动态规划中的“提纯”

中山纪念中学 陈启峰

【概述】

动态规划中优化的方法有很多种，总的来说可以分为两种方法，即**减少重复计算**和**减少冗余**。这两种“提纯”方法都体现了动态规划的本质特征。减少重复计算一般从策略上着手，先找出重复计算的值，然后增添新的状态来存储该状态值；而减少冗余一般可以从状态和策略去优化，删除无用的状态和策略。

通过下面这题的分析，希望能对大家起到抛砖引玉的作用。

【关键字】

减少重复计算、减少冗余

【正文】

问题简述——求两个序列的最长公共上升子序列（LCIS）：

给出两个长度分别为 n,m 的序列 A 和 B。

求出一个长度最长序列 C，满足：

① $C_1 < C_2 < C_3 \ldots\ldots < C_l$；

② C 既是 A 的子序列又是 B 的子序列；

题目分析：

当看到这题的三个关键字——最长、公共、上升时，我便不禁有似曾相识的感觉。如果抛开公共或者上升的要求，就是我们都熟悉的最长公共子序列（LCS）问题，和最长上升子序列（LIS）问题了！

动态规划的首要任务是确定好状态。回顾 LCS 和 LIS 的状态表示方法，有创意地综合这两种状态表示方法的特性，便可以得到这题的一种状态表示：

设 $x_{i,j}$ 表示 A_1 到 A_i 和 B_1 到 B_j 的 LCIS 的长度，其中要求 $A_i=B_j$、必须选取 A_i 和 B_j 作为一个匹配。于是得到一个状态转移方程 A

$$x_{i,j}=\max\begin{cases}1\\ \max\limits_{\substack{0<k<i,0<l<j,B_l<B_j,\\ \text{并且}x_{k,l}\text{有意义}}}\{x_{k,l}+1\}\end{cases}$$

这个动态规划的时间复杂度为 $O(n^4)$，显然是不尽人意的，需要进行“提纯”。

“提纯”方法 1——减少重复计算

仔细分析不同状态的策略，易知意义相同的值会被计算了多遍。比如：对于任意的两个状态 x_{i,j_1} 和 x_{i,j_2}，如果存在某个 l 使得 B_l<B_{j_1}、B_{j_2} 并且 l<j1、j2,那么在两个状态的策略选取时，$\{x_{k,l}|k<i\}$里状态都被计算一次。

计算某个状态集合，其实质是计算这个集合中状态值最大的状态，因此我们只需知道这个集合中的最大状态值就足够了。而形如$\{x_{k,l}|k<i\}$的集合被计算多次，也就是这些集合的最大状态值被重复计算了，所以可以添增一种状态表示——$f_{i,j}$ 表示 $\max\{x_{k,j}|k<=i\}$。于是得出双状态的状态转移方程 B：

$$x_{i,j}=\max\begin{cases}1\\ \max\limits_{0<k<j,B_k<B_j}\{f_{i-1,k}+1\}\end{cases}$$

$$f_{i,j}=\max\{0,f_{i-1,j},x_{i,j}\}$$

这样总的时间复杂度是 $O(n^3)$,如果使用二叉查找树还可以使时间复杂度降到 $O(n^2 \log n)$。显然这样的“提纯”已经除去很多杂质了，但可别忘记还有另一把锋利的刀哦~~。^_^

