浅谈数位类统计问题

山东省青岛第二中学
刘聪

引入

- 在信息学竞赛中，有一类与数位有关的区间统计问题。
- 例如：
- 求 $[1,n]$ 中所有数的各位数字之和。
- 求 $[1,n]$ 中各位数字之和为定值的数的个数。
- 将 $[m,n]$ 中的数字分组，每组包含连续的一段数，且数字之和不能超过定值，求分组数。
- ……

$n \leq 10^{18}$!

- 如何求解？

【例题 1】 Amount of degrees

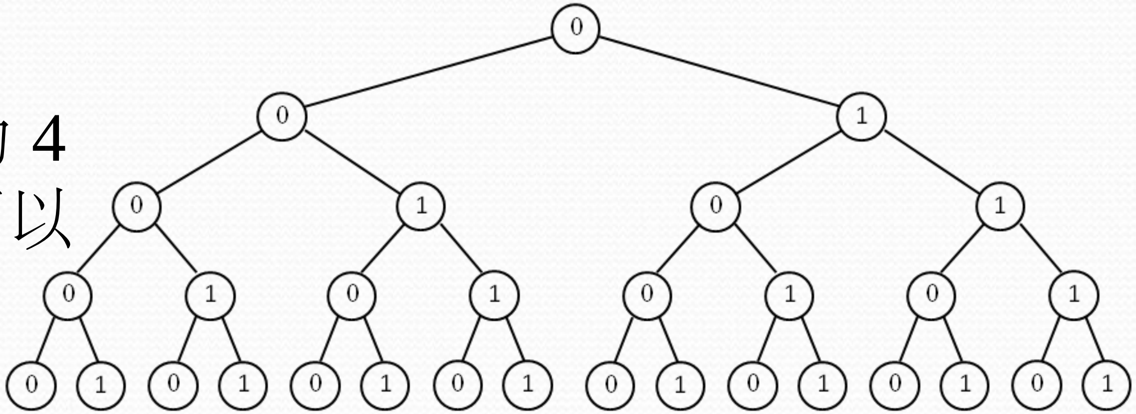
- 求给定区间 [X,Y] 中满足下列条件的整数个数：这个数恰好等于 K 个互不相等的 B 的整数次幂之和。
- 例如，设 X=15， Y=20， K=2， B=2，则有且仅有下列三个数满足题意：
  - $17 = 2^4+2^0$，
  - $18 = 2^4+2^1$，
  - $20 = 2^4+2^2$。
- 数据规模： $1 \le X \le Y \le 2^{31}-1$， $1 \le K \le 20$， $2 \le B \le 10$。

【例题 1 】 Amount of degrees

- 由于所求的幂互不相等，符合条件的数写成 B 进制一定只由 0 和 1 组成。
- 因此我们只考虑 $B=2$ 的情况。
- 此处区间满足区间减法，即 $count(X,Y)=count(0,Y)-count(0,X-1)$
- 因此我们唯一需要求的就是，给定 n ，求出所有不超过 n 的正整数中，其二进制表示恰好含有 K 个 "1" 的有多少。

【例题 1】 Amount of degrees

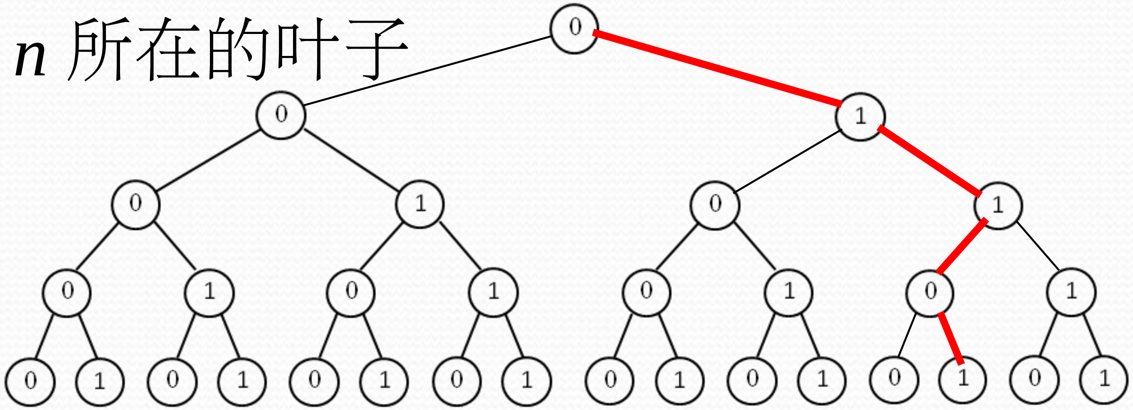
- 这里，我使用一棵完全二叉树来代表一个区间内的数。
- 例如右图中高度为 4 的完全二叉树就可以代表所有长度为 4 的二进制数。



