Game of Numbers解题报告

长郡中学 任瀚林

2015 年 10 月 18 日

§1 题面

§1.1 问题描述

(注：这是codechef上的官方翻译)

八神酱正在玩一个关于数的游戏。他拥有两个长度为N的数组，$A_1, A_2, \ldots, A_N$和$B_1, B_2, \ldots B_N$。

让我们维护两个二元组的集合S_1, S_2，初始时集合均为空。每次操作，他将会选择两个数对$(i, j), (p, q)$，满足(i, j)不在S_1中,(p, q)不在S_2中，$B_j > A_i, B_p < A_q, \gcd(A_i, B_j) \neq 1, \gcd(A_q, B_p) \neq 1$，且$\gcd(A_q, B_p)$与$\gcd(A_i, B_j)$不互质。如果这样的数对存在，他会将$(i, j), (p, q)$分别加入到集合$S_1, S_2$中。

八神酱想知道自己最多可以进行多少次操作，你能帮助他吗?

§1.2 输入格式

输入数据的第一行包含一个整数T——测试数据的组数。

对于每组测试数据，第一行包含一个整数N。接下来的一行包含N个整数$A_1, A_2, \ldots A_N$。接下来的一行包含N个整数$B_1, B_2, \ldots B_N$。

§1.3 输出格式

对于每组测试数据，输出一行表示结果。

§1.4 样例输入

```
2
4
2 5 6 14
3 4 7 10
2
2 3
5 7
```

§1.5 样例输出

```
3
0
```

§1.6 样例说明

样例1：以下是一种合法的操作方案。我们给出每次操作的$(i, j), (p, q)$：

第一轮：$(1, 2), (2, 3)$

第二轮：$(1, 4), (2, 4)$

第三轮：$(3, 4), (4, 4)$

样例2：没有合法的操作。

§1.7 数据规模和约定

测试点比例	$N \le$
5%	4
10%	15
15%	30
10%	50
10%	100
15%	200
15%	300
20%	400

对于所有测试数据，$1 \le N \le 400$，$1 \le T \le 10$，$1 \le A_i, B_i \le 10^9$。

§2 解题报告

§2.1 算法1

爆搜。

我们记录下S_1, S_2的内容就可以爆搜了。事实上不同的$\{S_1, S_2\}$个数只有$O(2^{N^2})$个。复杂度$O(T2^{N^2}N^4)$，期望得分5分。

见`source\GNUM1\GNUM.cpp`。

§2.2 算法2

我们发现一对数对$(i,j),(p,q)$能否被加入，与当前的S_1, S_2无关，只与A, B两个数组有关。

建立一个$N^2 \times N^2$的二分图，点表示数对，在可以一起加入的数对之间连边，会发现答案等于二分图匹配的大小。

所以可以使用匈牙利算法或Dinic算法跑二分图最大匹配，复杂度$O(TN^6)$或$O(TN^5)$，期望得分30分。以Dinic为例说明建图：

考虑所有的满足$A_i < B_j, \gcd(A_i, B_j) > 1$的数对$(i,j)$，将$\gcd(A_i, B_j)$加入数组$U$。同理考虑所有的满足$A_i > B_j, \gcd(A_i, B_j) > 1$的数对$(i,j)$，将$\gcd(A_i, B_j)$加入数组$V$。那么Dinic建出来的图会长成这样：

- S向U中的所有元素连边，容量为1；
- V中的所有元素向T连边，容量为1；
- 如果$\gcd(U_i, V_j) > 1$，那么U_i向V_j连边，容量为1。

见`source\GNUM21\GNUM.cpp` (匈牙利算法)或`source\GNUM22\GNUM.cpp` (Dinic算法)。

§2.3 算法3

用$\text{networkflow}(n,m)$表示$O(n)$个点$O(m)$条边的图上跑网络流的复杂度。

算法2的复杂度是$O(T\text{networkflow}(n^2, n^4))$，显然会超时。

我们考虑优化连边，$\gcd(U_i, V_j) > 1$说明存在一个质数p满足$p|U_i$且$p|V_j$。

我们新建若干个点表示质数，把每一个U_i向整除它的质数连边，容量为1；把每一个质数向被它整除的V_j连边，容量为1。

这样做是对的吗?我们设原图为G_1，新图为G_2，那么我们证明：$\text{maxflow}(G_1) = \text{maxflow}(G_2)$。为了证明此结论，我们证明对于$G_1$中的任意一个流$f$，$G_2$中存在一个流$f'$，它们的流量相等(i)；对于$G_2$中任意一个流$f'$，存在$G_1$中的一个流$f$，它们的流量相等(ii)。

