عارف معتمدی

۹۶۳۱۰۷۰

گزارش پروژه پایانی

موضوع انتخابی: پیاده سازی الگوریتم NPP در زمانبند RM

پروژه با زبان برنامه نویسی پایتون نوشته شد و بجای استفاده از فایل قرار داده شده، بنده تمام پروژه را از اول به تنهایی نوشتم (البته این مورد به این خاطر بود که تا اواسط پیاده سازی پروژه، از وجود این فایل مطلع نبودم و تا حد خوبی بیس این برنامه را نوشته بودم). لازم به ذکر است که این به حد کمتری شئی گرا است.

```
inputjson = None
with open("input.json", "r") as read_file:
    inputjson = json.load(read_file)

startTime = inputjson['startTime']
endTime = inputjson['endTime']

taskset = inputjson['taskset']

print('Task Set:')
for task in taskset:
    print('task',task['taskId'],': (\u03A6,T,C,D,\u0394) = ',end='')
    print('(',task['offset'],',',task['period'],',',task['wcet'],',',task['deadline'],',',task['sections'],')')
```

در این قسمت ابتدایی کد ما ابتدا فایل input.json را می‌خوانیم و مجموعه تسک ها را از آن استخراج می‌کنیم.

از اینجا به بعد به ازای هر واحد زمان از شروع (startTime) تا انتها (endTime) محاسبات را انجام می‌دهیم.

```
finished =[]
ongoingTasks = {}
semaphorInUse = []

flag = False
for current_time in range(startTime,endTime+1):

    print('****************************************')
    print('              Second:',current_time,'              ')
```

لازم به ذکر است که ما مجموعه taskهای در حال پردازش را در ongoingTasks داریم، اگر منبع مشترکی در حال استفاده باشد در semaphoreInUse و لیست کار های انتها یافته را در finished نگه می‌داریم. flag در حالت false نیز نشان دهنده feasible بودن این مجموعه است.

```
for task in taskset:
    if not(task['taskId']-1 in ongoingTasks):
        ongoingTasks[task['taskId']-1] = {}
    periodNum =  max(int((current_time-task['offset'] )/ task['period']),0)
    newTime = current_time - periodNum * task['period'] - task['offset']
    if not(periodNum in ongoingTasks[task['taskId']-1]):
        if newTime>= 0 and newTime<task['deadline']:
            if not([task['taskId']-1,periodNum] in finished):
                print('Task',task['taskId'],'created. Period Number:',periodNum)
                ongoingTasks[task['taskId']-1][periodNum] = {}
                ongoingTasks[task['taskId']-1][periodNum]['remaining'] = task['wcet']
```

در اینجا ما چک میکنیم اگر وارد پریود جدید یک تسکی شویم، آن را به مجموعه OngoingTasks اضافه میکنیم. هر job با شناسه ۲ متغیر taskID, periodNumber به این مجموعه اضافه میشود. PeriodNumber در واقع نشان‌دهنده i ـمین دوره پریود این تسک است.

```
deletingTasks = []
for task,val in ongoingTasks.items():
    for per,dic in val.items():
        if current_time >= ((per*taskset[task]['period'])+taskset[task]['offset']+ taskset[task]['deadline']) :
            print('removing task',task+1)
            deletingTasks.append([task,per])
            print('OH NO... OH NO... OH NO NO NO NO NO... RM-NPP ****FAILED**** !!!!!!!!!')
            flag = True
        elif dic['remaining'] <= 0:
            print('finished task',task+1)

            deletingTasks.append([task,per])

for (task,per) in deletingTasks:
    ongoingTasks[task].pop(per)
    finished.append([task,per])
```

در این قسمت چک می‌کنیم اگر هر کدام از تسک های ما به ددلاین رسیده بودند و یا کارشان با موفقیت به اتمام رسیده بود، از مجموعه ongoingTasks حذف می‌شوند. در صورت رسیدن یک جاب به ددلاین، flag برابر True شده و به این معنی است که مجموعه ما feasible نبوده و یک بار حد اقل ددلاین یک جاب را miss کرده.

