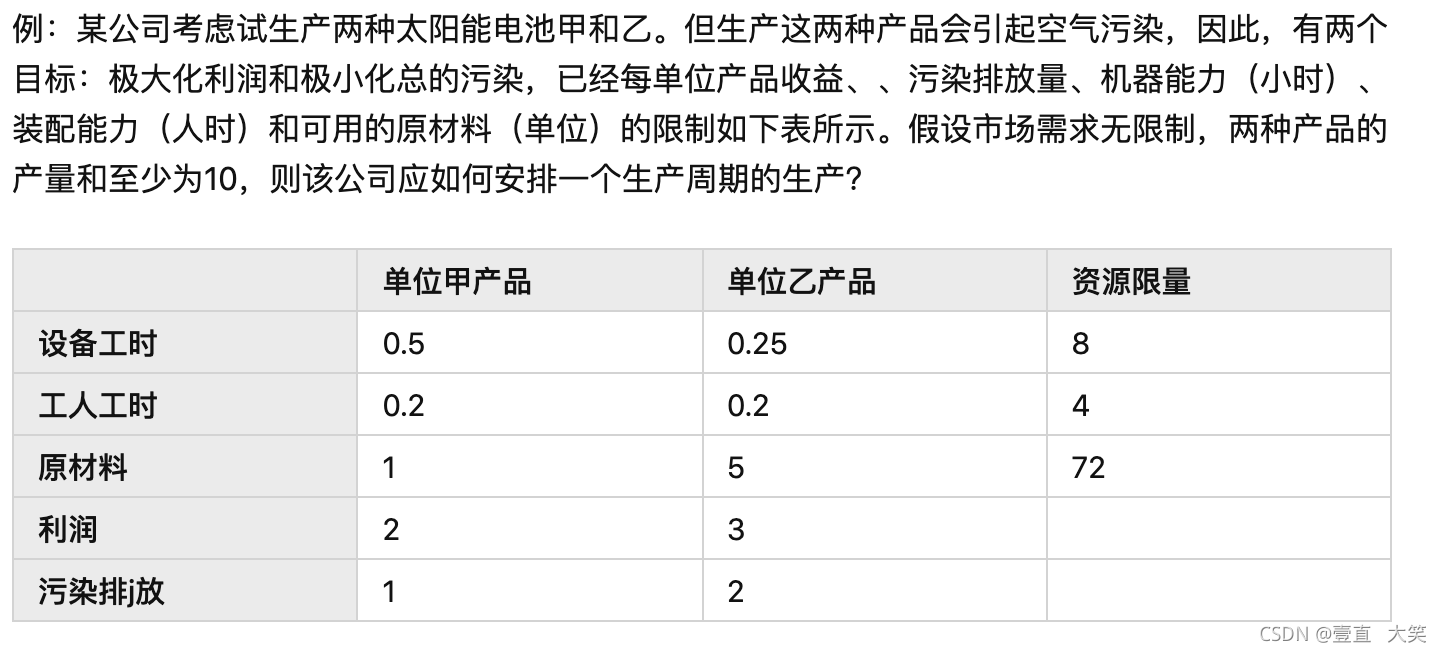
多目标规划

例题：



import numpy as np  
import cvxpy as cp  
  
c1 = np.array([-2, -3])  
c2 = np.array([1, 2])  
a = np.array([[0.5, 0.25], [0.2, 0.2], [1, 5], [-1, -1]])  
b = np.array([8, 4, 72, -10])  
x = cp.Variable(2, pos=True)  
  
# 1.线性加权法求解  
obj = cp.Minimize(0.5 \* (c1 + c2) @ x)  
con = [a @ x <= b]  
prob = cp.Problem(obj, con)  
prob.solve(solver='GLPK\_MI')  
print('\n======1.线性加权法======\n')  
print('解法一理想解：', x.value)  
print('利润：', -c1 @ x.value)  
print('污染排放：', c2 @ x.value)  
  
# 2.理想点法求解  
obj1 = cp.Minimize(c1 @ x)  
prob1 = cp.Problem(obj1, con)  
prob1.solve(solver='GLPK\_MI')  
v1 = prob1.value # 第一个目标函数的最优值  
obj2 = cp.Minimize(c2 @ x)  
prob2 = cp.Problem(obj2, con)  
prob2.solve(solver='GLPK\_MI')  
v2 = prob2.value # 第二个目标函数的最优值  
print('\n======2.理想点法======\n')  
print('两个目标函数的最优值分别为：', v1, v2)  
obj3 = cp.Minimize((c1 @ x - v1) \*\* 2 + (c2 @ x - v2) \*\* 2)  
prob3 = cp.Problem(obj3, con)  
prob3.solve(solver='CVXOPT') # GLPK\_MI 解不了二次规划，只能用CVXOPT求解器  
print('解法二的理想解：', x.value)  
print('利润：', -c1 @ x.value)  
print('污染排放：', c2 @ x.value)  
  
# 3.序贯法求解  
con.append(c1 @ x == v1)  
prob4 = cp.Problem(obj2, con)  
prob4.solve(solver='GLPK\_MI')  
x3 = x.value # 提出最优解的值  
print('\n======3.序贯法======\n')  
print('解法三的理想解：', x3)  
print('利润：', -c1 @ x3)  
print('污染排放：', c2 @ x3)