首先G_1中的流一定是这样的形式：不考虑S,T，中间的边中有流量的一定是一个匹配；在匹配中的点与S(或T)之间的连边有流量。对于任意一条属于匹配集合的边(x,y)，在G_2中找到x,y对应的点，由于$\gcd(U_x,V_y)>1$，所以存在质数p使得$p|U_x,p|V_y$，给G_2中的$S\to x\to p\to y\to T$加上1的流量。由于一个点x只在匹配中出现一次，所以G_2的任意一条边流量不会大于1。所以(i)成立。

让我们证明(ii)的正确性。考虑一个质数p，因为流入p的流量等于流出p的流量，所以所有满足$x\to p$有流量的x与所有满足$p\to y$有流量的y可以一一配对。设边$S\to x$有流量，那么流量一定是1，因此只存在一个质数p使得$x\to p$的边有流量。所以一个x只会被配对一次，且若x,y配对了那么$\gcd(x,y)\geq p>1$。同理一个y只会被配对一次。这样我们得到了U中部分点与V中部分点的一个匹配，且互相匹配的点权值的gcd大于1。这就对应着G_1中的一个流。故(ii)成立。

因为$U_i,V_j\leq 10^9$，所以可以认为一个数不同的质因数个数不超过$X=9$个。这样可能的质数点的个数就不超过XN。

边数是XN^2级别的。

复杂度$O(T\text{networkflow}(N^2,XN^2))$，根据实现优劣期望得分$70\sim 100$分。对于单位容量的图，Dinic算法的复杂度可以达到$O(m\min(n^{\frac{2}{3}},m^{\frac{1}{2}}))$，故可以认为本问题复杂度是$O(TX^{\frac{3}{2}}N^3)$的。而且网络流算法的运行时间一般达不到上界。

见`source\GNUM3\GNUM.cpp`。

§2.4 算法4

可以考虑将U或V中相等的元素合并，例如$U_{i_1}=U_{i_2}=\cdots=U_{i_x}$的时候，可以建立一个点代替$U_{i_1},U_{i_2},\ldots,U_{i_x}$，$S$向它的连边容量为$x$。$V$中的相等元素也可以同样处理。新的点与中间的质数点之间的连边容量可以设为$+\infty$。正确性较显然，故略去证明。

由于本人水平有限，并不会分析其复杂度，也没能将点数卡到$O(N^2)$。

见`source\GNUM4\GNUM.cpp`。

期望得分100分。

§3 数据生成

每个测试点有10组数据，所以可以考虑针对不同的算法，使用不同的数据生成器。见`programs\gen_data.cpp`。

`gen1(n, l, r)`：A_i,B_i在一个给定范围$[l,r]$内随机生成。这个生成器是最弱的，但是应该可以对付很多WA的程序。

`gen2(n, p)`：A_i,B_i是$[1,\lfloor\frac{10^9}{p}\rfloor]$内的随机数乘以$p$的结果。该生成器可以使算法2、3建图的点数达到上界。

`gen3(n, p[])`：考虑质数$p_1,p_2,\ldots,p_k(k>2)$，令P为它们的乘积。每次随机一个$1\leq i\leq k$，然后随机一个数使得它是$\frac{P}{p_i}$的倍数，但不是p_i的倍数。也可以把算法2、3建图的点数卡到上界。取$k=9$，p是前9小的质数，可以把算法3连边的复杂度卡到上界。

`gen4(n, x, y)`：考虑两个满足$\gcd(x,y)=1$的正整数(x,y)。生成的A_i均为x的倍数，B_i均为y的倍数。取较好的x,y可以把算法4的两边的点数都卡到1000左右。目前没有想到更好的卡算法4的方法。

前面90个测试点是4个生成器混用，后面10个测试点只使用了生成器$3,4$。

§4 参考文献

[1] codechef tutorial：`https://discuss.codechef.com/questions/47245/gnum-editorial`