# 分类

可以按照很多标准来给算法分类，例如：

1. 按实现方式划分

**递归算法与迭代算法**

**过程式算法与声明式（非过程式）算法**

**串行算法、并行算法、分布式算法**

**确定性算法与非确定性算法**

**精确算法与近似算法**

1. 按设计方式划分

**贪婪算法**

**分治算法**

**动态规划算法**

**线性规划算法**

**规约（转换并治理）算法**

1. 按其他方式划分

**按研究领域划分**

**按复杂度划分**

**随机化的算法**

**分支定界与回溯**

# 迭代法

迭代法是一种不断用旧值递推新值的过程，分精确迭代和近视迭代。是用来求方程和方程组近似根的方法。

迭代变量

迭代关系， 迭代关系选择不合理，会导致迭代失败

迭代过程控制，也就是迭代什么时候结束，不能无休止进行下去

# 递归

递归是一种设计和描述算法的有力工具。递归算法执行过程分递推和回归两个阶段

在递推阶段，将大的问题分解成小的问题在回归阶段，获得最简单问题的解后，逐级返回，依次得到稍微复杂情况的解，知道获得最终的结果

1）确定递归公式

2）确定边界条件

## 斐波那契数列

fib(n)=fib(n-1)+fib(n-2)

递归实现

非递归实现

## 其它案例

阶乘计算

梵塔问题 （三根针1，2，3表示，1号从小到大n个盘子，先要都移到3号上，不能出现大盘压小盘，找出移动次数最少的方案）

快速排序

递归运行效率较低，因为有函数调用的开销，递归多次也可能造成栈溢出。

# 穷举搜索法

或者叫蛮力法。对可能的解的众多候选按照某种顺序逐一枚举和检验。典型的问题如选择排序和冒泡排序。

## 背包问题

给定n个重量为 w1,w2,...,wn,定价为 v1,v2,...,vn 的物品，和一个沉重为W的背包，求这些物品中一个最有价值的子集，且能装入包中。

## 其它案例

选择排序

冒泡排序

# 动态规划

动态规划（Dynamic Programming，DP）技术通常与记忆化（memoization）技术结合起来使用。

复杂问题不能分解成几个子问题，而分解成一系列子问题；DP通常基于一个递推公式及一个(或多个)初始状态，当前子问题解由上一次子问题解推出。

状态 状态转移方程 递推关系

动态规划算法的关键在于解决冗余，以空间换时间的技术，需要存储过程中的各种状态。可以看着是分治算法+解决冗余使用动态规划算法的问题的特征是子问题的重叠性，否则动态规划算法不具备优势。

动态规划与分治的区别在于，分治法所要处理的那些子问题之间并没有依赖关系，而动态规划所要处理的子问题却是有所重叠的，因此，可以把已经解决的子问题保存到表格里，这就是记忆化技术。运用这种技术，算法可以把很多问题的复杂度由指数级别降至O(n2)、O(n3)这样的多项式级别。

动态规划技术与（分治算法中的）递归计数相比，其区别在于，它会把已经解决的子问题放在表格中，以免去重复的计算；而分治算法所要递归解决的那些子问题，彼此之间不重复。由此可见，并非所有的问题都适合用动态规划技术来解决。

## 基本步骤

1、划分问题

2、选择状态

3、确定决策并写出状态转移方程

4、写出规划方程

## 最长递增子序列

最长递增子序列（LIS Longest Increasing Subsequence）

## 其它案例

最短路径

# 贪心算法

贪心算法（greedy algorithm，贪婪算法）是分阶段执行的，每一个阶段都根据当前的情况来判断，而不考虑后续的发展。

一般来说，这种算法选出的解是局部最佳（local best）解。该算法预设了这样一个前提，就是认为全局最优解可以由局部最优解所推出。

即，贪心算法不追求最优解，只找到满意解。

## 特点

## 适用场合

## 赫夫曼编码

## 其它案例

找回零钱问题

装箱问题

# 回溯法

也叫试探法。是一种选优搜索法，按照选优条件搜索，当搜索到某一步，发现原先选择并不优或达不到目标，就退回重新选择。

## 一般步骤

1、针对问题，定义解空间（ 这时候解空间是一个集合，且包含我们要找的最优解）

2、组织解空间，确定易于搜索的解空间结构，通常组织成树结构 或 图结构

3、深度优先搜索解空间，搜索过程中用剪枝函数避免无效搜索

回溯法求解问题时，一般是一边建树，一边遍历该树；且采用非递归方法。

## 八皇后问题

8x8的国际象棋棋盘上放置8个皇后，使得任何一个皇后都无法直接吃掉其他的皇后。任意2个皇后都不能处于同一个 横线，纵线，斜线上。

分析

任意2个皇后不能同一行，也就是每个皇后占据一行，通用的，每个皇后也要占据一列

一个斜线上也只有一个皇后

## 其它案例

迷宫问题

# 分治算法

分治（divide and conquer，D&C）算法将一个难以直接解决的大问题，分割成一些规模较小的相同问题，各个击破，分而治之。

分治算法常用递归实现：

1）问题缩小的小规模可以很容易解决

2) 问题可以分解为规模较小相同问题

3）子问题的解可以合并为该问题的解

4）各个子问题相互独立，(如果这条不满足,转为动态规划求解）

## 步骤

分治法的步骤：

1、划分（divide）：把问题划分成多个子问题，这些子问题和原问题同属一类，但规模较小

2、递归（recursion）：递归地解决这些子问题

3、治理（conquer）：将子问题的答案适当地合并起来

## 大整数乘法

如 26542123532213598\*345987342245553677884

## 其它案例

快速排序

归并排序

最大子数组和

二叉搜索算法

# 总结

贪心法、分治法、动态规划都是将问题归纳为根小的、相似的子问题，通过求解子问题产生全局最优解。

贪心法

分治法

动态规划