لازم به ذکر است که در برنامه، بعد از رسیدن به ددلاین یک تسک، دیگر پردازش آن را ادامه نمی‌دهد و به عبارتی آن را از مجموعه job های در حال پردازش حذف میکند.

```
if semaphorInUse == []:

    rmList = []
    for task, dic1 in ongoingTasks.items():
        for per,dic2 in dic1.items():
            rmList.append((task,per,taskset[task]['period']))
    if len(rmList)>0:
        bestchoice = sorted(rmList ,key=lambda x: x[2])[0]
        ith_time = taskset[bestchoice[0]]['wcet'] - ongoingTasks[bestchoice[0]][bestchoice[1]]['remaining']

        if (ith_time ==0):
            ongoingTasks[bestchoice[0]][bestchoice[1]]['section'] = taskset[bestchoice[0]]['sections'][0]
            semaphorInUse= [taskset[bestchoice[0]]['sections'][0][1],bestchoice[0],bestchoice[1],taskset[bestchoice[0]]['sec
        else:
            counter = 0
            for sec in taskset[bestchoice[0]]['sections']:
                if ith_time == counter:
                    semaphorInUse= [sec[1],bestchoice[0],bestchoice[1],sec[0]]
                    break
                else:
                    counter += sec[1]

        ongoingTasks[bestchoice[0]][bestchoice[1]]['remaining'] -=1
        print('Executing task:',bestchoice[0]+1,', Period Number:',bestchoice[1],', semaphore:',semaphorInUse[3])
        semaphorInUse[0] -=1
        if semaphorInUse[0] <=0:
            semaphorInUse = []
    else:
        print('This time slot, CPU is free :)')
```

در این قسمت چک میشود اگر منبع مشترکی در حال استفاده نباشد، job ها را به ترتیب RM سورت میکنیم و بهترین را بر می‌گزینیم. در این نقطه ما چک میکنیم که الان باید به کدامین section این برنامه بپردازیم و آن section را در semaphore in use با ذکر taskID و periodNumber جاب، ذخیره می‌کنیم.

در انتها برنامه/جاب مورد نظر را ۱ واحد زمانی در section مشخص شده ران میکنیم. چک میکینم اگر آن section به انتها رسیده بود، semaphore In Use را خالی می‌کنیم.

اگر هم لیست مجموعه تسک ها خالی باشد، میگوییم در این اسلات زمانی، CPU ما خالی می‌باشد.

```
else:
    ongoingTasks[semaphorInUse[1]][semaphorInUse[2]]['remaining'] -=1
    print('Executing task:',semaphorInUse[1]+1,', Period Number:',semaphorInUse[2],', semaphore:',semaphorInUse[3])

    semaphorInUse[0] -=1
    if semaphorInUse[0] <=0:
        semaphorInUse = []
```

سپس در صورت در حال استفاده بودن منبعی و یا section یی از یک جاب، چون ما الگوریتم NPP را داریم و preemption برای مواقع استفاده از منبع مشترک وجود ندارد، باید تا انتهای آن section به انجام آن بپردازیم. در اینجا نیز ما ۱ واحد زمانی صرف انجام کار مد نظر می‌کنیم و چک میکنیم اگر section به انتها رسید، مجموعه semaphore in use را خالی می‌کنیم.

در انتها feasible بودن این الگوریتم برای مجموعه تسک های ورودی را چک می‌کنیم.

```
if flag:

    print('')
    print('')
    print('============================================================')
    print(' // Validating the Alghorithm:')
    print('          RM-NPP is NOT feasible for this task set')
    print('')

else:
    print('')
    print('')
    print('============================================================')
    print(' // Validating the Alghorithm:')
    print('          RM-NPP IS feasible for this task set')
    print('')
```

