



RAMAIAH
Institute of Technology

Real-Time Scheduling:

Earliest Deadline First (EDF) Algorithm

Shreya - 1MS22CS138

Shrinidhi Pawar - 1MS22CS141



RAMAIAH
Institute of Technology

Problem Statement

Real-Time Scheduling (Earliest Deadline First)

- Implement the Earliest Deadline First (EDF) scheduling algorithm
- Manage tasks in a real-time system efficiently
- Ensure tasks meet their deadlines to avoid missed deadlines



RAMAIAH
Institute of Technology

Solution Overview

- Developed a Linux Kernel module to simulate EDF and CFS scheduling
- Implemented process queues and scheduling logic in C
- Created a user-space Python GUI for live interactive visualization
- Visualization includes Gantt chart animations and scheduling metrics
- User controls: Start, Stop, and Restart to observe scheduling step-by-step



RAMAIAH
Institute of Technology

Architecture Flow



RAMAIAH
Institute of Technology

Tools and Technologies Used

- Kernel Module Programming: C language, Linux Kernel 5.x environment
- Process Management: Kernel linked lists and scheduling simulation
- User Interface: Python with Tkinter for GUI and Matplotlib for Gantt charts
- Build and Automation: Makefile for kernel module compilation
- Visualization: Real-time animations demonstrating scheduling decisions and metrics

Kernel Module - edf_scheduler1.c

Purpose:

- Simulates Earliest Deadline First (EDF) and Completely Fair Scheduler (CFS) algorithms
- Manages task scheduling and execution inside the Linux kernel module
- Tracks CPU usage and missed deadlines for each scheduler
- Provides logs via /proc for monitoring and comparison
- Enables real-time testing and visualization of scheduling behavior



Key Code Snippet

```
static void add_edf_process(int pid, int deadline, int exec_time, int arrival_time) {  
    struct edf_process *new_proc = kmalloc(sizeof(*new_proc), GFP_KERNEL);  
    struct edf_process *pos;  
  
    if (!new_proc) return;  
  
    new_proc->pid = pid;  
    new_proc->deadline = deadline;  
    new_proc->execution_time = exec_time;  
    new_proc->arrival_time = arrival_time;  
  
    list_for_each_entry(pos, &edf_ready_queue, list) {  
        if (deadline < pos->deadline) {  
            list_add_tail(&new_proc->list, &pos->list);  
            return;  
        }  
    }  
    list_add_tail(&new_proc->list, &edf_ready_queue);  
}
```



RAMAIAH
Institute of Technology

Key Code Snippet

```
// EDF Scheduler simulation
static void run_edf_scheduler(void) {
    struct edf_process *proc, *tmp;
    edf_current_time = 0;
    edf_missed_deadlines = 0;

    list_for_each_entry_safe(proc, tmp, &edf_ready_queue, list) {
        if (edf_current_time < proc->arrival_time)
            edf_current_time = proc->arrival_time;

        printk("Running P%d (Deadline: %d, Exec: %d) at time %d\n",
               proc->pid, proc->deadline, proc->execution_time, edf_current_time);

        edf_current_time += proc->execution_time;

        if (edf_current_time > proc->deadline) {
            edf_missed_deadlines++;
            printk("Deadline missed by P%d\n", proc->pid);
        }

        list_del(&proc->list);
        kfree(proc);
    }
}
```


Makefile Script

Purpose:

- Automates compilation of the kernel module `edf_scheduler1.c`
- Uses the current kernel build environment for compatibility
- Provides targets for building (all) and cleaning (clean) module files
- Simplifies module management with standard commands



RAMAIAH
Institute of Technology

Code Snippet

```
obj-m += edf_scheduler1.o

KDIR := /lib/modules/$(shell uname -r)/build
PWD := $(shell pwd)

all:
    $(MAKE) -C $(KDIR) M=$(PWD) modules

clean:
    $(MAKE) -C $(KDIR) M=$(PWD) clean
```

GUI - edf_scheduler1.py

Purpose:

