0/1 背包问题及其解法研究

黄波，蔡之华

（中国地质大学计算机学院，湖北 武汉 430074）

摘要：0/1 背包问题是实际当中经常遇到的一类经典 NP－hard 组合优化问题之一。本文分别从贪心方法、动态规划、回溯法、分枝－限界法，遗传算法这五种算法设计方法入手，概述了各种设计方法的基本原理，提出了求解 0/1 背包问题的算法思想，并对算法进行分析，提出了改进方法。

关键词：0/1 背包问题；贪心方法；动态规划；回溯法；分枝－限界法；遗传算法

中图分类号:TP301　　**文献标识码：**A　　**文章编号：**1009-3044(2007)07-2pppp-0c

0/1 Knapsack Problem and Its Solution Methods Study

Huang Bo , Cai Zhi-hua

（China University of Geosciences， Wuhan 430074，China）

Abstract: The 0/1 knapsack problem is a classic NP－hard problem in the combinational optimization1 which is often encountered in practice. This paper will introduce five algorithm design methods, which are Greedy method, Dynamic programming, Backtracking, Branch and bound, Genetic algorithm, summarize their basic tenets, give the solving algorithm thought to 0/1 knapsack problem, analyse the algorithms and put forward the improving methods.

Key words:0/1 knapsack problem; Greedy method, Dynamic programming,;Backtracking; Branch and bound;Genetic algorithm

1 引言

计算机科学是一种创造性思维活动的科学。计算机算法设计与分析是面向设计的、处于核心地位的一门学科。算法是一组有穷的规则，它规定了解决某一特定类型问题的一系列运算[1]。它由有限条可执行的、有确定结果的程序指令组成。算法设计是一件非常困难的工作，常用的算法设计方法有：分治法、贪心方法、动态规划、回溯法、分枝－限界法、基本检索与周游方法、遗传算法等。本文将分别就贪心方法、动态规划、回溯法、分枝－限界法，遗传算法这几种算法设计方法研究 0/1 背包问题的求解方法进行分析，并对各种算法的复杂度进行分析，提出可改进的地方。

2 常见算法设计方法概要

2.1 贪心方法[2]

贪心方法是一种不要求最优解，只希望得到较为满意解的方法。贪心方法一般可以快速得到满意的解，因为它省去了为找最优解要穷尽所有可能而必须耗费的大量时间。贪心方法常以当前情况为基础作最优选择，而不是考虑各种可能的整体情况，所以贪心方法不要回溯。但是，对于某些情况，实际上有解，而该算法不能找到解。对于找不到解的情况，程序只要穷举所有可能情况，就能找到解。

2.2 动态规划

动态规划方法建立在最优原则的基础上， 可以很好地解决许多用贪心方法无法解决的问题。和贪心方法一样， 在动态规划中， 可将一个问题的解决方案视为一系列决策的结果。不同的是， 在贪心方法中， 每采用一次贪心准则便做出一个不可撤回的决策， 而在动态规划中， 还要考察每个最优决策序列中是否包含一个最优子序列。即要求：无论过程的初始状态和初始决策是什么，其余的决策必须相对于初始决策所产生的状态构成一个最优决策

序列。

2.3 回溯法[3]

回溯法是一种系统地搜索问题解答的方法。为了实现回溯， 首先需要为问题定义一个解空间， 这个解空间必须至少包含问题的一个解（可能是最优的）。下一步是组织解空间以便它能被容易地搜索。典型的组织方法是图或树。一旦定义了解空间的组织方法，这个空间即可按照深度优先的方法从开始结点进行搜索， 利用限界函数避免移动到不可能产生解的子空间。

2.4 分枝－限界法

分枝限界是另一种系统地搜索解空间的方法，它与回溯法的主要区别在于对E结点的扩充方式。每个活结点有且仅有一次机会变成E结点。当一个结点变为E结点时， 则生成从该结点移动一步即可到达的所有新结点。在生成的结点中，抛弃那些不可能导出（最优）可行解的结点， 其余结点加入活结点表， 然后从表中选择一个结点作为下一个E结点。从活结点表中取出所选择的结点并进行扩充， 直到找到解或活动表为空， 扩充才结束。

