

1. 渐进时间复杂度

算法时间复杂度的本质是算法的执行时间，也就是算法中所有语句的频度之和。上一篇博客中说了，语句频度就是语句的执行次数，它与算法求解问题的规模大小息息相关。

假设对于给定的算法，目前问题规模为n，则语句频度可以表示成一个关于问题规模的函数 T(n)，那么算法时间复杂度也就可以用T(n)表示，其含义是算法在输入规模为n时的运行时间。

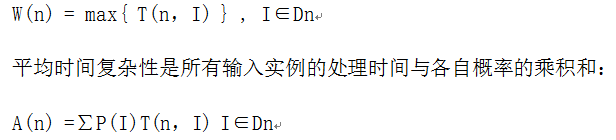
当问题规模很大时，精确的计算T(n)是很难实现而且也是没有必要的。对于算法时间性能的分析无需非要得到时间复杂度T(n)的精确值，它的变化趋势和规律也能清楚地反映算法的时间耗费。基于此，引入了渐进时间复杂度作为时间性能分析的依据，它的含义就是：在问题规模 n趋于无穷大时算法时间复杂度T(n)的渐进上界，即函数 T(n)的数量级(阶)。

算法时间复杂度和渐进算法时间复杂度在实际的算法分析过程中是不予区分的，渐进时间复杂度可以简称为时间复杂度，记为T(n)=O(f(n))。其中，通过统计算法中基本操作重复执行的次数就可近似地得到算法的执行效率，用O(n)表示，称为时间复杂度。

当问题的规模n趋向无穷大时，影响算法效率的重要因素是T(n)的数量级，而其他因素仅是使时间复杂度相差常数倍，因此可以用T(n)的数量级(阶)评价算法。时间复杂度T(n)的数量级(阶)称为渐进时间复杂性。

最坏情况下的时间复杂性和平均时间复杂性有什么不同

最坏情况下的时间复杂性和平均时间复杂性考察的是n固定时，不同输入实例下的算法所耗时间。最坏情况下的时间复杂性取的输入实例中最大的时间复杂度：



为什么要分析最坏情况下的算法时间复杂性

最坏情况下的时间复杂性决定算法的优劣，并且最坏情况下的时间复杂性较平均时间复杂性游可操作性

1. 算法重要特性是什么？

2. 算法分析的目的是什么？

3. 算法的时间复杂性与问题的什么因素相关？

4. 算法的渐进时间复杂性的含义？

5. 最坏情况下的时间复杂性和平均时间复杂性有什么不同？

6. 简述二分检索（折半查找）算法的基本过程。

7. 背包问题的目标函数和贪心算法最优化量度相同吗？

8. 采用回溯法求解的问题，其解如何表示？有什么规定？

9. 回溯法的搜索特点是什么？

10. n皇后问题回溯算法的判别函数place的基本流程是什么？

11. 为什么用分治法设计的算法一般有递归调用？

12. 为什么要分析最坏情况下的算法时间复杂性？

13. 简述渐进时间复杂性上界的定义。

14. 二分检索算法最多的比较次数？

15. 快速排序算法最坏情况下需要多少次比较运算？

16. 贪心算法的基本思想？

17. 回溯法的解（x1,x2,……xn）的隐约束一般指什么？

18. 阐述合并排序的分治思路

19. 快速排序的基本思想是什么。

20. 什么是直接递归和间接递归？消除递归一般要用到什么数据结构？

21. 试述分治法的基本思想。

22. 设计动态规划算法有哪些主要步骤？

23. 分治法与动态规划法的异同？

24. 备忘录方法和动态规划算法相比有何异同？简述之。

1. 输入、输出、确定性、有限性、可实现性。

2. 分析算法占用计算机资源的情况，对算法做出比较和评价，设计出更好的算法。 3. 算法的时间复杂性与问题的规模相关，是问题大小n的函数。

4．当问题的规模n趋向无穷大时，影响算法效率的重要因素是T(n)的数量级，而其他因素仅是使时间复杂度相差常数倍，因此可以用T(n)的数量级(阶)评价算法。时间复杂度T(n)的数量级(阶)称为渐进时间复杂性。

