**深 圳 大 学 实 验 报 告**

**课程名称： 算法分析与设计**

**实验项目名称： 背包问题**

**学院： 计算机与软件学院**

**专业： 软件工程**

**指导教师： 李荣华**

**报告人： 洪继耀 学号： 2014150120 班级： 02**

1. 实验目的

（1） 掌握动态规划算法设计思想。

（2） 掌握背包问题的动态规划解法。

1. 实验内容
2. 算法综述以及01背包问题的分阶段。

在求解多阶段决策的动态过程的优化问题， 我们常用动态规划算法，它的核心是：

**多阶段决策问题，求解的问题可以划分为一系列相互联系的阶段，在每个阶段都要做出决策，并且上一个阶段的决策的选择会影响下一个阶段的决策。我们分阶段地去求解，且记下每一个阶段的最优解，以后阶段的解都依赖前面阶段的解。**

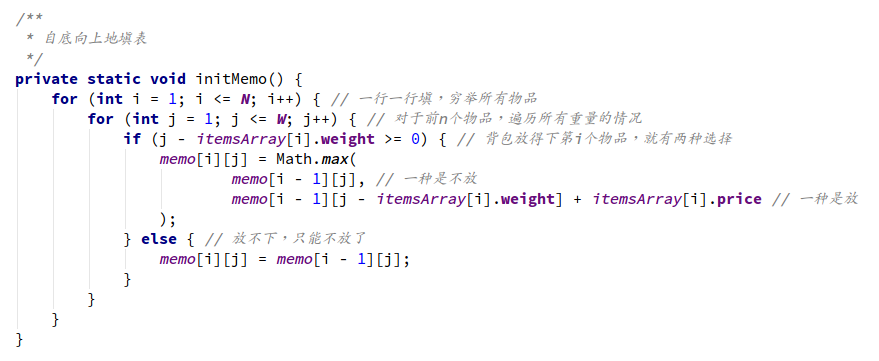
适用动态规划算法的问题的核心都是**分阶段。**

对于0-1背包问题，我们可以这样分阶段：

* 1. 第一阶段：考虑只放入1号物品，对于不同的背包承重(从1到W)，求最大价值，记录在备忘录中;
  2. 第二阶段：只考虑1号和2号物品，对于不同的背包承重，按照一定方法，求最大价值，记录在备忘录中；该问题求解在已知上个阶段最优解的基础之上进行，这个一定方法就是我们的状态转移方程（后面说）
  3. 以此类推，直至第n阶段。

1. 核心算法-填表。

2.1 以下给出算法的核心代码之一——填表：



memo[i][j]的意义是，只考虑前i件物品且背包重量为j时背包可背负的最大价值。

此处使用**带备忘录的自底向上方法，**计算memo数组（备忘录）中所有值。

2.2 状态转移方程：

我们用w[i]表示i的重量 v[i]表示其价值

在第i阶段，我们考虑第i个物品的时候，只有两种选择——放进背包，或者不放进背包。

于是我们考虑i-1阶段的最优解，从这个最优解出发：

A.此时当背包承重不足以放下当前物品，那么当然只能不放；

B.而当背包承重足够时，我们有放或不放两种选择：假设放i物品放入包中，那么背包负重就减少该物品的重量(j-w[i])，并且总价值增加该物品的价值，**而对于放进i后的背包的剩余空间，也就是一个容量为j-w[i]的背包，它能容纳的最高价值我们已经计算过了，我们只需比较物品i的价值和剩余部分的最高价值的和，以及不放i的情况下背包的最大价值，这两个价值哪个大，就可以知道放这个物品到底是有益还是有害。**

也就是说，新的最优解，要么是c[i-1][j-w[i]] + v[i]，也就是我们选择放进了物品i，并根据剩余的质量求出剩下部分的最优解，两值相加

要么是 c[i-1][j] 不放物品i 依旧是上个阶段的最优解

也就是说，状态转移的方程是

memo[i][j] = Max ( c[i-1][j-w[i]] + v[i] , c[i-1][j] )