تسک ست ۱)

```
{
    "startTime": 0,
    "endTime": 20,
    "taskset": [
        {
            "taskId": 1,
            "period": 25,
            "wcet": 3,
            "deadline": 25,
            "offset": 1,
            "sections": [[1,1],[0,2]]
        },
        {
            "taskId": 2,
            "period": 30,
            "wcet": 4,
            "deadline": 30,
            "offset": 0,
            "sections": [[1,2],[3,2]]
        }
    ]
}
```

```
F:\Courses\Embedded Real-time Systems\Project>python prj.py
Task Set:
task 1 : (Φ,T,C,D,Δ) = ( 1 , 25 , 3 , 25 , [[1, 1], [0, 2]] )
task 2 : (Φ,T,C,D,Δ) = ( 0 , 30 , 4 , 30 , [[1, 2], [3, 2]] )
****************************************
                Second: 0
Task 2 created. Period Number: 0
Executing task: 2 , Period Number: 0 , semaphore: 1
****************************************
                Second: 1
Task 1 created. Period Number: 0
Executing task: 2 , Period Number: 0 , semaphore: 1
****************************************
                Second: 2
Executing task: 1 , Period Number: 0 , semaphore: 1
****************************************
                Second: 3
Executing task: 1 , Period Number: 0 , semaphore: 0
****************************************
                Second: 4
Executing task: 1 , Period Number: 0 , semaphore: 0
****************************************
                Second: 5
finished task 1
Executing task: 2 , Period Number: 0 , semaphore: 3
****************************************
                Second: 6
Executing task: 2 , Period Number: 0 , semaphore: 3
****************************************
                Second: 7
finished task 2
This time slot, CPU is free :)
****************************************
                Second: 8
This time slot, CPU is free :)
****************************************
                Second: 9
This time slot, CPU is free :)
****************************************
                Second: 10
This time slot, CPU is free :)
****************************************
                Second: 11
This time slot, CPU is free :)
****************************************
                Second: 12
This time slot, CPU is free :)
****************************************
                Second: 13
This time slot, CPU is free :)
****************************************
                Second: 14
This time slot, CPU is free :)
****************************************
                Second: 15
This time slot, CPU is free :)
****************************************
                Second: 16
This time slot, CPU is free :)
****************************************
                Second: 17
This time slot, CPU is free :)
****************************************
                Second: 18
This time slot, CPU is free :)
****************************************
                Second: 19
This time slot, CPU is free :)
****************************************
                Second: 20
This time slot, CPU is free :)


================================================
 // Validating the Alghorithm:
          RM-NPP IS feasible for this task set
```

تسک ست ۲)

```
{
    "startTime": 0,
    "endTime": 20,
    "taskset": [
        {
            "taskId":1,
            "period":15,
            "wcet":3,
            "deadline":15,
            "offset":1,
            "sections":[[1,1],[0,2]]
        },{
            "taskId":2,
            "period":20,
            "wcet":4,
            "deadline":17,
            "offset":0,
            "sections":[[1,2],[3,2]]
        }
    ]
}
```

```
F:\Courses\Embedded Real-time Systems\Project>python prj.py
Task Set:
task 1 : (Φ,T,C,D,Δ) = ( 1 , 15 , 3 , 15 , [[1, 1], [0, 2]] )
task 2 : (Φ,T,C,D,Δ) = ( 0 , 20 , 4 , 17 , [[1, 2], [3, 2]] )
****************************************
                Second: 0
Task 2 created. Period Number: 0
Executing task: 2 , Period Number: 0 , semaphore: 1
****************************************
                Second: 1
Task 1 created. Period Number: 0
Executing task: 2 , Period Number: 0 , semaphore: 1
****************************************
                Second: 2
Executing task: 1 , Period Number: 0 , semaphore: 1
****************************************
                Second: 3
Executing task: 1 , Period Number: 0 , semaphore: 0
****************************************
                Second: 4
Executing task: 1 , Period Number: 0 , semaphore: 0
****************************************
                Second: 5
finished task 1
Executing task: 2 , Period Number: 0 , semaphore: 3
****************************************
                Second: 6
Executing task: 2 , Period Number: 0 , semaphore: 3
****************************************
                Second: 7
finished task 2
This time slot, CPU is free :)
****************************************
                Second: 8
This time slot, CPU is free :)
****************************************
                Second: 9
This time slot, CPU is free :)
****************************************
                Second: 10
This time slot, CPU is free :)
****************************************
                Second: 11
This time slot, CPU is free :)
****************************************
                Second: 12
This time slot, CPU is free :)
****************************************
                Second: 13
This time slot, CPU is free :)
****************************************
                Second: 14
This time slot, CPU is free :)
****************************************
                Second: 15
This time slot, CPU is free :)
****************************************
                Second: 16
Task 1 created. Period Number: 1
Executing task: 1 , Period Number: 1 , semaphore: 1
****************************************
                Second: 17
Executing task: 1 , Period Number: 1 , semaphore: 0
****************************************
                Second: 18
Executing task: 1 , Period Number: 1 , semaphore: 0
****************************************
                Second: 19
finished task 1
This time slot, CPU is free :)
****************************************
                Second: 20
Task 2 created. Period Number: 1
Executing task: 2 , Period Number: 1 , semaphore: 1


============================================================
 // Validating the Alghorithm:
         RM-NPP IS feasible for this task set
```