“提纯”方法 2——减少冗余

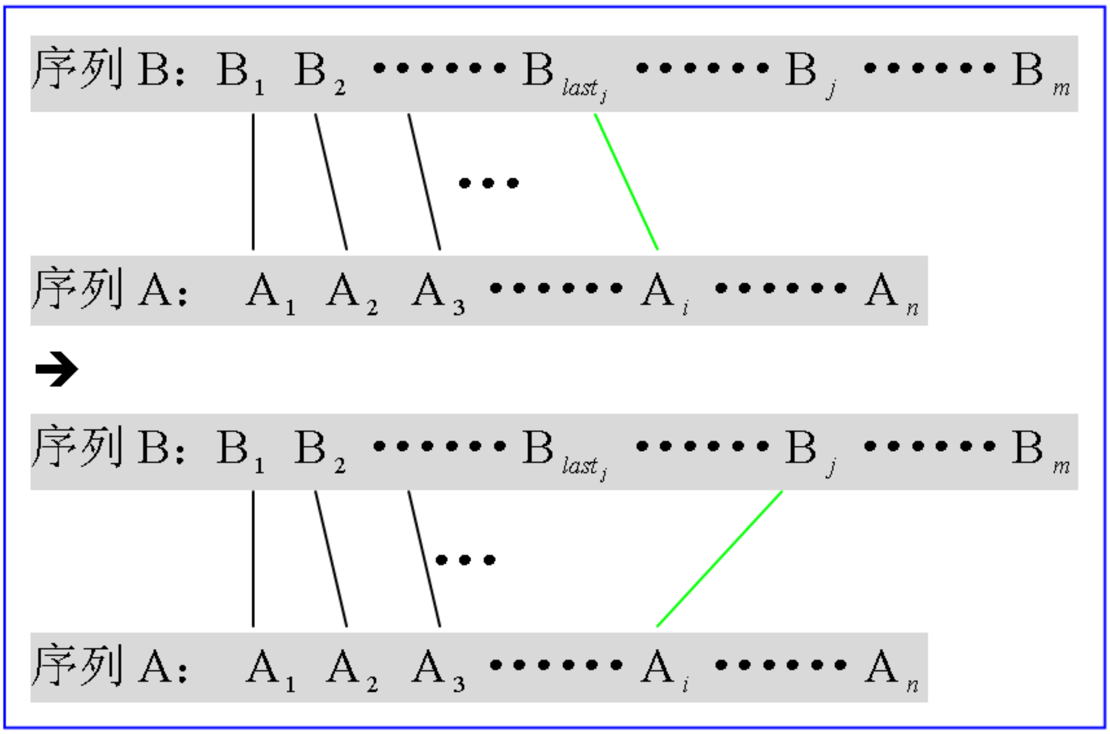
冗余常常隐藏得很隐蔽，就像豪宅里的白蚁，不易被发现，所以要花多点时间和心思去发挖掘。

在状态转移方程序 B 中，着眼于所有最优策略的选取范围，通过多次试验，不难发现对于任意的一个状态 $x_{i,j}$，如果其最优策略小于一个定值 $last_j=\max\{0,w|w<j$ 并且 $B_w=B_j\}$，那么 $x_{i,j}=x_{i,last_j}$。于是便产生一个猜想——是不是对于所有状态 $x_{i,j}$，若其最优策略小于 $last_j$，都有 $x_{i,j}=x_{i,last_j}$？下面通过证明来肯定这个猜想。

