数据结构

第 6 周 – 集合数据结构：表示方法与运算

Koczka Ferenc 博士 · 埃斯特哈齐·卡罗伊大学 · 2026

复习

- **数组**：直接访问 O(1)，插入/删除 O(n)
- 在有序数组中**二分查找**：O(log n)
- **排序**：冒泡、选择、插入 – $O(n^2)$
- **归并**（第 3 周练习 5）：把两个有序数组合并成一个有序数组，O(n + m)

今天的问题

- 数学中的集合在计算机中怎样表示?
- 在每种表示中集合运算有多快?

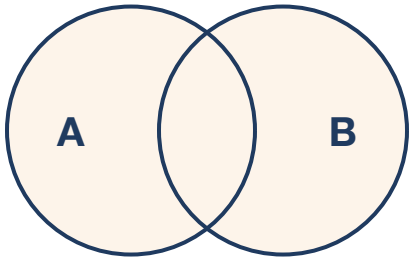
数学背景

集合：由互不相同的元素组成的**无序**整体。

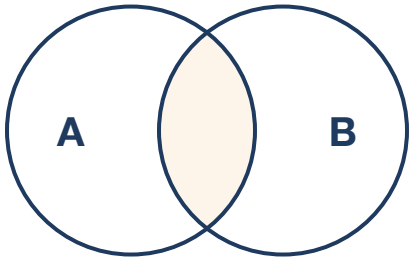
- 每个元素**至多出现一次**：{1, 2, 2} = {1, 2}
- 顺序**无关紧要**：{1, 2} = {2, 1}
- 记号：$x \in A$（属于）· $|A|$（元素个数）· $\varnothing$（空集）· **U**（全集）

运算	记号	含义
属于	$x \in A$	x 是否在 A 中
子集	$A \subseteq B$	A 的每个元素也是 B 的元素
并	$A \cup B$	属于 A **或** B 的元素
交	$A \cap B$	A **和** B 的公共元素
差	$A \setminus B$	属于 A 但**不**属于 B 的元素

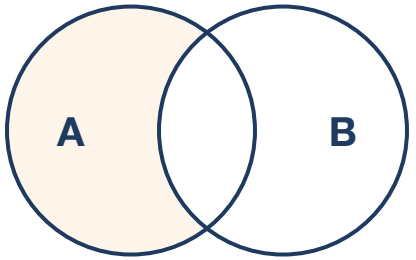
集合运算图示



A ∪ B（并集）



A ∩ B（交集）



A \ B（差集）

例 – U = 一周的七天：

A = {周一，周三，周四，周六}（训练日） · B = {周二，周三，周四}（上课日）

A ∪ B = {周一，周二，周三，周四，周六} · A ∩ B = {周三，周四} · A \ B = {周一，周六}

作为数据结构的集合

性质	集合
元素	同构
元素之间的关系	**无**（没有顺序）
大小	动态（可以增大和缩小）
重复元素	无

可能的表示方法（都基于数组）：

1. **位向量**（特征向量、布尔向量）
2. **无序**数组（序列）
3. **有序**数组（序列）
4. 哈希表（第 11 周）、二叉搜索树（第 15 周）

用位向量表示

位向量

如果**全集很小且已知**（$U = \{u_1, \ldots, u_n\}$），每个集合都是一个长度为 **n** 的布尔向量：**A[i] = true ⇔ $u_i \in A$**

	一	二	三	四	五	六	日	
U	1	1	1	1	1	1	1	
A	1	0	1	1	0	1	0	A = {周一，周三，周四，周六}
B	0	1	1	1	0	0	0	B = {周二，周三，周四}
A ∪ B	1	1	1	1	0	1	0	逐位：或
A ∩ B	0	0	1	1	0	0	0	逐位：与
A \ B	1	0	0	0	0	1	0	逐位：与非