- Provides a graphical interface to simulate and visualize EDF and CFS scheduling algorithms
- Displays live animated Gantt charts comparing the two schedulers step-by-step
- Offers user controls: Start, Stop, Restart, and Next Step for interactive simulation
- Reads real-time scheduler logs from the kernel module via /proc/edf_cfs_log
- Shows key metrics like missed deadlines and CPU utilization dynamically
- Helps users understand and compare the scheduling behavior visually and interactively



RAMAIAH
Institute of Technology

Key Code Snippet

```
def next_step(self):
    if self.current_step >= len(self.all_events):
        self.timer.stop()
        return
    event = self.all_events[self.current_step]
    scheduler, pid, start, end = event
    self.plot_gantt(upto=self.current_step + 1)
    self.current_step += 1

def start_simulation(self):
    self.load_logs()
    self.current_step = 0
    self.timer.start(1000) # 1-second interval

def stop_simulation(self):
    self.timer.stop()
```



RAMAIAH
Institute of Technology

Output Screenshots

```
cselab1@rit:~/Downloads/22CS138-141/edf_activity$ make
make -C /lib/modules/6.11.0-25-generic/build M=/home/cselab1/Downloads/22CS138-141/edf_activity modules
make[1]: Entering directory '/usr/src/linux-headers-6.11.0-25-generic'
warning: the compiler differs from the one used to build the kernel
The kernel was built by: x86_64-linux-gnu-gcc-13 (Ubuntu 13.3.0-6ubuntu2-24.04) 13.3.0
You are using: gcc-13 (Ubuntu 13.3.0-6ubuntu2-24.04) 13.3.0
CC [M] /home/cselab1/Downloads/22CS138-141/edf_activity/edf_scheduler1.o
MODPOST /home/cselab1/Downloads/22CS138-141/edf_activity/Module.symvers
LD [M] /home/cselab1/Downloads/22CS138-141/edf_activity/edf_scheduler1.ko
BTF [M] /home/cselab1/Downloads/22CS138-141/edf_activity/edf_scheduler1.ko
Skipping BTF generation for /home/cselab1/Downloads/22CS138-141/edf_activity/edf_scheduler1.ko due to unavailability of vmlinux
make[1]: Leaving directory '/usr/src/linux-headers-6.11.0-25-generic'
cselab1@rit:~/Downloads/22CS138-141/edf_activity$ sudo insmod edf_scheduler1.ko
[sudo] password for cselab1:
Sorry, try again.
[sudo] password for cselab1:
insmod: ERROR: could not insert module edf_scheduler1.ko: File exists
cselab1@rit:~/Downloads/22CS138-141/edf_activity$ echo 1 > /proc/edf_cfs_log
cselab1@rit:~/Downloads/22CS138-141/edf_activity$ cat /proc/edf_cfs_log
Starting EDF scheduling simulation...
Time 4: Running process P6 (Deadline: 6, Exec: 1)
-> P6 completed on time.
Time 5: Running process P4 (Deadline: 8, Exec: 2)
-> P4 completed on time.
Time 7: Running process P9 (Deadline: 9, Exec: 2)
-> P9 completed on time.
Time 9: Running process P2 (Deadline: 10, Exec: 2)
-> Deadline Missed by P2
Time 11: Running process P3 (Deadline: 12, Exec: 1)
-> P3 completed on time.
Time 12: Running process P8 (Deadline: 14, Exec: 4)
-> Deadline Missed by P8
Time 16: Running process P1 (Deadline: 15, Exec: 3)
-> Deadline Missed by P1
```




