Solutions For Claris' Contest # 4 Day 2

Claris

Hangzhou Dianzi University

2016 年 12 月 23 日

友好城市 (friend.c/cpp/pas)

给定一张 n 个点，m 条边的有向图。

友好城市 (friend.c/cpp/pas)

给定一张 n 个点，m 条边的有向图。

q 次询问，每次给定一个区间 $[l, r]$，问仅保留编号在这个区间内的边时，能互相到达的点对数。

友好城市 (friend.c/cpp/pas)

给定一张 n 个点，m 条边的有向图。

q 次询问，每次给定一个区间 $[l, r]$，问仅保留编号在这个区间内的边时，能互相到达的点对数。

- 对于 30% 的数据，$n \leq 50$，$m \leq 1000$，$q \leq 1000$。

友好城市 (friend.c/cpp/pas)

给定一张 n 个点，m 条边的有向图。

q 次询问，每次给定一个区间 $[l, r]$，问仅保留编号在这个区间内的边时，能互相到达的点对数。

- 对于 30% 的数据，$n \leq 50$，$m \leq 1000$，$q \leq 1000$。
- 对于 70% 的数据，$n \leq 150$，$m \leq 300000$，$q \leq 50000$，询问区间互不包含。

友好城市 (friend.c/cpp/pas)

给定一张 n 个点，m 条边的有向图。

q 次询问，每次给定一个区间 $[l, r]$，问仅保留编号在这个区间内的边时，能互相到达的点对数。

- 对于 30% 的数据，$n \leq 50$，$m \leq 1000$，$q \leq 1000$。
- 对于 70% 的数据，$n \leq 150$，$m \leq 300000$，$q \leq 50000$，询问区间互不包含。
- 对于 100% 的数据，$n \leq 150$，$m \leq 300000$，$q \leq 50000$。

30 分做法

- 对于 30% 的数据，$n \leq 50$，$m \leq 1000$，$q \leq 1000$。

30 分做法

- 对于 30% 的数据，$n \leq 50$，$m \leq 1000$，$q \leq 1000$。
- 暴力求出强连通分量，那么一个点数为 t 的强连通分量对答案的贡献为 $\frac{t(t-1)}{2}$。

30 分做法

- 对于 30% 的数据，$n \leq 50$，$m \leq 1000$，$q \leq 1000$。
- 暴力求出强连通分量，那么一个点数为 t 的强连通分量对答案的贡献为 $\frac{t(t-1)}{2}$。
- 时间复杂度 $O(q(n+m))$。

70 分做法

- 对于 70% 的数据，$n \leq 150$，$m \leq 300000$，$q \leq 50000$，询问区间互不包含。

70 分做法

- 对于 70% 的数据，$n \leq 150$，$m \leq 300000$，$q \leq 50000$，询问区间互不包含。
- 注意到区间互不包含，因此可以转化成 $O(m)$ 次添加或删除边的操作。

70 分做法

- 对于 70% 的数据，$n \leq 150$，$m \leq 300000$，$q \leq 50000$，询问区间互不包含。
- 注意到区间互不包含，因此可以转化成 $O(m)$ 次添加或删除边的操作。
- 除了 Tarjan 算法，求强连通分量还可以用 Kosaraju 算法，只需要在正反图各做一次 DFS 即可。

70 分做法

- 对于 70% 的数据，$n \leq 150$，$m \leq 300000$，$q \leq 50000$，询问区间互不包含。
- 注意到区间互不包含，因此可以转化成 $O(m)$ 次添加或删除边的操作。
- 除了 Tarjan 算法，求强连通分量还可以用 Kosaraju 算法，只需要在正反图各做一次 DFS 即可。
- 面对稠密图，Kosaraju 算法的瓶颈在于寻找与点 x 相连且未访问过的点。

