

# 组合型搜索与记忆化搜索

## Combination-based DFS & Memoization Search

课程版本 v5.2    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://zhuanglan.zhihu.com/jiuzhang>

官网: <http://www.jiuzhang.com>

- 请在随课教程中自学如下先修知识：
  - <http://www.jiuzhang.com/tutorial/algorithm/19>
  - 组合类搜索入门问题全子集问题 (Subset) 及其 4 种解法 (课前只需学习递归的 2 种解法)
  - 什么是 Deep copy, 为什么需要 Deep copy?
- 深度优先搜索 DFS 的判断条件 (什么时候用 DFS)
- 递归三要素
- 组合类搜索入门题
- 深度优先搜索的时间复杂度分析

# 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，基本可以确定是 DFS

除了二叉树以外的 90% DFS 的题，要么是排列，要么是组合

# 求所有方案可以用 BFS 么？

说说看你的想法

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。

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

# 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

# k Sum II

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

<http://www.jiuzhang.com/solution/k-sum-ii/>

找出所有  $k$  个数之和 = target 的组合

# Split String

<http://www.lintcode.com/problem/split-string/>

<http://www.jiuzhang.com/solution/split-string/>

# 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?

# 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/>

# 休息5分钟

Take a break

# 记忆化搜索 Memoization Search

注意不是 Memorization

将函数的计算结果保存下来，下次通过同样的参数访问时，直接返回保存下来的结果

问：

1. 对这个函数有什么限制条件没有？
2. 和系统设计中的什么很像？

记忆化搜索通常能够将指数级别的时间复杂度降低到多项式级别。



# Wildcard Matching

<http://www.lintcode.com/problem/wildcard-matching/>

<http://www.jiuzhang.com/solution/wildcard-matching/>

# Follow up: Regular Expression Matching

<http://www.lintcode.com/problem/regular-expression-matching/>

<http://www.jiuzhang.com/solution/regular-expression-matching/>

面试是一定不会让你做完整版的 Regular Expression 的  
所以一定是阉割版的

**Strong Hire:** 两个都答出来, 且写出来, Bug Free or Bug 很少

**Hire / Weak Hire:** 两个都答出来, 写完第一个, 第二个能基本在第一个的基础上改完, 允许有一些提示和少量 Bug

**No Hire:** 没写完, 或者需要很多提示

**Strong No:** 第一个都没写完

# Word Break II

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

<http://www.jiuzhang.com/solution/word-break-ii/>

**Strong Hire:** DFS+DP优化

**Hire / Weak Hire:** DFS 能写完, 且 Bug free or Bug 不多, 不需要提示 or 需要少量提示

**No Hire:** DFS 写不完, 或者需要很多提示

**Strong No:** 啥都想不出

# 记忆化搜索 = 动态规划(DP)

记忆化搜索是动态规划的一种实现方式

记忆化搜索是用搜索的方式实现了动态规划

因此记忆化搜索, 就是动态规划

# 学习动态规划？ Dynamic Programming

《九章算法强化班》

<https://www.jiuzhang.com/course/5/>

《动态规划专题班》

<https://www.jiuzhang.com/course/12/>

DFS算法的掌握, 主要在练习

一个题第一遍不顺利, 就要写第二遍, 第三遍

像 Word Break II 纯 DFS 版本 和 Regular Expression Matching 这样的问题, 要练到 30 分钟内在 LintCode 上 AC。做不到就反复再练。