2.3 例子

以一个n=5,w=10的背包问题作为例子，其生成的备忘录如下，memo[5][10]即为原问题的解：

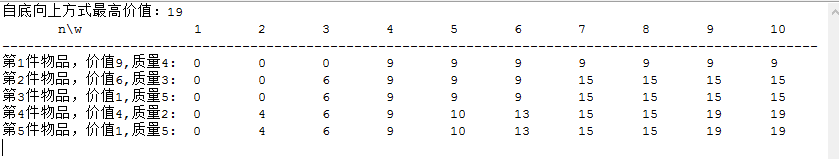
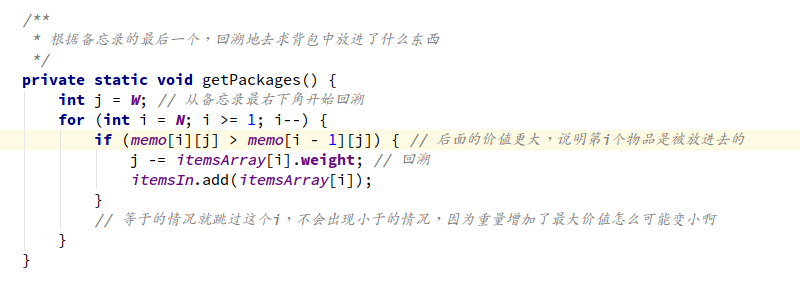


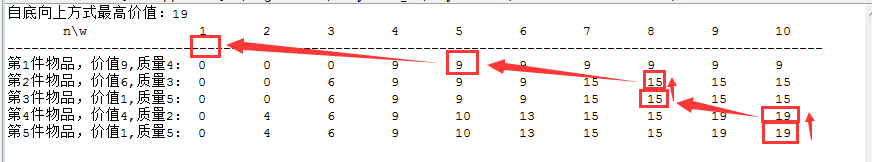
Figure 1 W=10,N=5的某背包问题

1. 如何知道最优解背包中包含了哪些物品

3.1以下给出算法的核心代码之二——回溯求物品：



3.2 算法解释



首先观察上表，可以看出右下角memo[5][10]为最高价值，那么假设我们在第10列往上看:

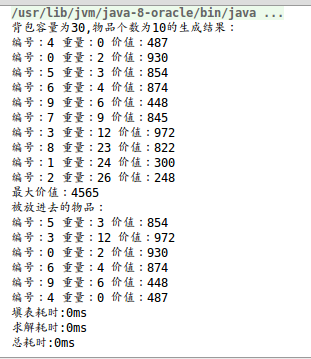
1. 如果memo[4][10]与memo[5][10]相等——意味着第五件物品没有被选择，或者有某件物品与之等价
2. 如果memo[5][10] 大于memo[4][10]——意味着考虑了第五个物品以后，承重为10的背包最高价值大于只考虑了前4个物品的最高价值，也就是这个物品被放入了背包。
3. 在此例中，memo[5][10]=memo[4][10],说明第五个物品没有被选择，接着同样的，比较memo[4][10]与memo[3][10]，发现memo[4][10]>memo[3][10]，意味着第四件物品被选择了，接下来，我们既然知道第四件物品被选择了，意味着背包容量少了v[4]，因而应该去到memo[3][w-v[4]]即memo[3][8]处继续往上回溯。

