基础算法

Shinetism

YALI

July 20,2019

什么是基础算法？

- 讲到基础算法，大家都能脱口而出：
- 枚举？贪心？分治？二分？倍增？分块？……
- 这些算法之所以称为基础，是因为它们经常作为一种解题思想，许多高级一些算法也是基于这样一些思想的，比如：
- 线段树、平衡树基于分治，网络流也有贪心思想，BSGS是一个简单的分块……
- 所以说，基础对于竞赛来说十分重要，大家要对基础算法深刻理解，才能灵活运用. 这一节中的题目也会~~比较水~~有较强的综合性.

目录

枚举

最基础的思想！

枚举

- 可以说几乎每一个循环都是一次枚举
- 它是一种最朴素的算法
- 接下来我将举出几个例子，为了让大家清晰地理解题意，这里提出几个数学相关的问题

T1 经典题

- Description:
  - 给出一个数 n , 判断 n 是否为质数.
- Restraint:
  - $1 \leq n \leq 10^{16}$

T1 经典题

- 朴素的枚举：
  - 依次枚举 $2\sim(n-1)$ 的所有数，如果有一个数是 n 的约数，那么这个数是合数，否则是质数.
  - Time Complexity: $O(n)$
- 聪明的枚举：
  - 依次枚举 $2\sim\lfloor\sqrt{n}\rfloor$ 的所有数，如果有一个数是 n 的约数，那么这个数是合数，否则是质数.
  - Time Complexity: $O(\sqrt{n})$
- 有一个更加精妙的算法——Miller Rabin 算法，但是它不是今天的主角，明天有时间再讲.

T2 又一经典题

- Description:
  - 给出一个数 n , 求 n 的所有约数的质因数个数的和.
- Restraint:
  - $1 \leq n \leq 10^{12}$
- Hint:
  - 一个数 n 的约数个数是 $O(n^{\frac{2}{3}})$

T2 又一经典题

- 聪明的枚举：
  - 首先用 $O(\sqrt{n})$ 的时间将 n 进行质因数分解. 接下来枚举每一个质因数的指数, 那么每次得到的数一定是 n 的约数, 加起来得到答案.
  - Time Complexity: $O(n^{\frac{2}{3}})$
- 这道题是个常见套路, 在各类数论题中需要用到.

贪心

局部最优即是全局最优！

贪心

- 贪心思想往往表现为一些与最优性有关的结论
- 与其他算法结合考察时也是难点
- 解决贪心有关的题目最为关键的是敢于猜测，猜测后的证明往往不那么重要，因为你可以造许多组数据验证正确性
- 贪心算法几乎没有规律，要有一双善于发现结论的眼睛
- 这里举一个例子

T3 「BZOJ2697」特技飞行

- Description:
  - 神犇航空开展了一项载客特技飞行业务. 每次飞行长 n 个单位时间, 每个单位时间可以进行一项特技动作, 可选的动作有 k 种, 每种动作有一个刺激程度 c_i . 定义某次动作的价值为 (距上次该动作的时间) $\times c_i$, 若为第一次进行该动作, 价值为0. 安排一种方案, 使得总价值最大.
- Restraint:
  - $1 \leq n \leq 1000, 1 \leq k \leq 300, 0 \leq c_i \leq 1000$

T3 「BZOJ2697」特技飞行

- 首先我们发现一个动作没有必要做三次或以上, 也没必要只做一次.
- 所以每个动作都是两次.
- 然后我们发现动作 i 的第二次出现不能够在动作 j 的第一次出现之前.
- 所以按照 c_i 从大到小排序, 从两边往中间放.

思维训练!

- Description:
  - 有 n 堆石头, 每堆有 a_i 个, 有 k 种颜色要求给所有石头涂色. 要求任意两堆石头中相同颜色的石头的数量之差小于等于 1.
- Restraint:
  - $1 \le n, k, ai \le 100$

思维训练!

