动态规划算法时间效率的优化

福州第三中学　　毛子青

动态规划算法的时间复杂度＝

状态总数＊每个状态转移的状态数＊每次状态转移的时间

一、减少状态总数

1、改进状态表示；（例一）

2、其他方法：选取恰当的规划方向等；

二、减少每个状态转移的状态数

1、根据最优解的性质减少决策量；

（例二）

2、其他方法：利用四边形不等式证明决策的单调性等；

三、减少状态转移的时间

1、减少决策时间　（例三）

方法：采用恰当的数据结构；

2、减少计算递推式的时间

方法：进行预处理，利用计算结果等；

例一、 Raucous Rockers 演唱组（USACO`96）

[问题描述]

现有 n 首由 Raucous Rockers 演唱组录制的歌曲，计划从中选择一些歌曲来发行 m 张唱片，每张唱片至多包含 t 分钟的音乐，唱片中的歌曲不能重叠。按下面的标准进行选择：

(1) 这组唱片中的歌曲必须按照它们创作的顺序排序；

(2) 包含歌曲的总数尽可能多。

输入 n， m， t，和 n 首歌曲的长度，它们按照创作顺序排序，没有一首歌超出一张唱片的长度，而且不可能将所有歌曲的放在唱片中。输出所能包含的最多的歌曲数目。

设 n 首歌曲按照创作顺序排序后的长度为 long[1..n]，则动态规划的状态表示描述为：

g[i, j, k]，(0≤i≤n， 0≤j≤m， 0≤k<t), 表示前 i 首歌曲，用 j 张唱片另加 k 分钟来录制，最多可以录制的歌曲数目。

状态转移方程为：

当 k≥long[i]， i≥1 时：

g[i, j, k]=max{g[i-1,j,k-long[i]]+1， g[i-1,j,k]}

当 k<long[i]， i≥1 时：

g[i, j, k]=max{g[i-1,j-1,t-long[i]]+1， g[i-1,j,k]}

规划的边界条件为：

当 0≤j≤m， 0≤k<t 时： g[0,j,k]=0;

问题的最优解为： g[n,m,0]。

算法的时间复杂度为： O(n*m*t)。

改进的状态表示描述为：

g[i,j]=(a, b)， 0≤i≤n， 0≤j≤i， 0≤a≤m， 0≤b≤t，表示在前 i 首歌曲中选取 j 首录制所需的最少唱片为： a 张唱片另加 b 分钟。

状态转移方程为：

g[i, j]=min{g[i-1,j]， g[i-1,j-1]+long[i]}

其中 (a, b)+long[i]=(a', b') 的计算方法为：

当 b+long[i] ≤t 时： a'=a; b'=b+long[i];

当 b+long[i] ＞ t 时： a'=a+1; b'=long[i];

规划的边界条件：

当 0≤i≤n 时， g[i,0]=(0,0)

题目所求的最大值是： answer=max{k| g[n, k]≤(m-1,t)}

算法的时间复杂度为： $O(n^2)$ 。

Back

例三、石子合并问题 (NOI`95)

[问题描述]

在一个操场上摆放着一圈 n 堆石子。现要将石子有次序地合并成一堆。规定每次只能选相邻的 2 堆石子合并成新的一堆，并将新的一堆的石子数记为该次合并的得分。

试编程求出将 n 堆石子合并成一堆的最小得分和最大得分以及相应的合并方案。

本例只考虑最大得分。

设各堆的石子数依次为 d[1..n]，则动态规划的状态表示为：

m[i,j]， $1 \le i, j \le n$，表示合并 d[i..j] 所得到的最大得分：

令 $t[i,j]=\sum_{k=i}^{j} d[k]$，则状态转移方程为：

$$m[i,j]=\max_{i \le k \le j-1}\{m[i,k]+m[k+1,j]+t[i,j]\} \qquad i<j$$

规划的边界条件为： m[i,i]=0

令 s[i,j]=k，表示合并的最优断开位置。

算法的时间复杂度为 $O(n^3)$。

合并第 i 堆到第 j 堆石子的最优断开位置 s[i,j] 要么等于 i，要么等于 j-1，也就是说最优合并方案只可能是：

{ (i) (i+1… j) }　　或　　{ (i… j-1) (j) }

证明：设合并第 i 堆到第 j 堆石子的最优断开位置 s[i,j]=p，且 i<p<j-1。

情况 1、 t[i, p]≤t[p+1,j]