تسک ست ۳)

```json
{
    "startTime":0,
    "endTime":20,
    "taskset":[
        {
            "taskId":1,
            "period":15,
            "wcet":3,
            "deadline":15,
            "offset":1,
            "sections":[[1,1],[0,2]]
        },{
            "taskId":2,
            "period":20,
            "wcet":4,
            "deadline":17,
            "offset":0,
            "sections":[[1,2],[3,2]]
        },{
            "taskId":3,
            "period":10,
            "wcet":4,
            "deadline":12,
            "offset":0,
            "sections":[[2,2],[3,2]]
        }
    ]
}
```

```
****************************************
               Second: 10
Task 3 created. Period Number: 1
Executing task: 2 , Period Number: 0 , semaphore: 3
****************************************
               Second: 11
finished task 2
Executing task: 3 , Period Number: 1 , semaphore: 2
****************************************
               Second: 12
Executing task: 3 , Period Number: 1 , semaphore: 2
****************************************
               Second: 13
Executing task: 3 , Period Number: 1 , semaphore: 3
****************************************
               Second: 14
Executing task: 3 , Period Number: 1 , semaphore: 3
****************************************
               Second: 15
finished task 3
This time slot, CPU is free :)
****************************************
               Second: 16
Task 1 created. Period Number: 1
Executing task: 1 , Period Number: 1 , semaphore: 1
****************************************
               Second: 17
Executing task: 1 , Period Number: 1 , semaphore: 0
****************************************
               Second: 18
Executing task: 1 , Period Number: 1 , semaphore: 0
****************************************
               Second: 19
finished task 1
This time slot, CPU is free :)
****************************************
               Second: 20
Task 2 created. Period Number: 1
Task 3 created. Period Number: 2
Executing task: 3 , Period Number: 2 , semaphore: 2


============================================================
 // Validating the Alghorithm:
         RM-NPP IS feasible for this task set
```

```
F:\Courses\Embedded Real-time Systems\Project>python prj.py
Task Set:
task 1 : (Φ,T,C,D,Δ) = ( 1 , 15 , 3 , 15 , [[1, 1], [0, 2]] )
task 2 : (Φ,T,C,D,Δ) = ( 0 , 20 , 4 , 17 , [[1, 2], [3, 2]] )
task 3 : (Φ,T,C,D,Δ) = ( 0 , 10 , 4 , 12 , [[2, 2], [3, 2]] )
****************************************
               Second: 0
Task 2 created. Period Number: 0
Task 3 created. Period Number: 0
Executing task: 3 , Period Number: 0 , semaphore: 2
****************************************
               Second: 1
Task 1 created. Period Number: 0
Executing task: 3 , Period Number: 0 , semaphore: 2
****************************************
               Second: 2
Executing task: 3 , Period Number: 0 , semaphore: 3
****************************************
               Second: 3
Executing task: 3 , Period Number: 0 , semaphore: 3
****************************************
               Second: 4
finished task 3
Executing task: 1 , Period Number: 0 , semaphore: 1
****************************************
               Second: 5
Executing task: 1 , Period Number: 0 , semaphore: 0
****************************************
               Second: 6
Executing task: 1 , Period Number: 0 , semaphore: 0
****************************************
               Second: 7
finished task 1
Executing task: 2 , Period Number: 0 , semaphore: 1
****************************************
               Second: 8
Executing task: 2 , Period Number: 0 , semaphore: 1
****************************************
               Second: 9
Executing task: 2 , Period Number: 0 , semaphore: 3
****************************************
```

