蚁群算法求解多选择整数背包问题

□ 田志波

（茂名学院 广东·茂名 525000）

摘 要：本文针对一维、二维多选择整数背包问题的数学模型，采用动态规划和蚁群算法对其进行求解，并对蚁群算法作了适当的改进。随机数据实验表明，随着问题规模的扩大，动态规划算法的计算复杂度将急剧增大，造成求解困难，而基本蚁群算法及改进蚁群算法能够快速有效地求得问题近优解，且改进蚁群算法解的质量比原算法平均提高了 2.8%。

关键词：多选择整数背包 动态规划 蚁群算法

中图分类号：TP14 **文献标识码**：A **文章编码**：1007-3973(2009)06-101-03

1 引言

背包问题(Knapsack Problem)是运筹学中一个典型的优化难题，属于 NP-hard 问题[1]。对于规模较大的背包问题，精确求解会产生组合爆炸，求解过程将会花费漫长的时间，怎样能快速有效的求解背包问题一直是人们研究的课题。

目前该问题的算法大致可分为两类：精确算法，如动态规划、分支定界、隐枚举法、回溯法和切割平面法等，由于该类算法计算量是指数时间的，故当规模较大时，在实效上不实用；启发式算法，如模拟退火(SA)、噪声算法(NM)、约束满意(TA)、神经网络(NN)、遗传算法(GA)、贪婪算法、Lagrange 算法、禁忌搜索(TS)[2]等；以及上述这些方法的结合，如将启发式贪婪算法与简单遗传算法相结合构成的一种混合遗传算法[3]，其解的质量和求解性能较简单遗传算法和贪婪算法都有很大的改善。

2 问题描述及其数学模型

2.1 一维多选择整数背包问题描述

设有一只背包，最多可容纳总重量为 W 的物品。现有 n 件物品可供选择装入背包中，这 n 种物品的编号分别为 1,2,…,n。各物品的重量、价值分别为 $w_1,w_2,\cdots,w_n$ 和 $p_1,p_2,\cdots,p_n$，每种物品装入个数为 $x_j(x_j\geqslant 0$ 且为整数，$j\in\{1,2,\cdots,n\})$，每件物品均不能拆开。求如何选择装入的物品(各几件)，可使总价值 z 达到最大。(其中 W，$w_1,w_2,\cdots,w_n$ 和 $p_1,p_2,\cdots,p_n$ 均为整数)

其数学模型为：

$$\max \quad z=\sum_{j=1}^{n}p_jx_j$$

$$\text{s.t.} \quad \sum_{j=1}^{n}w_jx_j\leq W$$

$$j\in\{1,2,\cdots,n\},\ x_j\geq 0 \text{ 且 } x_j,W,w_j,p_j \text{均为整数}$$

2.2 二维多选择整数背包问题描述

设有一只背包，最多可容纳总重量为 W、总体积为 V 的物品。现有 n 件物品可供选择装入背包中，这 n 种物品的编号分别为 1,2,…,n。各物品的价值、重量、体积分别为 $p_1,p_2,\cdots,p_n$，$w_1,w_2,\cdots,w_n$，$v_1,v_2,\cdots,v_n$，每种物品装入个数为 $x_j(x_j\geqslant 0$ 且为整数，$j\in\{1,2,\cdots,n\})$，每件物品均不能拆开。求如何选择装入的物品(各几件)，可使总价值 z 达到最大。

其数学模型为：

$$\max \quad z=\sum_{j=1}^{n}p_jx_j$$

$$\text{s.t.} \quad \sum_{j=1}^{n}w_jx_j\leq W$$

$$\sum_{j=1}^{n}v_jx_j\leq V$$

$$j\in\{1,2,\cdots,n\},\ x_j\geq 0 \text{ 且 } x_j,W,V,w_j,v_j,p_j \text{ 均为整数}$$

3 动态规划算法求解多选择整数背包问题

动态规划是运筹学的一个分支，它是解决在时间过程中，依次分阶段的选取一些决策，来解决整个多阶段决策的动态过程最优化的一种数学规划方法。其主旨思想是将一个多级决策的问题巧妙地转化为一系列互相联系的单阶段决策问题，然后逐个加以解决。

