تمرین 5: الگوریتم زمانبندی Least Laxity را برای هر مجموعه وظیفه aperiodic دلخواهی پیاده سازی کنید. زمانبندی وظایف در خروجی کد شما باید به صورت بصری نمایش داده شود.

ملیکا علیزاده 401106255

در برنامه زیر الگوریتم زمانبندی Least Laxity را برای هر مجموعه وظیفه aperiodic دلخواهی پیاده سازی شده است.

در کلاس task به معرفی وظیفه و ویژگی آن می‌پردازیم. و در متود laxity مقدار آن را محاسبه می‌کنیم.

```python
class Task:
    def __init__(self, task_id, arrival_time, execution_time, deadline):
        self.task_id = task_id
        self.arrival_time = arrival_time
        self.execution_time = execution_time
        self.deadline = deadline
        self.remaining_time = execution_time

    def laxity(self, current_time):
        return self.deadline - (current_time + self.remaining_time)
```

در تابع llf_scheduler الگوریتم را پیاده‌سازی می‌کنیم. به این شکل که در هر لحظه از زمان تسک‌هایی که تازه وارد شده‌اند به لیست active_tasks اضافه می‌شوند و تسک‌هایی که اجراشان تمام شده حذف می‌شوند. و تسکی که کمترین laxity را دارد انتخاب و از زمان باقی‌مانده آن تسک یک واحد کم می‌شود و در لیست زمانبندی ثبت می‌شود. در صورتی که تسکی برای اجرا وجود نداشته باشد مقدار Idle ثبت می‌شود.

```python
def llf_scheduler(tasks, total_time):
    time_line = []
    active_tasks = []
    all_tasks = sorted(tasks, key=lambda t: t.arrival_time)
    arrived_index = 0

    for current_time in range(total_time):
        # Add new arrived tasks
        while arrived_index < len(all_tasks) and all_tasks[arrived_index].arrival_time == current_time:
            active_tasks.append(all_tasks[arrived_index])
            arrived_index += 1

        # Remove finished tasks
        active_tasks = [t for t in active_tasks if t.remaining_time > 0]

        if active_tasks:
            # Sort by least laxity
            active_tasks.sort(key=lambda t: t.laxity(current_time))
            current_task = active_tasks[0]
            current_task.remaining_time -= 1
            time_line.append(current_task.task_id)
        else:
            time_line.append("Idle")

    return time_line
```

در تابع plot_schedule خروجی زمان‌بندی را به‌صورت نمودار تصویری نمایش می‌دهیم. هر تسک با رنگی متفاوت نمایش داده می‌شود و زمان‌هایی که CPU بیکار است با رنگ خاکستری مشخص می‌شود.

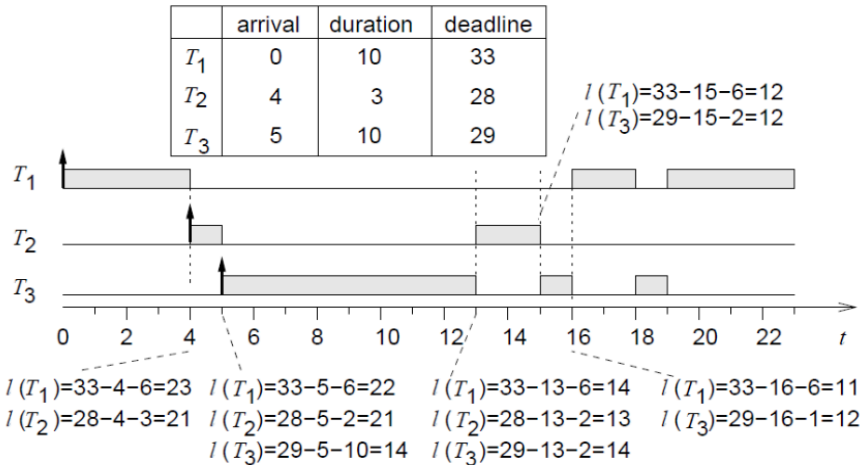
```python
def plot_schedule(time_line, total_time):
    colors = {}
    unique_tasks = set(time_line)
    for i, task in enumerate(unique_tasks):
        if task != "Idle":
            colors[task] = f"C{i}"

    fig, ax = plt.subplots(figsize=(15, 2))
    for time, task in enumerate(time_line):
        if task != "Idle":
            ax.add_patch(mpatches.Rectangle((time, 0), 1, 1, color=colors[task]))
        else:
            ax.add_patch(mpatches.Rectangle((time, 0), 1, 1, color="gray", alpha=0.5))

    ax.set_xlim(0, total_time)
    ax.set_ylim(0, 1)
    ax.set_xticks(range(0, total_time + 1))
    ax.set_yticks([])
    ax.set_xlabel("Time")
    ax.set_title("LLF Scheduling for Aperiodic Tasks")

    legend_handles = [mpatches.Patch(color=colors[task], label=f"Task {task}") for task in unique_tasks if task != "Idle"]
    legend_handles.append(mpatches.Patch(color="gray", alpha=0.5, label="Idle"))
    ax.legend(handles=legend_handles, loc="upper right", ncol=3)
    plt.show()
```

برای تست این کد از مثال اسلاید زیر استفاده شده است تا درستی الگوریتم پیاده‌سازی شده چک شود.




```python
# Example usage according to slides
tasks = [
    Task(task_id=1, arrival_time=0, execution_time=10, deadline=33),
    Task(task_id=2, arrival_time=4, execution_time=3, deadline=28),
    Task(task_id=3, arrival_time=5, execution_time=10, deadline=29),
]

total_time = max(task.deadline for task in tasks)
schedule_llf = llf_scheduler(tasks, total_time)

print("LLF Schedule:", schedule_llf)
plot_schedule(schedule_llf, total_time)
```

خروجی:

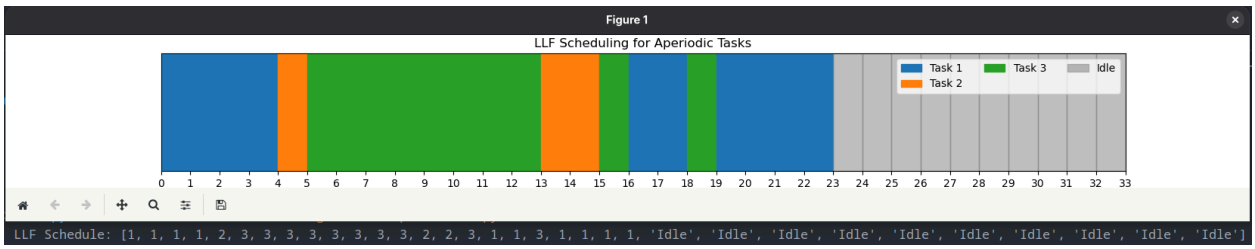


Figure 1
LLF Scheduling for Aperiodic Tasks
Task 1
Task 2
Task 3
Idle
0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31 32 33
LLF Schedule: [1, 1, 1, 1, 2, 3, 3, 3, 3, 3, 3, 3, 3, 2, 2, 3, 1, 1, 3, 1, 1, 1, 1, 'Idle', 'Idle', 'Idle', 'Idle', 'Idle', 'Idle', 'Idle', 'Idle', 'Idle', 'Idle']