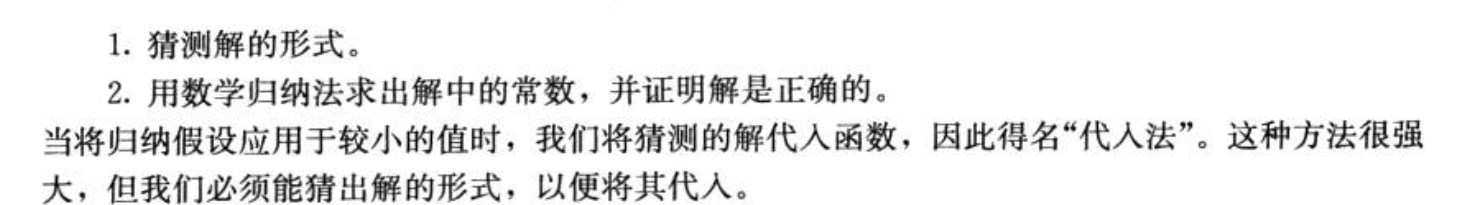
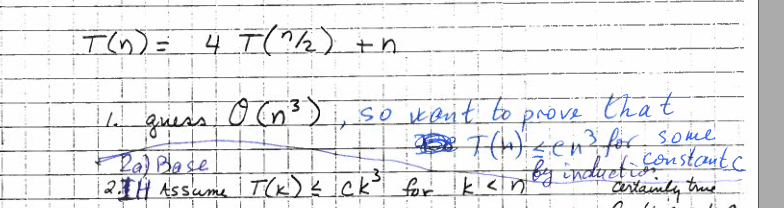
sorting recurrences

