BZOJ2424:[HAOI2010]订货 题解

GodCowC

December 8, 2015

题目

Description

某公司估计市场在第i个月对某产品的需求量为U_i，已知在第i月该产品的订货单价为d_i，上个月月底未销完的单位产品要付存贮费用m，假定第一月月初的库存量为零，第n月月底的库存量也为零，问如何安排这n个月订购计划，才能使成本最低？每月月初订购，订购后产品立即到货，进库并供应市场，于当月被售掉则不必付存贮费。假设仓库容量为S。

Input

第1行：$n, m, S(1 \leq n \leq 50, 0 \leq m \leq 10, 0 \leq S \leq 10000)$
第2行：$U_1, U_2, \cdots, U_i, \cdots, U_n(0 \leq U_i \leq 10000)$
第3行：$d_1, d_2, \cdots, d_i, \cdots, d_n(0 \leq d_i \leq 100)$

Output

只有1行，一个整数，代表最低成本

SampleInput

```
3 1 1000
2 4 8
1 2 4
```

SampleOutput

```
34
```

Hint

Source

Day1

题解

这是一道费用流练手题，唯一的难点在于建图。我们可以把每个月抽象为一个点，即第i个点表示第i个月，再增加一个源点s和一个汇点t，对于题中的信息，我们可以抽象成如下模型：

对于需求，我们让第i个点到汇点t的流量限制为U_i，单位费用为0；对于订购，我们让源点s到第i个点的流量限制为$+\inf$，单位费用为d_i；对于存贮，我们让第i个点到第$i+1$个点的流量限制为S，单位费用为m。

之后，我们就可以跑一遍费用流了。这里的模型构建还是很容易的，毕竟费用的定义就是生活中的费用，流量恰好对应了货物数量。

代码

```
#include <iostream>
#include <cstdio>
#include <cctype>
#include <cmath>
#include <algorithm>
#define MAXN 60
#define INF 100000000
using namespace std;
int readint()
{
    int ans=0;
    char c;
    while (!isdigit(c=getchar()));
    do
    {
        ans=ans*10+c-'0';
        c=getchar();
    } while (isdigit(c));
    return ans;
}
int n,m,S;
int s,t;
int ans;
struct Node
{
    int pre,dis,delta;
    Node(int p=0,int d=0,int x=0):pre(p),dis(d),delta(
       x){}
};
struct Edge
{
    int flow,cap,fee;
```

```
    Edge(int f=0,int c=0,int fee=0):flow(f),cap(c),fee
        (fee){}
};
Node argpath[MAXN];
Edge graph[MAXN][MAXN];
bool find()
{
    for (int i=1;i<=t;i++)
        argpath[i]=Node(0,INF,INF);
    argpath[s]=Node(0,0,INF);
    bool quit;
    do
    {
        quit=true;
        for (int i=1;i<=t;i++)
            if (argpath[i].dis<INF)
                for (int j=1;j<=t;j++)
                    if (graph[i][j].flow<graph[i][j].
                        cap)
                        if (argpath[i].dis+graph[i][j
                            ].fee<argpath[j].dis)
                        {
                            argpath[j].dis=argpath[i].
                                dis+graph[i][j].fee;
                            argpath[j].pre=i;
                            argpath[j]= Node(i,argpath
                                [i].dis+graph[i][j].fee
                                ,min(argpath[i].delta,
                                graph[i][j].cap-graph[i
                                ][j].flow));
                            quit=false;
                        }
    } while (!quit);
    return (argpath[t].dis<INF);
}
void change()
{
    int i=t;
    while (i!=s)
    {
        int j=argpath[i].pre;
        graph[j][i].flow+=argpath[t].delta;
        graph[i][j].flow=-graph[j][i].flow;
        i=j;
    }
    ans+=argpath[t].dis*argpath[t].delta;
```

```
}
int main()
{
    n=readint();
    m=readint();
    S=readint();
    s=n+1,t=n+2;
    for (int i=1;i<n;i++)
    {
        graph[i][i+1]=Edge(0,S,m);
        graph[i+1][i]=Edge(0,0,-m);
    }
    for (int i=1;i<=n;i++)
    {
        int u=readint();
        graph[i][t]=Edge(0,u,0);
        graph[t][i]=Edge(0,0,0);
    }
    for (int i=1;i<=n;i++)
    {
        int d=readint();
        graph[s][i]=Edge(0,INF,d);
        graph[i][s]=Edge(0,0,-d);
    }
    ans=0;
    while (find())
        change();
    cout<<ans<<endl;
    return 0;
}
```