RAMAIAH
Institute of Technology

Output Screenshots

```
cselab1@rit: ~/Downloads/22CS138-1...  x  cselab1@rit: ~/Downloads/22CS138-1...  x  cselab1@rit: ~/Downloads/22CS138-14...  
-> Deadline Missed by P5  
EDF scheduling simulation completed.  
  
Starting CFS scheduling simulation...  
Time 4: Running process P6 (vruntime: 2, Exec: 1)  
-> P6 completed on time.  
Time 5: Running process P4 (vruntime: 5, Exec: 2)  
-> P4 completed on time.  
Time 7: Running process P9 (vruntime: 7, Exec: 2)  
-> P9 completed on time.  
Time 9: Running process P1 (vruntime: 10, Exec: 3)  
-> Deadline Missed by P1  
Time 12: Running process P8 (vruntime: 12, Exec: 4)  
-> P8 completed on time.  
Time 16: Running process P10 (vruntime: 14, Exec: 1)  
-> P10 completed on time.  
Time 17: Running process P3 (vruntime: 15, Exec: 1)  
-> Deadline Missed by P3  
Time 18: Running process P7 (vruntime: 18, Exec: 3)  
-> Deadline Missed by P7  
Time 21: Running process P2 (vruntime: 20, Exec: 2)  
-> Deadline Missed by P2  
Time 23: Running process P5 (vruntime: 25, Exec: 5)  
-> Deadline Missed by P5  
CFS scheduling simulation completed.  
  
--- Statistics ---  
EDF Missed Deadlines: 6  
CFS Missed Deadlines: 5  
Total Missed Deadlines: 11  
EDF CPU Utilization: 85%  
CFS CPU Utilization: 85%  
Overall CPU Utilization: 85%  
cselab1@rit: ~/Downloads/22CS138-141/edF_activity$
```



