一类动态规划问题的优化方法

广西柳州铁路第一中学　林涛

【摘要】

目前动态规划问题已屡见不鲜，其方程大家都能熟练的列出来，但是在算法优化方面往往遇到困难，这里就近年来几道典型题目讨论一类动态规划问题的优化方法。这些动态规划题目的优化多少都得到了 IOI2002batch 的启发，大家可以先看看俞玮的解题报告。

【正文】

有一类动态规划问题，其决策变量存在某些单调性，针对这一特点，可以在决策变量的选取上进行优化，从而降低算法时间复杂度，以下通过几个例子来说明其应用。

欧元（Balkan2003）

问题描述：

给一数列 a_1、a_2、...、a_n 和 T，要求把数列划分成若干段，每段有一个权值 F，如将 a_i、...、a_j 分为一段，则该段权值计算如下：

$F = Sum[i,j] \times j - T$，其中 $Sum[i,j] = \sum_{k=i}^{j} a_k$

求一种划分方案，使各段权值之和最大。

其中 $1 \le n \le 34567$，并且 $-1000 \le ai \le 1000 (i = 1,2,...,n)$。

分析：

很容易想到动态规划的方法，令 $F[i]$ 为数列前 i 项的最优划分权值和，$t[i]$ 为数列前 i 项之和，则

状态转移方程为：$\underset{0<i\le n}{F[i]} = \underset{0\le k<i}{Max}\{F[k] + (t[i] - t[k]) \times i - T\}$

边界条件：$F[0]=0$。

整个算法的时间复杂度为 $O(n^2)$。能否进一步优化?

我们首先对数列最优划分方案进行分析。假设有某段数列：a_i、...、a_j（$j<n$），如果存在 $i\le k\le j$，使得 a_i、...、a_{k-1} 之和为负，a_k、...、a_j 之和为正，那么这个划分一定不是最优的。因为我们可以在保持其它划分不变的情况下，将

a_i、...、a_{k-1}独立出来，而 a_k、...、a_j 并入下一个划分的数列，这样，a_i、...、a_{k-1}所乘的数变小，a_k、...、a_j所乘的数就变大了，总权值就变大了。

因此我们有这样的结论：如果 a_i、...、a_j（$j<n$）是最优划分方案中的子方案那么必定有：$t[j] \le t[i-1]$，否则必然可以找到上述的 k，使得划分更优。

接下来我们对 $F[i]$的决策变量 k 进行分析：

令 $S[k,i]=F[k]+(t[i]-t[k])\times i-T$，则对于任意 $0\le k1<k2<i$ 有：

$$S[k1,i]=F[k1]+(t[i]-t[k1])\times i-T, S[k2,i]=F[k2]+(t[i]-t[k2])\times i-T$$
$$\to\ S[k1,i]-S[k2,i]=F[k1]-F[k2]+(t[k2]-t[k1])\times i$$

（1）当 $t[k2]\ge t[k1]$时：

我们可以知道 $a_{k1+1}\ldots a_{k2}$的和为正，所以无论数列 $a_1\ldots a_{k2}$ 如何划分，都必然有：

1、某段数列的和为正；

2、某段数列的尾部为正。

无论哪种情况成立，都说明这不是最优划分，即 $k2$没有意义。

令 $g(j,k)=(F[k]-F[j])/(t[j]-t[k])$对于 $0<k1<k2<k3<\ldots<kn$，我们可以维护一个队

（2）当 $t[k2]<t[k1]$时：

若 $S[k1,i]-S[k2,i]=F[k1]-F[k2]+(t[k2]-t[k1])\times i>0$,

即 $i<(F[k2]-F[k1])/(t[k1]-t[k2])$,

则对于任意 $j\le i$, 有 $S[k1,j]-S[k2,j]>0$,

可见决策变量是单调的。

列，使得 $t[k1]>t[k2]>t[k3]>\ldots>t[kn]$，保证：

$$i<g(k2,k1)<g(k3,k2)<g(k4,k3)<\ldots<g(kn,kn-1)$$

那么有 $S[k1,i]>S[k2,i]>S[k3,i]>\ldots>S[kn,i]$。

则 $F[i]=F[k1]+(t[i]-t[k1])\times i$。

实现：

队列的维护操作：

（1）加入一个新元素 k。因为 $k>kn$，所以把 k 放置于队列尾，若 $g(k,kn)<g(kn,kn-1)$，则 $n\leftarrow n-1$，直到 $g(kn,kn-1)<g(k,kn)$。

（2）删除元素：若 $g(k2,k1)<i$，则删除 $k1$，直到 $g(k2,k1)>i$。

因为加入的元素和删除的元素不超过 n，所以算法的时间复杂度为 $O(n)$。

最大平均数（USACO）

问题描述：

给一数列 a_1，a_2，...，a_n，求长度至少为 L 的一段，使这其平均数最大。

分析：

假设以 *ai* 为结尾的序列中，从第 $F[i]$个数开始的序列平均数最大。当 $F[i]$知道时，考虑求 $F[i+1]$。

假设 $F[i+1]<F[i]$，那么就表示 $F[i+1]$到 $F[i]$-1 这段的平均数大于 $F[i]$到 i+1 的平均数，然而由定义知道 $F[i+1]$到 $F[i]$-1 的平均数不可能大于 $F[i]$到 i 的平均数，因此以 i+1 为结尾的数列平均数不会超过以 i 为结尾的，所以是没有意义的。

