月考题解

陈宇琪

2020 年 5 月 27 日

9.1 层次遍历

队列模拟

- 首先，将 1 号节点加入队列。
- 每次从队列中取出元素，如果为-1，则弹出后继续，否则输出队列顶部元素。
- 如果顶部元素不是-1，将其取出，并将其左右儿子节点加入队列。

9.2 医疗排队

解法 1：STL 模拟

- 维护 set<pair<int,int> > s 和 map<int,int> mp，其中 set 中每个元素 pair<int,int> 为“挂号”的编号和病人的编号，map 记录病人的编号到“挂号”的编号的映射。
- 对于 1 操作：取出“挂号”编号最小的病人，即 s.begin()->second。给这个病人分配新的挂号的编号，即当前最大的挂号编号 +1，即 s.end()->first+1，更新这个病人的编号，即 mp[s.begin()->second]=s.end()->first+1，从 set 中删除首元素，并插入一个新的节点。
- 对于 2 操作：根据 mp 找到这个病人现在的编号，即 mp[x]，进而找到在 set 中的节点 make_pair(mp[x],x)，将这个节点删除，并将 x 的“挂号”编号更新为当前最小的挂号编号-1，即 s.begin()->first-1，更新 set 和 mp。
- 对于 3 操作：直接输出 s.begin()->second。

9.2 医疗排队

解法 2：链表模拟

- 维护一个双链表，记录当前的排队的顺序。同时，保存每个病人的节点的地址。同时，动态维护队列的首尾指针。
- 对于 1 操作：将队列首元素移到队列末尾（注意在这个过程中不是删除这个节点，而是将这个节点移动到另一个地方，这一点很关键！），同时更新首尾指针。
- 对于 2 操作：找到这个病人对应的节点的地址，由于是双链表，所以可以将这个节点从链表中“脱离”出来，然后放在链表首部，同时更新首指针。
- 对于 3 操作：输出链表头部元素。
- 注意在 2 操作时，将链表中一个元素移动出来有 3 种情况：移动头元素，中间元素和尾元素。对于第三种情况，移动出来后还要更新尾指针。

9.3 数独问题

搜索

- 直接暴力搜索即可，在搜索过程中每一次搜索之前先判断当前这个格子填入这个数是否合法（即这个格子所在的行、列、9 宫格是否已经有这个数）。
- 关于字典序：按照行的顺序填写数独，每次搜索优先填小的数，再考虑大的数，找到一个合法解之后直接输出答案并结束程序。
- 关于复杂度：在大多数情况下，数独的搜索效率还是很高的，因为当大多数格式确定时候，方案几乎唯一了。
- 对于数独问题的算法优化可以参考 dancing link 算法。

12.1 数的变换

33 分解法：搜索

- 使用 dfs 算法遍历所有可能的情况，由于数的范围只有 10，所以最终的结果不会超过 long long 的范围。
- 注意在选择最优解的时候不能取模，只有在输出的时候才可以取模。
- 复杂度计算：假设每个数被加了 x_i 次，则所有可能的方案数为满足 $x_1 + ...x_n = m$，且 x_i 为非负数的所有情况数。利用离散数学的知识，可以知道这个方案数为 C_{m+n-1}^{n-1}，由于对每个方案要计算所有数的乘积，所以复杂度为 $O(n \times C_{m+n-1}^{n-1})$。

12.1 数的变换

67 分解法：贪心 + 暴力

- 注意到将 1 变成 2，结果变大了 2 倍；而将 2 变成 3，结果才变大了 $\frac{3}{2}$ 倍。
- 每次选择数列中最小的数，将其变成原来的数 $+1$ 即可。
- 注意不是将原来的数列中最小的数变成这个数加 m。
- 复杂度为 $O(n \times m)$。

12.1 数的变换

100 分解法：贪心 +STL

- “每次选择数列中最小的数，将其变成原来的数 $+1$”，这个操作可以理解成“每次选择最小的数，修改它的值，再放回序列中”，这个可以利用 priority_queue 实现这个操作。
- 注意优先级队列默认是大根堆，需要修改成小根堆。

12.2 单峰序列

30 分解法：二进制枚举

- 暴力枚举每个数取/不取，用 $O(n)$ 复杂度判断一个序列是不是单峰序列。
- 复杂度为 $O(n \times 2^n)$。

12.2 单峰序列

60 分解法：动态规划/LIS

- 令 $f[i]$ 表示前缀 1 到 i 的最长上升子序列，$g[i]$ 表示后缀 i 到 n 的最长下降子序列，则答案为 $max\{f[i]+g[i]-1\}$。
- 计算后缀最长下降子序列等价于求反转序列的最长上升子序列。
- 复杂度为 $O(n^2)$。

12.2 单峰序列

100 分解法：LIS

- 计算序列的最长上升子序列可以做到复杂度为 $O(n \times \log(n))$ 的算法。
- 可以使用基于二分的 LIS 算法，或者使用树状数组优化动态规划算法。

12.3 高级并查集

30 分解法：模拟

- 使用 set 或者数组模拟即可。
- 使用 set 模拟复杂度为 $O(n \times m \times \log(n))$。
- 使用数组模拟复杂度为 $O(n \times m)$。

12.3 高级并查集

50&70 分解法：并查集模板

- 50 分做法：在并查集模板基础上维护每个并查集大小。
- 70 分做法：每次遇到 4 操作，用 $O(n)$ 复杂度重新建立并查集。
- 维护每个并查集大小：对每个并查集记录一个 sz，初始每个并查集的 sz 都是 1，假设合并节点 x 和 y，则执行两步操作，首先执行 $fa[findset(x)] = findset(y)$ 来合并并查集，然后执行 $sz[findset(y)] += sz[findset(x)]$ 即可。

12.3 高级并查集

100 分解法：并查集

- 动态维护每个节点的当前编号（初始每个节点的编号就是 i）。
- 对于 4 操作，为这个节点新开一个节点 tmp，更新这个节点的编号。
- 假设用 idx 数组记录每个节点的编号，则 4 操作总共三步操作，首先执行 $sz[findset(x)]--$ 来将 x 从原先集合中移除，然后执行 $idx[x] = ++tmp$ 来给 x 分配新的节点，最后初始化 $idx[x]$ 节点，执行 $fa[idx[x]] = idx[x], sz[idx[x]] = 1$。
- 注意如果 x 就是集合的根节点，这样操作也是正确的。