用模拟退火算法并行求解整数规划问题

（1991 年 6 月 27 日收到）

谢云①

（荆州师范专科学校　湖北江陵 434104）

摘　　要

以 0—1 背包问题为例，描述了用模拟退火算法并行求解整数规划问题的方法。这种算法具有比目前的近似算法远为优越的试验性能，较圆满地解决了 0—1 背包问题这一著名的 NP 完全问题。还讨论了将该算法用于求解一般的整数规划问题的推广途径。所述算法有广泛的应用价值。

关键词：整数规划问题，0—1 背包问题，模拟退火算法，并行算法，PW（k）算法。

一、引　　言

整数规划问题是实际应用中常见的一类问题，0—1 背包问题即为一典型例子，它是一个 NP 完全问题[1,2]。目前最好的近似算法是由动态规划法改进而得的 PW(k)算法[1-3]，这是一个 ε-近似算法，对确定的问题规模 n 及整数 k＞0，其时间复杂性为 $O(n^{k+1})$，最大相对误差为 $\frac{1}{k+1}$[2,3]。

模拟退火算法是近年提出的一种适合解大规模组合优化问题，尤其是解 NP 完全问题的随机近似算法[4]，曾成功地解决了货郎担问题（TSP）等 NP 完全问题[4,5]，其显著优点之一是适合大规模并行实现[6,7]。

本文先用模拟退火算法解 0—1 背包问题，可在比 PW（k）算法少得多的时间内求出与之等质的近似解。然后给出一个高效的并行算法，较圆满地解决了这一 NP 完全问题。最后讨论用该算法解一般的整数规划问题的推广途径。

二、解 0—1 背包问题的模拟退火算法描述

一个 0—1 背包问题即：给定一个可装重量 M 的背包及 n 件物品，物品 i 的重量和价值分别为 W_i 和 C_i，i＝1，…，n。要选若干件物品装入背包，使其价值之和为最大[1,3]。

求解的模拟退火算法描述为：

1. 数学模型

$$max\ \ z = C_1X_1 + \cdots + C_nX_n \tag{1}$$

$$m = W_1X_1 + \cdots + W_nXn \leqslant M \tag{2}$$

$$X_i \in \{0,1\}, \quad i = 1,\cdots,n \tag{3}$$

2. 解空间

① 作者现为武汉大学软件工程国家重点实验室访问学者。

由所有可行解组成，即为

$$\{(X_1,\cdots,X_n)\mid\sum_{i=1}^{n}W_iX_i\leqslant M,\quad X_i\in\{0,1\}\}\tag{4}$$

3. 目标函数

即（1）式

$$z=C_1X_1+\cdots+C_nX_n$$

需求其最大值。

4. 初始解

可任选解空间（4）中一向量为初始解。例如为简单起见，不妨就选零向量

$$(O,\cdots,O)\tag{5}$$

为初始解。

5. 新解的产生

随机选取物品 i：若 i 不在背包中，则将其直接装入背包，或同时从背包中随机取出另一物品 j；若 i 在背包中，则将其取出，并同时随机装入另一物品 j。

6. 目标函数差及背包重量差

设当前解及新解对应的目标函数值和背包重量分别为 z，z_1 和 m，m_1，则关于新解的目标函数差及背包重量差分别为

$$\triangle z=z_1-z\tag{6}$$

$$\triangle m=m_1-m\tag{7}$$

7. 新解的接收概率

新解的接收概率是扩充了可行性测定的 Metropolis 准则[4,5]

$$\begin{cases}0 & m+\triangle m>M\\ 1 & m+\triangle m\leqslant M\text{且}\triangle+>0\\ exp(\triangle z/t) & \text{否则}\end{cases}\tag{8}$$

其中 t 为算法引入的控制参数。

8. 冷却计划表[1,5]