- 设$\mathrm{m} = \min_{i=1}^{n} a_i$，每一堆石子前 $\min(a_i, m+1)$ 个涂成第一种颜色
- 剩下的颜色每堆涂一个(如果某堆涂完了就不涂了)

分治

分裂+合并！

分治

- 分治思想是特别特别特别重要的思想，是许多算法最为巧妙的地方，也是许多题目解题的关键
- 分治的主体过程如下：
  - 将一个规模为 n 的问题转化为常数个规模小于 n 的问题
  - 将这些子问题递归解决
  - 将子问题的答案合并为原问题答案
- 可以分治解决的问题的答案具有可合并性，且子问题之间互不干扰

T4 归并排序

- Description:
  - 用分治的思想给 n 个数排序.
- Restraint:
  - $1 \le n \le 10^6, -10^9 \le a_i \le 10^9$

T4 归并排序

- 将 n 个数字排序，可以把这 n 个数字分为大小相近的两部分，两部分分别排序，然后把两个有序的数列合并.
- 我们当然也可以分为三部分、四部分，但是从稍后的时间复杂度分析中你可以发现它们的时间复杂度没有任何区别.

T5 二维偏序问题

- Description:
  - 给出 n 个有序对 (a_i, b_i) . 对于每一个 (a_i, b_i) , 求出满足 $a_j < a_i$ 且 $b_j < b_i$ 的 j 的个数.
- Restraint:
  - $1 \le n \le 10^6, -10^9 \le a_i, b_i \le 10^9$

T5 二维偏序问题

- 我们想想如何处理一维偏序问题：
  - 将数列排序，对应的编号就是它的答案了
- 我们可以尝试性地将这个思想运用到二维偏序中
  - 将有序对按照 a_i 排序
  - 接下来就变为了逆序对问题，可以运用归并排序解决

分治的时间复杂度？

- 分治的时间复杂度通常不是很直观，我们来思考这样几个抽象问题的时间复杂度.
- 我们用 $T(n)$ 表示规模为 n 的问题的时间复杂度.
- 如果一个问题被分为两个规模为 $\frac{n}{2}$ 的问题，合并复杂度有 $O(n)$，那么：
    - $T(n) = 2T\left(\frac{n}{2}\right) + O(n)$，可以得到 $T(n) = O(n\log n)$
- 如果一个问题被分为两个规模为 $\frac{n}{2}$ 的问题，合并复杂度有 $O(n\log n)$，那么：
    - $T(n) = 2T\left(\frac{n}{2}\right) + O(n\log n)$，可以得到 $T(n) = O(n\log^2 n)$
- 如果是 $T(n) = T\left(\frac{n}{2}\right) + O(n\log n)$，就可以得到 $T(n) = O(n\log n)$
- 这一系列问题的时间复杂度可以由主定理得出，主定理较为复杂，这里不做讲解.

二分

一半一半找答案！

二分

- 讲到二分，大家肯定最先想到二分查找.
  - Tips: 二分查找可以用C++自带的 std::lower_bound 和 std::upper_bound
- 可是二分这种思想远远不止二分查找.
- 我们来看看两道题

T6 「BZOJ1816」[CQOI2010] 扑克牌

- Description:
  - 你有 n 种牌, 牌上写着 $1\sim n$, 第 i 种牌的数目为 c_i . 另外有一种特殊的牌：JOKER，它的数目是 m . 你可以用 $1\sim n$ 各一张来组成一套牌, 也可以用一张JOKER和除了某一种牌以外的其他牌各一张组成一套牌. 你的任务是组成尽量多的套牌. 每张牌最多只能用在一副套牌里(可以有牌不使用).
- Restraint:
  - $2 \le n \le 50, 0 \le m, c_i \le 500{,}000{,}000$

T6 「BZOJ1816」[CQOI2010] 扑克牌

- 我们二分答案.
- 所谓二分答案, 就是将答案的区间通过二分一步一步缩小.
- 二分答案步骤如下:
  - 外层二分答案, 每次检测区间中值 mid 是否可作为答案
  - 通过可行或不可行, 将答案范围缩小为 mid 左或者右
- 举个例子, 比如刚才这道题.

