第 9 课
递归回溯

薛浩

xuehao0618@outlook.com

今日话题

- 回顾
- 递归回溯
- 递归组合问题

回顧

简单递归问题

```
if( 最简单的情况 ) {
    直接处理问题;
    返回处理结果。
} else {
    把问题分成一个或多个相同形式的子问题;
    一个个解决这些子问题;（采用相同的逻辑）
    整合这些子问题的结果;
    返回最终结果。
}
```

递归自相似问题

根据一些规则，不断由简单变向复杂的过程。

使用阶（Order）来定义其复杂性。

递归自相似问题

根据一些规则，不断由简单变向复杂的过程。

使用阶（Order）来定义其复杂性。

递归自相似问题

根据一些规则，不断由简单变向复杂的过程。

使用阶（Order）来定义其复杂性。

递归子集问题

包含/排除模式

```
if( 最简单的情况 ) {
    直接处理问题;
    返回处理结果。
} else {
    选择其中一个元素，降低问题规模;
    包含该元素的处理结果;              // include
    不包含该元素的处理结果;            // exclude
    返回最终结果。
}
```

递归排列问题

选择/探索/不选择模式

```
1  if( 最简单的情况 ) {
2      直接处理问题;
3      返回处理结果。
4  } else {
5      for (遍历每一个可能的选项) {
6          选择其中一个选项;            // Choose
7          递归探索包含该选项的分支; // Explore
8          切换下一个选项               // Unchoose
9      }
10 }
```

今日话题

- ~~回顾~~
- 递归回溯
- 递归组合问题

递归回溯

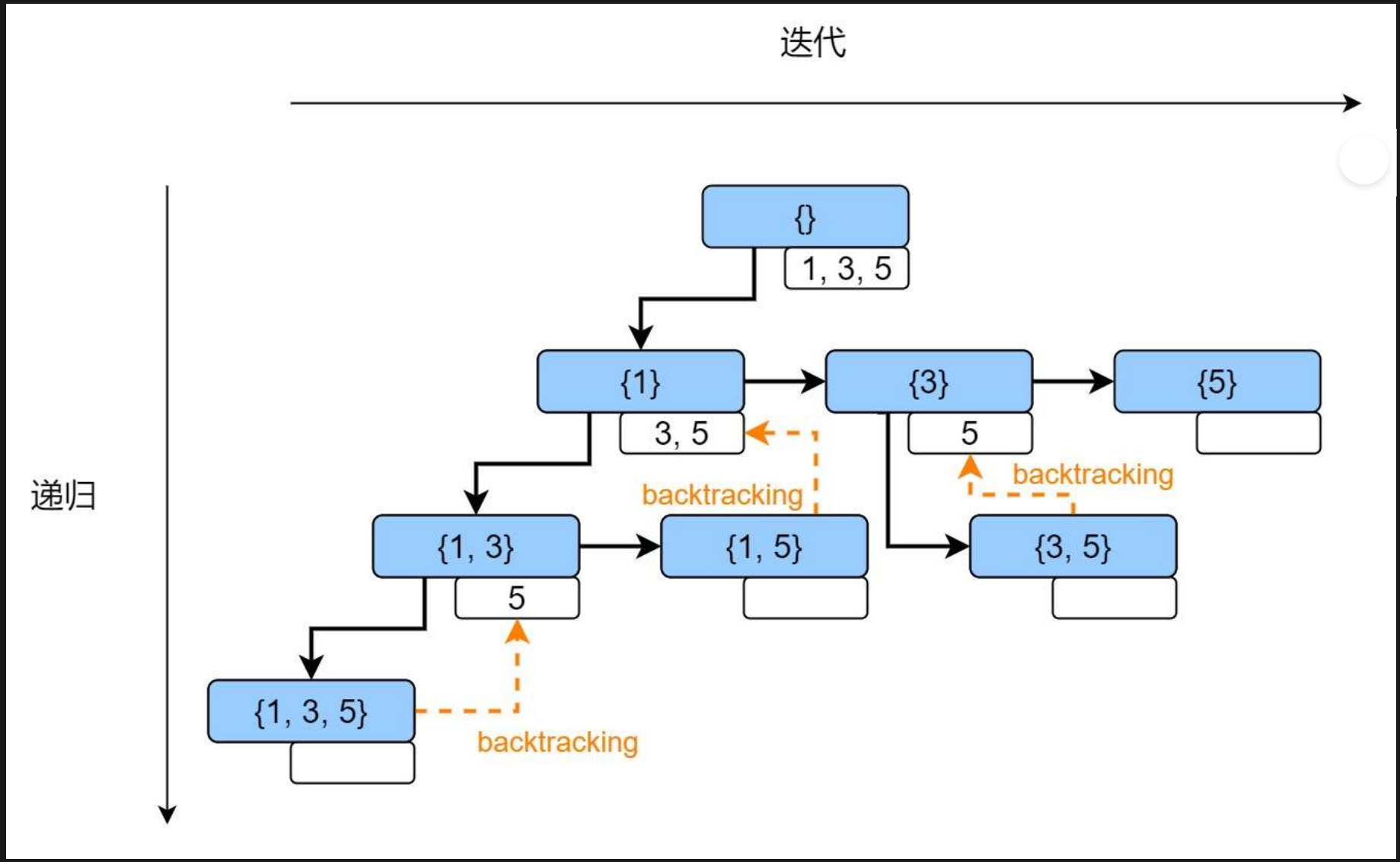
递归回溯

用于寻找一些计算问题的解决方案，特别是约束满足问题。

DFS

回溯的一种实现方式，常以决策树的形式存在。

另一种回溯逻辑


迭代
递归
{}
1, 3, 5
{1}
3, 5
{3}
5
{5}
{1, 3}
5
{1, 5}
{3, 5}
{1, 3, 5}
backtracking
backtracking
backtracking

今日话题

- ~~回顾~~
- ~~递归回溯~~
- 递归组合问题

递归组合问题

递归子集问题的一个特例，通过约束删除部分递归分支。

DFS 解法

```
 1 bool canMakeSumRec(Vector<int> &v, int index, int target) {
 2     if (index == v.size()) {
 3         if (target == 0)
 4             return true;
 5     } else {
 6         if (canMakeSumRec(v, index + 1, target - v[index]))
 7             return true;
 8         if (canMakeSumRec(v, index + 1, target))
 9             return true;
10     }
11     return false;
12 }
```

小试牛刀

按照新的回溯策略，改写 8.2 节的例题。

今日话题

- ~~回顾~~
- ~~递归回溯~~
- ~~递归组合问题~~

下一次课

- 算法分析

THE END