包括如下四个参数：

t_0——控制参数 t 的初值；

α——控制参数 t 的衰减函数；

L——Markov 链的长度；

S——停止准则，常用的停止准则是，在相继的 K 个 Markov 链中对解无任何改变时即停止算法运行。

9. 伪程序

综合以上各条，可给出算法的如下伪程序。其中变量 n，M，W，C，X，z，m 和 t_0，L 以及 α，S 的含义同上。Random 为（0，1）内均匀分布的伪随机数。Initiate 为初始化过程。

```
Procedure Knapsack (n; M; W; C; Var X; Var z; Var m);
  begin
    Initiate (n, M, W, C, X, z, m, t0, L);
    t: =to;
    repeat
      for k: =1 to L do
        begin
          随机选取物品 i;
            if X [i] =0
              then if m+W [i] <=M
                Then begin X [i]: =1; z: =z+C [i]; m: =m+W [i] end
                else begin
                  随机选取物品 j 满足 X [j] =1;
                  △z: =C [i] -C [j]; △m: =W [i] -W [j]
                  if (m+△m<=M) and ( (△z>0) or (Exp (△z/t) >Random))
                    then begin X [i]: =1; X [j]: =0; z: =z+△z; m: =m+△m end
                  end
              else begin
                随机选取物品 j 满足 X [j] =0;
                △z: =C [j] -C [i]; △m: =W [j] -W [i];
                if (m+△m<=M) and ( (△z>0) or (Exp (△z/t) >Random))
                  then begin X [i]: =0; X [j]: =1; z: =z+△z; m: =m+△m end
                end
      end;
    t: =α (t)
  until 停止准则被 S 满足
end;
```

三、计算机模拟结果

用上述算法及 k=2 时的 PW (k) 算法即 PW (2) 算法对 n 从 10 到 1000 的 15 个 0—1 背包问题实例的计算机模拟结果见表 1。所有模拟均在一台 IBM PC/XT 微型计算机上用 TURBO PASCAL 4.0 实现。模拟退火算法的冷却计算表选为：$t_0=0.5$，$\alpha(t)=0.9t$，$L=10n$，S 为 $K=1$。

从表 1 可以看出，两种算法所得解的质量相当（按平均情况相差小于 1.72%），所需时间却相差很大：PW (k) 算法的时间随 n 的增大而剧增（n 增大一倍，时间增大 2^{k+1} 倍），而模拟退火算法增加很少。如表 1 中 n=1000 时，两者相差竟达 554 倍。

表 1 两种算法的模拟结果对比

n	PW (2) 算法		模拟退火算法			
			10 次模拟平均		最好情况	
	z	time (s)	z	time (s)	z	time (s)
10	37	0.16	37.0	0.626	37	0.49
20	131	0.99	129.8	2.278	131	2.42
30	387	2.97	383.3	3.713	387	4.17
40	596	6.70	595.3	6.658	596	4.07
50	1116	12.80	1107.1	7.754	1117	5.88
60	1375	21.59	1351.8	6.260	1372	5.05
70	2085	33.73	2062.2	8.392	2080	9.06
80	2639	49.54	2603.2	6.430	2624	6.92
90	3363	69.97	3347.4	7.339	3363	7.97
100	3745	95.07	3730.5	8.709	3745	9.61
200	15045	736.00	14945.4	17.438	14992	29.33
300	36787	2477.97	36604.5	25.985	36715	36.52
400	64067	5803.43	63807.0	36.823	63919	38.78
500	99067	11368.65	98604.6	47.526	98779	83.77
1000	411827	88553.85	410514.4	159.789	410701	228.33

四、算法的并行实现

为进一步提高解的质量，可以借助更精细的冷却计划表，但运行时间将会随之增长[4]。提高效率的办法是“选择大量地并行”[8]。根据上述算法对解的可行性的要求，下面给出基于协同试验并行策略[7]的并行算法伪程序。其中 p 为并行计算机中处理机的台数，而语句“goto 10”实际上表示处理机控制之间的跳转。根据实际模拟计算，若将 Markov 链长 L 取为串行算法中长度的 $\frac{1}{p}$，则当 p 分别为 2，4，8 和 16 时，上述十五个实例的最大加速[9]的平均值分别可达 1.78，2.58，3.82 和 6.41，且解的质量并不降低。

```
procedure Knapsack (n; M; W; C; Var X; Var z; Var m);
  begin
    Initiate (n, M, W, C, X, z, m, t0, L);
    t: =t0;
    repeat
      for k: =1 to L do
        begin
          for r: =1 to p do IN PARALLEL
```

```
              begin
                随机选取物品 i;
                if X [i] =0 then
                  if m+W [i] <=M then begin Δm: =W [i]; Δz: =C [i]; goto 10 end
                    else begin
                        随机选取物品 j 满足 X [j] =1;
                        Δm: =W [i] -W [i]; Δz: =C [i] -C [j];
                        if (m+Δm<=M) and ( (Δz>0) or (Exp (Δz/t) >Random))
                          then goto 10
                      end
                else begin
                        随机选取物品 j 满足 X [j] =0;
                        Δm: =W [j] -W [i]; Δz: =C [j] -C [i];
                        if (m+Δm<=M) and ( (Δz>0) or (Exp (Δz/t) >Random))
                          then goto 10
                      end
            end;
            goto 20;
10:         接收新解并校正 z 和 m 的值；同时中断其余处理机运行；
20:       end
        t: =α (t)
      until 停止准则被 S 满足
    end;
```

