**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 二维0-1背包问题 指导教师 李展

实验项目编号 实验七 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 4 月 29 日

1. 问题描述

给定n个物品和一个背包。物品i的重量是wi，体积是bi，其价值为vi，背包容量为c，容积为d。问应如何选择装入背包中的物品，使得装入背包中物品的总价值最大？其中物品只能选择放和不放。

1. 算法思路

该问题可以迅速列出限制条件：

s.t. ;

容易证明该问题具有最优子结构；可以得到状态迁移方程

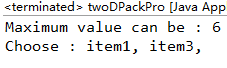
当每件物品只可以取一次时变量j和k采用逆序的循环，当物品有如完全背包问题时采用顺序的循环，当物品有如多重背包问题时拆分物品。

1. 测试结果

1.c = 7, d = 8, w = {2,1,4,1,4}, k = {3,4,6,1,2}, v = {6,5,7,3,8}



2.c = d = 6, w = k = {4,3,2}, v = {5,2,1}



1. 实验总结

本算法与一维的0-1背包问题相比，采取了三维的数组来当备忘录。

根据状态迁移方程，可以得到算法的递归式：

由此可以算出m(n,c,d)的最优值解。时间复杂度为O(ncd)。

要算出物品的选择，需要调用traceBack算法；同一维0-1背包问题，m[1,c,d]的值为问题的最优值，然后每次分别j-wi、k-bi，逐渐递减算出选择了哪些物品。时间复杂度为O(c+d+n)。

六、附录 （程序代码）

**public** **class** twoDPackPro {

**public** **void** twoDPro(**int** n, **int** c, **int** d, **int**[] val, **int**[] vol, **int**[] wei, **int**[][][]m) {

**for**(**int** i = vol[n]; i <= c; i ++)

**for**(**int** j = wei[n]; j <= d; j ++) {

m[n][i][j] = val[n];

}

**for**(**int** i = n - 1; i > 0; i --) {

**int** jMax = Math.*min*(vol[i] - 1, c);

**int** wMax = Math.*min*(wei[i] - 1, d);

**for**(**int** j = 1; j <= jMax; j ++)

**for**(**int** w = 1; w <= d; w ++)

m[i][j][w] = m[i + 1][j][w];

**for**(**int** j = jMax; j <= c; j ++)

**for**(**int** w = 1; w <= wMax; w ++)

m[i][j][w] = m[i + 1][j][w];

**for**(**int** j = vol[i]; j <= c; j ++)

**for**(**int** w = wei[i]; w <= d; w ++)

m[i][j][w] = Math.*max*(m[i + 1][j][w], m[i + 1][j - vol[i]][w - wei[i]] + val[i]);

}

}

**void** traceBack(**int**[][][]m, **int**[] x, **int**[]vol, **int**[]wei, **int** n, **int** maxVol, **int** maxWei) {

**for**(**int** i = 1; i < n; i ++) {

**if**(m[i][maxVol][maxWei] == m[i+1][maxVol][maxWei]) x[i] = 0;

**else** {

x[i] = 1;

maxVol -= vol[i];

maxWei -= wei[i];

}

}

x[n] = (m[n][maxVol][maxWei] > 0)? 1: 0;

}

**public** **static** **void** main(String[] args) {

**int** maxVol = 7;

**int** maxWei = 8;

**int**[] vol = {0, 2,1,4,1,4};

**int**[] wei = {0, 3,4,6,1,2};

**int**[] val = {0, 6,5,7,3,8};

**int**[][][] m = **new** **int**[vol.length][maxVol+1][maxWei+1];

**int**[] x = **new** **int**[vol.length];

**new** twoDPackPro().

twoDPro(vol.length-1,maxVol,maxWei,val,vol,wei,m);

**new** twoDPackPro().

traceBack(m,x,vol,wei,vol.length-1,maxVol,maxWei);

System.***out***.println("Maximum value can be : " + m[1][maxVol][maxWei]);

System.***out***.print("Choose :");

**for**(**int** i = 1; i < val.length; i ++)

**if**(x[i] > 0) System.***out***.print(" item" + i +",");

}

}