第二节

九章算法

数据结构(上)--并查集与字典树

讲师：侯卫东

加班主任，进班级答疑群
快速获取面试资料/课程福利

关注公众号，了解大厂资讯

版权声明

九章的所有课程均受法律保护，不允许录像与传播录像

一经发现，将被追究法律责任和赔偿经济损失

今日大纲

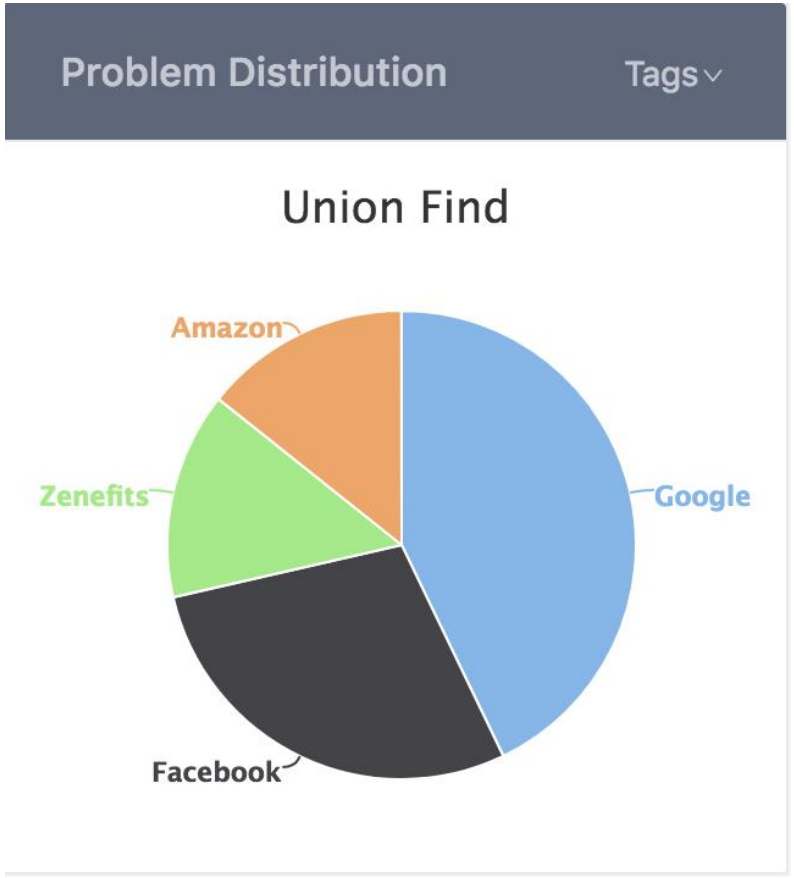
面试大厂的必备数据结构

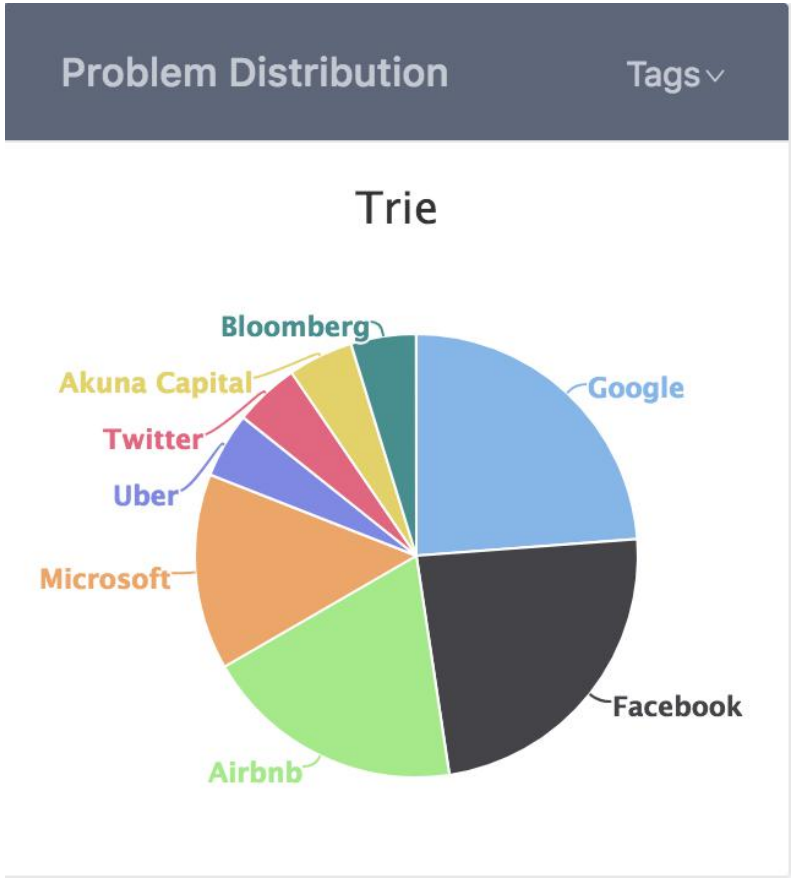
并查集 Union Find

- 可以解决什么问题
- 代码模板
- 例题

字典树 Trie

- 可以解决什么问题
- 代码模板
- 例题





并查集 Union Find