由于 i<p，所以可以设 q=s[i,p]。于是最优合并方案为：

{ [(i…q) (q+1...p)] (p+1…j) }

它的得分 F_1=m[i, q]+m[q+1,p]+m[p+1,j]+t[i, j]+t[i, p]

我们可以构造如下的合并方案：

{ (i…q) [(q+1...p) (p+1…j)] }

它的得分 F_2=m[i, q]+m[q+1,p]+m[p+1,j]+t[i, j]+t[q+1,j]

由于 q<p，所以 t[i, p]≤t[p+1,j]<t[q+1,j]

所以 $F_1<F_2$，这与合并第 i 堆到第 j 堆石子的最优断开位置 s[i, j]=p 矛盾

状态转移方程优化为：

$$m[i,j]=\max\{m[i+1,j], m[i,j-1]\}+t[i,j] \qquad i<j$$

规划的边界条件是：　　$m[i,i]=0$

算法的时间复杂度 $O(n^2)$。

例三、 LOSTCITY (NOI`2000)

[问题描述]

现给出一张单词表、特定的语法规则和一篇文章：

文章和单词表中只含 26 个小写英文字母 a...z 。

单词表中的单词只有名词，动词和辅词这三种词性，且相同词性的单词互不相同。单词的个数不超过 1000 ，单词的长度均不超过 20 。

语法规则可简述为：名词短语：任意个辅词前缀接上一个名词；动词短语：任意个辅词前缀接上一个动词；句子：以名词短语开头，名词短语与动词短语相间连接而成。

文章的长度不超过 5k 。且已知文章是由有限个句子组成的，句子只包含有限个单词。

编程将这篇文章划分成最少的句子，在此前提之下，要求划分出的单词数最少。

我们分别用 v,u,a 表示动词，名词和辅词，给出的文章用 L[1..M] 表示，则状态表示描述为：

F(v,i)：表示将 L 的前 i 个字符划分为以动词结尾（当 i<>M 时，可带任意个辅词后缀）的最优分解方案下划分的句子数与单词数；

F(u,i)：表示将 L 的前 i 个字符划分为以名词结尾（当 i<>M 时，可带任意个辅词后缀）的最优分解方案下划分的句子数与单词数。

状态转移方程为：

F(v,i)=min{ F(u,j)+(0,1)，　L(j+1..i) 为动词；

　　F(v,j)+(0,1)，　L(j+1..i) 为辅词，　i<>M；　}

F(u,i)=min{ F(u,j)+(1,1)，L(j+1..i) 为名词；

　　F(v,j)+(0,1)，L(j+1..i) 为名词；

　　F(u,j)+(0,1)，L(j+1..i) 为辅词，　i<>M；　}

边界条件：　F(v,0)=(1,0)；　　F(u,0)=(∞, ∞)；

问题的解为：　min{ F(v,M), F(u,M) }；

采用不同的方法查找字符串的比较：

设单词表的规模为 N （ N 的最大值为 1000 ）

设文章的长度为 M （ M 的最大值为 5000 ）

	顺序查找	二分查找	哈希表	检索树
算法的时间复杂度	O(20*M*N)	$O(20*M*\log_2 N)$	O(20*M)	O(M)
最坏情况下的比较次数	10^8	10^6	10^5	$5*10^3$
Input.009	超时	1.27s	0.32s	0.05s
Input.010	超时	1.33s	0.33s	0.05s

（测试环境： Pentium 200 / 32MB ）

Back