- 其范围为 $[0,2^4-1]$
- 每个叶子就代表了一个数。

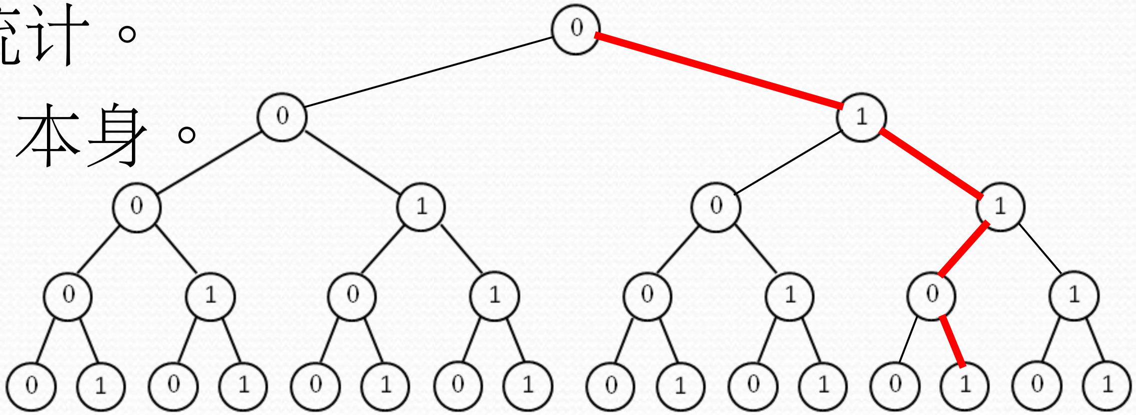
【例题 1】 Amount of degrees

- 假设 K=3 ， n=13， 其二进制为 1101 。
- 我们从根 " 走 " 到 n 所在的叶子
- 每次“向右走”时，左侧的子树都是完整的完全二叉树，且所有小于 n 的整数都包含在其中。



【例题 1 】Amount of degrees

- 因此，我们实际上只需对这三棵子树进行 统计。
- 当然，不能忘记 n 本身。



这棵树中有多少数含 3 个 1？

这棵树中有多少数含 2 个 1？
（因为祖先已有 1 个 1）

这棵树中有多少数含 1 个 1？

【例题 1 】 Amount of degrees

- 很容易递推求出这个问题:
- 设 $f[h,s]$ 表示在高度为 h 的完全二叉树包含的数中（范围是 $[0,2^h-1]$ ），二进制中恰含 s 个 1 的数有多少。
  - $f[h,s] = \underbrace{f[h-1,s]}_{\text{左子树中的个数}} + \underbrace{f[h-1,s-1]}_{\text{右子树中的个数}}$

【例题 1 】 Amount of degrees

- 剩下的问题就是，如何将任意进制问题转化为二进制问题。
- 只需贪心将 n 的 B 进制表示转化为只含 01 ： 找到 n 的左起第一位大于 1 的数位，将它以及它右边所有数位赋为 1 。
- 递推求 f 的时间复杂度为 $O(\log^2(n))$ ， 每次询问的时间复杂度为 $O(\log(n))$ ， 因此总时间复杂度为 $O(\log^2(n))$ 。
- 问题得到完美解决！