2.5 遗传算法[4]

遗传算法是一类借鉴生物界自然选择和自然遗传机制的随机化的搜索算法，由美国 J.Holland 教授提出，其主要特点是群体搜索策略、群体中个体之间的信息交换和搜索不依赖于梯度信息。因此它尤其适用于处理传统搜索方法难于解决的复杂和非线性问题。遗传算法是一种群体型操作，该操作以群体中所有个体为对象。选择、交叉和变异是遗传算法的三个主要算子，他们构成了遗传算法的主要操作，使遗传算法具有了其它传统方法所没有的特性。

3 0/1 背包问题

3.1 问题的提出

给定 n 种物品和 1 个背包。物品 i 的重量是 wi，其价值为 vi，背包的容量为 C。问应如何装入背包中的物品，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、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。该问题称为 0/1 背包问题。

3.2 问题符号化

0/1 背包问题的符号化表示是，给定 c>0，w_i>0，v_i>0，1≤i≤n，要求找到一个 n 元 0－1 向量（X1，X2，…，Xn），Xi＝0 或 1，1≤i≤n，使得 $\sum_{i=1}^{n} W_i X_i \le C$，而且 $\sum_{i=1}^{n} V_i X_i$ 达到最大。

数学模型为：$\max \sum_{i=1}^{n} V_i X_i$

约束条件：$\begin{cases} \sum_{i=1}^{n} W_i X_i \le C \\ X_i = 0 \text{或} 1, 1 \le i \le n \end{cases}$

4 求解 0/1 背包问题

4.1 利用贪心方法求解 0/1 背包问题

4.1.1 算法思想

贪心方法采用逐步构造最优解的方法。在每个阶段，都在一定的标准下作出一个看上去最优的决策。决策一旦做出，就不可更改，做出贪心决策的依据称为贪心准则。0/1 背包问题有三种贪心准则：

（1）效益贪心准则：从剩余的物品中， 选出可以装入背包的价值最大的物品， 利用这种规则， 价值最大的物品首先被装入(假设有足够容量) ， 然后是下一个价值最大的物品， 如此继续下去。这种策略不能保证得到最优解。

（2）重量贪心准则: 从剩下的物品中选择可装入背包的重量最小的物品。但这种规则在一般情况下不一定能得到最优解。

（3）单位效益最高贪心准则：从剩余物品中选择可装入包的 vi/wi 值最大的物品， 这种策略也不能保证得到最优解， 但它是一个直觉上近似的解。

4.1.2 算法分析及改进

利用以上贪心方法求解0/1背包问题的时间复杂度为$O(2^n)$。由于以上几种准则很难得到最优解，因此可以建立一种启发式方法来求解，即：使求解的结果与最优解的值之差在最优值的X%（X<100）之内。首先将最多K件物品放入背包，如果这K件物品重量大于C，则放弃它。否则，剩余的容量用来考虑将剩余物品按vi/wi 递减的顺序装入。通过考虑由启发式产生的解法中最多为K件物品的所有可能的子集来得到最优解。

4.2 利用动态规划求解0/1背包问题

4.2.1 算法思想

对于0/1背包问题，可以通过作出变量X1，X2，…，Xi的一个决策序列来得到它的解。而对变量X的决策就是决定它是取0值还是取1值。假定决策这些X的次序为Xn，Xn－1，…，X1。在对Xn做出决策之后，问题处于下列两种状态之一：背包的剩余容量是C，没有产生任何效益；剩余容量是C－c，效益值增长了v。显然，剩余下来对Xn－1，Xn－2，…，X1的决策相对于决策X所产生的问题状态应该是最优的，否则Xn，…，X1就不可能是最优决策序列[1]。如果设$f_j(X)$是KNAP（1，j，X）最优解的值，那么$f_n(C)$就可以表示为：

$$f_n(C)=\max\{f_{n-1}(C), f_{n-1}(C-W_n)+V_n\}$$

对于任意的$f_i(X)$，（i>0），则有：$f_i(X)=\max\{f_{i-1}(X), f_{i-1}(X-W_i)+V_i\}$

4.2.2 算法分析及改进

利用动态规划求解0/1背包问题的复杂度为O(min{nc，2^n})。动态规划主要是求解最优决策序列，当最优决策序列中包含最优决策子序列时， 可建立动态规划递归方程， 它可以帮助高效地解决问题。