一种用于支持集合**快速合并和查找**操作的数据结构

O(1) 合并两个集合 - Union

O(1) 查询元素所属集合 - Find

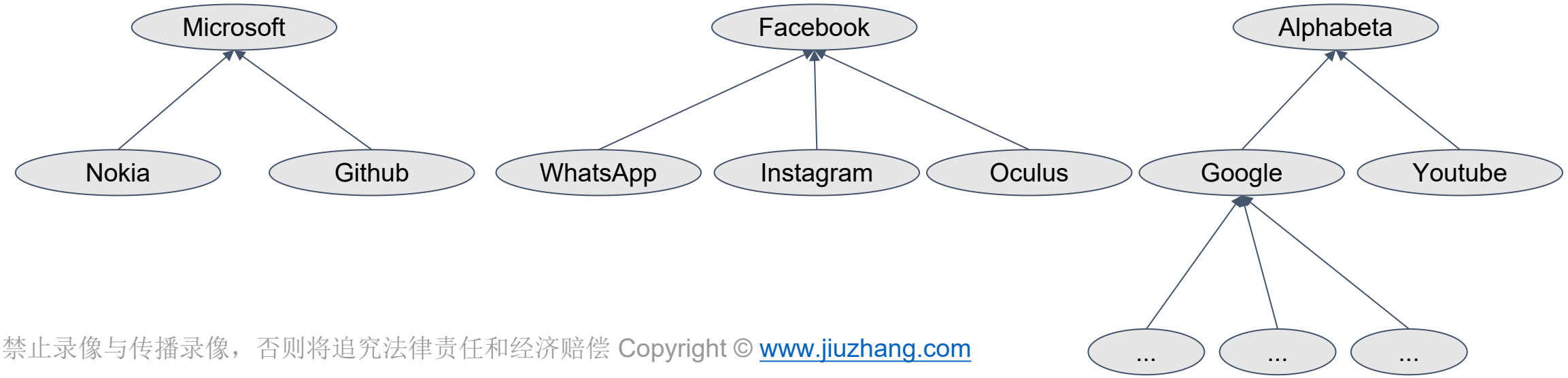
并查集

公司并购 —— 合并两个集合

查询子公司所在集团 —— 查询所在集合

判断两个子公司是否在同一家集团

Union Find 是一棵多叉树



如何实现 Union Find

底层数据结构

- 父亲表示法，用一个数组/哈希表记录每个节点的父亲是谁。
- father[“Nokia”] = “Microsoft”
- father[“Instagram”] = “Facebook”

