# 算法策略的总结

策略是面向问题的，算法是面向实现的。

# 一、不同算法策略特点小结

## 1、贪心策略

贪心策略一方面是求解过程比较简单的算法，另一方面它又是对能适用问题的条件要求最严格（即适用范围很小）的算法。

贪心策略解决问题是按一定顺序，在只考虑当前局部信息的情况下，就做出一定的决策，最终得出问题的解。

即：通过局部最优决策能得到全局最优决策

## 2、递推策略

递推也是由当前问题的逐步解决从而得到整个问题的解，依赖于信息间本身的递推关系，每一步不需要决策参与到算法中，更多用于计算

## 3、递归策略

递归常常用于分治算法、动态规划算法中。

递归是利用大问题与其子问题间的递推关系来解决问题的。

能采用递归策略的算法一般有以下特征：

（1）为求解规模为N的问题，设法将它分解成规模较小的问题，然后从这些小问题的解方便地构造出大问题的解

（2）并且这些规模较小的问题也能采用同样的分解和综合方法，分解成更小的问题，并从这些更小的问题的解构造出规模较大问题的解

（3）特别的，当规模N = 1时，能直接得解

## 4、枚举策略

对问题所有的解逐一尝试，从而找出问题的真正解。一般用于决策类问题，很难找到大、小规模之间的关系，也不易对问题进行分解。

## 5、递归回溯策略

类似于枚举，通过尝试遍历问题各个可能解的通路，当发现此路不通时，回溯到上一步继续尝试别的通路。

## 6、分治策略

分治一般用于较复杂的问题，必须可以逐步被分解为容易解决的独立的子问题，这些子问题解决后，进而将它们的解"合成"，就得到较大问题的解，最终合成为总问题的解。

## 7、动态规划策略

与贪心类似，也是通过多阶段决策过程来解决问题。每个阶段决策的结果是一个决策结果序列，这个结果序列中，最终哪一个是最优的结果，取决于以后每个阶段的决策，当然每次决策结果序列都必须进行存储。因此是"高效率，高消费的算法"。

同时，它又与递归法类似，当问题不能分解为独立的阶段，却又符合最优化原理时，就可以使用动态规划法，通过递归决策过程，逐步找出子问题的最优解，从而决策出问题的解。

# 二、算法策略间的关系

## 1、对问题进行分解的算法策略——分治法与动态规划法

共同点：（1）分治法与动态规划法实际上都是递归思想的运用

（2）二者的根本策略都是对问题进行分解，找到大规模与小规模的关系，然后通过解小规模的解，得出大规模的解

不同点： 适用于分治法的问题分解成子问题后，各子问题间无公共子子问题，而动态规划法相反。

动态规划法 = 分治算法思想 + 解决子问题间的冗余情况

## 2、多阶段逐步解决问题的策略——贪心算法、递推法、递归法和动态规划法

贪心算法：每一步都根据策略得到一个结果，并传递到下一步，自顶向下，一步一步地做出贪心决策。

动态规划算法：每一步决策得到的不是一个唯一结果，而是一组中间结果（且这些结果在以后各步可能得到多次引用），只是每一步都使问题的规模逐步缩小，最终得到问题的一个结果。

递推、递归法：注重每一步之间的关系，决策的因素较少。递推法是根据关系从前向后推导，从小规模问题的结论推解出大问题的解。而递归法是根据关系从后向前使大问题转化为小问题，最后同样由小规模问题的解推解出大问题的解。

## 3、全面逐一尝试、比较——蛮力法、枚举法、递归回溯法

蛮力策略（即枚举和递归回溯）：

当问题找不到信息间的相互关系、也不能将问题分解为独立的子问题，就只有把全部解都列出来之后，才能判定和推断出问题的解。

蛮力策略适用于规模不大的问题。

（1）枚举法：实现依赖于循环。所以一个枚举法只针对一个特定问题规模的情况，例如：八重循环嵌套解八皇后问题的算法。

（2）递归回溯法：适用于任意指定规模的情况，例如：递归回溯法解N皇后问题。

### 4、算法策略的中心思想

用算法策略将解决问题的过程归结为：用算法的基本工具“循环机制和递归机制”实现。

## 三、算法策略侧重的问题类型

一般常遇到的问题分为四类：

（1）判定性问题：可用递推法、递归法

（2）计算问题：可用递推法、递归法

（3）最优化问题：贪心算法、分治法、动态规划法、枚举法

（4）构造性问题：贪心算法、分治法、广度优先搜索、深度优先搜索