【例题 1 】 Amount of degrees

- 实际上，我们也可以只用“位”的思想来考虑这个问题，而不使用树的思想。
- 我们每次找到 *n* 的 "1" 数位，将它赋为 0 ，则其右面的数位可任取 0 、 1 。
- 可根据组合数算出满足题意的数的个数。
- 穷举所有“ 1” 数位，计算组合数，即可完成询问。
- 事实上， f 的递推公式正是组合数公式！

$$f[h,s]=f[h-1,s]+f[h-1,s-1]$$

$$C[n,k]=C[n-1,k]+C[n-1,k-1]$$

- 与树的思想异曲同工！

【例题 1 】 Amount of degrees

- 使用高度为 h 的完全 B 叉树表示所有长度为 h 的 B 进制数，思考起来更加直观，在处理较复杂问题时优点明显。
- 当然，最终程序我们并不真正建树，它的作用只是帮助我们思考。
- 无论从树结构考虑还是从数位的角度考虑，基本思想都是:

逐位确定

【例题 4 】Tickets

- 有一位售票员给乘客售票。对于每位乘客，他会卖出多张连续的票，直到已卖出的票的编号的数位之和不小于给定的正数 K 。然后他会按照相同的规则给下一位乘客售票。
- 初始时，售票员持有的票的编号是从 L 到 R 的连续整数。请你求出，售票员可以售票给多少位乘客。
- 数据规模：$1 \le L \le R \le 10^{18}$，$1 \le K \le 1000$ 。

【例题 4 】Tickets

- 有了例 1 的基础，本题基本思路也是从特殊到一般：
- 能否将原问题的区间拆分为一些长度为 10^k 的子区间，再合并得到原区间的答案？
- 亦即，将原树拆分为若干完整的完全十叉数，根据子树的信息合并得到原问题的答案。
- 一个比较棘手的问题是，这棵树的前几个元素可能会并入之前的小组，而不是作为一个新的小组。
- 换言之，只使用树的高度无法表示一组状态。

添加状态！

【例题 4 】Tickets

- 设 *f*[*h*,*sum*,*rem*] 表示对于高度为 *h* 的完全十叉树，其每个数的数位之和需要额外添加 *sum* ，在这棵树之前的最后一个小组剩余空间为 *rem* ，所能得到的小组情况。
- *f* 返回值有两个： *f*[*h*,*sum*,*rem*].*a* 记录这棵树被划分成的小组数量， *f*[*h*,*sum*,*rem*].*b* 记录这些小组中的最后一组的剩余空间，以便与下一组合并
- 同例 1 一样，这里 *f* 的值也由其
- for i:=0 to 9 do f[h,sum,rem]:= merge(f[h,sum,rem] , f[h-1,sum+i,f[h,sum,rem].b]);
- 也就是之前得到的 "*b*" 值作为下一棵子树的 "*rem*" 值。

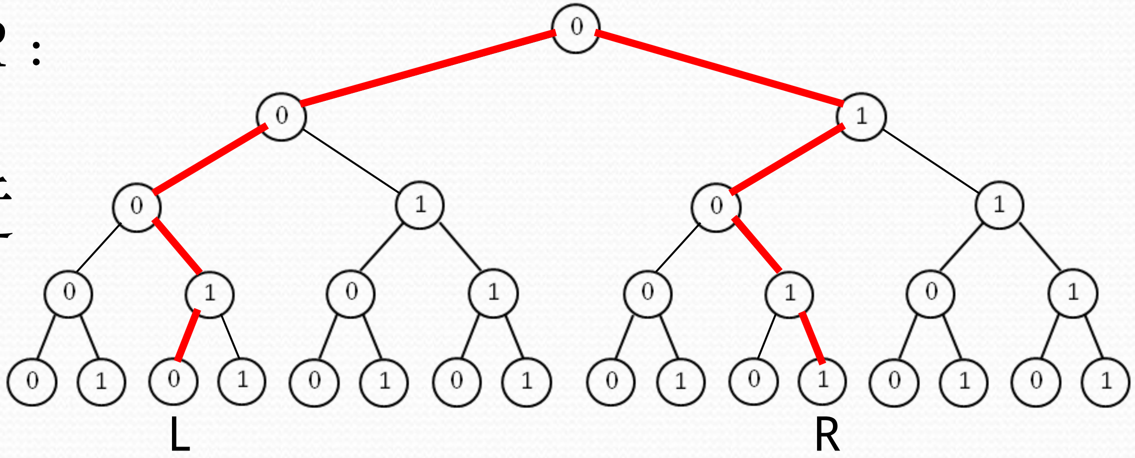
```
function merge(x,y);
  x.a:=x.a+y.a;
  x.b:=y.b;
  return x;
```

