# 并查集 Disjoint Set

跳跃性

适用场景：

- 组团、配对问题
- Group or not?

## 基本操作

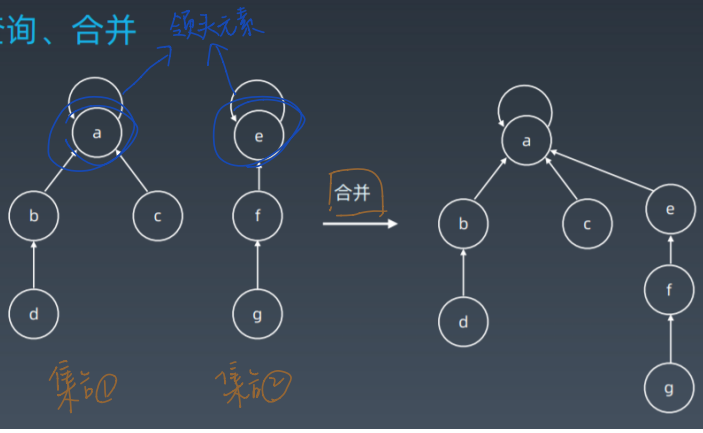
- makeSet(s)：建立一个新的并查集，其中包含 s 个单元素集合。
- unionSet(x, y)：把元素 x 和元素 y 所在的集合合并，要求 x 和 y 所在的集合不相交，如果相交则不合并。
- find(x)：找到元素 x 所在的集合的代表，该操作也可以用于判断两个元素是否位于同一个集合，只要将它们各自的代表比较一下就可以了。

## 初始化

(一开始) 每一个元素，拥有一个 parent 数组指向自己 —素→ 它自己的话就是自己的集合

```
parent[i] = i
```


## 查询、合并




如何查询：

看它的 parent 再看它的 parent 就一直往上，直到它的 parent 等于它自己的时候，说明找到了它的领头元素 (i.e. 集合的代表元素)

找出集合的领头元素

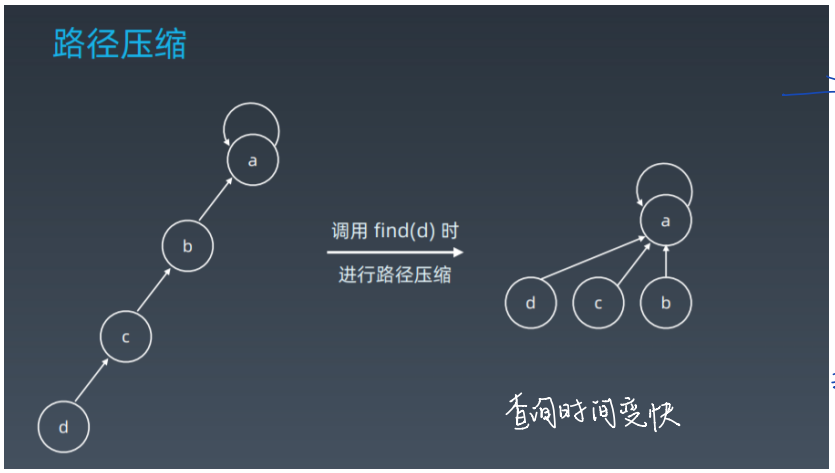
↓

将 parent[e] 指向 a

or 或

将 parent[a] 指向 e

(a的parent指向e)




d 的 parent 是 c

c 的 parent 是 b

b 的 parent 是 a

⇓

可以直接把这条路上的所有元素都指向 a

## Java 实现

```
class UnionFind {
  private int count = 0; // size
  private int[] parent; // parent
  public UnionFind(int n) {
    count = n;
    parent = new int[n];
    for (int i = 0; i < n; i++) {
      parent[i] = i; //初使化
    }
  }
  public int find(int p) {
    while (p != parent[p]) {
      parent[p] = parent[parent[p]];
      p = parent[p];
    }
    return p;
  }
  public void union(int p, int q) {
    int rootP = find(p);
    int rootQ = find(q);
    if (rootP == rootQ) return;
    parent[rootP] = rootQ;
    count--;
  }
}
```

初使化：parent[i] = i

寻找结点p所在的集合编号 → 怎么找集合

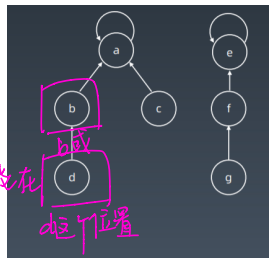
不断判断，parent[p] ≠ p 的时候，继续往上找，parent[p] 就等于 parent[p] 的 parent

然后把 parent[p] 赋给 p 本身

怎么找它集合和它的集合的领头元素

！不能直接用 parent[p] 结果返回去

∵ 大多数情况




可能在这个位置

它要做的一件事：

不断地往上找 parent，最后找到 parent 等于 parent[p] 的情况下，

同时，把 p 它的爷爷结点直接赋给现在的 parent[p]

Union 合并

首先调 find 找出它的 p 的集合所在的领头元素，找出后赋给 rootP

另外找 q 赋给 rootQ，

两个相等不用管；

不相等 → 就把其中一个可以写 rootP 或在括号里面，也可以把两个换一下

二、要不断地往上找它的 parent，再找它的 parent

直到 parent[i] = i 的时候说明：

找到了它所在集合的领头元素

可以把 rootP的parent 弄成是 rootQ 即可
也可以把 rootQ的parent 弄成是 rootP

⇓

最后的话，就把这两个集合给并在一起，再 count 一下就行了

↳ 指的是里面的独立的集合就减少了一个