تسک ست ۴ (در اینجا باید infeasible شود):

```
{
    "startTime":0,
    "endTime":20,
    "taskset":[
        {
            "taskId":1,
            "period":15,
            "wcet":3,
            "deadline":15,
            "offset":1,
            "sections":[[1,1],[0,2]]
        },{
            "taskId":2,
            "period":20,
            "wcet":4,
            "deadline":17,
            "offset":0,
            "sections":[[1,2],[3,2]]
        },{
            "taskId":3,
            "period":10,
            "wcet":6,
            "deadline":12,
            "offset":0,
            "sections":[[2,4],[3,2]]
        }
    ]
}
```

خروجی:

```
**************************************
              Second: 11
Executing task: 3 , Period Number: 1 , semaphore: 2
**************************************
              Second: 12
Executing task: 3 , Period Number: 1 , semaphore: 2
**************************************
              Second: 13
Executing task: 3 , Period Number: 1 , semaphore: 2
**************************************
              Second: 14
Executing task: 3 , Period Number: 1 , semaphore: 2
**************************************
              Second: 15
Executing task: 3 , Period Number: 1 , semaphore: 3
**************************************
              Second: 16
Task 1 created. Period Number: 1
Executing task: 3 , Period Number: 1 , semaphore: 3
**************************************
              Second: 17
removing task 2
OH NO... OH NO... OH NO NO NO NO NO... RM-NPP ****FAILED**** !!!!!!
finished task 3
Executing task: 1 , Period Number: 1 , semaphore: 1
**************************************
              Second: 18
Executing task: 1 , Period Number: 1 , semaphore: 0
**************************************
              Second: 19
Executing task: 1 , Period Number: 1 , semaphore: 0
**************************************
              Second: 20
Task 2 created. Period Number: 1
Task 3 created. Period Number: 2
finished task 1
Executing task: 3 , Period Number: 2 , semaphore: 2


============================================================
 // Validating the Alghorithm:
         RM-NPP is NOT feasible for this task set
```

```
F:\Courses\Embedded Real-time Systems\Project>python prj.py
Task Set:
task 1 : (Φ,T,C,D,Δ) = ( 1 , 15 , 3 , 15 , [[1, 1], [0, 2]] )
task 2 : (Φ,T,C,D,Δ) = ( 0 , 20 , 4 , 17 , [[1, 2], [3, 2]] )
task 3 : (Φ,T,C,D,Δ) = ( 0 , 10 , 6 , 12 , [[2, 4], [3, 2]] )
**************************************
              Second: 0
Task 2 created. Period Number: 0
Task 3 created. Period Number: 0
Executing task: 3 , Period Number: 0 , semaphore: 2
**************************************
              Second: 1
Task 1 created. Period Number: 0
Executing task: 3 , Period Number: 0 , semaphore: 2
**************************************
              Second: 2
Executing task: 3 , Period Number: 0 , semaphore: 2
**************************************
              Second: 3
Executing task: 3 , Period Number: 0 , semaphore: 2
**************************************
              Second: 4
Executing task: 3 , Period Number: 0 , semaphore: 3
**************************************
              Second: 5
Executing task: 3 , Period Number: 0 , semaphore: 3
**************************************
              Second: 6
finished task 3
Executing task: 1 , Period Number: 0 , semaphore: 1
**************************************
              Second: 7
Executing task: 1 , Period Number: 0 , semaphore: 0
**************************************
              Second: 8
Executing task: 1 , Period Number: 0 , semaphore: 0
**************************************
              Second: 9
finished task 1
Executing task: 2 , Period Number: 0 , semaphore: 1
**************************************
              Second: 10
Task 3 created. Period Number: 1
Executing task: 2 , Period Number: 0 , semaphore: 1
**************************************
```