5. 最坏情况下的时间复杂性和平均时间复杂性考察的是n固定时，不同输入实例下的算法所耗时间。最坏情况下的时间复杂性取的输入实例中最大的时间复杂度：

W(n) = max{ T(n，I) } ,   I∈Dn

平均时间复杂性是所有输入实例的处理时间与各自概率的乘积和：

A(n) =∑P(I)T(n，I)  I∈Dn

6. 设输入是一个按非降次序排列的元素表A[i：j] 和x，选取A[(i+j)/2]与x比较，如果

A[(i+j)/2]=x，则返回(i+j)/2，如果A[(i+j)/2]<x，则A[i：(i+j)/2-1]找x，否则在A[ (i+j)/2+1：j] 找x。上述过程被反复递归调用。

7. 不相同。目标函数：获得最大利润。最优量度：最大利润/重量比。 8. 问题的解可以表示为n元组：（x1,x2,……xn），xi∈Si, Si为有穷集合，xi∈Si, （x1,x2,……xn）具备完备性，即（x1,x2,……xn）是合理的，则（x1,x2,……xi）(i<n)一定合理。 9. 在解空间树上跳跃式地深度优先搜索，即用判定函数考察x[k]的取值，如果x[k]是合理的就搜索x[k]为根节点的子树，如果x[k]取完了所有的值，便回溯到x[k-1]。 10.将第K行的皇后分别与前k-1行的皇后比较，看是否与它们相容，如果不相容就返回false，测试完毕则返回true。

11.子问题的规模还很大时，必须继续使用分治法，反复分治，必然要用到递归。

12.最坏情况下的时间复杂性决定算法的优劣，并且最坏情况下的时间复杂性较平均时间复杂性游可操作性。

13.T(n)是某算法的时间复杂性函数，f(n)是一简单函数，存在正整数No和C，n〉No，有T(n)<f(n)，这种关系记作T(n)=O(f(n))。 14.二分检索算法的最多的比较次数为 log n 。

15.最坏情况下快速排序退化成冒泡排序，需要比较n2

次。

16.是一种依据最优化量度依次选择输入的分级处理方法。基本思路是：首先根据题意，选取一种量度标准；然后按这种量度标准对这n个输入排序，依次选择输入量加入部分解中。如果当前这个输入量的加入，不满足约束条件，则不把此输入加到这部分解中。 17.回溯法的解（x1,x2,……xn）的隐约束一般指个元素之间应满足的某种关系。

18.讲数组一分为二，分别对每个集合单独排序，然后将已排序的两个序列归并成一个含n个元素的分好类的序列。如果分割后子问题还很大，则继续分治，直到一个元素。 19.快速排序的基本思想是在待排序的N个记录中任意取一个记录，把该记录放在最终位置后，数据序列被此记录分成两部分。所有关键字比该记录关键字小的放在前一部分，所有比它大的放置在后一部分，并把该记录排在这两部分的中间，这个过程称作一次快速排序。之后重复上述过程，直到每一部分内只有一个记录为止。

20.在定义一个过程或者函数的时候又出现了调用本过程或者函数的成分，既调用它自己本身，这称为直接递归。如果过程或者函数P调用过程或者函数Q，Q又调用P，这个称为间接递归。消除递归一般要用到栈这种数据结构。

21.分治法的基本思想是将一个规模为n的问题分解为k个规模较小的子问题，这些子问题互相独立且与原问题相同。递归地解这些子问题，然后将各个子问题的解合并得到原问题的解。 22.设计动态规划算法的主要步骤为：