查询所在集合

- 用所在集合最顶层的老大哥节点来代表这个集合

合并两个集合

- 找到两个集合中最顶层的两个老大哥节点 A 和 B
- father[A] = B // or father[B] = A 如果无所谓谁合并谁的话

代码模板 - 初始化

使用哈希表或者数组来存储每个节点的父亲节点

如果节点不是连续整数的话，就最好用哈希表来存储

最开始所有的父亲节点都指向自己

```
int[] f = new int[n];
for (int i = 0; i < n; ++i) {
    f[i] = i;
}
```

注：

也可以将父亲节点指向空

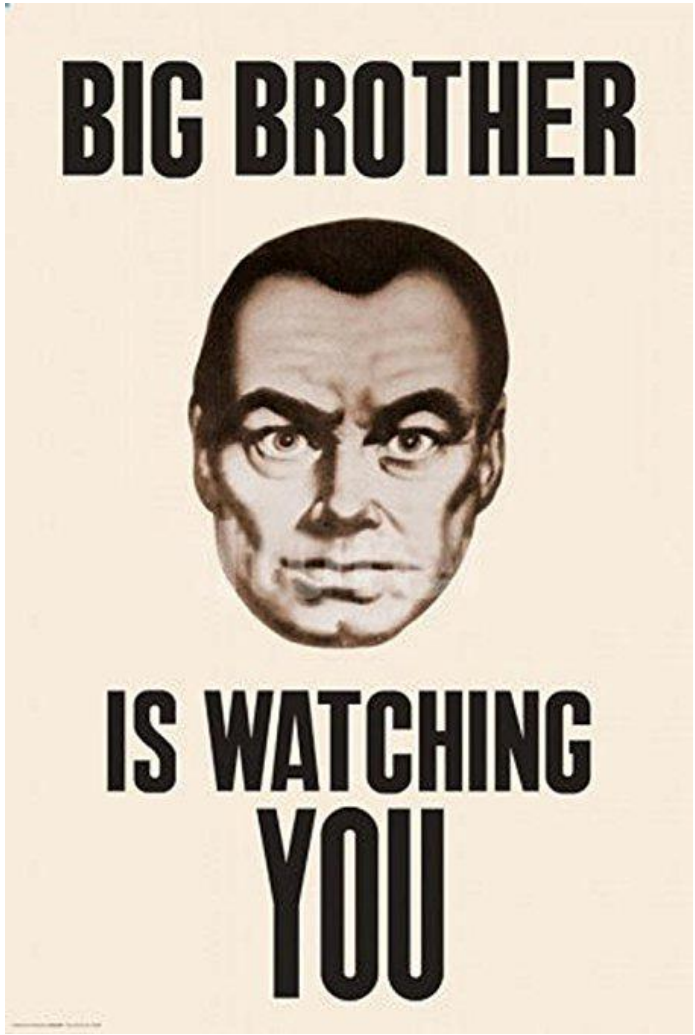
代码模板 - 查找老大哥

沿着父亲节点一路往上走就能找到老大哥

下面这份代码有什么问题？

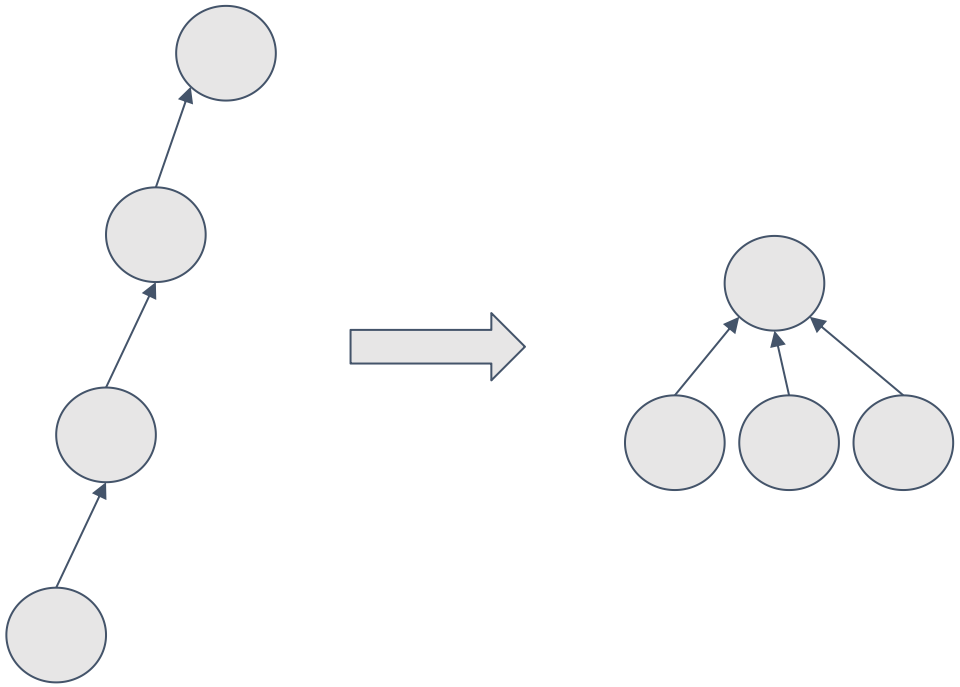
```
int find(int x, int[] f) {
    while (f[x] != x) {
        x = f[x];
    }

    return x;
}
```



代码模板 - 路径压缩

路径压缩 —— 在找到老大哥以后，还需要把一路上经过的点都指向老大哥



```java
int find(int x, int[] f) {
    int j, k, fx;
    j = x;
    while (f[j] != j) {
        j = f[j];
    }

    while (x != j) {
        fx = f[x];
        f[x] = j;
        x = fx;
    }

    return j;
}
```

代码模板 - 集合合并

找到两个元素所在集合的两个老大哥 A 和 B

将其中一个老大哥的父指针指向另外一个老大哥

```
void merge(int x, int y, int[] f) {
    int fx = find(x, f);
    int fy = find(y, f);
    if(fx != fy) {
        f[fx] = fy;
    }
}
```

x	$\lg^* x$
$(-\infty, 1]$	0
$(1, 2]$	1
$(2, 4]$	2
$(4, 16]$	3
$(16, 65536]$	4
$(65536, 2^{65536}]$	5

时间复杂度

都是 O(log* n)，约等于 O(1)

https://en.wikipedia.org/wiki/Proof_of_O(log*n)_time_complexity_of_union%E2%80%93find

https://en.wikipedia.org/wiki/Iterated_logarithm

Connecting Graph

http://www.lintcode.com/problem/connecting-graph/

https://www.jiuzhang.com/solution/connecting-graph/

LintCode 589

给定n个图中的节点，一开始节点之间没有边。需要支持如下操作：

1. connect(a,b)， 连接a点和b点
2. query(a,b)， 询问a点和b点是否在图中连通

例子

5 // n = 5
query(1, 2) 输出 false
connect(1, 2)
query(1, 3) 输出 false
connect(2, 4)
query(1, 4) 输出 true

