## 1.对搜索树的新的认识

刚才在写代码的时候我有了一个对回溯法的搜索空间树的理解的一个新的想法：在以前我一直强调的是搜索树的边（因为边构成了最终的解，有问题中有多少个需要赋值的对象就要有多少条边），而忽视了结点的意义。

今天在对最大团问题的程序写注释的时候，我发现其实搜索树中的结点也有其实际的含义，解释如下：

回溯法对解空间进行搜索时所采用的搜索策略是深度优先搜索，因此如果可能的话，一次从头结点到叶子结点就能够得到问题的一个解。因此对于回溯法的搜索树可以这样理解：结点对应于问题中的结点，每个结点发出的边则对应于对该结点进行赋值。

还有一个问题：在搜索树中边的个数等于问题中需要赋值的结点的个数，那么结点的个数就等于边的个数加一，这是什么原因呢？

对问题中的最后一个结点（注意这是解空间树的倒数第二个结点）进行赋值后，表示我们已经得到了问题的完整解，则我们可以是搜素过程通知下来。具体的实现方法是：让回溯函数继续往下搜索一次，则搜索层次（也就是BachTrack的参数）会加一，在回溯函数写一条if判断语句，一旦搜索层次比实际问题中需要赋值的结点数大一的时候，表示搜索已经完成，可以推出当前路径的搜索，进入其它的路径。

## 2.为什么最终的结果一定是最优值

也就是说：是否应该像迭代法求最值那样，添加一个判断？

if (i >= n){//问题中的结点的编号是0，1，2.....(n-1)

for (int j = 0; j < n; j++){

bestx[j] = x[j];

}

bestcn = cn;//这类似迭代的思想：只要

return;

}

## 3.对比m着色的代码和对大团的代码：在最大团的if（i>n）中有return，而m着色中则没有，why？

显然这两个问题的性质是完全不同的：m着色是满足性搜索，只要符合条件的解都搜索出来。而最大团问题是最值搜索，需要找出最值。