Substition method



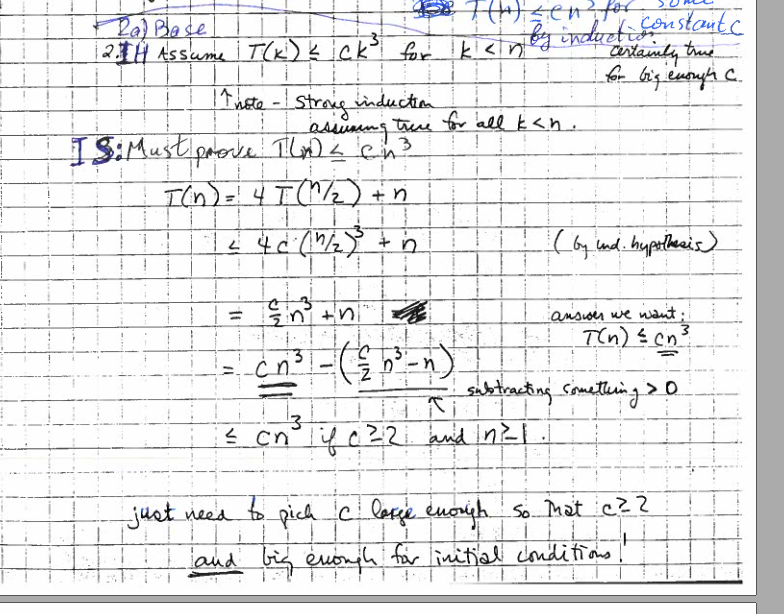


第一步：我们猜他的上线是n^3, //大多数情况都猜O

第二步：Induction数学归纳法

假设小于等于c \*猜的那个数

假设对所有k小于n成立



假设对于任意k都成立

第一步：我们猜他的上线是n^3, //大多数情况都猜O

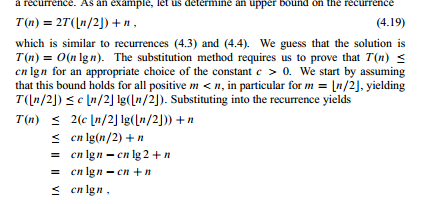
第二步：Induction数学归纳法

T(n)实际用时， c\*xxn ， 猜测用时

假设小于等于c \*猜的那个数

假设对所有k小于n成立

//注意和传统数学归纳法不同的是，k是下标，所以是k+1这样推的



而这里T(n)<=2T(n/2)+n是我们的已知事实

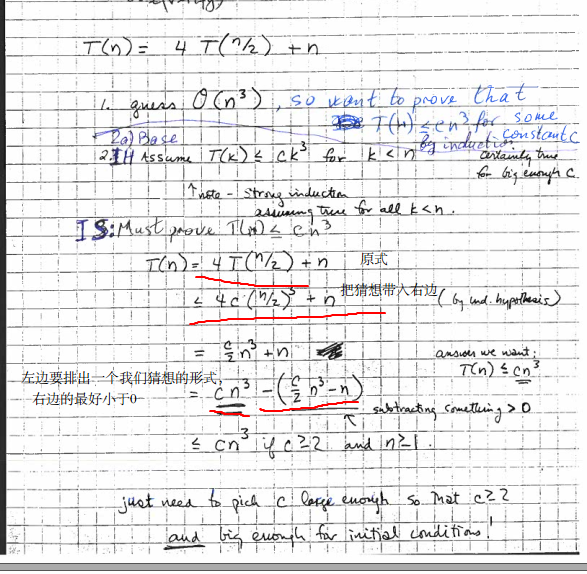
将右边的T(N/2)带入我们的推论

2T(N/2)+N<=2(cn/2lg(n/2))+n

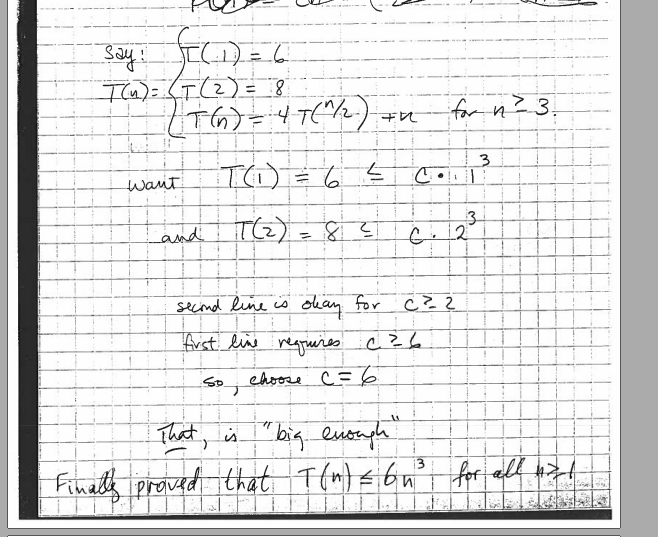
<=cnlgn-cnlg2+n

=cnlgn-cn+n //因为clg2还是constant