位向量 – 运算

伪代码

```
Function Member(A, i)          // O(1)
  Return A[i]
End function

Procedure Union(A, B, C, n)     // O(n)
  For i ← 1 to n
    C[i] ← A[i] or B[i]
  End for
End procedure

// 交:C[i] ← A[i] and B[i]
// 差:C[i] ← A[i] and not B[i]

Function Subset(A, B, n)        // A ⊆ B ?
  i ← 1
  While i ≤ n and (not A[i] or B[i])
    i ← i + 1
  End while
  Return i > n
End function
```

Java

```java
boolean[] in = new boolean[7];

BitVectorSet union(BitVectorSet o) {
    BitVectorSet r = new BitVectorSet();
    for (int i = 0; i < in.length; i++)
        r.in[i] = in[i] || o.in[i];
    return r;
}
```

运算	步数
属于、插入、删除	**O(1)**
并、交、差	O(|U|)
子集	O(|U|)
元素个数	O(|U|)

位掩码：一位 = 一个元素

如果 $|U| \le 64$，整个集合可以放进**一个整数**：第 **i** 位表示 u_i 是否属于 A。

运算	Java	Go	例（A = 0101101,B = 0001110）
并	`a | b`	`a | b`	0101111
交	`a & b`	`a & b`	0001100
差	`a & ~b`	`a &^ b`	0100001
属于（i）	`(a >> i & 1) == 1`	`a>>i&1 == 1`	
子集	`(a & ~b) == 0`	`a&^b == 0`	
元素个数	`Integer.bitCount(a)`	`bits.OnesCount(a)`	4

一次集合运算 = **一条处理器指令**。应用：权限（读/写/执行）、星期、各种设置。

位向量 – 优点与局限

优点：

- 最快：属于判断 **O(1)**，用位掩码时集合运算只需一步
- 实现简单

局限：

- 全集必须**很小且事先已知**
- 所需空间与 |U| 成正比，而不是与元素个数成正比
  例如 0 … 10^9 之间的整数集合：10^9 位 ≈ **125 MB** – 即使其中只有 3 个元素！
- 元素必须用**序号**来标识（周一 → 0，周二 → 1,…）

用无序数组表示

无序数组

元素存放在数组中，**顺序任意**，没有重复。

A = `[8, 2, 11, 5]` · B = `[9, 5, 14, 3, 11]`

- **属于：**顺序查找 – O(n)
- **插入：**先检查是否已经存在（O(n)），再追加到末尾 – O(1)
- **删除：**找到它（O(n)），然后**把最后一个元素移到它的位置** – 不需要移动其他元素，因为顺序无关紧要！

> 所需空间与元素个数成正比 – 不需要全集，任何元素都可以存放。

无序数组 – 运算

子集：A ⊆ B

```
Function Subset(A, n, B, m)
  i ← 1
  While i ≤ n and Member(B, m, A[i])
    i ← i + 1
  End while
  Return i > n
End function
```

交：A ∩ B

```
Procedure Intersection(A, n, B, m, C, k)
  k ← 0
  For i ← 1 to n
    If Member(B, m, A[i]) then
      k ← k + 1 ; C[k] ← A[i]
    End if
  End for
End procedure
```