解法

使用并查集，每次对于a和b，找到各自的老大哥节点A和B，其中需要路径压缩

如果A不等于B，将A树指向B树

Connecting Graph II

http://www.lintcode.com/problem/connecting-graph-ii/

https://www.jiuzhang.com/solution/connecting-graph-ii/

LintCode 590

给定n个图中的节点，一开始节点之间没有边。需要支持如下操作：

1. connect(a,b)， 连接a点和b点
2. query(a)， 询问a点所在连通块的节点个数

例子

5 // n = 5
query(1) 输出 1
connect(1, 2)
query(1) 输出 2
connect(2, 4)
query(1) 输出 3

解法

使用并查集，每次对于a和b，找到各自的老大哥节点A和B，其中进行路径压缩

老大哥节点记录下自己的子树的节点个数

如果A不等于B，将A树指向B树。B树老大哥节点更新节点个数

FollowUp：每个点有权值，问A点所在连通块的权值总和/最大权值

Connecting Graph III

http://www.lintcode.com/problem/connecting-graph-iii/

https://www.jiuzhang.com/solution/connecting-graph-iii/

LintCode 591

给定n个图中的节点，一开始节点之间没有边。需要支持如下操作：

1. connect(a,b)， 连接a点和b点
2. query()， 询问连通块数目

例子

5 // n = 5
query() 输出 5
connect(1, 2)
query() 输出 4
connect(2, 4)
query() 输出 3

解法

使用并查集，每次对于a和b，找到各自的老大哥节点A和B，其中进行路径压缩

如果A不等于B，将A树指向B树。连通块数目减1

FollowUp：每个点有权值，问当前所有连通块的最大平均权值

Number of Islands II

https://www.lintcode.com/problem/number-of-islands-ii/

https://www.jiuzhang.com/solution/number-of-islands-ii/

LintCode 434

给定一个mxn矩阵，一开始每个格子都是大海。然后给定一些格子要依次改成岛屿，需要返回每次一个格子改成岛屿后，当前连通岛屿的个数

例子

输入：n = 3, m = 3
改成岛屿的格子坐标：A = [(0,0),(0,1),(2,2),(2,1)]

输出： [1, 1, 2, 2]

解法

使用并查集，每个格子作为一个节点。当一个格子变成岛屿，和它的四个邻居依次连接，相当于在图中加四条边

并查集在二维中的拓展

Graph Valid Tree

https://www.lintcode.com/problem/graph-valid-tree/

https://www.jiuzhang.com/solution/graph-valid-tree/

LintCode 178

给定n个节点和一些无向边，判断是否形成一棵树。

例子

输入： n=5, edge=[[0, 1], [0, 2], [0, 3], [1, 4]]

输出：true

输入： n=5, edge= [[0, 1], [1, 2], [2, 3], [1, 3], [1, 4]]

输出：false

解法

- 使用并查集，将所有边加入

- 形成树有两个条件：
  - n-1条边
  - 最后只有一个连通块

Accounts Merge

https://www.lintcode.com/problem/accounts-merge/

http://www.jiuzhang.com/solution/accounts-merge/

LintCode 1070

给定一些账户，每个账户有一个用户名和一些关联邮箱。如果两个账户含有相同的关联邮箱，则这两个账户同属于一个人。但是不同的人可能有相同的用户名。输出合并后的账户，一个人一个账户。

输入： accounts = [["John", "johnsmith@mail.com", "john00@mail.com"],
["John", "johnnybravo@mail.com"],
["John", "johnsmith@mail.com", "john_newyork@mail.com"],
["Mary", "mary@mail.com"]]

输出： [["John", 'john00@mail.com', 'john_newyork@mail.com', 'johnsmith@mail.com'], ["John", "johnnybravo@mail.com"], ["Mary", "mary@mail.com"]]

解法

- 一种做法是将每个原始账户作为一个节点，但是两个账户可以连接取决于它们共享至少一个邮箱，处理起来比较麻烦

- 我们可以将每个邮箱作为一个节点，同一个原始账户中的邮箱之间连边。并用老大哥节点存储用户名

- 灵活定义并查集的节点

其他 Union Find 的练习题

https://www.lintcode.com/problem/set-union/

类似 Accounts Merge，只是少了用户名

https://www.lintcode.com/problem/maximum-association-set

需要输出最大的连通块。可以在并查集合并过程中打擂台，也可以在最后每个点找一次老大哥节点。

跟连通性有关的问题

都可以使用 BFS 和 Union Find

什么时候无法使用 Union Find?

跟连通性有关的问题

都可以使用 BFS 和 Union Find

什么时候无法使用 Union Find?