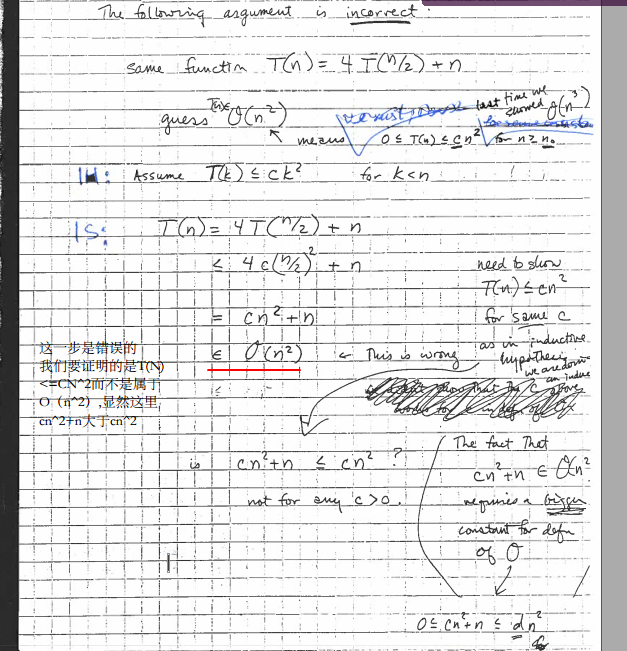
<=cnlgn

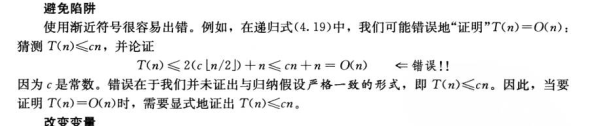


C只要big enough就行了，因为·算法我们更关注n到n^2的差距，而不是n到2n的差距



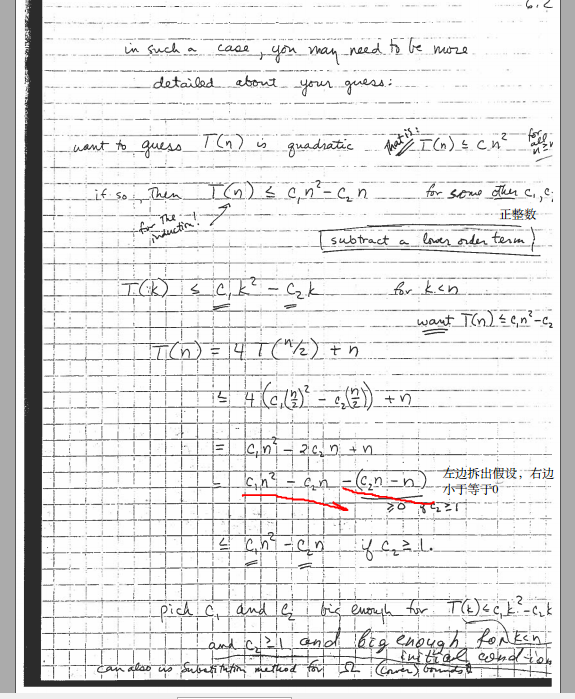
例如这里，我们设置的是T（n）最高是cn^3，选c=1,那么实际上是小于6的，但我们不在乎，可以把c=6





所以在这种情况下我们或许需要进一步精确我们的假设

注意，减去一格低阶项是一个很常见的想法，如果如果我们得到了一个O（n^2）这样一个差的不远的东西·，我们第一个就该猜这个

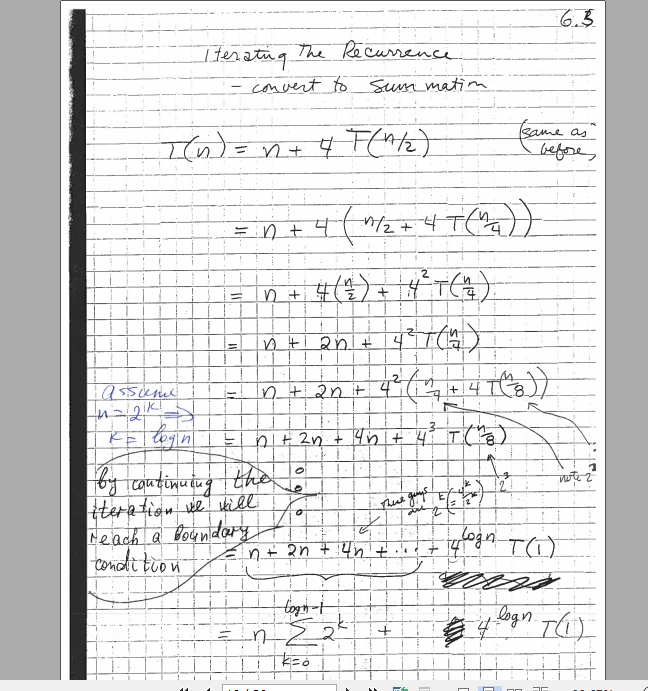


假设上限位c1n^2-c2n

iterating法 迭代法

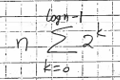
这种方法不用猜一个解，但是需要更强的数学基础

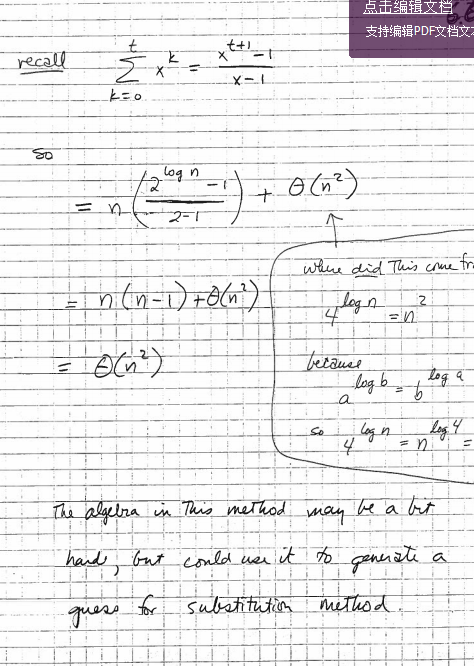
我们通过迭代把整个式子展开求和



T(N/2)自然等于n/2+4T(n/4)

