GOA COLLEGE OF ENGINEERING

"Bhausaheb Bandodkar Technical Education Complex"

Experiment No: 7

Aim: To implement Earliest Deadline First Algorithm

Theory:

Earliest Deadline First (EDF) is an optimal dynamic priority scheduling algorithm used in real-time systems. It can be used for both static and dynamic real-time scheduling. EDF uses priorities to the jobs for scheduling. It assigns priorities to the task according to the absolute deadline. The task whose deadline is closest gets the highest priority. The priorities are assigned and changed in a dynamic fashion. EDF is very efficient as compared to other scheduling algorithms in real-time systems. It can make the CPU utilization to about 100% while still guaranteeing the deadlines of all the tasks. EDF includes the kernel overload. In EDF, if the CPU usage is less than 100%, then it means that all the tasks have met the deadline. EDF finds an optimal feasible schedule. The feasible schedule is one in which all the tasks in the system are executed within the deadline. If EDF is not able to find a feasible schedule for all the tasks in the real-time system, then it means that no other task scheduling algorithms in real-time systems can give a feasible schedule. All the tasks which are ready for execution should announce their deadline to EDF when the task becomes runnable. EDF scheduling algorithm does not need the tasks or processes to be periodic and also the tasks or processes require a fixed CPU burst time. In EDF, any executing task

Code:

```
class Job:
def __init__(self, num, length, deadline):
self.length = int(length)
self.deadline = int(deadline)
self.num = num

def __str__(self):
return f"Job_num:{self.num}_len:{self.length}_deadline:{self.deadline}"

def __repr__(self):
return f"Job_num:{self.num}_len:{self.length}_deadline:{self.deadline}"


def orderJobs(jobList):
jobOrder = []
while(len(jobList) > 0):
earliestDeadline = 1000000
earliestJob = -1
for idx, job in enumerate(jobList):
if job.deadline < earliestDeadline:
earliestDeadline = job.deadline
```

```
earliestJob = idx
jobOrder.append(job)
del jobList[earliestJob]
return jobOrder


def feasabilityCheck(jobOrder):
currTime = 0
for job in jobOrder:
currTime += job.length
if currTime > job.deadline:
return False
return True


def main():
n = int(input("Enter number of jobs: "))
jobList = []
for i in range(1, n+1):
currJob = input(
f"Enter the length and deadline of job {i} separated by spaces: \n")
jobList.append(Job(i, *currJob.split(" ")))
jobOrder = orderJobs(jobList)
if feasabilityCheck(jobOrder):
print("Job order will be:")
for job in jobOrder:
print(job)
else:
print("Jobs cannot be scheduled so all jobs complete before deadline")


main()
```

GOA COLLEGE OF ENGINEERING

"Bhausaheb Bandodkar Technical Education Complex"

Output:

```
deeprajb in < MacBookAir.local > Sem8Stuff/MS/Exps  main  ×2 ×10via  v18.4.0  v3.10.4
❯ python3 Exp7.py
Enter number of jobs: 2
Enter the length and deadline of job 1 separated by spaces:
10 20
Enter the length and deadline of job 2 separated by spaces:
5 4
Jobs cannot be scheduled so all jobs complete before deadline
```

```
deeprajb in < MacBookAir.local > Sem8Stuff/MS/Exps  main  ×2 ×10via  v18.4.0  v3.10.4  took 20
s
❯ python3 Exp7.py
Enter number of jobs: 2
Enter the length and deadline of job 1 separated by spaces:
10 20
Enter the length and deadline of job 2 separated by spaces:
5 6
Job order will be:
Job_num:2_len:5_deadline:6
Job_num:1_len:10_deadline:20
```

Conclusion: A program implementing Earliest Deadline First Algorithm was successfully implemented and executed .