并： A 的全部元素 + B 中不属于 A 的元素

差： A 中不属于 B 的元素

运算	步数
属于	O(n)
插入	O(n)
删除	O(n)
子集	**O(n · m)**
并、交、差	**O(n · m)**

对每个元素都要在另一个集合中顺序查找 → 平方阶！

用有序数组表示

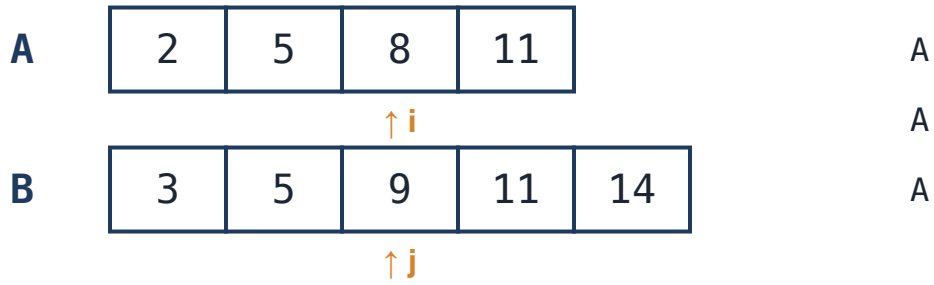
有序数组

元素按**递增顺序**存放，没有重复。

A = `[2, 5, 8, 11]` · B = `[3, 5, 9, 11, 14]`

- **属于：二分查找** – O(log n)
- **插入 / 删除：**用二分查找确定位置，但需要**移动**元素 – O(n)
- **集合运算：**对两个有序数组进行**归并** – 只需一次同步遍历

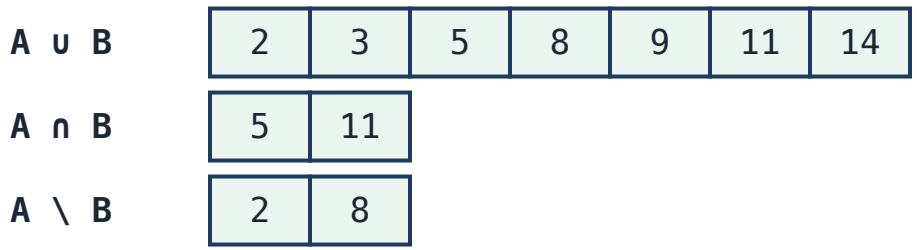
归并：集合运算的基础



A[i] < B[j] → 取 A[i],i ← i + 1

A[i] = B[j] → 只取一次，两者都前进

A[i] > B[j] → 取 B[j],j ← j + 1



三种运算：都只需一次同步遍历

O(n + m)

用归并求并集

伪代码

```
Procedure Union(A, n, B, m, C, k)
  i ← 1 ; j ← 1 ; k ← 0
  While i ≤ n and j ≤ m
    k ← k + 1
    If A[i] < B[j] then
      C[k] ← A[i] ; i ← i + 1
    else if A[i] > B[j] then
      C[k] ← B[j] ; j ← j + 1
    else                          // 相等
      C[k] ← A[i] ; i ← i + 1 ; j ← j + 1
    End if
  End while
  While i ≤ n                     // A 的剩余部分
    k ← k + 1 ; C[k] ← A[i] ; i ← i + 1
  End while
  While j ≤ m                     // B 的剩余部分
    k ← k + 1 ; C[k] ← B[j] ; j ← j + 1
  End while
End procedure
```

Java

```java
int i = 0, j = 0, k = 0;
while (i < a.length && j < b.length) {
    if (a[i] < b[j])      c[k++] = a[i++];
    else if (a[i] > b[j]) c[k++] = b[j++];
    else { c[k++] = a[i++]; j++; }
}
while (i < a.length) c[k++] = a[i++];
while (j < b.length) c[k++] = b[j++];
```

交：只在相等的分支中写入 C，剩余部分不复制。

差：只在 `A[i] < B[j]` 的分支中写入，并复制 A 的剩余部分。

用归并判断子集

```
Function Subset(A, n, B, m)              // A ⊆ B ?
  i ← 1 ; j ← 1
  While i ≤ n and j ≤ m and A[i] ≥ B[j]
    If A[i] = B[j] then i ← i + 1
    j ← j + 1
  End while
  Return i > n
End function
```

- 如果 `A[i] < B[j]`，那么 A[i] **一定不**在 B 中（B 后面的元素更大）→ 不是子集
- 步数：**O(n + m)**

各种表示方法的比较

运算	位向量	无序数组	有序数组
属于	**O(1)**	O(n)	O(log n)
插入、删除	**O(1)**	O(n)	O(n)
子集	O(|U|)	O(n · m)	**O(n + m)**
并、交、差	O(|U|)	O(n · m)	**O(n + m)**
空间	|U| 位	n 个元素	n 个元素
前提	U 小且已知	–	元素可排序

