T214 - Re:Zero

Bryan Chung {kobebryan9}

2021-04-18

Background

Problem Idea: Bryan Chung

Preparation by Bryan Chung, Fuzen Ng and Percy Wong

Problem Restatement

There are N hidden triples representing N monsters

{starting position$_i$, X_i, T_i} for $0 \le i \le N - 1$

You may make queries by asking positions at day D, $0 \le D \le 10^{15}$

position$_i$ at day D = starting position$_i$ + $\lfloor D / T_i \rfloor * X_i$

N positions at day D will be returned

You need to find all $<X_i, T_i>$ pairs by asking as few queries as possible

Statistics

Task	Attempts	Max	Mean	Std Dev
T214 - Re:Zero	31	100	23.799	30.17

Subtasks
100: 1 93.1: 2 86.2: 1 72.4: 1 28.64: 2 28.155: 1 21.442: 2 9.148: 18

First solved by **dbsgame** at 4:47

香港電腦奧林匹克競賽
Hong Kong Olympiad in Informatics

CONSTRAINTS

- $1 \le N \le 1000$

For all $0 \le i \le N-1$:

- $-10^{18} \le$ monster i's starting position $\le 10^{18}$
- $-1000 \le X_i \le 1000$
- $X_i \ne 0$
- $1 \le T_i \le 100$

SCORING

Condition	Score
$130 < W$	0
$8 \le W \le 130$	$72 - 6.5 \cdot \sqrt{W - 7.5}$
$4 \le W \le 7$	$100 - 6.9 \cdot (W - 3)$
$W \le 3$	100

W = number of queries used

Solution for ~9.5 points

Just ask queries for 100/101 times

If position of a monster between two consecutive days is different,

X[i] and T[i] can be found directly

X[i] equals to the difference between the positions

T[i] equals to the first day that has different position than Day 0

How to avoid multiple crashing?

Let say if a monster is at position 0 at Day 0 and it is at position 8 at Day 8.

Is there sufficient information for us to determine what the hidden pair is?

How to avoid multiple crashing?

Let say if a monster is at position 0 at Day 0 and it is at position 8 at Day 8.

Is there sufficient information for us to determine what the hidden pair is?

Possibilities: <1, 1>, <2, 2>, <4, 3>, <4, 4>, <8, 5>, <8, 6>, <8, 7>, <8, 8>

Observation

Notice that T[i] is small ($1 \leq T[i] \leq 100$)

And D is extremely large ($0 \leq D \leq 10^{15}$)

Can we get more information with larger numbers?

Approach 1 - LCM

Recall Least Common Multiple (LCM) of a group of numbers is the minimum value such that every number in the group is its factor.

If monsters' T[i] are ranged from A to B, then at Day LCM(A to B), all of them will move at the start of the Day.

Where at Day LCM(A to B) - 1, All monsters' T[i] will not move at the start of the Day. (Except T[i] = 1)

Approach 1 - (Solution for ~44 points)

We can group every 8 consecutive numbers and query for the day of their LCM and their LCM -1.

As the positions of the monsters between two days must be different if its T[i] is one of the 8 consecutive numbers, we can calculate the X[i] and T[i] respectively by referring their positions at Day 0.

T[i] = LCM / ((position at Day LCM - position at Day 0) / X[i])

Approach 1 - (Solution for ~52 points)

Instead of grouping 8 consecutive numbers, we can group more consecutive numbers as long as their LCM is $\leq 10^{15}$ and query for the day of their LCM and their LCM -1.

As the positions of the monsters between two days must be different if its T[i] is one of the consecutive numbers, we can calculate the X[i] and T[i] respectively as well.

Approach 1 - (Solution for ~72 points)

Notice that grouping consecutive numbers is not efficient as they don't have much common factors.

Let P[i] be the primes from 2 to 97, P'[i] be max($P[i]^k$) where $P[i]^k \leq 100$ and k is a positive integer. (i.e. P'[i] = 64, 81, 25, ...)

Less queries could be achieved by splitting P'[i] into few groups, and then query their LCM and LCM - 1. (hint: similar to Sieve of Eratosthenes)

We can still calculate X[i] and T[i] in the same way.

Approach 2 - Unique Intervals

If we guarantee in every nubmer in a interval, all values will update at most once, then we can get the X[i] values.

Take interval 51 to 100 as an example.

Let's assume that we have already handled all $1 \leq T[i] \leq 50$.

For analyzing interval 51 - 100, we can query for day 50 and day 100, and the difference is the corresponding X[i].

Approach 2 - (Solution for ~28.6 points)

Ask queries for 53 times from D = 0 to 50, D = 100
and D = any large number Y (e.g. 1000000).

If position of a monster between two consecutive days is different,
X[i] and T[i] can be found directly if $1 \le T[i] \le 50$.

We can then analyze $51 \le T[i] \le 100$ using the method mentioned, by getting
X[i] = position on Day 100 - position on Day 0 (or 50)

Calculate the value of T[i] using the query of the large number Y.

T[i] = Y / ((position on Day Y - position on Day 0) / X[i])

Approach 2 - (Solution for ~62 points)

Ask queries for 10 times from D = 0, 1, 2, 4, 8, 16, 32, 64, 128

and D = any large number Y (e.g. 1000000).

If we process the range of two consecutive values from small to large, numbers in each interval will update at most once.

So we can get the X[i] and calculate the value of T[i] using the query of the large number Y.

T[i] = Y / ((position on Day Y - position on Day 0) / X[i])

Approach 2 - (Solution for ~64 points)

Ask queries for 9 times from D = 0, 1, 3, 7, 15, 31, 63, 127

and D = any large number Y (e.g. 1000000).

X[i] and T[i] can be calculated by the exact same way.

Merging both Approaches - (Solution for 93.1 points)

Note that LCM(1 to 33) ≤ 10^{15}.

We can query 4 times with D = 50, 100, LCM(1 to 33) and LCM(1 to 33) - 1.

For T[i] in range 1 to 33, It can be calculated by approach 1.

For T[i] in range 34 to 100, notice that each value would update at most once in between Day 50 and Day 100, it can be calculated by approach 2.

Full Solution - Mapping + Random Generator

Knowing the optimal W = 3 (0, A, B), we may find a pair of numbers <A, B> such that for each i:

Let y = A / T[i], z= B / T[i], <y / GCD(y, z) , z / GCD(y, z)> represents a unique T[i].

In other words, all <y / GCD(y, z) , z / GCD(y, z)> should be distinct in order to map into exactly one T[i] from 1 to 100.

Full Solution - Mapping + Random Generator

Let y = A / T[i], z = B / T[i], <y / GCD(y, z) , z / GCD(y, z)> represents a unique T[i].

Let E = position on Day y - position on Day 0,

F = position on Day z - position on Day 0,

We can map back <E / GCD(E, F) , F / GCD(E, F)> to find the corresponding T[i].

Full Solution - Mapping + Random Generator

If we choose A with some multiple of primes - 1

Eg : (81* 64 * 49 * 25 * 11 * 13 * 17 * 19 - 1)

We can cut down many possibilities as it is hard to occur collisions.

A suitable B can be easily found offline by random (under 0.1s run time)

Challenge: what if the scoring function also depends on D (smaller is better)?

Trainer’s best -> A and B are both smaller than 30000.

Other Solutions

The mapping technique mentioned above can generate by purely random + mapping to get 4 - 10 queries.

Gamegame fulled with a math construction for Pair A and B.