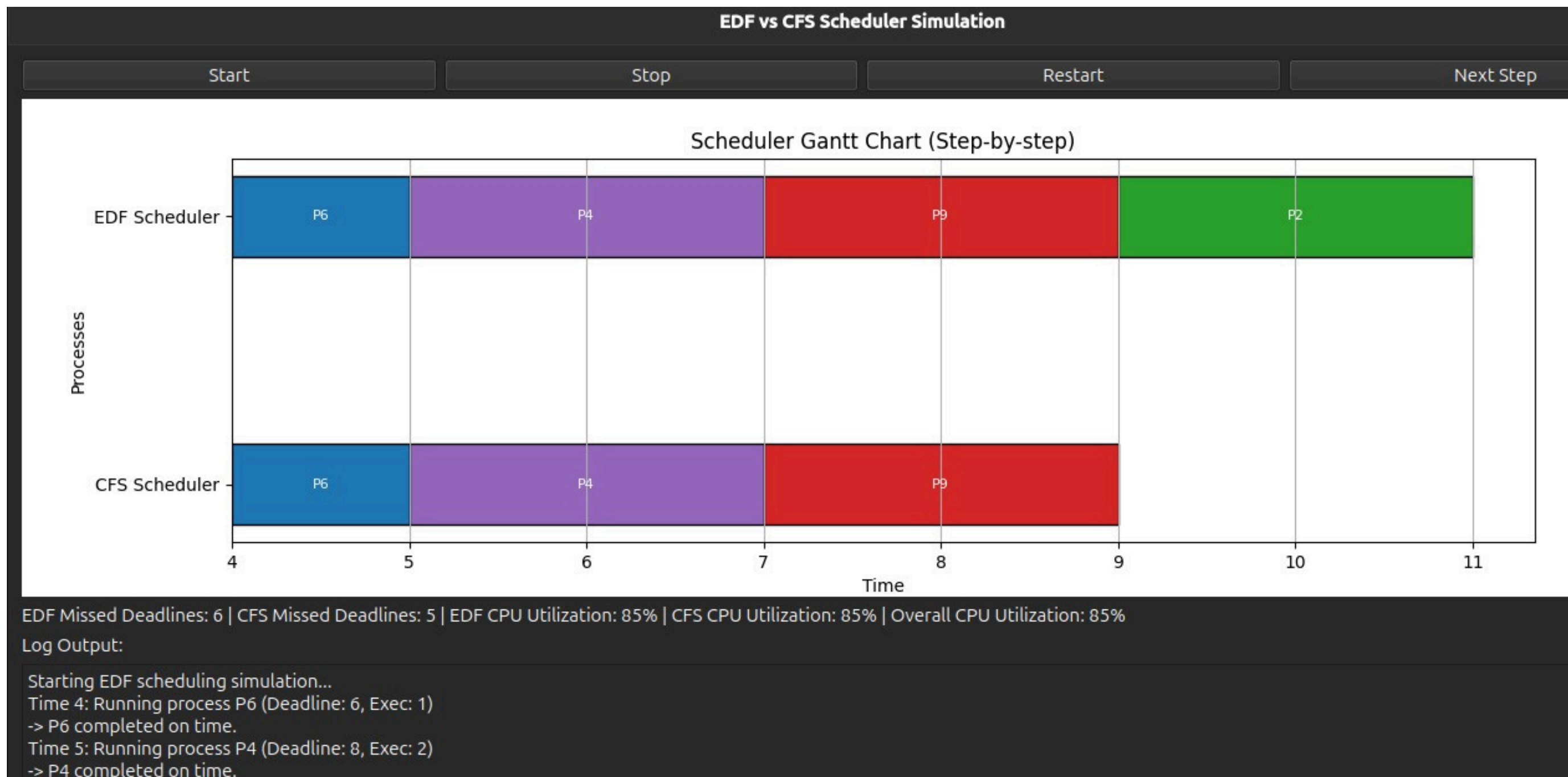
RAMAIAH
Institute of Technology

Output Screenshots



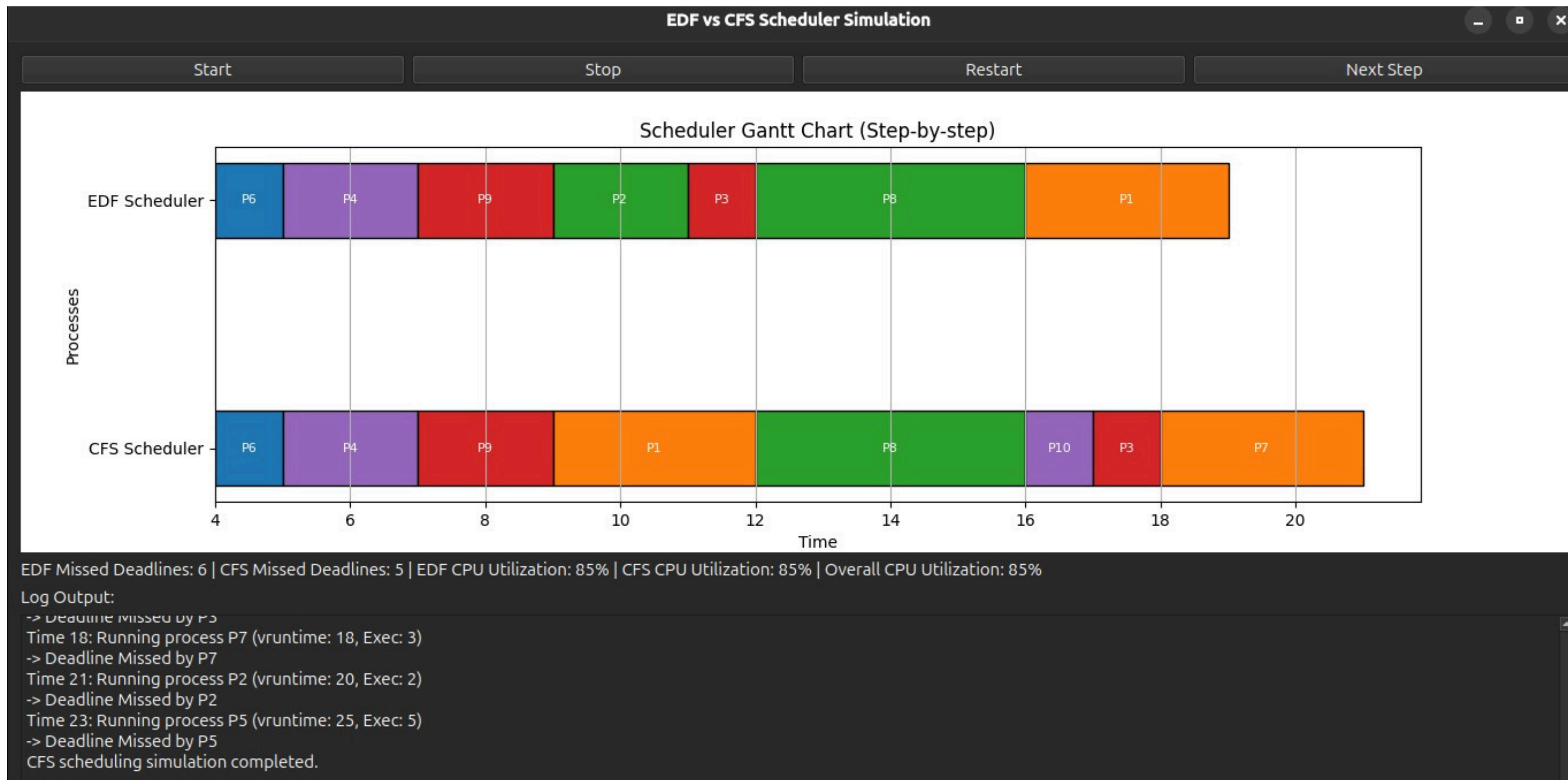


Output Screenshots





Output Screenshots





RAMAIAH
Institute of Technology

Output Screenshots

Log Output:

```
Starting EDF scheduling simulation...
Time 4: Running process P6 (Deadline: 6, Exec: 1)
-> P6 completed on time.
Time 5: Running process P4 (Deadline: 8, Exec: 2)
-> P4 completed on time.
Time 7: Running process P9 (Deadline: 9, Exec: 2)
-> P9 completed on time.
Time 9: Running process P2 (Deadline: 10, Exec: 2)
-> Deadline Missed by P2
Time 11: Running process P3 (Deadline: 12, Exec: 1)
-> P3 completed on time.
Time 12: Running process P8 (Deadline: 14, Exec: 4)
-> Deadline Missed by P8
Time 16: Running process P1 (Deadline: 15, Exec: 3)
-> Deadline Missed by P1
Time 19: Running process P10 (Deadline: 16, Exec: 1)
-> Deadline Missed by P10
Time 20: Running process P7 (Deadline: 18, Exec: 3)
```



RAMAIAH
Institute of Technology

Output Screenshots

Log Output:

```
Starting CFS scheduling simulation...
Time 4: Running process P6 (vruntime: 2, Exec: 1)
-> P6 completed on time.
Time 5: Running process P4 (vruntime: 5, Exec: 2)
-> P4 completed on time.
Time 7: Running process P9 (vruntime: 7, Exec: 2)
-> P9 completed on time.
Time 9: Running process P1 (vruntime: 10, Exec: 3)
-> Deadline Missed by P1
Time 12: Running process P8 (vruntime: 12, Exec: 4)
-> P8 completed on time.