> 没有“最好”的表示方法：选择取决于**全集的大小**和**最常用的运算**。

内置集合

Java

```java
Set<String> a = new HashSet<>();  // 平均 O(1)
a.add("Wuhan");
a.contains("Wuhan");    // true
a.remove("Wuhan");

Set<String> u = new HashSet<>(a);
u.addAll(b);            // 并
u.retainAll(b);         // 交
u.removeAll(b);         // 差

TreeSet<Integer> t = new TreeSet<>(); // O(log n)
t.first(); t.last();    // 有序!
```

Go – 没有集合类型，用 map

```go
type Set map[string]struct{}

s := Set{}
s["Wuhan"] = struct{}{}      // 插入
_, ok := s["Wuhan"]          // 属于
delete(s, "Wuhan")           // 删除

// 常用的简写:
seen := map[int]bool{}
if !seen[x] {
    seen[x] = true
}
```

`HashSet` 和 Go 的 `map`：**哈希表**（第 11 周）· `TreeSet`：**平衡二叉搜索树**（第 15 周）

小结

- **集合**：由互不相同的元素组成的无序整体；运算：$\in$、$\subseteq$、$\cup$、$\cap$、$\setminus$
- **位向量 / 位掩码**：全集小且已知时最快 – O(1)、O(|U|)
- **无序数组**：简单，但集合运算是 O(n · m)
- **有序数组**：二分查找 O(log n)，集合运算用**归并**，O(n + m)
- 内置：Java `HashSet` / `TreeSet`,Go `map[T]struct{}`

下周：栈 – 后进先出原则、数组实现、后缀表达式求值。

复习题

1. 全集 U = {1, 2, …, 8}。写出位向量：A = {2, 3, 5, 7},B = {1, 2, 3, 4}!计算 A ∪ B、A ∩ B、A \ B!
2. 判断对错：在无序数组表示中，求交集需要 O(n + m) 步。
3. 在纸上进行归并：A = [1, 4, 6, 9],B = [2, 4, 9, 10]!写出 A ∪ B、A ∩ B 和 A \ B!
4. 为什么用位向量表示学号（8 位数字）的集合不现实?
5. 用位掩码：`a = 0b1010`，`b = 0b0110`。`a & b`、`a | b`、`a & ~b` 分别是多少?

实践课

✎ 示例程序

程序	Java	Go	演示内容
位向量	`BitVectorSet.java`	`w06/bitvectorset`	星期集合、位掩码运算
无序	`UnorderedSet.java`	`w06/unorderedset`	O(n · m) 的运算、用最后一个元素填补的删除
有序	`SortedSet.java`	`w06/sortedset`	二分查找、归并
内置	`BuiltinSets.java`	`w06/builtin`	`HashSet` 、 `TreeSet` 、Go `map`

✎ 练习

1. **去重：**通过排序把数组变成集合！（`[4,1,4,2,1]` → `[1,2,4]`）
2. **对称差：**A △ B = (A \ B) ∪ (B \ A) – 在有序数组上只用一次归并！
3. **是否不相交？**用 O(n + m) 步判断两个有序集合是否没有公共元素！
4. **幂集：**用位掩码输出 **n** 元集合的所有子集！共有多少个？
5. **不同的字母：**一段文本中出现了多少个不同的英文字母？（位向量）
6. **全字母句：**一个句子是否包含了全部英文字母？
7. **共同好友：**用内置集合（map）求两个好友名单的交集。

参考答案：`Exercises06.java`、`w06/exercises`

✎ 课后作业

1. 实现一个始终保持有序的 `SortedSet` 类，带有 `add(x)` 和 `remove(x)` 操作！（提示：用二分查找确定位置。）
2. **多重集**（包）：一个元素可以出现多次。你会用什么代替位向量来表示它？实现并和交（计数的最大值和最小值）！
3. 用**一次**同步遍历求三个有序集合的交集！
4. 在 1 000 000 个介于 1 和 100 000 之间的数中，有多少个**不同**的数？用位向量求解，并与基于排序的解法比较！