**华南农业大学**

**《算法分析与设计》课程实验**

**实验7：0-1背包问题**

年级专业： 21信息与计算科学

学生姓名： 吴嘉豪

学生学号： 202125810321

指导老师： 赵峰

实验时间： 2024年05月20日

一、实验内容

用回溯法与分支限界法解决0-1背包问题，比较并分析试验结果。

二、实验步骤

1、编写回溯法与分支限界法求解0-1背包问题的程序；

2、用测试样例测试算法的正确性；

3、随机生成5、10、20、30、40个物品的数据（重量与价值），设置适当的背包承重，分别用两种算法求解，记录运行时间与访问的结点个数；

4、比较两种算法的运行结果，尝试对比较结果进行解释；

5、填写实验报告。

三、实验结果

2

根据该测试用例易知， 算法正确性

文本

描述已自动生成

4

对于这两个算法我都进行了剪枝， 当前状态的重量如果大于最大背包容量就必须要进行判断下去， 但是我在实现因为 分支限界法 的过程中， 我是将新的元素先反正nq队列中， 然后再赋值给q队列。 这导致了分支限界法稍慢。

同时对于回溯法，来说， 我只使用了四个整形变量 index，currentValue， TotalValue，currentHeavey。（不涉及数组的拷贝）。 来维护 这也是他快的原因

文本

描述已自动生成

文本

描述已自动生成

四、源程序代码

#*include*<iostream>

#*include*<string>

#*include*<algorithm>

#*include*<ctime>

#*include*<functional>

#*include*<queue>

#*include*<vector>

using namespace std;

int n;

void *solve*(){

    vector<int> *value*(n);

    vector<int> *price*(n);

*for*(int i = 0; i < n; ++i) value*[*i*]* = *rand*() % 10 + 5;

*for*(int i = 0; i < n; ++i) price*[*i*]* = *rand*() % 10 + 1;

    int capcity = *rand*() % 50 + 20;

    long long startclock = 0;

    long long endclock = 0;

    startclock = *clock*();

// *选择的物品*

    vector<int> chose;

    int totalValue = 0;

    int currentValue = 0;

    int currentHeavy = 0;

    function<void(int)> dfs = [&](int index){

*if*(currentHeavy > capcity) *return* ;

*if*(index == n){

            totalValue = *max*(totalValue, currentValue);

*return* ;

        }

*dfs(*index + 1*)*;

        currentValue += price*[*index*]*;

        currentHeavy += value*[*index*]*;

*dfs(*index + 1*)*;

        currentHeavy -= value*[*index*]*;

        currentValue -= price*[*index*]*;

    };

*dfs(*0*)*;

    endclock = *clock*();

    cout*<<*"现有物品重量为"*<<endl*;

*for*(int i = 0; i < n; ++i) cout*<<*value*[*i*]<<*" ";

    cout*<<endl*;

    cout*<<*"现有物品的价值为"*<<endl*;

*for*(int i = 0; i < n; ++i) cout*<<*price*[*i*]<<*" ";

    cout*<<endl*;

    cout*<<*"背包可容纳重量为: "*<<*capcity*<<endl*;

    cout*<<*"---------------对于回溯算法来说-------------"*<<endl*;

    cout*<<*"对于回溯算法来说， 最优的值为："*<<*totalValue*<<endl*;

    cout*<<*"花费的时间为："*<<*(endclock - startclock)*<<endl*;

    currentValue = 0;

    currentHeavy = 0;

    totalValue = 0;

    startclock = *clock*();

    queue<pair<int,int>>q;

    q.*push*(*make\_pair*(0, 0));

*for*(int i = 0; i < n; ++i){

        int v = value*[*i*]*;

        int p = price*[*i*]*;

        queue<pair<int,int>>nq;

*while*(q.*size*()){

// *可以拿也可以不拿*

            auto pItem = q.*front*();

            nq.*push*(pItem);

            pItem.first += v;

            pItem.second += p;

*if*(pItem.first <= capcity){

                nq.*push*(pItem);

            }

            q.*pop*();

        }

        q *=* nq;

    }

*while*(q.*size*() > 0){

        auto p = q.*front*();

        q.*pop*();

        totalValue = *max*(totalValue, p.second);

    }

    endclock = *clock*();

    cout*<<*"---------------对于分支限界法来说-------------"*<<endl*;

    cout*<<*"对于分支限界法来说, 最优值为: "*<<*totalValue*<<endl*;

    cout*<<*"花费的时间为："*<<*(endclock - startclock)*<<endl*;

}

int *main*(){

    int t = 5;

*srand*(*time*(nullptr));

    int testN[5] = {5, 10, 20, 30, 40};

*while*(t--) {

        n = testN[4 - t];

*solve*();

    }

*return* 0;

}

五、评语及评分

|  |
| --- |
| 评语及评分  评阅人签名： |