70 分做法

- 面对稠密图，Kosaraju 算法的瓶颈在于寻找与点 x 相连且未访问过的点。

70 分做法

- 面对稠密图，Kosaraju 算法的瓶颈在于寻找与点 x 相连且未访问过的点。
- 考虑用 bitset 来保存边表 $g[x]$，以及未访问过的点集 S，那么取出 $g[x]$ *and* S 内的所有 1 即可。

70 分做法

- 面对稠密图，Kosaraju 算法的瓶颈在于寻找与点 x 相连且未访问过的点。
- 考虑用 bitset 来保存边表 $g[x]$，以及未访问过的点集 S，那么取出 $g[x]$ *and* S 内的所有 1 即可。
- 这样一来，求强连通分量的复杂度降低为 $O(\frac{n^2}{32})$。

70 分做法

- 面对稠密图，Kosaraju 算法的瓶颈在于寻找与点 x 相连且未访问过的点。
- 考虑用 bitset 来保存边表 $g[x]$，以及未访问过的点集 S，那么取出 $g[x]$ *and* S 内的所有 1 即可。
- 这样一来，求强连通分量的复杂度降低为 $O(\frac{n^2}{32})$。
- 时间复杂度 $O(m + \frac{qn^2}{32})$。

100 分做法

- 当询问区间没有特殊性质的时候，70 分算法的瓶颈在于取出所有在 $[l, r]$ 内的边。

100 分做法

- 当询问区间没有特殊性质的时候，70 分算法的瓶颈在于取出所有在 $[l, r]$ 内的边。
- 考虑将边分成 $\sqrt{m}$ 块，用 bitset 保存每块的边表，然后用 ST 表维护横跨任意两块的边表。

100 分做法

- 当询问区间没有特殊性质的时候，70 分算法的瓶颈在于取出所有在 $[l, r]$ 内的边。
- 考虑将边分成 $\sqrt{m}$ 块，用 bitset 保存每块的边表，然后用 ST 表维护横跨任意两块的边表。
- 那么对于每个询问，只需要在 ST 表中查询一次，然后往左往右暴力添加 $O(\sqrt{m})$ 条边即可。

100 分做法

- 当询问区间没有特殊性质的时候，70 分算法的瓶颈在于取出所有在 $[l, r]$ 内的边。
- 考虑将边分成 $\sqrt{m}$ 块，用 bitset 保存每块的边表，然后用 ST 表维护横跨任意两块的边表。
- 那么对于每个询问，只需要在 ST 表中查询一次，然后往左往右暴力添加 $O(\sqrt{m})$ 条边即可。
- 时间复杂度 $O\left(\frac{\left(\sqrt{m}\log m+q\right)n^2}{32}+q\sqrt{m}\right)$。

最长路径 (path.c/cpp/pas)

对于所有的 $1 \leq k \leq n$，求从 1 出发最长简单路径长度恰好为 k 的 n 个点带标号竞赛图个数。

最长路径 (path.c/cpp/pas)

对于所有的 $1 \leq k \leq n$，求从 1 出发最长简单路径长度恰好为 k 的 n 个点带标号竞赛图个数。

- 对于 60% 的数据，$n \leq 8$。

最长路径 (path.c/cpp/pas)

对于所有的 $1 \leq k \leq n$，求从 1 出发最长简单路径长度恰好为 k 的 n 个点带标号竞赛图个数。

- 对于 60% 的数据，$n \leq 8$。
- 对于 100% 的数据，$n \leq 2000$。

60 分做法

- 对于 60% 的数据，$n \leq 8$。

60 分做法

- 对于 60% 的数据，$n \leq 8$。
- 暴力枚举所有竞赛图然后打表即可。

60 分做法

- 对于 60% 的数据，$n \leq 8$。
- 暴力枚举所有竞赛图然后打表即可。
- 时间复杂度 $O\left(2^{\frac{n(n-1)}{2}}\right)$。