五、推　　广

为运用上述算法解其它的整数规划问题，可从如下几个方面推广：

1. 将约束条件（3）改为

$$X_i \geqslant 0 \text{ 为整数}, i=1,\cdots,n$$

并在算法中的"随机选取物品 i"之后增加一个"随机决定将 i 放入还是取出"的操作，而将对 X 的判断、限制及赋值等运算作相应改动，即可用于求解一般的整数背包问题。

2. 将约束条件（2）改为

$$W_{1j}X_1 + \cdots + W_{nj}X_n \leqslant M_j, j=1,\cdots,h$$

同时对解空间（4）、背包重量差（7）、接收概率（8）及算法作相应地扩充，即可用于求解 h 个约束的 0－1 背包问题。

3. 将目标函数（1）改为

$$min\ \ z = C_1X_1 + \cdots + C_nX_n$$

且随之改变初始解（5）和接收概率（8），则可用于求解最小化的 0－1 规划问题。

4. 根据具体情况综合上述 1～3 中的某几条，可将该算法用于求解任何单目标的 0－1 规划乃

至整数规划问题。

六、结 论

综上所述，本文描述的模拟退火算法较圆满地解决了0－1背包问题这一NP完全问题，又可推广为求解一般的整数规划问题的算法，其效率和质量可通过并行实现及冷却计划表的细化而提高。因此，该算法使用灵活，运行效率高，所得解的质量较好，且应用广泛，不失为解整数规划问题的一种较好的方法。

另外，注意到算法并未用到问题的“线性”这一特征，因此实际上还可用于求解非线性规划问题。对此本文不再赘述。

致谢：本文得到作者的导师、武汉大学并行算法研究室康立山教授的指导及该室同仁的帮助，在此一并致谢！

参考文献：

[1]张泽增,NPC理论导引,贵州人民出版社,贵阳,1989,77－178

[2]赵瑞清,孙宗智,计算复杂性概论,气象出版社,北京,1989,161－171

[3]卢开澄,组合数学:算法与分析(下册),清华大学出版社,北京,1983,365～369

[4]E. H. L. Aarts, J. H. M. Korst, *Simulated Annealing and Boltzmann Machines*, Wiley, Chichester, 1989, 13-94

[5]W. H. Press et al., *Numerical Recipes: The Art of Scientific Computing*, Cambridge University Press, London, 1986, 326-334

[6]康立山,陈毓屏,解货郎担问题的异步并行模拟退火算法,自然科学进展——国家重点实验室通讯,1990,试刊(2),182－188

[7]谢云,模拟退火算法并行实现的若干策略,91′杭州国际科学计算学术讨论会交流论文

[8]康立山,陈毓屏,异步并行算法展望,自然杂志,8(1985),1,27

[9]陈景良,并行数值方法,清华大学出版社,北京,1983,6

The Application of the Simulated Annealing Algorithm to Solving Integer Programming Problems in Parallel

(received Jun. 27,1991)

Xie Yun

(Jingzhou Normal College, Jiangling, Hubei, 434104)

Abstract

A way of applying the simulated annealing algorithm to the integer programming problems (with zero-one knapsack problem as an example) is presented. The empirical performance of the algorithm is much better than the conventional approximate algorithms. The algorithm has solved successfully the zero-one knapsack problem which is a well-known NP-complete problem. The approaches for applying the algorithm to solve general integer programming problems are discussed.

Key words: Integer programming problem, Zero- one knapsack problem, Simulated annealing algorithm, Parallel algorithm, PW(k)-algorithm.