D.从而我们知道，第4,2,1个物品被选择了

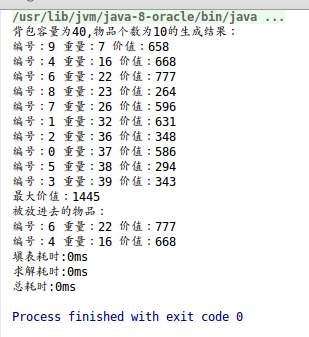
1. 随机生成n个物品的体积与价值，并对给出的三组数据进行验证。

**第一组：**

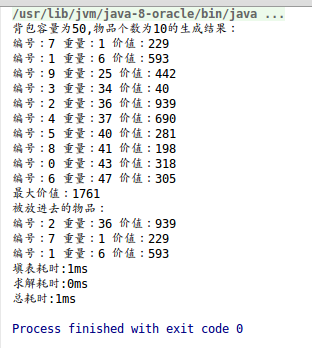
（n=10,W=30）



（n=10,W=40）

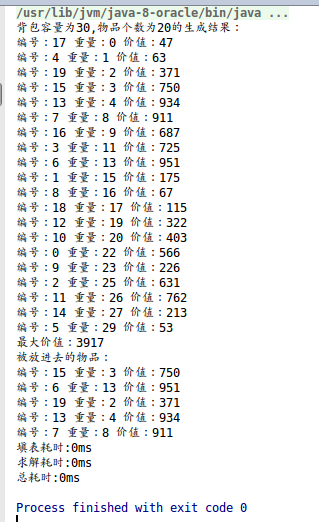


（n=10,W=50）

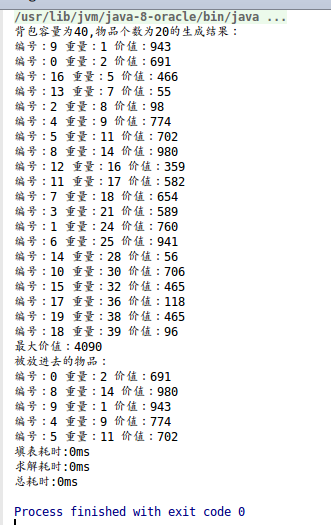


**第二组：**

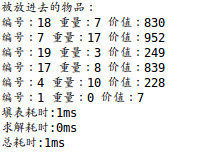
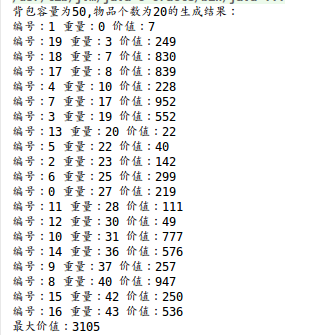
（n=20,W=30）



（n=20,W=40）

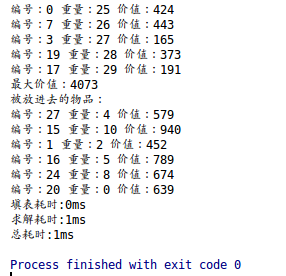
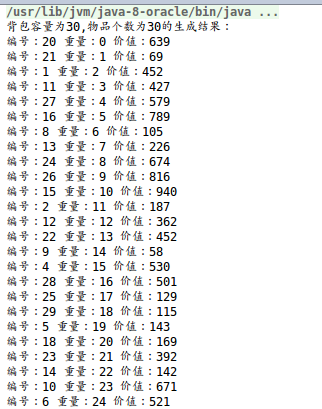


（n=20,W=50）

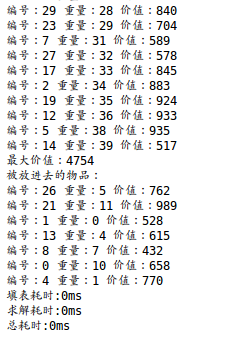
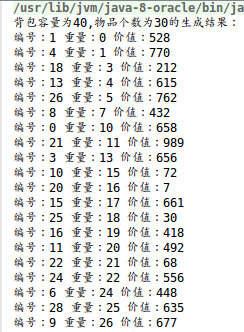


**第三组：**

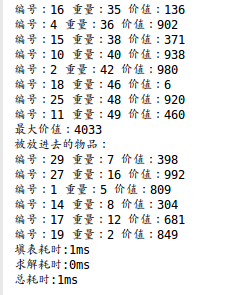
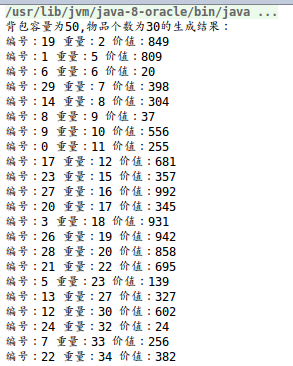
（n=30,W=30）



（n=30,W=40）

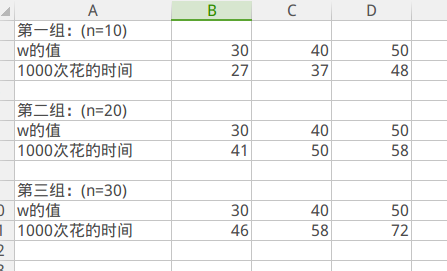


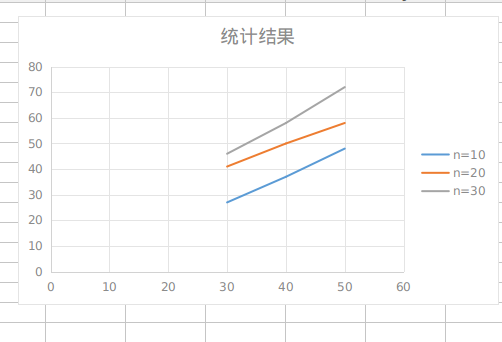
（n=30,W=50）



1. 数据处理分析

分别重复1000次实验，算出总时间：





可以看到，曲线接近是十分完美的直线。在n固定的情况下，算法的运行时间随着w的增加而线性增加，这也与理论符合。因为该算法主要就是计算memo这个二维数组，该数组的大小是n\*w，当n确定，算法运行时间自然是线性增加的。

四、实验总结

通过本次实验我理解了动态规划算法设计的思想。其核心其实就是分治，并比分治多了一步解决冗余。

|  |
| --- |
| 指导教师批阅意见：  成绩评定：  指导教师签字：  年 月 日 |
| 备注： |

注：1、报告内的项目或内容设置，可根据实际情况加以调整和补充。

2、教师批改学生实验报告时间应在学生提交实验报告时间后10日内。