100 分做法

Lemma 1

强连通竞赛图必然存在哈密顿回路。

100 分做法

Lemma 1

强连通竞赛图必然存在哈密顿回路。

Lemma 2

竞赛图按强连通分量缩点之后是一条链。

100 分做法

Lemma 1

强连通竞赛图必然存在哈密顿回路。

Lemma 2

竞赛图按强连通分量缩点之后是一条链。

- 因此将竞赛图缩点之后，从 1 出发的最长简单路径长度等于 1 所在强连通分量之前的点数之和加上 1 所在强连通分量的点数。

100 分做法

- 设 $f[i]$ 表示 i 个点的带标号竞赛图个数，$g[i]$ 表示 i 个点的带标号强连通竞赛图个数。

100 分做法

- 设 $f[i]$ 表示 i 个点的带标号竞赛图个数，$g[i]$ 表示 i 个点的带标号强连通竞赛图个数。
- $f[n] = 2^{\frac{n(n-1)}{2}}$

100 分做法

- 设 $f[i]$ 表示 i 个点的带标号竞赛图个数，$g[i]$ 表示 i 个点的带标号强连通竞赛图个数。
- $f[n] = 2^{\frac{n(n-1)}{2}}$
- 对于 g 的计算，考虑容斥，枚举拓扑序最小的强连通块的大小 i，则有 $g[n] = f[n] - \sum_{i=1}^{n-1} C(n,i)g[i]f[n-i]$

100 分做法

- 设 $f[i]$ 表示 i 个点的带标号竞赛图个数，$g[i]$ 表示 i 个点的带标号强连通竞赛图个数。
- $f[n] = 2^{\frac{n(n-1)}{2}}$
- 对于 g 的计算，考虑容斥，枚举拓扑序最小的强连通块的大小 i，则有 $g[n] = f[n] - \sum_{i=1}^{n-1} C(n,i)g[i]f[n-i]$
- 枚举 1 所在强连通分量之前的点数之和 i，1 所在强连通分量的点数 j，那么

  $ans[i+j] += C(n-1,i)C(n-i-1,j)f[i]g[j]g[n-i-j]$

100 分做法

- 设 $f[i]$ 表示 i 个点的带标号竞赛图个数，$g[i]$ 表示 i 个点的带标号强连通竞赛图个数。
- $f[n] = 2^{\frac{n(n-1)}{2}}$
- 对于 g 的计算，考虑容斥，枚举拓扑序最小的强连通块的大小 i，则有 $g[n] = f[n] - \sum_{i=1}^{n-1} C(n,i)g[i]f[n-i]$
- 枚举 1 所在强连通分量之前的点数之和 i，1 所在强连通分量的点数 j，那么

  $ans[i+j] += C(n-1,i)C(n-i-1,j)f[i]g[j]g[n-i-j]$
- 时间复杂度 $O(n^2)$。

异或与区间加 (xor.c/cpp/pas)

给定一个长度为 n 的序列 a 和一个整数 k，要求维护一个长度为 n 的数列 b。

异或与区间加 (xor.c/cpp/pas)

给定一个长度为 n 的序列 a 和一个整数 k，要求维护一个长度为 n 的数列 b。

m 次操作，每次给定 st, en, w，将所有左右端点 l, r 都在 $[st, en]$ 内且 $a[l]$ 到 $a[r]$ 的异或和为 k 的区间，将 b 对应区间内每个数都加上 w。

异或与区间加 (xor.c/cpp/pas)

给定一个长度为 n 的序列 a 和一个整数 k，要求维护一个长度为 n 的数列 b。

m 次操作，每次给定 st, en, w，将所有左右端点 l, r 都在 $[st, en]$ 内且 $a[l]$ 到 $a[r]$ 的异或和为 k 的区间，将 b 对应区间内每个数都加上 w。

输出 m 次操作完毕之后的 b 序列。

异或与区间加 (xor.c/cpp/pas)

给定一个长度为 n 的序列 a 和一个整数 k，要求维护一个长度为 n 的数列 b。

m 次操作，每次给定 st, en, w，将所有左右端点 l, r 都在 $[st, en]$ 内且 $a[l]$ 到 $a[r]$ 的异或和为 k 的区间，将 b 对应区间内每个数都加上 w。

