The House - Editorial

Task Author: Ahmet Kaan Avcı

Problem Recap

In this interactive problem, Arben is tasked with finding an empty $S \times S$ square subgrid (where $S \geq M$) on an $N \times N$ plot of land. The grid has a strict constraint: there is exactly one boulder in every row and every column. We are allowed to query rectangular regions to count the number of boulders inside them using the "Gurëmatës". To get a perfect score, we must find the required empty subgrid in $Q \leq 11$ queries.

Before diving into the solution, we must first answer a fundamental question: **What is the maximum guaranteed empty square size, M?**

Theorem 1. *For any $N \times N$ grid with exactly one boulder per row and column, there always exists an empty square subgrid of size $M \times M$, where $M = \lfloor \sqrt{N-1} \rfloor$.*

Proof Sketch (Upper Bound): Let $s = \lceil \sqrt{N} \rceil$. Consider a worst-case boulder placement where boulders are placed at (i, p_i) such that $p_i = \lfloor \frac{i-1}{s} \rfloor + ((i-1) \bmod s) \cdot s + 1$. If we analyze this grid, we can see that no empty square of size $s \times s$ can fit without hitting a boulder. Therefore, the guaranteed empty square size M must be strictly less than s, meaning $M \leq \lceil \sqrt{N} \rceil - 1$, which simplifies to $M = \lfloor \sqrt{N-1} \rfloor$.

Note: This establishes that we cannot guarantee a square larger than M. The constructive proof that an empty $M \times M$ square ***always*** *exists is provided by our optimal algorithm below.*

An incredibly useful consequence of this formula is that $M^2 \leq N-1$, which means $M^2+1 \leq N$. We will heavily abuse this property in our optimal solution!

Subtask 1 & 2: The Naive Approaches

Subtask 1 ($O(N^2)$ Queries): We can find the maximum possible empty square by querying every single 1×1 cell to reconstruct the grid layout. Once the grid is fully known, we can build a 2D prefix sum array in $O(N^2)$ time and easily binary search for the largest empty square. For $N \leq 40$, this takes up to 1600 queries, which earns a partial score.

Subtask 2 ($O(N \log N)$ Queries): Instead of checking every cell, we can find the exact location of the boulder in each row independently. For any row i, we can binary search the column by querying prefix rectangles `ask(i, 1, i, mid)`. Once all N boulders are found, we proceed with the offline 2D prefix sum solution. For $N = 500$, this takes around $500 \times \lceil \log_2 500 \rceil \approx 4500$ queries. This gives a much better multiplier, but is still far from the $Q \leq 11$ bound needed for a perfect score.

Subtask 3: $O(\log_2 M)$ Queries (The Optimal Solution)

Notice that an $M^2 \times M^2$ subgrid can be perfectly partitioned into exactly M^2 disjoint $M \times M$ blocks. Let's make a single query to the top-left $M^2 \times M^2$ region: `ask(1, 1,` M^2`,` M^2`)`.

There are two cases:

- **Case 1: The query returns $\le M^2 - 1$.**
  We have M^2 blocks but strictly fewer than M^2 boulders. By the Pigeonhole Principle, **at least one $M \times M$ block is completely empty**. Our target region is found!
- **Case 2: The query returns exactly M^2.**
  This is the beautiful part. If the top-left $M^2 \times M^2$ region has exactly M^2 boulders, it means *all* the boulders of the first M^2 rows are trapped in the first M^2 columns, and vice versa.
  Now, remember our inequality: $M^2 + 1 \le N$. What happens if we shift our region exactly 1 column to the right? Consider the region `Rows: 1 to` M^2`, Cols: 2 to` M^2+1.
  - We lose the boulder from column 1 (which was inside the first M^2 rows).
  - Do we gain a boulder from column M^2+1? No! Because all the boulders for the first M^2 rows were already accounted for in columns $1 \dots M^2$.

  Therefore, this shifted $M^2 \times M^2$ region contains **exactly $M^2 - 1$ boulders**. Since it can also be divided into M^2 blocks, by the Pigeonhole Principle, at least one $M \times M$ block in this shifted region is completely empty!

Extracting the Answer

Once we have our target $M^2 \times M^2$ region (either the original or the shifted one) containing $< M^2$ boulders, we just need to pinpoint the empty block using a 2D binary search over the $M \times M$ blocks:

1. **Binary Search the Row-Blocks:** There are M row-blocks. We query the upper half. If the number of boulders in it is strictly less than the number of $M \times M$ blocks it contains, an empty block must be there, so we recurse upwards. Otherwise, we recurse downwards. This takes $\lceil \log_2 M \rceil$ queries. 2. **Binary Search the Col-Blocks:** We now have a horizontal strip of size $M \times M^2$ with $< M$ boulders. We do the same binary search on the M column-blocks. This takes another $\lceil \log_2 M \rceil$ queries.

Total Query Complexity: $1 + 2\lceil \log_2 M \rceil$ queries.

For the maximum constraint $N = 500$, $M = \lfloor \sqrt{499} \rfloor = 22$.
$\lceil \log_2 22 \rceil = 5$.
Total queries in the worst case: $1 + 5 + 5 = \mathbf{11}$ queries.

This algorithmic construction proves that an $M \times M$ empty subgrid is always achievable!