* It is more flexible.
* It allows different processes to move between different queues.
* It prevents starvation by moving a process that waits too long for the lower priority queue to the higher priority queue.

Disadvantages of Multilevel Feedback Queue Scheduling:

* The selection of the best scheduler, it requires some other means to select the values.
* It produces more CPU overheads.
* It is the most complex algorithm.

A diagram of a few different options

Description automatically generated with medium confidence

Now let us suppose that queues 1 and 2 follow[round robin](https://www.geeksforgeeks.org/program-round-robin-scheduling-set-1/) with time quantum 4 and 8 respectively and queue 3 follow [FCFS](https://www.geeksforgeeks.org/program-for-fcfs-cpu-scheduling-set-1/).

Depending on its priority, the operating system can place a newly launched process into any one of the three queues mentioned above. For instance, the operating system would prefer not to have a background process assigned to queues 1 and 2, which have a higher priority. It will immediately add it to queue 3, which has a lesser priority. Assume that our present procedure for consideration has a high priority and is assigned to queue 1.

The process in queue 1 runs for four units. If it finishes within these units or provides CPU for I/O operations inside these units, its priority remains unchanged. If the process re-enters the ready queue, it begins executing in queue 1 once more.

If a process in queue 1 does not complete in 4 units then its priority gets reduced and it is shifted to queue 2.

Above points 2 and 3 are also true for queue 2 processes but the time quantum is 8 units. In a general case if a process does not complete in a time quantum then it is shifted to the lower priority queue.

In the last queue, processes are scheduled in an FCFS manner.

A process in a lower priority queue can only execute only when higher priority queues are empty.

A process running in the lower priority queue is interrupted by a process arriving in the higher priority queue.