

深度优先搜索 Depth First Search

课程版本 v4.2 主讲 令狐冲



扫描二维码关注微信/微博
获取最新面试题及权威解答

微信: [ninechapter](#)

微博: <http://www.weibo.com/ninechapter>

知乎: <http://zhuoanlan.zhihu.com/jiuzhang>

官网: <http://www.jiuzhang.com>

- Recursion
- Combination
- Permutation
- Graph
- Non-Recursion

什么时候使用 DFS？

回顾：还记得什么时候使用 BFS 么？

独孤九剑 —— 破鞭式

碰到让你找所有方案的题，一定是DFS

90%DFS的题，要么是排列，要么是组合

问题模型: 求出所有满足条件的“组合”。

判断条件: 组合中的元素是顺序无关的。

时间复杂度: 与 2^n 相关。

我们在第一节课中讲过的全子集问题(<http://www.lintcode.com/problem/subsets/>)就是典型的组合搜索问题。

一般来说, 如果面试官不特别要求的话, DFS都可以使用递归(Recursion)的方式来实现。

递归三要素是实现递归的重要步骤:

- 递归的定义
- 递归的拆解
- 递归的出口

令狐师兄带你写代码之 Combination Sum

<http://www.lintcode.com/problem/combination-sum/>

<http://www.jiuzhang.com/solutions/combination-sum/>

问：和subsets的区别有哪些？

与 Subsets 比较

- Combination Sum 限制了组合中的数之和
 - 加入一个新的参数来限制
- Subsets 无重复元素, Combination Sum 有重复元素
 - 需要先去重
- Subsets 一个数只能选一次, Combination Sum 一个数可以选很多次
 - 搜索时从 index 开始而不是从 index + 1

Combination Sum II

<http://www.lintcode.com/problem/combination-sum-ii/>

<http://www.jiuzhang.com/solutions/combination-sum-ii/>

问：如何去重？

Palindrome Partitioning

<http://www.lintcode.com/problem/palindrome-partitioning/>

<http://www.jiuzhang.com/solutions/palindrome-partitioning/>

问:有什么可以优化的地方?

休息5分钟

调查问卷

问题模型: 求出所有满足条件的“**排列**”。

判断条件: 组合中的元素是顺序“**相关**”的。

时间复杂度: 与 $n!$ 相关。

Permutations

<http://www.lintcode.com/problem/permutations/>

<http://www.jiuzhang.com/solutions/permutations/>

Permutations II

<http://www.lintcode.com/problem/permutations-ii/>

<http://www.jiuzhang.com/solutions/permutations-ii/>

问：如何去重？

N Queens

<http://www.lintcode.com/problem/n-queens/>

<http://www.jiuzhang.com/solutions/n-queens/>

通用的DFS时间复杂度计算公式

$O(\text{答案个数} * \text{构造每个答案的时间})$

<http://www.jiuzhang.com/qa/2994/>

Search in a Graph

图中的搜索

Word Ladder

<http://www.lintcode.com/problem/word-ladder/>

<http://www.jiuzhang.com/solutions/word-ladder/>

Word Ladder II

<http://www.lintcode.com/problem/word-ladder-ii/>

<http://www.jiuzhang.com/solutions/word-ladder-ii/>

Stack - Non Recursion

要诀：基本上都会用上栈(Stack)

Tree Traversal

<http://www.jiuzhang.com/solutions/binary-tree-preorder-traversal/>

<http://www.jiuzhang.com/solutions/binary-tree-inorder-traversal/>

<http://www.jiuzhang.com/solutions/binary-tree-postorder-traversal/>

<http://www.jiuzhang.com/solutions/binary-search-tree-iterator/>

Combination

<http://www.jiuzhang.com/solutions/subsets/>

Permutation

<http://www.jiuzhang.com/solutions/permutations/>

- 什么时候用 DFS？
 - 求所有方案时
- 怎么解决DFS？
 - 不是排列就是组合
- 复杂度怎么算？
 - $O(\text{答案个数} * \text{构造每个答案的时间复杂度})$
- 非递归怎么办？
 - 必“背”程序