4.3 利用回溯法求解0/1背包问题

4.3.1 算法思想

0/1 背包问题的解空间由各分量 Xi 分别取 0 或 1 值的2^n个不同的 n 元向量组成，它有两种可能的树结构，一种对应于元组大小固定的表示，另一种对应于元组大小可变的表示。用其中任一种树结构都可导出背包问题的回溯算法。在选取限界函数上，需要遵循的基本原则[1]：即无论使用哪种限界函数，都应有助于杀死某些活结点，使这些活结点实际上不再

扩展。0/1 背包问题的限界函数可以取成能产生某些值的上界的函数。如果扩展给定活结点和它的任一子孙所导致最好可行解值的上界不大于迄今所确定的最好解之值，就可以杀死此活结点。左孩子表示一个可行的结点，无论何时都要移动它；当右子树可能含有比当前最优解还优的解时，移动它。一种决定是否要移动右子树的简单方法是 r 为还未遍历的对象的收益之和，将 r 加到 cp(当前节点所获收益)之上，若(r+cp) ≤bestp（目前最优解的收益），则不需搜索右子树。

4.3.2 算法分析及改进

由于计算上界函数需要O(n)时间，在最坏情况下有O(2^n)个右孩子结点需要上界函数，故计算0/1背包问题的回溯算法所需的计算时间复杂度为O(n2^n)。对回溯法的改进主要是对判断是否移动右子树上，一种更有效的方法是按效益密度vi/wi对剩余对象排序，将对象按密度递减的顺序去填充背包的剩余容量，当遇到第一个不能全部放入背包的对象时，就使用它的一部分。

4.4 利用分枝－限界法求解 0/1 背包问题

4.4.1 算法思想[3]

常用的分枝－限界法有最小耗费或最大收益法，FIFO（先进先出）法等。0/1背包问题的分枝－限界算法可以使用最大收益算法定界函数来计算活结点的收益上限upprofit，使得以活结点为根的子树中的任一结点的收益值都不可能超过upprofit，活结点的最大堆使用upprofit作为关键值域。在子集树中执行最大收益分枝定界搜索的函数首先初始化活结点的最大堆，并使用一个数组bestx 来记录最优解。由于需要不断地利用收益密度来排序，物品的索引值会随之变化，因此必须将函数所生成的结果映射回初始时的物品索引。函数中的循环首先检验E结点左孩子的可行性，如它是可行的，则将它加入子集树及活结点队列（即最大堆）；仅当结点右孩子的定界值指明可能找到一个最优解时才将右孩子加入子集树和队列中。

4.4.2 算法分析及改进

回溯法比分枝限界在占用内存方面具有优势。回溯法占用的内存是0（解空间的最大路径长度），而分枝限界所占用的内存为0（解空间大小）。对于一个子集空间，回溯法需要0(n)的内存空间，而分枝限界则需要0(2^n)的空间。虽然最大收益或最小耗费分枝限界在直觉上要好于回溯法，并且在许多情况下可能会比回溯法检查更少的结点，但在实际应用中，它可能会在回溯法超出允许的时间限制之前就超出了内存的限制。

4.5 利用遗传算法求解 0/1 背包问题

4.5.1 算法思想

(1) 编码方案：用遗传算法来求解0/1背包问题，一种很自然的编码方案是将待求解的各量X表示成长为n的二进制字符串X[i]，j = 1，2,…, n, x[i]=1表示物体j放入背包, x[j]=0表示物体j不放入背包.