维多选择整数背包问题的动态规划算法设计如下：

(1)划分阶段 k：将供选择的物品按 1,2,…,n 排序，每个阶段可装入一种物品；

(2)确定决策变量：x_k，装入第 k 种物品的件数；s_k，背包中允许装入前 k 种物品的总重量；

(3)建立状态转移方程：$s_k=s_{k-1}+w_kx_k$

决策集合为：$D(s_k)=\{x_k|0\leqslant x_k\leqslant[s_k/w_k],xk \text{ 为整数}\}$

$[s_k/w_k]$表示不超过 s_k/w_k 的最大整数

(4)建立递归方程：

$$f_k(s_k)=\max_{x_k=0,1,2\ldots[s_k/w_k]}\{\max\{f_{k-1}(s_k),f_k(s_k-w_kx_k)+p_kx_k\}\}$$

(5)递推求解：逐步计算出 $f_1(s_1),f_2(s_2),\cdots,f_n(s_n)$，最后求得，$f_n(W)$即为所求的最大价值。

二维多选择整数背包问题的动态规划算法设计如下：

(1)划分阶段 k：将供选择的物品按 1,2,…,n 排序，每个阶段可装入一种物品；

(2)确定决策变量：x_k，装入第 k 种物品的件数；s_k，背包中允许装入前 k 种物品的总重量；r_k，背包中允许装入前 k 种物品的总体积；

(3)建立状态转移方程：$s_k=s_{k-1}+w_kx_k$，$r_k=r_{k-1}+v_kx_k$

决策集合为：$D(s_k,r_k)=\{x_k|0\leqslant x_k\leqslant\min\{[s_k/w_k],[r_k/v_k]\},x_k \text{ 为整数}\}$

$[s_k/w_k],[r_k/v_k]$表示不超过 $s_k/w_k,r_k/v_k$ 的最大整数

(4)建立递归方程：

$$f_k(s_k,r_k)=\max_{x_k=0,1,2\ldots\min\{[s_k/w_k],[r_k/v_k]\}}\{\max\{f_{k-1}(s_k,r_k),f_k(s_k-w_kx_k,r_k-v_kx_k)+p_kx_k\}\}$$

(5)递推求解：逐步计算出 $f_1(s_1,r_1),f_1(s_1,r_2),\cdots,f_1(s_1,r_n),\cdots,f_n(s_n,r_1),f_n(s_n,r_2),\cdots,f_n(s_n,r_n)$，最后求得，$f_n(W,V)$即为所求的最大价值。

4 蚁群算法求解多选择整数背包问题

蚂蚁算法(ant algorithm)[4]是一种源于大自然中生物世

界的新型的拟生态系统算法，1991 年，由意大利学者 Dorigo 等人首先提出。研究发现，蚂蚁在寻找食物源时，能在其走过的路径上释放一种蚂蚁特有的分泌物——信息素(pheromone)，使得一定范围内的其他蚂蚁能够察觉到并由此影响他们以后的行为。当一些路径上通过的蚂蚁越来越多时，其留下的信息素轨迹(trail)也越来越多，以致信息素强度增大，后来蚂蚁选择该路径的概率也越高，从而更增加了该路径的信息素强度。借鉴旅行商问题的蚁群算法求解过程，可以用图 1 来形象地描述多选择整数背包问题。

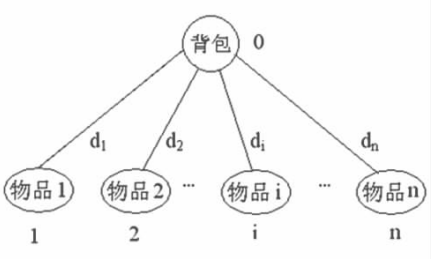

图 1　多选择背包问题的描述

4.1　问题的基本蚁群算法实现

如图 1 所示，设节点 0 代表背包，节点 1−n 分别代表各物品，从节点 0 到任意节点 j(j≠0)的路径权值为 d_j。可以将背包(节点 0)看作蚂蚁寻优的起点，任一物品 j(节点 j，j≠0)看作可供蚂蚁选择的一个食物源，d_j 可以理解为从寻优起点到食物源的距离。一维、二维多选择整数背包问题的蚁群算法实现的主要区别就在于 d_j 的表达式不同，其表达式分别为：

一维背包问题　$d_j=w_j/p_j$

二维背包问题　$d_j=\sqrt{w_j^2+v_j^2}/p_j$

对于任意一只蚂蚁，其从位置 i 转移到位置 j 的概率表达式为：

$$P_{ij}^{k}(t)=\frac{\tau_{ij}^{\alpha}(t)\eta_{ij}^{\beta}}{\sum_{j=1}^{n}\tau_{ij}^{\alpha}\eta_{ij}^{\beta}}\quad(i=0,\ j\in\{1,2,...,n\})$$