در اینجا می‌بینیم که سر واحد زمانی ۱۷، تسک ۲ به ددلاین خود رسید و قبل اتمام کارش، حذف شد. در نتیجه یک deadline برای ما missشده و به همین خاطر RM-NPP برای این تسک ست feasible نیست.

تسک ست ۵)

```
{
    "startTime":0,
    "endTime":20,
    "taskset":[
        {
            "taskId":1,
            "period":15,
            "wcet":3,
            "deadline":15,
            "offset":1,
            "sections":[[1,3]]
        },{
            "taskId":2,
            "period":20,
            "wcet":6,
            "deadline":17,
            "offset":0,
            "sections":[[1,6]]
        }
    ]
}
```

```
              Second: 11
This time slot, CPU is free :)
****************************************
              Second: 12
This time slot, CPU is free :)
****************************************
              Second: 13
This time slot, CPU is free :)
****************************************
              Second: 14
This time slot, CPU is free :)
****************************************
              Second: 15
This time slot, CPU is free :)
****************************************
              Second: 16
Task 1 created. Period Number: 1
Executing task: 1 , Period Number: 1 , semaphore: 1
****************************************
              Second: 17
Executing task: 1 , Period Number: 1 , semaphore: 1
****************************************
              Second: 18
Executing task: 1 , Period Number: 1 , semaphore: 1
****************************************
              Second: 19
finished task 1
This time slot, CPU is free :)
****************************************
              Second: 20
Task 2 created. Period Number: 1
Executing task: 2 , Period Number: 1 , semaphore: 1


==================================================
 // Validating the Alghorithm:
         RM-NPP IS feasible for this task set
```

```
F:\Courses\Embedded Real-time Systems\Project>python prj.py
Task Set:
task 1 : (Φ,T,C,D,Δ) = ( 1 , 15 , 3 , 15 , [[1, 3]] )
task 2 : (Φ,T,C,D,Δ) = ( 0 , 20 , 6 , 17 , [[1, 6]] )
****************************************
              Second: 0
Task 2 created. Period Number: 0
Executing task: 2 , Period Number: 0 , semaphore: 1
****************************************
              Second: 1
Task 1 created. Period Number: 0
Executing task: 2 , Period Number: 0 , semaphore: 1
****************************************
              Second: 2
Executing task: 2 , Period Number: 0 , semaphore: 1
****************************************
              Second: 3
Executing task: 2 , Period Number: 0 , semaphore: 1
****************************************
              Second: 4
Executing task: 2 , Period Number: 0 , semaphore: 1
****************************************
              Second: 5
Executing task: 2 , Period Number: 0 , semaphore: 1
****************************************
              Second: 6
finished task 2
Executing task: 1 , Period Number: 0 , semaphore: 1
****************************************
              Second: 7
Executing task: 1 , Period Number: 0 , semaphore: 1
****************************************
              Second: 8
Executing task: 1 , Period Number: 0 , semaphore: 1
****************************************
              Second: 9
finished task 1
This time slot, CPU is free :)
****************************************
              Second: 10
This time slot, CPU is free :)
****************************************
```

در اینجا چون هر تسک ۱ سکشن دارد، با شروع هر سکشن، به علت npp، باعث میشود تسک تا انتها برود.

اضافه و یا کم کردن offset نیز باعث تاخیر در شروع کار یک تسک شده. نمیتوان به جرعت گفت باعث بهبود و یا بدتر کردن کار خواهد شد یا خیر.

با تشکر از توجه شما