# 经典动态规划问题

**钢条切割问题**：对于一段长为n的钢条，有一个长度-价格的对应表，如何切割钢条使得最后总价值最大？

这个问题的解法同样包括分治的思想，但是有一点改变：它不是仅考虑两个规模完全相同的问题，而是考虑覆盖了所有规模组合的子问题并且从中取出最大值。即f(n)=max{f(i)+f(n-i)};

**矩阵连乘问题**：矩阵乘法满足结合律，因此在一个矩阵乘法链中可以任意加括号来改变最终得到结果所花费的次数，求一个方案使得最终的乘法次数较少。

A1A2…An，设Ai规模为pi\*pi+1.

解法同样考查所有的子问题，用A[i..j]表示i到j的连乘问题，则显然A[1..n]的问题可以考虑成A[i..k],A[k+1..n]的问题，然后合并他们。问题是，k是否确定？所以，最终的递归方程看起来也如此：f(i..j)=min{f(i..k)+f(k+1…j)}.

其实，最初我也考虑过分治算法的模型，连乘问题的最优解可以这样产生，设i..j的中点是k，则i..j的最优解或者是i..k,k+1..j的形式，这种形式的特点是Ak,Ak+1一定是不加括号的，或者是除去Ak,Ak+1,加入合并Ak，Ak+1后的新矩阵的解，此时问题规模减少1，这种形式的特点是Ak，Ak+1一定是加括号的。这也是可行的，子问题之间并不重叠。其实采用任意的k值仍然可以将该问题分解为两类并且规模都将减少，但是分治的一个重要思想是使得子问题的规模尽量相同。由此，我们也注意到：**构造分治算法的理念是1.找到关于某些元素的条件{是，否}，依据这个条件将问题二划分2.二划分之后子问题的规模差不多并且相对于原问题规模减少。**而**构造动态规划算法的理念是对所有规模的子问题求解并比较。**

**01背包问题：**有n个物品，一个背包，每个物品的价值为pi，重量为wi，背包容量为C，求一个选择方案，使得放入的物品的价值最大。

**解法：**可以对第c个物品进行考察，以放入和不放入为标准来划分问题，于是，这个问题就化成了两个子问题。从递归方程上说，其复杂度应当为2^n，但是递归方程只有两个参数，而把这两个参数都遍历完的解至多是O(c\*n)

# 问题实例

**最长回文子序列：**

给定一个字符串，求这个字符串的最长回文序列，这个字串不必是连续的。如character返回carac.

**字符串编辑距离：**

　　对于序列S和T，它们之间距离定义为：对二者其一进行几次以下的操作(1)删去一个字符；(2)插入一个字符；(3)改变一个字符。每进行一次操作，计数增加1。将S和T变为同一个字符串的最小计数即为它们的距离。给出相应算法。

# 没有参考解的问题收集