需要拆开两个集合的时候无法使用Union Find

并查集总结

1. 合并两个集合
2. 查询某个元素所在集合
3. 判断两个元素是否在同一个集合
4. 获得某个集合的元素个数
5. 统计当前集合个数

关键操作：快速寻找老大哥节点

课间休息五分钟

字典树 Trie

又名 Prefix Tree

来自单词 Retrieval，发音与 Tree 相同

Trie 的考点

实现一个 Trie

比较 Trie 和 Hash 的优劣

字符矩阵类问题使用 Trie 比 Hash 更高效

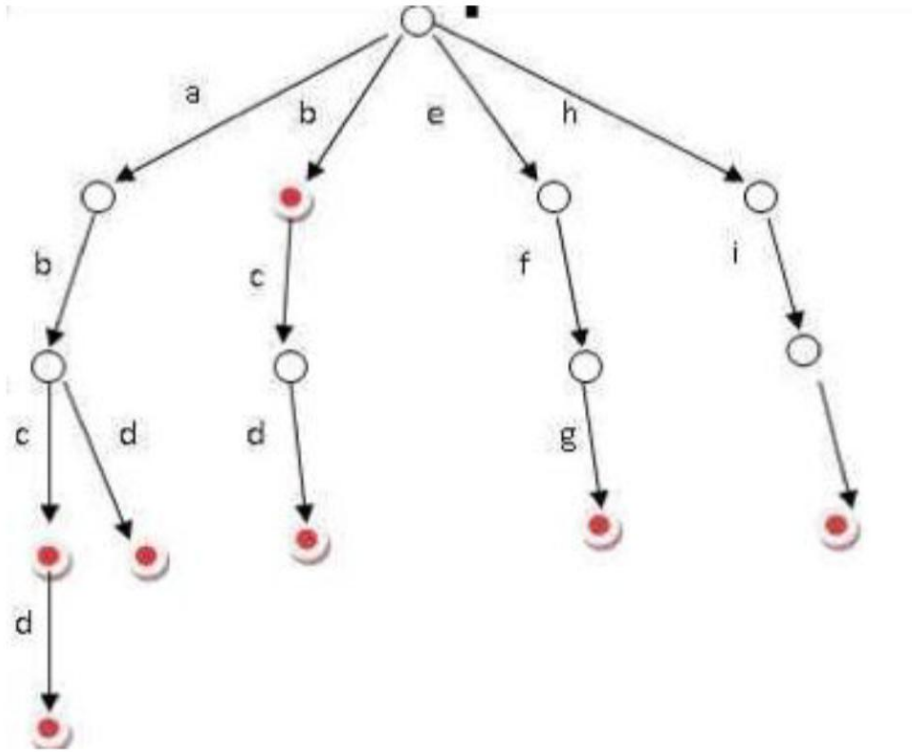
Implement Trie

https://www.lintcode.com/problem/implement-trie/

https://www.jiuzhang.com/solution/implement-trie/

LintCode 442

假设有 [b，abc，abd，bcd，abcd，efg，hii] 这7个单词，查找abc 在不在字典里面



Implement Trie - 代码模板

需要一个新的类 TrieNode 代表 Trie 中的节点

Insert 插入一个单词

```java
class TrieNode{
    public TrieNode[] sons;
    public boolean isWord;
    public String word;

    public TrieNode() {
        int i;
        sons = new TrieNode[26];
        for (i = 0; i < 26; ++i) {
            sons[i] = null;
        }

        isWord = false;
    }

    static public void Insert(TrieNode p, String word) {
        int i;
        char[] s = word.toCharArray();
        for (i = 0; i < s.length; ++i) {
            int c = s[i] - 'a';
            if (p.sons[c] == null) {
                p.sons[c] = new TrieNode();
            }

            p = p.sons[c];
        }

        p.isWord = true;
        p.word = word;
    }
}
```