证明：

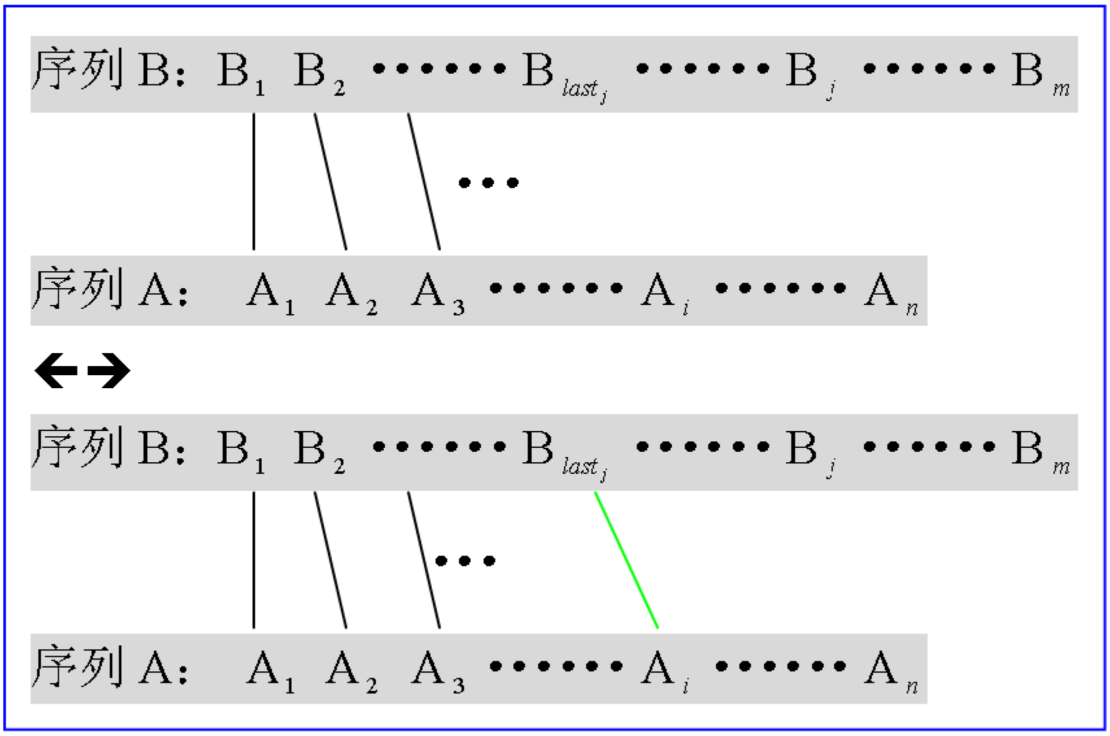
对于任意一个状态 $x_{i,j}$,$x_{i,last_j}$ 是一个可行解。因为这可由 $x_{i,last_j}$ 的最优方案构

造出来，只要将匹配$(i,last_j)$换成(i,j)就可以了，如下图：



所以$x_{i,j} \geq x_{i,last_j}$。

接着，对于任意的$f_{i-1,k}$ $(k<last_j)$，有 g 个匹配的$f_{i-1,k}$相应的方案，有 g+1 个匹配的$x_{i,last_j}$相应的方案一一对应。因为只要在前者的方案上加上匹配$(i,last_j)$。如下图：



所以 $x_{i,last_j}$ ≥任意的 $f_{i-1,k}+1(k<last_j)$。

综上两个结论，计算 $f_{i-1,k}+1(k<last_j)$等效于计算 $x_{i,last_j}$，也就是对于所有状态 $x_{i,j}$，若其最优策略小于 $last_j$，都有 $x_{i,j}=x_{i,last_j}$。

有了上面有力的证明以后，对状态转移方程 B 进行改进，就得到了更优的状态转移方程 C

$$x_{i,j}=\max\begin{cases}1\\ \max\limits_{last_j<k<j,B_k<B_j}\{f_{i-1,k}+1\}\\ x_{i,last_j}\end{cases}$$

$$f_{i,j}=\max\{0,f_{i-1,j},x_{i,j}\}$$

$$last_j=\max\{0,w|w<j \text{ 并且 } B_w=B_j\}$$

这个动态规划的总时间复杂度仅是 $O(n^2)$，时间复杂度较之前已经降下了很多。

总结

动态规划的“提纯”并没有固定的形式，常常需要一种灵感。这种灵感并非空中楼阁，而是来源于平时的思考和积累。只要能常总结、常思考、常创新并持

之以恒，就一定能学好动态规划。

【感谢】

感谢宋新波老师、郭华阳向我提很多宝贵的意见。

【参考文献】

《充分利用问题性质——例析动态规划的“个性化”优化》——项荣璟 2003 论文

《减少冗余与算法优化》——胡伟栋 2004 论文