输出 m 次操作完毕之后的 b 序列。

- 对于 20% 的数据，$n, m \leq 50$，$a_i \leq 2^6$。

异或与区间加 (xor.c/cpp/pas)

给定一个长度为 n 的序列 a 和一个整数 k，要求维护一个长度为 n 的数列 b。

m 次操作，每次给定 st, en, w，将所有左右端点 l, r 都在 $[st, en]$ 内且 $a[l]$ 到 $a[r]$ 的异或和为 k 的区间，将 b 对应区间内每个数都加上 w。

输出 m 次操作完毕之后的 b 序列。

- 对于 20% 的数据，$n, m \leq 50$，$a_i \leq 2^6$。
- 对于 40% 的数据，$n, m \leq 4000$，$a_i \leq 2^{30}$。

异或与区间加 (xor.c/cpp/pas)

给定一个长度为 n 的序列 a 和一个整数 k，要求维护一个长度为 n 的数列 b。

m 次操作，每次给定 st, en, w，将所有左右端点 l, r 都在 $[st, en]$ 内且 $a[l]$ 到 $a[r]$ 的异或和为 k 的区间，将 b 对应区间内每个数都加上 w。

输出 m 次操作完毕之后的 b 序列。

- 对于 20% 的数据，$n, m \leq 50$，$a_i \leq 2^6$。
- 对于 40% 的数据，$n, m \leq 4000$，$a_i \leq 2^{30}$。
- 对于 70% 的数据，$n, m \leq 150000$，$a_i \leq 2^6$。

异或与区间加 (xor.c/cpp/pas)

给定一个长度为 n 的序列 a 和一个整数 k，要求维护一个长度为 n 的数列 b。

m 次操作，每次给定 st, en, w，将所有左右端点 l, r 都在 $[st, en]$ 内且 $a[l]$ 到 $a[r]$ 的异或和为 k 的区间，将 b 对应区间内每个数都加上 w。

输出 m 次操作完毕之后的 b 序列。

- 对于 20% 的数据，$n, m \leq 50$，$a_i \leq 2^6$。
- 对于 40% 的数据，$n, m \leq 4000$，$a_i \leq 2^{30}$。
- 对于 70% 的数据，$n, m \leq 150000$，$a_i \leq 2^6$。
- 对于 100% 的数据，$n, m \leq 150000$，$a_i \leq 2^{30}$。

20 分做法

- 对于 20% 的数据，$n, m \leq 50$，$a_i \leq 2^6$。

20 分做法

- 对于 20% 的数据，$n, m \leq 50$，$a_i \leq 2^6$。
- 对于每个操作，暴力枚举所有区间即可。

20 分做法

- 对于 20% 的数据，$n, m \leq 50$，$a_i \leq 2^6$。
- 对于每个操作，暴力枚举所有区间即可。
- 时间复杂度 $O(mn^3)$。

40 分做法

- 对于 40% 的数据，$n, m \leq 4000$，$a_i \leq 2^{30}$。

40 分做法

- 对于 40% 的数据，$n, m \leq 4000$，$a_i \leq 2^{30}$。
- 利用前缀异或和可以将区间异或和以及区间加的复杂度降低至 $O(1)$。

40 分做法

- 对于 40% 的数据，$n, m \leq 4000$，$a_i \leq 2^{30}$。
- 利用前缀异或和可以将区间异或和以及区间加的复杂度降低至 $O(1)$。
- 将操作按右端点从小到大排序。

40 分做法

- 对于 40% 的数据，$n, m \leq 4000$，$a_i \leq 2^{30}$。
- 利用前缀异或和可以将区间异或和以及区间加的复杂度降低至 $O(1)$。
- 将操作按右端点从小到大排序。
- 枚举 l，从 n 到 l 枚举所有 r，同时用双指针维护出所有包含了当前区间的所有操作的 w 的和。