Add and Search Word

https://www.lintcode.com/problem/add-and-search-word/

https://www.jiuzhang.com/solution/add-and-search-word/

LintCode 473

支持两种字符串操作：

addWord(word): 加入一个词

search(word)：搜索一个词，其中可能有”.”，代表任何单个字符

例子：

addWord("bad")

addWord("dad")

addWord("mad")

search("pad") → false

search("bad") → true

search(".ad") → true

解法

- addWord使用Trie

- searchWord在Trie中DFS，一旦需要”.”字符就遍历所有儿子节点

编程

Word Squares

https://www.lintcode.com/problem/word-squares/

https://www.jiuzhang.com/solution/word-squares/

LintCode 634

- 给出一系列 **不重复的单词**，找出所有用这些单词能构成的 单词平方。
- 单词平方是一个kxk的单词方阵：从第 k 行的单词和第 k 列的单词相同

```
b a l l
a r e a
l e a d
l a d y
```

- 例子：
- 输入：["area","lead","wall","lady","ball"]
- 输出
  例如，[["wall","area","lead","lady"],["ball","area","lead","lady"]]

- 限定：单词个数<=1000，单词长度在1到5之间

分析

- 直接搜索，时间复杂度最高1000^5
- 查找可以剪枝的部分

分析

- 剪枝一:
  - 第一个词填了ball后，第二个词必须以a开头
  - 第二个词填了area后，第三个词必须以le开头
  - 以其他开头的就没必要搜下去了

b	a	l	l
a			
l			
l			

分析

- 剪枝一:
  - 怎么实现?
    - 用Hash or Trie树记录下以某个前缀开头的有哪些单词
    - 比如以l开头的有lead lady，以le开头的有lead，以lea开头的有lead
    - 每次只用从特定开头的单词中继续往后搜

b	a	l	l
a			
l			
l			

lead lady	lead	lead	lead
l	le	lea	lead

分析

- 剪枝二:
  - 第一个词填了ball
  - 第二个词想填area的话
  - 字典中必须有以le la开头的单词，否则没有的话就不能填area

b	a	l	l
a	r	e	a
l	e		
l	a		

Word Search II

https://www.lintcode.com/problem/word-search-ii/

https://www.jiuzhang.com/solution/word-search-ii/

LintCode 132

- 给定一个小写字母矩阵和一个字典。找到字典中所有在矩阵中出现的词。一个词可以在矩阵中任意位置开始，然后向上下左右一个方向走一步。一个格子在一个词里只能用一次。

- 例子：
- 输入：

  [“doaf”,
  “agai”,
  “dcan”]

  {"dog", "dad", "dgdg", "can", "again"}
- 输出： {"dog", "dad", "can", "again"}

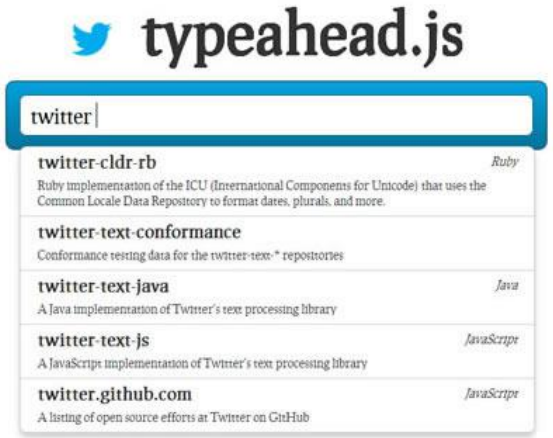
题解

- 用Trie存储字典里所有词
- 在矩阵中dfs时，在Trie里对应节点向下走
- Trie可以帮助剪枝

编程

Typeahead

Trie 在系统设计中的运用



总结

- 并查集Union Find
  - 路径压缩寻找老大哥节点
  - 动态合并集合与查询节点所在集合
  - 不能分拆集合

- 字典树Trie
  - 合并所有公共的前缀
  - 动态插入与查询单词
  - 不能查询非前缀（如字符串一部分）

谢谢！

扫描二维码关注微信小程序/公众号
获取第一手求职资料