由以上的结论，我们在计算最大平均数时，只需关心当 $F[i+1]>=F[i]$时，以 i+1 为结尾的序列的平均数即可。

假设我们已经求出 $F[i]$，在以后的求解过程中，由于序列开始位置不断增大，我们有必要考虑什么时候该增大，该增大多少。假设当前求出的最大平均数为 x，而当前的开始位置为 *p*，末尾位置为 *i*，我们考虑 *p* 到 *i-L* 这一段，假如从 *p* 到某一位置的平均数小于当前的最大平均数，那么它们是没有任何意义的，直接删去。

我们假设 *p* 到 *p*、*p+1*、*p+2*…的平均数分别为 *x0*、*x1*、*x2*…，如果 $x_i>x_{i+1}$，那么删去 *p* 到 *p+i* 之后必然删去 *p+i+1*，而删去 *p+i+1* 后，是否要删除 *p+i+2* 就要重新算。

但是重新算是没有必要的，因为我们预先就可以避免。重新计算的原因是由于序列中出现 $x_i>x_{i+1}$的现象，所以我们维护一个序列 P_1，P_2，P_3，...，令 *xi* 为 P_i到 P_{i+1}-1 的平均数，使他们满足 *x1<x2<x3<x4<*…。由于可删除的序列不得超过 *i-L*，所以当 *i* 增大时，可删除的序列会增加，当序列增加时，我们就需要维护序列 *P* 的性质。假设原有序列 P_1，P_2，P_3，...，*Pn*，可删除位置新增加到了 P_{n+1}-1，那么如果 x_n比 x_{n-1}大，序列性质不变，就不必理会；如果 x_n比 x_{n-1}小，我们就将 *Pn* 到 P_{n+1}-1 的序列并入 P_{n-1}到 P_n-1 的序列，同时将 *Pn* 删除，反复进行这一操作。

这一操作的实质是：如果最后序列的平均数小于倒数第二序列的平均数，那么把他们并成一个序列。由于在序列中，每个元素只被加入或者删除一次，算法的复杂度为 $O(n)$。

大家还可以参考周源在冬令营提到的，利用数形结合的方法解决此题。

新型计算机（OIBH）

问题描述：

给了一个序列，要求更改序列后满足：从序列第一个数开始，读入一个数 *i* 后，接着读出后面紧跟着的 *i* 个数，再读出一个数 *j*，接着读出后面紧跟着的 *j* 个数，...，如此反复，刚好能使整个数列恰好读完，更改序列的费用为所有数字更改前后绝对值之差的和。要求找出一种更改费用最小的方案。

分析：

初次读完题目，我们很容易想到动态规划的方程，令 $F[i]$为从 *i* 开始恰好读完的最小费用，$F[i]=\underset{k>i}{Min}\{F[k]+|ai-(k-i-1)|\}$。其算法的复杂度显然是 $O(n^2)$。

令$S[i,k]=F[k]+|ai+i+1-k|$，
如果对于$k<j$有：$S[i,k]<S[i,j]$,
则$|ai+i+1-k|-|ai+i+1-j|<F[j]-F[k]$ 。

这个不等式的解集可能为$a_i+i+1<x$。由这个单调性，我们试着像前两题一样构造序列 $k1>k2>k3\ldots$ 使当 $S[n,k_{i+1}]<S[n,k_i]$ 的解集为 $a_i+i+1<x_i$，并使得 $x1>x2>x3>x4\ldots$，有了这样一个序列我们在求解 F[i]时，就不用枚举决策变量 k，而是二分查找一个最大的 j 使 $a_i+i+1<xj$，那么决策变量为 k_j。当然，序列 $k1$，$k2$，$k3$，... 是不断增加的，增加一个元素 k_{n+1} 时，我们解出 $S[n,k_{n+1}]<S[i,k_n]$ 的解集为 $a_i+i+1<x_{n+1}$，如果 $x_{n+1}<x_n$ 那么序列性质不变否则，删除 k_n，不断重复这个操作，使序列重新满足 $x1>x2>x3>x4\ldots$。

维护操作的复杂度是 $O(n)$，整题复杂度就是 $O(nlogn)$。

其实本题还有更好的图论解法，复杂度可以降到 $O(n)$。

总结

这几题写出动态规划方程并不难，但是一般的方法下，决策变量需要枚举，本文着眼于决策变量的一些单调性，在决策变量的选取方面，作了进一步的优化，从而达到降低算法时间复杂度的目的。几题的解法有一些共同点，也是这一类题目优化的共同点：

1. 通过动态规划方程，发现决策变量的一些单调性。
2. 利用这个单调性维护一个体现该单调性的队列。
3. 利用这个队列进行决策。