上面展开我们可以发现，

n+2n+4n+...= 

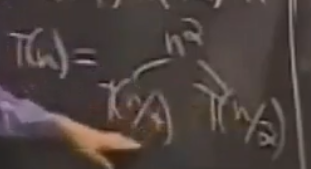


这个方法有点吃数学，跳过

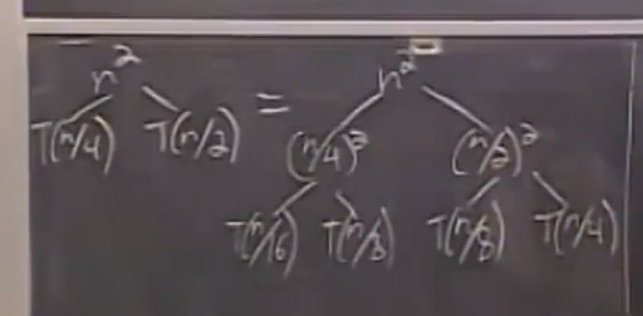
recursion-tree方法

例如T(n)=T(n/4)+T(n/2)+n^2

第一步，把非recursion项提取出来，作为root，把recursion项作为leaf,构成初始树

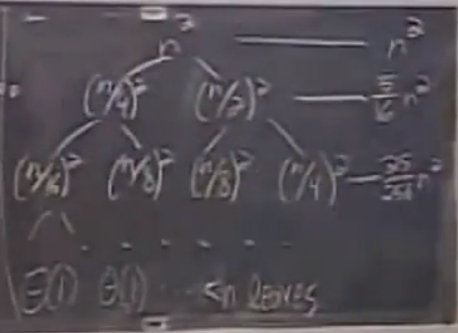
 //左边是n./4，画质不行

然后递归展开叶节点

，我们可以无限递归，最终底部变成常数叶节点，

但是我们是不知道常数叶节点多少的，因为不同的左右子树递归速度不一样

但我们没必要完全展开树



每层都横着加起来

而这一列我们加起来的数加起来，实际上等于整个树的值，而整个树的和，就等于T（n）//它是由T（n）拓展出来的

1+5/16+25/256+.....5^k/16^k

甚至这里也没必要仔细算

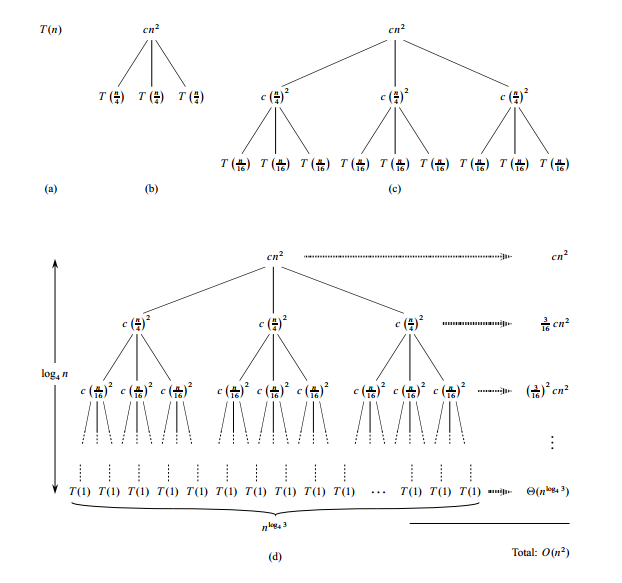
因为他小于1+1/2+1/4+1/8+....=2

所以O(n^2)就是一个合理的upper bound

你可以把每一个节点，看成是这一次recursion所需要的代价，

这个方法主要目的是得到一个好的guess，然后我们可以用substition 法来猜测

书里的例子是T=3T(n/4)+cn^2



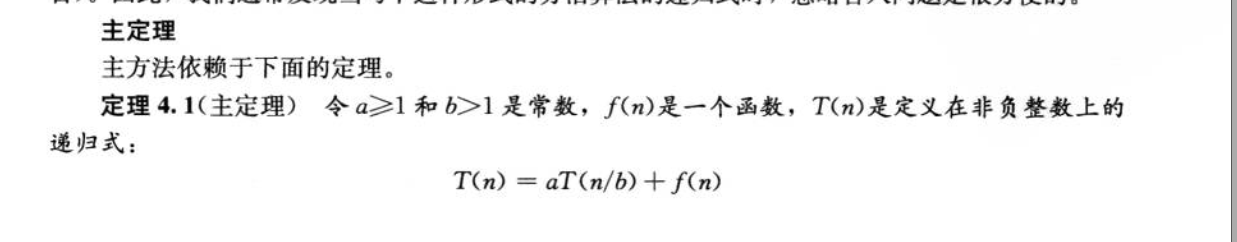