T6 「BZOJ1816」[CQOI2010] 扑克牌

- 我们将初始答案范围设为 $[0, C]$ 其中 C 为牌数的最大值.
- 判断 mid 是否可行: (这里用到了贪心思想)
  - 每次找到牌数最小的牌, 将牌数加一, 重复 mid 次, 最后找到最小值.
  - 如果最小值大于或等于 mid 则可行, 否则不可行.
- 如果可行把左端点设为 mid , 否则将右端点设为 $mid - 1$.
- 持续到左右端点相同时停止.

倍增

二进制！ 指数级！

倍增

- 倍增是一种十分稳定的 $O(m\log n)$ 级别的算法, 在序列上和树上均能够使用, 但是灵活性不强, 对信息类型要求较严格.
- 在序列上, 通常使用 $f(i,j)$ 表示区间 $[i,i+2^j-1]$ 的信息.
- 在树上, 通常使用 $f(i,j)$ 表示从节点 i 往上长度为 2^j 的路径的信息.

T7 模板题

- Description:
  - 给出一个 n 个点的图，每一个点 i 有一条有向边指向 a_i . 有 m 次询问，每次询问从结点 t 走 k 步之后到达哪个点.
- Restraint:
  - $1 \leq n \leq 10^5, 0 \leq m \leq 10^5, k \leq 10^{18}$

T7 模板题

- 直接预处理从 i 节点走 2^j 步到达哪个点, 每次查询时根据 k 的二进制位来跳跃.
- Time Complexity: $O(n\log k + m\log k)$

T8 又是一道模板题

- Description:
  - 给定一棵大小为 n 的有根点权树,支持以下操作:
    - 换根
    - 修改点权
    - 查询以 x 为根的子树的最小值
- Restraint:
  - $1 \le n \le 10^5, 0 \le \ m \le 10^5, k \le 10^{18}$

T8 又是一道模板题

- 首先树转序列，求出这棵树 dfs 序列
- 然后建立倍增数组，记录区间最小值
- 每次查询分类讨论询问点和根节点的关系

分块

灵活的根号！灵活的除法！

分块

- 分块是指将数据分为多块，每一块维护整体的信息，或者更加方便求出整体信息
- 整块地处理能够降低时间复杂度
- 调节块的大小可以平衡时间复杂度

T9 「BZOJ2453」维护队列

- Description:
  - 长度为 n 的序列, 每一个位置有一个颜色. 现在要支持 m 次以下操作:
    - 查询 $[l, r]$ 中有多少种颜色
    - 更改某一位置的颜色
- Restraint:
  - $1 \le n \le 10000, 0 \le\ m \le 10000,$ 修改次数 $\le 1000, 1 \le$ 颜色标号 $\le 10^6$

T9 「BZOJ2453」维护队列

- 如何查询区间 $[l, r]$ 的颜色数？
- 对每个位置记录前一次该颜色出现的位置 $pre(i)$，每次查询 $[l, r]$ 中 $pre(i)$ 小于 l 的数目.
- 同时要支持修改？
- 我们分块，块内按照 $pre(i)$ 排序，如果需要修改直接重构块.
- 假设块的大小为 B，则时间复杂度为 $O\left(n\log B + m_1\frac{n}{B}\log B + m_2 B\log B\right)$.
- 取后面两项相等，得到 $B = \sqrt{10n}$，具体可以造几组数据手动调整.

T10 「BZOJ3687」「FJ2014集训」简单题

- Description:
  - 给出数集 U , 且 $n = |U|$. 定义 $sum(S)$ 表示数集 S 中所有数字的和.
  - 现在求对于所有 U 的子集 S , $sum(S)$ 的异或和.
- Restraint:
  - $1 \leq n \leq 1000, \sum a_i \leq 2000000$

T10 「BZOJ3687」 「FJ2014集训」简单题

- 我们弄一个长度为 $\sum a_i$ 的 $bitset$, 如果第 i 位有值, 表示有奇数个子集和为 i .
- 然后每一位异或起来就可以了.
- Time Complexity: $O(n\frac{\sum a_i}{\omega})$.

完结撒花～