动态规划感想

早就听闻它的大名，在算法设计与分析课上系统地学习动态规划之前，我已经了解过一点它的内容。

我接触动态规划算法，是从求斐波那契数列开始的。虽然求斐波那契第n项的值并不是一个严格意义上的动态规划问题，但是它涉及到了动态规划的核心思想，状态转移方程，或者叫动态规划方程。这也是动态规划里面最难想到的一步，比如矩阵连乘积的最优乘法次序问题中“在中间的位置断开矩阵”。

我根据网络上的博客，以及自己的刷题经验，也总结了求解动态规划问题的一般步骤。

1. **定义原问题和子问题。**子问题是和原问题相似但规模更小的问题。
2. **定义状态。**状态在计算机的物理表示就是数组，数学上可以认为是某个函数的自变量，常见的有一元自变量和二元自变量。DP数组的元素就是这个状态对应子问题的求解结果。
3. **寻找状态转移方程。**这一步是动态规划的核心步骤。其实不用一开始就编程，可以先在草稿纸上推演，得到数学上的函数表达式，然后再编程实现，就很简单了。最核心的部分还是找到动态规划的方程。
4. **回过去递归找解。**得到结果之后，题目往往需要我们回过去找如何得到这个最优解，这时候就要递归地回去找临界的点，然后层层递归。

动态规划问题有一些独特的性质。

1. **最优子结构性质。**如果问题的最优解所包含的子问题的解也是最优的，我们就称该问题具有最优子结构性质（即满足最优化原理）。这个性质往往在证明中需要使用。
2. **子问题重叠性质。**子问题重叠性质是指在用递归算法自顶向下对问题进行求解时，每次产生的子问题并不总是新问题，有些子问题会被重复计算多次。动态规划算法正是利用了这种子问题的重叠性质，对每一个子问题只计算一次，然后将其计算结果保存在一个表格中，当再次需要计算已经计算过的子问题时，只是在表格中简单地查看一下结果，从而获得较高的效率。
3. **无后效性。**子问题的解一旦确定，就不再改变，不受在这之后、包含它的更大的问题的求解决策影响。这个性质也经常用在证明之中。

课堂上的几个动态规划问题，还是相当有难度的。第一个矩阵连乘积的最优乘法次序问题，三重循环的迭代变量很容易把人搞晕，要把握住，第一层的r表示的是每一条对角线，第二层i代表了一条对角线中的每一个元素，同时也是行号，此时根据r和i可以计算出j，也就是列号，第三层k代表的是矩阵断开的位置。第二个，最长公共子序列问题，这是比较常规的二维dp问题，相对容易。第三个，01背包问题，经典的优化问题，也比较容易，但是后期讲的跳跃点集算法我还没有搞明白。第四个，凸多边形最优三角剖分，将其转化为矩阵连乘积的想法，非常巧妙。第五个，多边形游戏，对我来说十分困难，尤其是三维的数组，课后还需要花些时间去消化。

其实动态规划也没有想象中的那么难，熟能生巧，只要多花时间练习和思考，肯定能够掌握这种算法的解题精髓和思想。