（1）找出最优解的性质，并刻划其结构特征。（2）递归地定义最优值。（3）以自底向上的方式计算出最优值。（4）根据计算最优值时得到的信息，构造最优解。

23.分治法与动态规划法的相同点是：将待求解的问题分解成若干个子问题，先求解子问题，然后从这些子问题的解得到原问题的解。

两者的不同点是：适合于用动态规划法求解的问题，经分解得到的子问题往往不是互相独立的。而用分治法求解的问题，经分解得到的子问题往往是互相独立的。

24.备忘录方法是动态规划算法的变形。与动态规划算法一样，备忘录方法用表格保存已解决的子问题的答案，在下次需要解此问题时，只要简单地查看该子问题的解答，而不必重新计算。

备忘录方法与动态规划算法不同的是，备忘录方法的递归方式是自顶向下的，而动态规划算法则是自底向上递归的。因此，备忘录方法的控制结构与直接递归方法的控制结构相同，区别在于备忘录方法为每个解过的子问题建立了备忘录以备需要时查看，避免了相同的子问题的重复求解，而直接递归方法没有此功能。

写出图着色问题的回溯算法的判断X[k]是否合理的过程。

解：i←0

while i<k do

if G[k,i]=1 and X[k]= X[i] then

return false

i←i+1

repeat

if i= k then return true

对于下图，写出图着色算法得出一种着色方案的过程。

解：K←1

X[1] ←1 , 返回 true

X[2]←1,返回false; X[2]←X[2]+1=2, 返回 true

X[3]←1 ,返回false; X[3]←X[3]+1=2, 返回false;X[3]←X[3]+1=3, 返回 true X[4]←1, 返回false; X[4]←X[4]+1=2, 返回false;X[4]←X[4]+1=3, 返回 true 找到一个解 （1，2，3，3）

写出归并排序算法对下列实例排序的过程。

(6,2,9,3,5,1,8,7)

解：调用第一层次 6,2,9,3 5,1,8,7 分成两个子问题 调用第二层次 6,2 9,3 5,1 8,7 分成四个子问题

调用第三层次 6 2 9 3 5 1 8 7 分成八个子问题

调用第四层次 只有一个元素返回上一层

第三层归并 2 ,6 3, 9 1,5 7,8 返回上一层

第二层归并 2 ,3,6, 9 1,5,7,8 返回上一层

第一层归并 1, 2 ,3, 5 ,6, 7, 8,9 排序结束，返回主函数

快速排序算法对下列实例排序，算法执行过程中，写出数组A第一次被分割的过程。

A=(65,70,75,80,85,55,50,2)

解：第一个分割元素为65

(1) (2) (3) (4) (5) (6) (7) (8) i p

65 2 75 80 85 55 50 70 3 7 65 2 50 80 85 55 75 70 4 6 65 2 50 55 85 80 75 70 4 6

55 70 75 80 85 65 50 2

写出用背包问题贪心算法解决下列实例的过程。

P=(18,12,4,1)

W=(12,10,8,3)

M=25

解: 实例符合P(i)/W(i)≥P(i+1)/W(i+1)的顺序。

CU←25,X←0

W[1]< CU: x[1]←1; CU←CU-W[1]=13;

W[2]< CU: x[2]←1; CU←CU-W[2]=3;

W[3]>CU: x[3]←CU/ W[3]=3/8;

实例的解为：（1，1，3/8，0）

有一个有序表为{1，3，9，12，32，41，45，62，75，77，82，95，100}，当使用二分查找值为82的结点时，经过多少次比较后查找成功并给出过程。

解：有一个有序表为{1，3，9，12，32，41，45，62，75，77，82，95，100}，当使用二分查找值为82的结点时，经过多少次比较后查找成功并给出过程。

一共要要执行四次才能找到值为82的数。

使用prim算法构造出如下图G的一棵最小生成树。

dist(1,2)=6;dist(2,5)=3;dist(5,6)=6;dist(6,4)=2;dist(4,1)=5;

dist(1,3)=1;dist(2,3)=5;dist(3,4)=5;dist(3,6)=4;dist(5,3)=6

解：使用普里姆算法构造出如下图G的一棵最小生成树。

dist(1,2)=6;dist(2,5)=3;dist(5,6)=6;dist(6,4)=2;dist(4,1)=5;

dist(1,3)=1;dist(2,3)=5;dist(3,4)=5;dist(3,6)=4;dist(5,3)=6

用分支限界法解装载问题时，对算法进行了一些改进，下面的程序段给出了改进部分；试说明斜线部分完成什么功能，以及这样做的原因，即采用这样的方式，算法在执行上有什么不同。