τ_{ij} 表示路径中留下的信息素强度；η_{ij} 表示该转移对于蚂蚁 k 的吸引度。在求解背包问题时 $\eta_{ij}=1/d_{ij}$，α 为路径中信息素强度的重要性(α≥0)；β 为吸引度的重要性(β≥0)。由于背包问题属于特例，每只蚂蚁从寻优起点(节点 0)出发只需一步就可到达 1−n 当中的任意一个食物源，因此在上式中有(i=0，j=1,2,…,n)。

整个背包问题的求解过程是由多只蚂蚁完成的。每当一只蚂蚁以较大的概率选中食物源时，如果，背包装入该物品 j 满足约束条件，变量 xj 将加 1；否则将给从 i 到 j 的路径以罚值(flag=−1)，以使后续的蚂蚁在本次背包求解中不再选择本路径。当所有目的节点都受罚以后将结束一次求解过程。记录最优解，然后对路径中信息素强度进行更新，更新的方程式为：

$$\Delta\tau_{ij}=Q*w_j*x_j$$

$$\tau_{ij}=(1-\rho)*\tau_{ij}+\Delta\tau_{ij}\quad(i=0,\{1,2,...,n\})$$

ρ 为挥发系数，Q 为正常数，x_j 为经过该路径的蚂蚁数量。

为算法不会限于局部最优，该问题求解中使用了变异策略，即设蚂蚁 k 将要从 i 转移到的目标节点 j，j 由下面的式子决定，

$$j=\begin{cases}p & \text{若 random}\leqslant\text{mutation}\\ m & \text{否则(m 为随机选择的其他目标节点)}\end{cases}$$

基本蚁群算法求解一维、二维多选择整数背包问题的具体算法步骤如下：

步骤一

(1)nc=0(蚂蚁寻优代数或搜索次数)；

(2)各 τ_{ij}、$\triangle\tau_{ij}$ 及 mu 初始化(0.8mu<1)；

(3)计算从源点(节点 0)到各目标节点(1,2,…,n)的概率值，并将其排序；

(4)清零各节点在上一代所置的罚值标志；

(5)K=1。

步骤二

(1)从节点(1,2,…,n)找出具有最大概率值的目标节点。

(2)若该节点未被置罚值且蚂蚁未发生变异，则沿着较大的概率方向移动；否则随机选择其他方向移动。

(3)对于蚂蚁选择的节点，若未使背包超重，则该节点的记数加 1；否则设置罚值(flag=−1)，以使后续的蚂蚁不再选择该路径。

(4)K=K+1。

步骤三

若还有目标节点没有受罚且 K<ANT_NUMBER，则返回步骤二；否则记录当前最优解，更新路径信息素强度。

步骤四

若 nc 还没有达到最大迭代次数并且解的值还有较大变化，nc=nc+1，k=1，返回到步骤二；否则，输出最优解，终止程序。

4.2　问题的改进蚁群算法实现

观察以上算法实验结果，不难发现，对于一维多选择整数背包问题来说，基本蚁群算法性能较好，而对于求解二维背包问题，其性能并不令人满意，所求最大价值与最优解的偏差较大。针对此种现象，这里将提出对于求解多选择整数背包问题的基本蚁群算法的改进算法，详细介绍如下：

(1)对变异系数法的改进：将式(5−4)改变为下式，

$$j=\begin{cases}p & \text{若 random}\leqslant\text{mutation}\\ m & \text{否则}\end{cases}$$

基本算法中，当 random>mutation 时，蚂蚁将随机访问其余任一个节点，这样虽然可以使寻优范围扩大，但同时也会严重降低利用启发式所求概率值寻优的效果，使实际解偏离最优解较多；若将其改进为选择概率排序列表中的下一节点，则会使解的质量在较短的时间内得到很大程度的提高。

(2)对变异系数设定的改进：寻优初始，希望可以得到较

好的解，因此，会将变异系数尽量设置大一些，如在[0.8,1]之间，以便使蚁群能够根据概率值排序列表进行节点选择；但随着程序的执行的代数的增大，我们更希望能跳出局部最优，扩大求解范围，因此，这时便需要变异系数具有较小的值。这里将原算法改进为：在程序初始化时，将各节点变异系数均设为 1.0，在程序的执行过程中有规律的改变各节点的变异系数。

(3)强制概率法：即为了扩大寻优范围，在程序执行过程中，如果未来几代（根据规模人工给定）的蚂蚁寻优后，均未获得超过现有最大价值量的可行解，就强制将下一代所求得的所有路径的概率值与 1 作差，然后再进行排序，对程序执行过程进行大规模扰动。