他把3T(n/4)拆分成三个T（n/4）

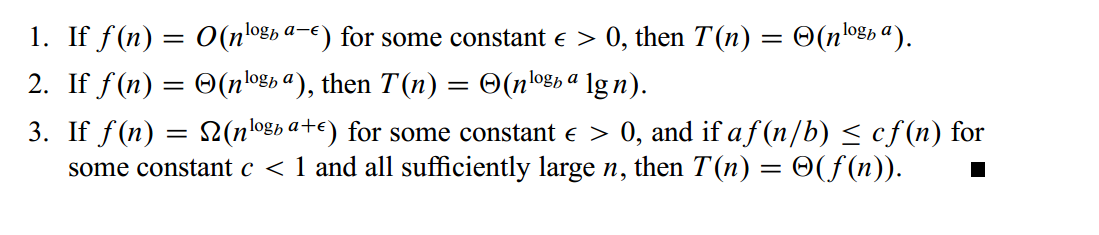
Master theorem

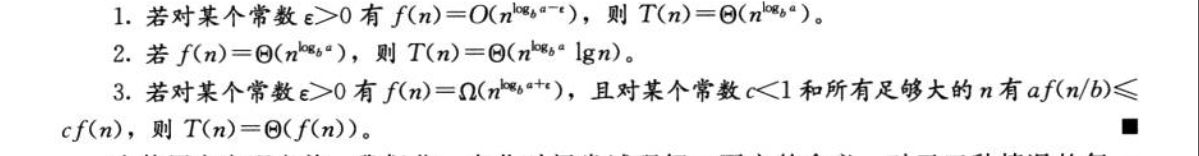
更多像recursion tree的一个应用，但更精确，但是master method限制也更多，只能应用于特定的RECURSION式子

只有这个形式才能用master method

a要大于等于1，b要大于1,f（n）是渐进正函数





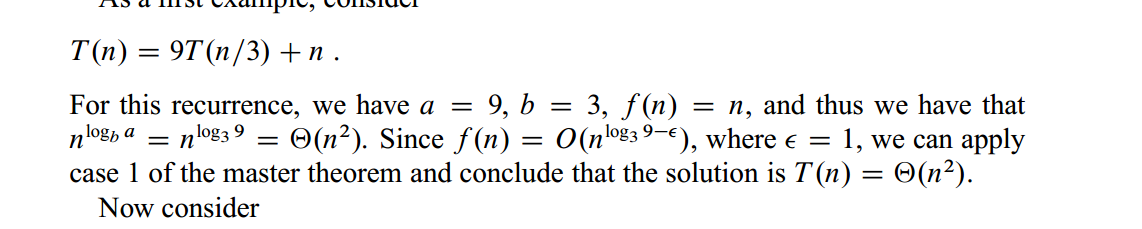


可能这个EPSILON比较令人费解，但要记住，我们比较的核心是n^logba

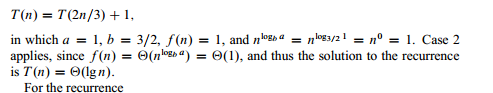
第一种情况就是fn比nlogba小

第二种情况就是同一类型，例如n^2与2n^2

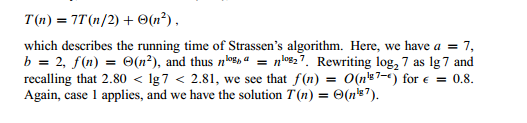
第三种情况就是fn大于nlogba



n^logba=n^2



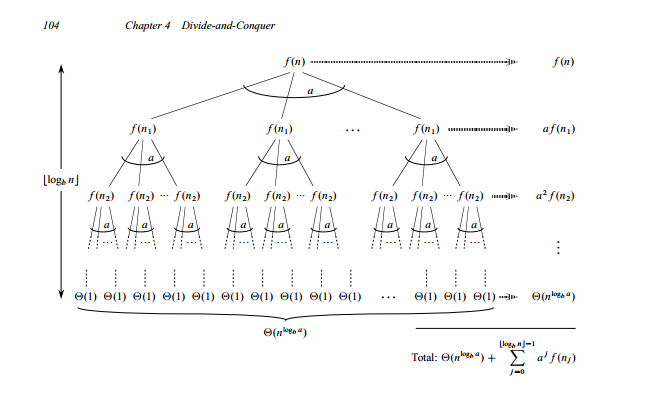
logxx 1=0



2,.8>2

实际上这就是个树theorem





f(n)就是第一次recur所需要的常熟代价

fn1实际上就是n/b

n2就是n/b^2

每次能分出a项，

一共是n，每次除到1，相当于b^x=n, 那么x也就是高度等于logbn

那么每次分出a项，a^x个常数，根据log的性质，等于n^log b a