40 分做法

- 对于 40% 的数据，$n, m \leq 4000$，$a_i \leq 2^{30}$。
- 利用前缀异或和可以将区间异或和以及区间加的复杂度降低至 $O(1)$。
- 将操作按右端点从小到大排序。
- 枚举 l，从 n 到 l 枚举所有 r，同时用双指针维护出所有包含了当前区间的所有操作的 w 的和。
- 那么对于一个合法的区间，直接进行区间加即可。

40 分做法

- 对于 40% 的数据，$n, m \leq 4000$，$a_i \leq 2^{30}$。
- 利用前缀异或和可以将区间异或和以及区间加的复杂度降低至 $O(1)$。
- 将操作按右端点从小到大排序。
- 枚举 l，从 n 到 l 枚举所有 r，同时用双指针维护出所有包含了当前区间的所有操作的 w 的和。
- 那么对于一个合法的区间，直接进行区间加即可。
- 时间复杂度 $O(n(n+m))$。

70 分做法

- 对于 70% 的数据，$n, m \leq 150000$，$a_i \leq 2^6$。

70 分做法

- 对于 70% 的数据，$n, m \leq 150000$，$a_i \leq 2^6$。
- 求出前缀异或和 s，设 $A[i] = s[i-1]$，$B[i] = s[i]$，那么一个区间合法等价于 $A[l]$ xor $B[r] = k$。

70 分做法

- 对于 70% 的数据，$n, m \leq 150000$，$a_i \leq 2^6$。
- 求出前缀异或和 s，设 $A[i] = s[i-1]$，$B[i] = s[i]$，那么一个区间合法等价于 $A[l]$ xor $B[r] = k$。
- 设 $f[i]$ 表示对 b 序列操作的前缀和，那么区间 $[l, r]$ 加上 w 等价于 $f[l] += w$，$f[r+1] -= w$。

70 分做法

- 对于 70% 的数据，$n, m \leq 150000$，$a_i \leq 2^6$。
- 求出前缀异或和 s，设 $A[i] = s[i-1]$，$B[i] = s[i]$，那么一个区间合法等价于 $A[l]$ xor $B[r] = k$。
- 设 $f[i]$ 表示对 b 序列操作的前缀和，那么区间 $[l, r]$ 加上 w 等价于 $f[l] += w$，$f[r+1] -= w$。
- 枚举所有 $A[l]$ 的值 p，那么对应的 B 也知道了，我们只关心值是 p 和 p xor k 的左右端点。

70 分做法

- 对于每个感兴趣的 l，对应的 $f[l]$ 要加上多少呢？

70 分做法

- 对于每个感兴趣的 l，对应的 $f[l]$ 要加上多少呢？
- 考虑将每个操作 $[st, en], w$ 拆成 2 份：

70 分做法

- 对于每个感兴趣的 I，对应的 $f[I]$ 要加上多少呢？
- 考虑将每个操作 $[st, en], w$ 拆成 2 份：
- 1.$[1, en], w$ 同时 $[1, st-1], -w$

70 分做法

- 对于每个感兴趣的 l，对应的 $f[l]$ 要加上多少呢？
- 考虑将每个操作 $[st, en], w$ 拆成 2 份：
- 1.$[1, en], w$ 同时 $[1, st-1], -w$
- 2.$l < st, st \leq r \leq en, -w$

70 分做法

- 对于每个感兴趣的 l，对应的 $f[l]$ 要加上多少呢？
- 考虑将每个操作 $[st, en], w$ 拆成 2 份：
- 1.$[1, en], w$ 同时 $[1, st-1], -w$
- 2.$l < st, st \le r \le en, -w$
- 类型 1：对于每个 r，求出其后面对应操作的 w 之和，那么每个 l 要加上的就是它们后面可行 r 的贡献之和。

