AtCoder ARC 094 D - Worst Case (700 点)

二分探索 マッチング 場合分け 解を変形していく(最適性を失わずに) AtCoder700点 AtCoder ABC/ARC-D

最適性を失わずに解を変形していくことで、都合のいい解の形だけ考えればいいようにする系の問題。

問題へのリンク

問題概要 (ARC 094 D / ABC 093 D)

$10^{10^{10}}$101010 人の参加者が 2 回のプログラミングコンテストに参加しました。順位がつくのですが、同順位はないものとします。参加者のスコアとは、2 回のコンテストの順位を掛け合わせた値です。

以下の QQ 回のクエリに答えよ:

- 整数 A, BA,B が与えられて、高橋君は 1 回目のコンテストで AA 位、2 回目のコンテストで BB 位をとったとして、高橋君よりスコアの小さい参加者の人数として考えられる最大値を求めよ。

制約

- $1 \leq Q \leq 100$1≤Q≤100
- $1 \leq A, B \leq 10^9$1≤A,B≤109

考えたこと

最大マッチング問題には違いない。

マッチングとしてはきっと Greedy に解ける系な気がする。少し考えてみると、

- 既に最大マッチングが得られていたら、まず左を 1, 2, 3, ..., A-1, A+1, A+2, ... という感じで連番になるように変形して、右も 1, 2, ..., B-1, B+1, B+2, ... という感じで連番になるように変形することができる (そうしても解が悪化しない)

ことがわかる。よって、左右ともに連番なもののみ考えればよくて、またマッチングのさせ方は、互いに逆順で OK。

つまり

- 左: 1, 2, ..., x のうち A を除いたもの
- 右: 1, 2, ..., x のうち B を除いたもの

を、左の最小と右の最大からスタートして順にマッチングさせていったときに、その最大値が AB-1 以下となるような最大の x を求めればいい。これを二分探索することにした。

その最大値の求め方としては、もし A と B を除く必要がなかったら、x/2 と x-x/2 の積でいいのだが、A と B が飛ばされるのを考える必要があるのが少し厄介。

でも実はほとんどの場合、実は x/2 と x-x/2 の積で OK。

なぜなら、A と B を一旦無視してマッチングを組み立てたときに

- A = B でない限り、最適な x では、A と B を飛ばしてマッチさせても x/2 と x-x/2 は組み合わされる

ことがわかるからである。

- A = B のときだけは例外で、でもそのときは二分探索するまでもなく、x = 2A-1 が最適だとわかる (答えは 2A-2)

計算量は、$O(Q \log \max(A, B))$O(Qlogmax(A,B))

```
#include <iostream>
using namespace std;

long long solve(long long a, long long b) {
    if (a == b) return (a-1)*2;
    long long lim = a*b - 1;
    long long low = 1, high = 1LL<<31;
    while (high - low > 1) {
        long long mid = (low + high) / 2;
        long long mul = (mid+1)/2 * (mid+1 - (mid+1)/2);
        if (mul > lim) high = mid;
        else low = mid;
    }
    return low - 1;
}

int main() {
    int Q; cin >> Q;
    for (int q = 0; q < Q; ++q) {
        long long a, b; cin >> a >> b;
        cout << solve(a, b) << endl;
    }
}
```

けんちょん (drken) (id:drken1215) 1年前