Time 16: Running process P10 (vruntime: 14, Exec: 1)
-> P10 completed on time.
Time 17: Running process P3 (vruntime: 15, Exec: 1)
-> Deadline Missed by P3
Time 18: Running process P7 (vruntime: 18, Exec: 3)
-> Deadline Missed by P7
Time 21: Running process P2 (vruntime: 20, Exec: 2)
```



RAMAIAH
Institute of Technology

Output Screenshots

```
--- Statistics ---  
EDF Missed Deadlines: 6  
CFS Missed Deadlines: 5  
Total Missed Deadlines: 11  
EDF CPU Utilization: 85%  
CFS CPU Utilization: 85%  
Overall CPU Utilization: 85%
```



RAMAIAH
Institute of Technology

Challenges Faced

- Handling real-time data reading and updates
- Ensuring smooth step-by-step animation
- Managing synchronization between EDF and CFS events
- Designing an intuitive and responsive UI
- Debugging kernel-user space communication



RAMAIAH
Institute of Technology

Enhancements

- Added start, stop, restart, and step-by-step controls
- Integrated live Gantt chart animations for better visualization
- Displayed comparative metrics for EDF and CFS schedulers
- Improved user interaction for sequential process execution
- Enhanced error handling for proc file operations

Conclusion

- Successfully implemented EDF real-time scheduling with interactive visualization
- Enabled comparison with CFS scheduler through live Gantt charts and metrics
- Provided user-friendly controls for better understanding of scheduling behavior
- Project demonstrates effective simulation and analysis of scheduling algorithms