解答：斜线标识的部分完成的功能为：提前更新bestw值；

这样做可以尽早的进行对右子树的剪枝。具体为：算法Maxloading初始时将bestw设置为0，直到搜索到第一个叶结点时才更新bestw。因此在算法搜索到第一个叶子结点之前，总有bestw=0，r>0 故Ew+r>bestw总是成立。也就是说，此时右子树测试不起作用。

为了使上述右子树测试尽早生效，应提早更新bestw。又知算法最终找到的最优值是所求问题的子集树中所有可行结点相应重量的最大值。而结点所相应得重量仅在搜索进入左子树是增加，因此，可以在算法每一次进入左子树时更新bestw的值。

写出多段图最短路经动态规划算法求解下列实例的过程，并求出最优值。

各边的代价如下：

C(1,2)=3， C(1,3)=5 ，C(1,4)=2

C(2,6)=8 ，C(2,7)=4 ，C(3,5)=5 ，C(3,6)=4， C(4,5)=2，C(4,6)=1

C(5,8)=4， C(6,8)=5 ，C(7,8)=6

解：Cost(4,8)=0

Cost(3,7)= C(7,8)+0=6 ，D[5]=8

Cost(3,6)= C(6，8)+0=5, D[6]=8

Cost(3,5)= C(5，8)+0=4 D[7]=8

Cost(2,4)= min{C(4，6)+ Cost(3,6), C(4，5)+ Cost(3,5)}

= min{1+ 5, 2+4}=6 D[4]=6

Cost(2,3)= min{C(3，6)+ Cost(3,6) }

= min{4+5}=9 D[3]=5

Cost(2,2)= min{C(2，6)+ Cost(3,6), C(2，7)+ Cost(3,7)}

= min{8+5, 4+6}=10 D[2]=7

Cost(1,1)= min{C(1,2)+ Cost(2,2), C(1,3)+ Cost(2,3), C(1,4)+ Cost(2,4)} = min{3+10, 5+9,2+6}= 8

D[1]=4

1→4→6→8

能采用贪心算法求最优解的问题，一般具有的重要性质为：最优子结构性质与贪心选择性质

动态规划算法的基本要素为：最优子结构性质与重叠子问题性质

回溯法的效率不依赖于问题的解空间的形式

有11个待安排的活动，它们具有下表所示的开始时间与结束时间，如果以贪心算法求解这些活动的最优安排（即为活动安排问题：在所给的活动集合中选出最大的相容活动子集合），得到的最大相容活动子集合为活动（ {1，4，8，11} ）。

3. 所谓贪心选择性质是指（所求问题的整体最优解可以通过一系列局部最优的选择，即贪心选择来达到）。

4. 所谓最优子结构性质是指（问题的最优解包含了其子问题的最优解）。 5. 回溯法是指（具有限界函数的深度优先生成法）。

6. 用回溯法解题的一个显著特征是在搜索过程中动态产生问题的解空间。在任何时刻，算法只保存从根结点到当前扩展结点的路径。如果解空间树 中从根结点到叶结点的最长路径的长度为h(n)，则回溯法所需的计算空间通常为（O(h(n))）。

7. 回溯法的算法框架按照问题的解空间一般分为（子集树）算法框架与（排列树）算法框架。

8. 用回溯法解0/1背包问题时，该问题的解空间结构为（子集树）结构。

9.用回溯法解批处理作业调度问题时，该问题的解空间结构为（排列树）结构。

用回溯法解图的m着色问题时，使用下面的函数OK检查当前扩展结点的每一个儿子所相应的颜色的可用性，则需耗时（渐进时间上限）（O（mn））