



***Liaoning Normal University***

**算法设计分析实验报告**

**课程名称：计算机算法设计分析**

**专业：计算机科学与技术（师范）**

**班级： （1） 班**

**学号：201621012299**

**姓名：周惠馨**

**指导教师：王相海**

**2018年11月18日**

1. **问题的提出**

给定n种物品和一背包。物品i的重量是wi，体积是bi，其价值为vi，背包的容量为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？在选择装入背包的物品时，对每种物品i只有两种选择，即装入背包或不装入背包。不能将物品i装入背包多次，也不能只装入部分的物品i。试设计一个解此问题的动态规划算法，并分析算法的计算复杂性。

1. **算法的基本思想**

动态规划有两种基本思路，一种是自顶而下的备忘录算法，它采用递归的方式，一步步缩小问题的规模，找到边界(n==0)以后，处理好边界（bestP = (c >= W[n]) ? P[n] : 0;），然后一步步翻译，返回更大问题的解。每获得一个子问题的答案，便保存到备忘录数组p[n][c]，避免充分计算提升了效率。另一种思路是自底向上的动态规划算法，它从最小的子问题（只包含0号物品）开始处理，然后依次增加可装物品的数量，用数组p[n][c]记录每一个已处理子问题的答案，由于0-1背包问题具有最优子结构，每个问题的最优解都是由其子问题的最优解组成，故这样依次递增问题的规模，可以得到最终需要的最优解。

1. **算法的程序实现**

#include "stdio.h"

#define n 4

int w[n],v[n],a[n],c;

float t[n],s;

void sort()

{

int i,j,b,m;

for(i=0;i<n;i++) a[i]=i;

for(i=0;i<n;i++)

{

b=i;

for(j=i+1;j<=n;j++)

if(t[b]<t[j])

b=j; //用b指出每趟在无序区段的最大元素

s=t[i];

t[i]=t[b];

t[b]=s;

m=a[i];a[i]=a[b];a[b]=m;

}

}

int main()

{

int i,j,sc;

float sumv;

printf("请输入物品的重量:\n");

for(i=0;i<n;i++)

scanf("%d",&w[i]);

printf("请输入物品的价值:\n");

for(i=0;i<n;i++)

scanf("%d",&v[i]);

printf("请输入背包的容量:\n");

scanf("%d",&c);

for(i=0;i<n;i++)

t[i]=(float)v[i]/(float)w[i];

for(i=0;i<n;i++)

printf("%f ",t[i]);

printf("\n");

sort();

sumv=0;

sc=c;

for(i=0;i<n;i++)

{

if(sc>=w[a[i]])

{

sc-=w[a[i]];

sumv=sumv+v[a[i]];

}

else

{

if(sc<w[a[i]])

break;

}

}

sumv=sumv+v[a[i]]\*sc/w[a[i]];

printf("背包中物品的最大价值为：%f",sumv);

getch();

return 0;

}

1. **算法的复杂度分析**

无论是自顶向下的备忘录算法还是自底向上的动态规划算法，都可以在O(n^3)时间内求解。他们有O(n^2)个备忘记录项（或子问题），这些记录项的初始化耗费O(n^2)时间。每个记录项只填入一次，每次填入时，耗费O(n)时间。因而填入O(n^2)各记录项总共耗费O(n^3)时间。对每个子问题，两种方法都只解一次，并记录答案，再碰到该子问题时，不重新求解而是取用已有答案。它们节省了计算量，提高了算法的效率。自底向上的动态规划算法按顺序计算了每个子问题的解；自顶向下的备忘录算法采用递归的方式只计算了确实需要求解的子问题，效率更高（当然，由于需要调用递归函数，自有了一些额外的开销），也更便于理解。

1. **算法运行结果与分析（拷屏黏贴）**





1. **结束语（讨论）**

在初始化时除了f[0]为0其它f[1...v]均设为-∞，这样就可以保证最终得到的f[N]是一种恰好装满背包的最优解。如果不能恰好满足背包容量，即不能得到f[v]的最优值，则此时f[v]=-∞，这样就能表示没有找到恰好满足背包容量的最优值。求小于等于背包容量的最优解，即不一定恰好装满背包；如果并没有要求必须吧背包装满，二是只希望价值尽量大，初始化时应该将f[0...v]全部设为0。