# 算法分析第六章作业

2. 工作分配问题。设有 n 件工作需要分配给 n 个人去完成。将工作 i 分配给第 j 个人完成所需要的费用为 $c_{ij}$。试设计一个算法，为每一个人分配一件不同的工作，并使总费用达到最小。

见附件 WorkAssignment.py

```python
WorkAssignment.py > ...
def dfs(current_cost, work_num, work_list, assigned_work, min_assigned_work):
    return

    for j in range(len(work_list[work_num-1])): # work_list[i][j] 表示第 i 件工作分配给第 j 个人的费用
        if assigned_work[j] == -1: # 如果第 j 个人还没有分配工作
            assigned_work[j] = work_num-1
            dfs (current_cost + work_list[work_num-1][j], work_num-1, work_list, assigned_work, min_assigned_work)
            assigned_work[j] = -1

    return min_cost

if __name__ == '__main__':
    n = 3
    work_list = [
        [1, 2, 3],
        [4, 5, 2],
        [3, 1, 4]
    ]
    min_assigned_work = [-1] * n
    min_cost = 0
    for i in range(n):
        min_cost += work_list[i][i]
    dfs(0, n, work_list,  [-1] * n, min_assigned_work)
    print(min_assigned_work)
    for i in range(n):
        print(f"第 {i+1} 个人分配的工作为 {min_assigned_work[i]}, 费用为 {work_list[min_assigned_work[i]][i]}")
    print("总费用为", min_cost) # 输出结果为 4
```

```
PROBLEMS  OUTPUT  TERMINAL  DEBUG CONSOLE  PORTS                Python

PS D:\研究生\研一上课程\12-26算法分析与设计\作业程序> cd d:/研究生/研一上课程/12-26算法分析与设计/作业程序
PS D:\研究生\研一上课程\12-26算法分析与设计\作业程序> & D:/ProgramFilesFolder/05-Anaconda3/python.exe d:/研究生/研一上课程/12-26算法分析与
程序/WorkAssignment.py
[0, 2, 1]
第 1 个人分配的工作为 0, 费用为 1
第 2 个人分配的工作为 2, 费用为 1
第 3 个人分配的工作为 1, 费用为 2
总费用为 4
```

3. 最佳调度问题。假设有 n 个任务要由 k 个可并行工作的机器来完成。完成任务 i 需要的时间为 $t_i$。试设计一个算法找出完成这 n 个任务的最佳调度，使得完成全部任务的结束时间最早。

见附件 BestManagePlan.py

给出了调度，由于同一个机器上的调度先后顺序并不重要，所以给出每个任务在第几号机器上执行即可

结束时间为 k 个机器的完成时间的 max

BestManagePlan.py > ...

```python
# 假设有 n 个任务要由 k 个可并行工作的机器来完成。完成任务 i 需
# 要的时间为 ti。试设计一个算法找出完成这 n 个任务的最佳调度，使得完成全部任务的结束
# 时间最早。

n = 5
k = 3
task_time_list = [3, 2, 4, 1, 5]
task_plan_list = [-1] * n
k_time_list = [0] * k
min_total_time = sum(task_time_list)

def dfs(current_task, k_time_list, task_time_list, total_time, current_plan_list):
    global min_total_time

    # print(current_task)
    if total_time >= min_total_time: # 剪枝
        return

    if current_task == n:
        # print(total_time)
        if total_time < min_total_time:
            min_total_time = total_time
            task_plan_list[:] = current_plan_list[:]
        return

    current_task_time = task_time_list[current_task]
    for i in range(0, k):
        current_plan_list[current_task] = i
        k_time_list[i] += current_task_time
        dfs (
            current_task + 1,
            k_time_list
```

PROBLEMS OUTPUT TERMINAL DEBUG CONSOLE PORTS Pytho

```
PS D:\研究生\研一上课程\12-26算法分析与设计\作业程序> cd d:/研究生/研一上课程/12-26算法分析与设计/作业程序
PS D:\研究生\研一上课程\12-26算法分析与设计\作业程序> & D:/ProgramFilesFolder/05-Anaconda3/python.exe d:/研究生/研一上课程/
程序/BestManagePlan.py
[0, 0, 1, 1, 2]
5
                                                    & D:/ProgramFilesFolder/05-Anaconda3/python.exe d:/研究生/研一上课程/
程序/BestManagePlan.py \12-26算法分析与设计\作业程序>
[0, 0, 1, 1, 2]
5
```

另外，可以用贪心算法，先对每个任务的执行时间进行排序，从大到小排序，再将任务分配给当前最早空闲的机器，时间复杂度 O(nlogn + n*k)。

5. 最小重量机器设计问题。设某一机器由 n 个部件组成，每一种部件都可以从 m 个不同的供应商处购得。设 $w_{ij}$ 是从供应商 j 处购得的部件 i 的重量，$c_{ij}$ 是相应的价格。试设计一个算法，给出总价格不超过 c 的最小重量机器设计。

dp[i][j]表示到第 i 个部件花费 j 的部件重量的最小值，时间复杂度为 O(nmc)
代码见 MinWeightMechine-DP.py

DFS 的做法：
dfs(i, current_cost)
对于第 i 个部件，遍历供应商，对于每个供应商手里的部件 i 有拿或不拿两种
代码见 MinWeightMechine-DFS.py
代码执行见下图：

WorkAssignment.py | MinWeightMechine-DP.py | MinWeightMechine-DFS.py

MinWeightMechine-DFS.py > ...

```python
    [1, 2, 3],
    [5, 1, 7]
]

weights = [
    [1, 2, 3],
    [4, 1, 6],
    [7, 3, 9],
    [5, 5, 2]
]
min_weight = sum(weights[i][j] for i in range(n) for j in range(m))
total_cost = c

def dfs(index, current_weight, current_cost):
    global min_weight, total_cost

    if current_cost > total_cost:
        return

    if index == n:
        if current_weight < min_weight:
            min_weight = current_weight
        return

    for j in range(m):
        dfs(index+1, current_weight+weights[index][j], current_cost+prices[index][j])

dfs(0, 0, 0)
print(min_weight)
```

PROBLEMS OUTPUT TERMINAL DEBUG CONSOLE PORTS

```
10
PS D:\研究生\研一上课程\12-26算法分析与设计\作业程序> cd d:/研究生/研一上课程/12-26算法分析与设计/作业程序
PS D:\研究生\研一上课程\12-26算法分析与设计\作业程序> & D:/ProgramFilesFolder/05-Anaconda3/python.exe d:/研究生/研一上课程/12-26算法分析与设计/作业程序/MinWeightMechine-DFS.py
10
```