70 分做法

- 对于每个感兴趣的 l，对应的 $f[l]$ 要加上多少呢？
- 考虑将每个操作 $[st, en], w$ 拆成 2 份：
- 1.$[1, en], w$ 同时 $[1, st-1], -w$
- 2.$l < st, st \le r \le en, -w$
- 类型 1：对于每个 r，求出其后面对应操作的 w 之和，那么每个 l 要加上的就是它们后面可行 r 的贡献之和。
- 类型 2：求出每种操作区间内可行 r 的个数，乘以自己的权值 w，那么每个 l 要减去的就是它们后面区间的贡献之和。

70 分做法

- 对于每个感兴趣的 l，对应的 $f[l]$ 要加上多少呢？
- 考虑将每个操作 $[st, en], w$ 拆成 2 份：
- 1.$[1, en], w$ 同时 $[1, st-1], -w$
- 2.$l < st, st \le r \le en, -w$
- 类型 1：对于每个 r，求出其后面对应操作的 w 之和，那么每个 l 要加上的就是它们后面可行 r 的贡献之和。
- 类型 2：求出每种操作区间内可行 r 的个数，乘以自己的权值 w，那么每个 l 要减去的就是它们后面区间的贡献之和。
- 可以用前缀和以及递推做到 $O(n+m)$。

70 分做法

- 对于每个感兴趣的 r，对应的 $f[r+1]$ 要减去多少呢？

70 分做法

- 对于每个感兴趣的 r，对应的 $f[r+1]$ 要减去多少呢？
- 依然将每个操作 $[st, en], w$ 拆成 2 份：

70 分做法

- 对于每个感兴趣的 r，对应的 $f[r+1]$ 要减去多少呢？
- 依然将每个操作 $[st, en], w$ 拆成 2 份：
- 1.$[1, en], w$ 同时 $[1, st-1], -w$

70 分做法

- 对于每个感兴趣的 r，对应的 $f[r+1]$ 要减去多少呢？
- 依然将每个操作 $[st, en], w$ 拆成 2 份：
- 1.$[1, en], w$ 同时 $[1, st-1], -w$
- 2.$l < st, st \leq r \leq en, -w$

70 分做法

- 对于每个感兴趣的 r，对应的 $f[r+1]$ 要减去多少呢？
- 依然将每个操作 $[st, en], w$ 拆成 2 份：
- 1.$[1, en], w$ 同时 $[1, st-1], -w$
- 2.$l < st, st \leq r \leq en, -w$
- 类型 1：对于每个 r，求出其前面可行 l 的个数，乘以其后面对应操作的 w 之和，即可得到它应该减去多少。

70 分做法

- 对于每个感兴趣的 r，对应的 $f[r+1]$ 要减去多少呢？
- 依然将每个操作 $[st, en], w$ 拆成 2 份：
- 1.$[1, en], w$ 同时 $[1, st-1], -w$
- 2.$l < st, st \le r \le en, -w$
- 类型 1：对于每个 r，求出其前面可行 l 的个数，乘以其后面对应操作的 w 之和，即可得到它应该减去多少。
- 类型 2：求出每种操作区间前面可行 l 的个数，那么每个 r 要加上的就是经过了这个点的区间的贡献之和。

70 分做法

- 对于每个感兴趣的 r，对应的 $f[r+1]$ 要减去多少呢？
- 依然将每个操作 $[st, en], w$ 拆成 2 份：
- 1.$[1, en], w$ 同时 $[1, st-1], -w$
- 2.$l < st, st \le r \le en, -w$
- 类型 1：对于每个 r，求出其前面可行 l 的个数，乘以其后面对应操作的 w 之和，即可得到它应该减去多少。
- 类型 2：求出每种操作区间前面可行 l 的个数，那么每个 r 要加上的就是经过了这个点的区间的贡献之和。
- 这也可以用前缀和以及递推做到 $O(n+m)$。