5 算法性能实验

经过数据实验，蚁群算法的参数设定为：ANT_NUMBER=n，α=1，β=5，ρ=0.8，Q=6，num=10。

改进算法中，蚂蚁变异系数初始时均为 1.0，然后每循环一代变异系数减小 1.0/20000。

本论文中以随机产生的方式分别得到了不同规模的一维、二维背包数据各 10 组。其中，各物品的重量、体积、价值参数均为[5, 25]之间的随机整数，其平均值为 15。背包的重量及体积约束值的计算式为：W=V=15*N*80%。

算法均采用 VC++6.0 实现，其运行平台为 Windows2000，处理器为 PentiumⅣ系列主频 2.4GHZ，内存为 512MB。

算法性能实验结果如下表。

表 1 一维多选择整数背包问题的算法性能实验结果

数量	重量约束	动态规划		基本蚁群	
		最大价值	时间（秒）	最大价值	时间（秒）
30	360	1	0.000	0.985	0.297
50	600	1	0.000	0.967	0.660
100	1200	1	0.000	0.929	3.898
200	2400	1	0.006	0.951	5.543
300	3600	1	0.025	0.949	8.176
400	4800	1	0.040	0.956	9.668
500	6000	1	0.059	0.986	9.820
1000	12000	1	0.250	0.995	0.998
1500	18000	1	0.599	1.000	0.066
2000	24000	1	1.056	1.000	0.109
2500	30000	1	1.593	1.000	0.179
3000	36000	1	3.243	1.000	0.250
3500	42000	1	8.725	1.000	0.339
4000	48000	1	15.174	1.000	0.441
4500	54000	1	24.099	1.000	0.570
平均		1	3.658	0.981	2.734

分析表 1：求解质量上，随着规模的扩大，基本蚁群算法解的质量也随之提高，基本蚁群算法所求得的近似解平均为最优解的 98.1%；求解时间上，基本蚁群算法求解平均时间比动态规划快 0.924 秒；求解规模上，基本蚁群算法受问题规模影响较小，均可在较短的时间内求得较好的近似解。

表 2 二维多选择整数背包问题的算法性能实验结果

数量	重量约束	体积约束	动态规划		基本蚁群		改进蚁群	
			最大价值	时间（秒）	最大价值	时间（秒）	最大价值	时间（秒）
5	60	60	1	0.000	1.000	0.000	1.000	0.000
10	120	120	1	0.000	0.999	0.002	1.000	0.009
20	240	240	1	0.033	0.984	0.009	0.999	0.054
30	360	360	1	0.142	0.952	0.017	0.987	0.067
40	480	480	1	0.354	0.957	0.035	0.989	0.097
50	600	600	1	0.685	0.957	0.048	0.978	0.040
60	720	720	1	1.234	0.952	0.062	0.986	0.061
70	840	840	1	1.804	0.943	0.106	0.990	0.081
80	960	960	1	2.576	0.957	0.093	0.991	0.111
90	1080	1080	1	7.512	0.957	0.123	0.986	0.123
100	1200	1200	1	35.750	0.954	0.239	0.991	0.092
110	1320	1320	1	88.176	0.954	0.156	0.990	0.061
120	1440	1440	1	145.148	0.954	0.176	0.992	0.106
平均			1	21.801	0.963	0.082	0.991	0.069

分析表 2 得：求解质量上，基本蚁群算法表现相对一维背包较差，然而，算法改进后解的质量提高了 2.8%，可见，改进蚁群算法更适于二维背包问题；求解时间和规模上，蚁群算法仍比动态规划具有优势。

6 结论

本文以一维、二维多选择整数背包问题为例，利用动态规划及蚁群算法对问题进行实验与分析，并针对求解二维多选择整数背包时近似解与精确解存在的较大偏差，对基本蚁群算法作了三点改进，使解的质量平均提高了 2.8%。可以推断，蚁群算法一定会是一种快速有效求解背包问题的启发式算法，特别是对于求解多维背包问题，也必定行之有效。

参考文献：

[1] 高天，翟延慧，王梦光．特殊多维 0-1 背包问题的约束简化方法——不等式单约束生法[J]．东北师大学报自然科学版，2002,34(3):21-25.

[2] P.Cappanera, M.Trubian.A local search based heuristic for the demand constrained multidimensional knapsack problem [A].Working paper presented at Congress Adaptive Memory and Evolution: Tabu Search and Scatter Search [C], Oxford, USA, 2001.

[3] 李娟，方平，周明．一种求解背包问题的混合遗传算法[J]．南昌航空工业学院学报，1998,(3):31-35.

[4] 马良．来自昆虫世界的寻优策略——蚂蚁算法[J]．自然杂志，1999,21(3):161-163.