**暨南大学本科实验报告专用纸**

课程名称 算法分析与设计实验 成绩评定

实验项目名称 最小重量机器设计 指导教师 李展

实验项目编号 实验X 实验项目类型 综合性 实验地点

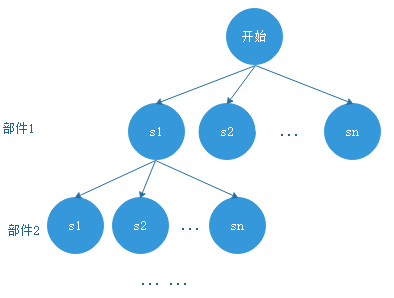
学生姓名 张印祺 学号 2018051948

学院 信息科学技术 系 计算机科学 专业 网络工程

实验时间 2020 年 5 月 20日

1. 问题描述

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。设是供应商j处购得的部件i的重量，cij是相应的价格。试设计一种算法，给出总价格不超过c的最小重量机器设计。

1. 算法思路

在算法开始时将部件一对应的所有供应商加入队列，然后对队列中进行以下操作：

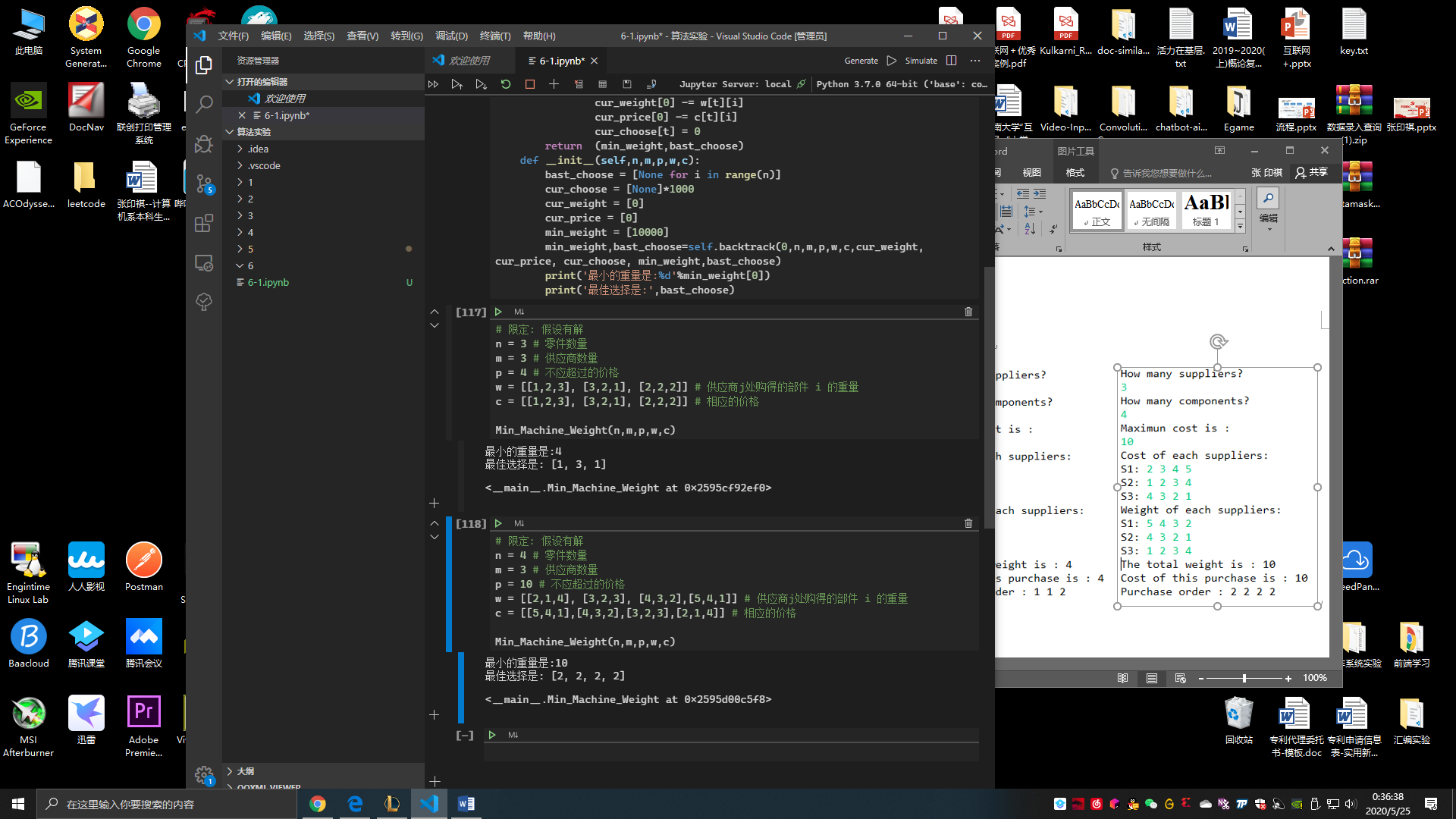
利用回溯法求解问题，首先可以为该题的二维数组w、c、n、m、p赋初值，构造回溯法中的解空间树

根据回溯法的要求，进行的是深度遍历，同时满足重量尽可能小，价格不能超过p。

1. 流程图



1. 测试结果



1. 实验总结

利用分支限界法，每次选择子结点都要进行分支判断。

假设不考虑价格限制，每个部件在哪购买都是相互独立的，对于每个部件只需要考虑提供最轻部件的供应商。考虑上价格限制后，除了考虑质量最轻，还要考虑购买这个部件后是否还有足够的经费购买其他部件。因此要先记录总的最低经费，购买完一个部件后便比较一次剩余经费是否低于剩余最低经费需求，如果是，则经费不足以购买全部部件，该结点不进行展开。

T(n) = o(。

1. 源代码

class Min\_Machine\_Weight():

    def backtrack(self,t,n,m,p,w,c,cur\_weight, cur\_price, cur\_choose, min\_weight,best\_choose):

    # 此处的t代表每一次遍历i供应商的零件

    #    global cur\_weight, cur\_price, cur\_choose, min\_weight, p, w, c

        if t == n:

            # 遍历到叶子结点

            if cur\_weight[0] < min\_weight[0]:

                min\_weight[0] = cur\_weight[0]

                for j in range(n):

                    if cur\_choose[j]==None:

                        cur\_choose[j]=0

                    best\_choose[j] = cur\_choose[j]+1

                return

        else:

            for i in range(m): # 遍历供应商

                if cur\_choose[t]==None:

                    cur\_choose[t]=0

                cur\_choose[t] = i

                cur\_weight[0] += w[t][i]

                cur\_price[0] += c[t][i]

                if cur\_weight[0] < min\_weight[0] and cur\_price[0] <= p:

                    # 该供应商的重量小于局部最优解 同时价格满足要求 则遍历其子树

                    self.backtrack(t+1,n,m,p,w,c,cur\_weight, cur\_price, cur\_choose, min\_weight,best\_choose)

                cur\_weight[0] -= w[t][i]

                cur\_price[0] -= c[t][i]

                cur\_choose[t] = 0

        return  (min\_weight,best\_choose)

    def \_\_init\_\_(self,n,m,p,w,c):

        best\_choose = [None for i in range(n)]

        cur\_choose = [None]\*1000

        cur\_weight = [0]

        cur\_price = [0]

        min\_weight = [10000]

        min\_weight,best\_choose=self.backtrack(0,n,m,p,w,c,cur\_weight, cur\_price, cur\_choose, min\_weight,best\_choose)

        print('最小的重量是:%d'%min\_weight[0])

        print('最佳选择是:',best\_choose)