70 分做法

- 因为 $a_i < 2^6$，因此暴力枚举所有可能的 A 即可。

70 分做法

- 因为 $a_i < 2^6$，因此暴力枚举所有可能的 A 即可。
- 时间复杂度 $O(2^6(n+m))$。

100 分做法

- 考虑另一种暴力：

100 分做法

- 考虑另一种暴力：
- 枚举每个右端点 r，然后枚举所有 A 的值为 $B[r]$ xor k 的左端点 l，求出贡献，进行区间加。

100 分做法

- 考虑另一种暴力：
- 枚举每个右端点 r，然后枚举所有 A 的值为 $B[r]$ xor k 的左端点 l，求出贡献，进行区间加。
- 如何快速求出完全包含了 $[l, r]$ 的操作的权值之和？

100 分做法

- 考虑另一种暴力：
- 枚举每个右端点 r，然后枚举所有 A 的值为 $B[r]$ xor k 的左端点 l，求出贡献，进行区间加。
- 如何快速求出完全包含了 $[l, r]$ 的操作的权值之和？
- 考虑将所有操作按右端点从大到小排序，然后从 n 到 1 枚举 r，同时加入所有右端点不小于 r 的操作，那么需要查询 $st \leq l$ 的操作的权值之和。

100 分做法

- 考虑另一种暴力：
- 枚举每个右端点 r，然后枚举所有 A 的值为 $B[r]$ xor k 的左端点 l，求出贡献，进行区间加。
- 如何快速求出完全包含了 $[l, r]$ 的操作的权值之和？
- 考虑将所有操作按右端点从大到小排序，然后从 n 到 1 枚举 r，同时加入所有右端点不小于 r 的操作，那么需要查询 $st \leq l$ 的操作的权值之和。
- 加入只有 $O(m)$ 次，考虑牺牲加入的复杂度，来让查询做到最优。

100 分做法

- 将序列分块，每一块维护要加上多少的标记，那么加入一个数的时候，只需要将其所在块内后面每个数暴力修改，然后将后面所有块暴力打上标记即可。每次加入的复杂度为 $O(\sqrt{n})$，查询复杂度仅为 $O(1)$。

100 分做法

- 将序列分块，每一块维护要加上多少的标记，那么加入一个数的时候，只需要将其所在块内后面每个数暴力修改，然后将后面所有块暴力打上标记即可。每次加入的复杂度为 $O(\sqrt{n})$，查询复杂度仅为 $O(1)$。
- 枚举每个右端点 r，然后枚举所有 A 的值为 $B[r]$ xor k 的左端点 l，如果这种 l 的出现次数不超过 $\sqrt{n}$，那么就可以用上述方法计算贡献。

100 分做法

- 将序列分块，每一块维护要加上多少的标记，那么加入一个数的时候，只需要将其所在块内后面每个数暴力修改，然后将后面所有块暴力打上标记即可。每次加入的复杂度为 $O(\sqrt{n})$，查询复杂度仅为 $O(1)$。
- 枚举每个右端点 r，然后枚举所有 A 的值为 $B[r]$ xor k 的左端点 l，如果这种 l 的出现次数不超过 $\sqrt{n}$，那么就可以用上述方法计算贡献。
- 如果这种 l 的出现次数超过了 $\sqrt{n}$，那么这种 l 最多只有 $\sqrt{n}$ 种，用 70 分算法解决即可。

100 分做法

- 将序列分块，每一块维护要加上多少的标记，那么加入一个数的时候，只需要将其所在块内后面每个数暴力修改，然后将后面所有块暴力打上标记即可。每次加入的复杂度为 $O(\sqrt{n})$，查询复杂度仅为 $O(1)$。
- 枚举每个右端点 r，然后枚举所有 A 的值为 $B[r]$ xor k 的左端点 l，如果这种 l 的出现次数不超过 $\sqrt{n}$，那么就可以用上述方法计算贡献。
- 如果这种 l 的出现次数超过了 $\sqrt{n}$，那么这种 l 最多只有 $\sqrt{n}$ 种，用 70 分算法解决即可。
- 时间复杂度 $O((n+m)\sqrt{n})$，常数很小。

Thank you!