(2) 适应度函数的设计：基于上述编码，按照所提出的罚函数处理约束条件的方法构造背包问题的适应度函数 $f_v(X)$： $f_v(X)=\sum_{i=1}^{n} X_i V_i - V_a(X)$。这里，对于所有满足条件 $\sum_{i=1}^{n} W_i X_i \le C$ 的可行解 X，惩罚函数$V_a(X)=0$。惩罚函数可以使用对数函数，线性函数，

二次函数等，分别随侵犯的程度增加而顺序采用[6]。

（3）选择操作：根据选择概率选择染色体，将上述的个体作为第 0 代。对于每个染色体，计算累积概率 Qi，从区间（0，Qi）种产生一个随机数 r，若 Qi－1<r<Qi，则选择第 i 个染色体，重复上述操作，复制染色体。

（4）交叉操作：判断染色体是否为活的染色体，若为活的染色体，则将染色体进行交叉，一般采用单点交叉方式。

（5）变异操作：染色体变异采用位点变异的方式。使变异后的适应度至少大于等于原适应度，否则重新选择变异位进行变异，如此重复若干次，若适应度值没有提高，则变异失败。

（6）当某代得到的结果满足要求或当前代数等于结束代数时算法结束得到结果，否则重复上述步骤（3）、（4）。

4.5.2 算法分析及改进

利用遗传算法求解背包问题具有与传统方法不同的效益，它可以获得最优解，解决传统算法设计方法所不能解决的问题。该算法的复杂度为：$O(n\,2^n)$。对于遗传算法的改进体现在基于惩罚函数修正方法和译码方法上，通过修正得到最优解。

5 各种算法设计方法比较

通过以上对 0/1 背包问题的求解分析，我们可以看到各种算法设计方法有各自不同的特点，针对不同的问题领域，它们有不同的效率。对于求解 0/1 背包问题，各算法的时间复杂、优缺点以及改进方法的比较如下表所示：

设计方法	时间复杂度	优点	缺点	改进方法
贪心方法	$O(2^n)$	速度较快	很难得到最优解	启发式方法
动态规划	$O(\min\{nc, 2^n\})$	可求得最优决策序列	速度较慢	建立动态规划递归方程
回溯法	$O(n\,2^n)$	能够获得最优解	时间复杂度较高	判断右子树时，按效益密度 vi/wi 对剩余对象排序
分枝－限界法	$O(2^n)$	速度较快，易求解	占用的内存较大，效率不高	最大收益或最小耗费分枝限界法，FIFO 法
遗传算法	$O(n\,2^n)$	能够解决传统算法不能解决的问题，能够获得最优解	速度较慢，算法不成熟	基于惩罚函数修正方法和译码方法

表：各种算法设计方法比较

6 结束语

以上是对贪心方法，动态规划，回溯法，分枝－限界法，遗传算法这五种常用算法设计方法原理的概要介绍，并具体应用于求解 0/1 背包问题中，以及对各种设计方法的算法分析。通过论述研究，得出各种算法在求解特定问题领域的优缺点和改进方法。

参考文献：
[1]余祥宣，崔国华，邹海明.计算机算法基础[M].武汉：华中科技大学出版社，2003.
[2]于秀霞. 求解背包问题的新型算法[J].长春大学学报.2002,4.
[3]陈莹， 廖利. 0/1 背包问题[J].电脑知识与技术.2006,5.
[4]王莉，绍定宏，陆金桂.基于遗传算法的背包问题求解[J].计算机仿真.2006,3.
[5]李鸣山，郑海虹.0－1 背包问题的多重分枝－限界算法[J].武汉测绘科技大学学报.1995,3.
[6]霍红卫，许进，保铮.基于遗传算法的0/1背包问题求解[J].西安电子科技大学学报.1998,8.
[7]李肯立，李庆华，戴光明，周炎涛.背包问题的一种自适应算法[J].计算机研究与发展.2004,7.
[8]闫建红.对背包问题的计算机算法研究[J].电脑开发与应用.2005,8.
[9]刘玉娟，王相海. 0－1背包问题的两种扩展形式及其解法[J].计算机应用研究.2006,1.

基金项目：湖北省人文基地资助项目 2004B0011。
作者简介：黄波,男，河南新县人，硕士，研究方向为数据挖掘及其应用;蔡之华,男，博士，教授，博士生导师，中国计算机学会高级会员，中国电子学会高级会员，武汉市科技评估咨询专家，《计算机工程》编辑部理事会理事。主要研究方向为数据挖掘、演化计算及应用、并行计算等。