【例题 4 】 Tickets

- 这样，我们就解决了单个子树的情况。
- 接下来的问题就是如何将原树分解为若干子树。
- 本题不满足区间减法，因此对于 $[L,R]$ 这个区间，需要直接找它们中间的子树。
- 为了方便起见，这里以二叉树为例进行介绍，其思想与十叉树一致。

【例题 4 】 Tickets

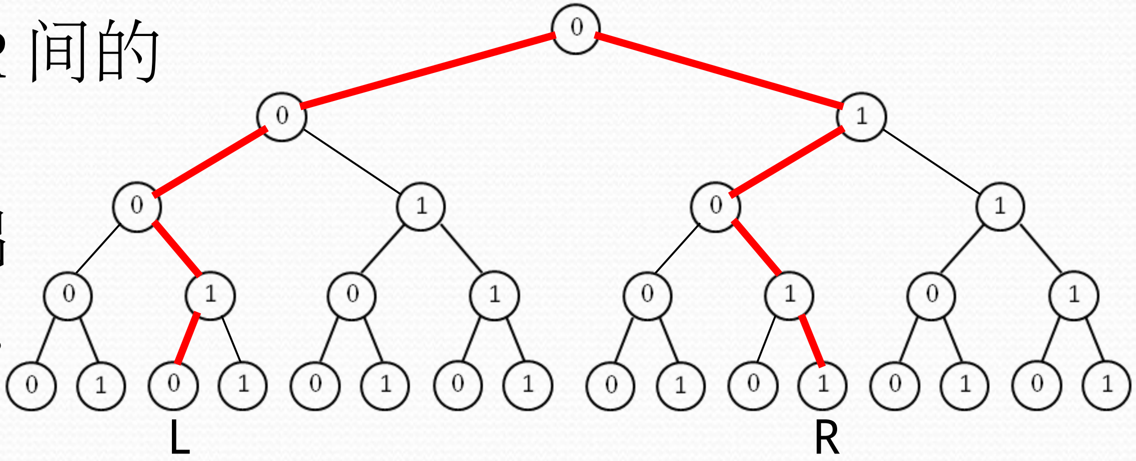
- 假设 L=2,R=11 。
- 让我们从 L 走到 R :
- 首先从 L 向上走，直到 L 和 R 的最近公共祖先
- 统计途中节点右侧的兄弟子树



- 再向下走到 R
- 统计途中节点左侧的兄弟子树
- 需要注意的是，在 LCA 的下一层，两条路径是兄弟关系，需要统计它们之间的子树。

【例题 4 】Tickets

- " 抠出 " 这些子树：
- 它们包含了 L 和 R 间的连续整数
- 我们只要分别求出每个子树，再加以合并即可！



- 这样，我们就解决了这道题目。

【例题 4 】Tickets

- 回顾解题过程：在处理完整子树时，为了表示状态，我们添加了新的一维。
- 由于问题不满足区间减法，我们改“走下去”为“走上去”、“走过去”、“走下来”。
- 其余步骤与例 1 相似：将待求区间分解为若干子区间（子树），每个完整子树利用递推求出，最后合并出完整答案。
- 递推求 f 的复杂度为 $O(k\log^2(R))$，需要询问的子树最多有 $O(\log(R))$ 个，因此总复杂度是 $O(k\log^2(R))$。
- 建议采用记忆化搜索实现。

总结

- 以上，我们通过两道例题了解了数位类的区间统计问题及其基本处理方法。
- 只要本着“逐位确定”的思想，往往就能找到突破口，设计出 $\log(n)$ 级别复杂度的算法。
- 使用树结构来思考问题，更加直观，化抽象为具体，有利于我们的思考。

数形结合！

谢谢大家
欢迎提问！