Kick Start 2018 - Round E

Analysis: Board Game

Board Game: Analysis

Small dataset

Bahu needs to determine the best possible card distribution given that Bala will choose a distribution uniformly at random. In the Small dataset, **N** = 3, so there are only 9! / 3! / 3! / 3! = 1680 different ways for a player to distribute their cards. (If there are multiple cards with the same strength value, some of these distributions may be practically equivalent, but we will treat them as different for simplicity.) We can enumerate all 1680 possible distributions U_i for Bahu, and all 1680 possible distributions A_j for Bala. (These variables are named after the last letters of the players' names.) For each U_i, we find the fraction of all A_js that lose against that U_i. The largest such fraction is our answer. The time complexity is on the order of 1680^2 times a small constant factor.

Large dataset

In the Large dataset, it is possible that **N** = 5. In that case, there are 15! / 5! / 5! / 5! = 756756 different distributions. We cannot use the above strategy with a 756756^2 time factor.

Only the sums of the cards in the three battlefields matter; let them be U1, U2, U3 for Bahu and A1, A2, A3 for Bala. Then Bahu wins if at least two of the following inequalities are satisfied: U1 > A1, U2 > A2, U3 > A3.

We can deal with the "at least two" part of that criterion by using the [inclusion-exclusion principle](). Then, for each U_i:

The number of A_js satisfying the above criterion =
The number of A_js satisfying U1 > A1 and U2 > A2 +
The number of A_js satisfying U1 > A1 and U3 > A3 +
The number of A_js satisfying U2 > A2 and U3 > A3 -
2 × the number of A_js satisfying U1 > A1 and U2 > A2 and U3 > A3.

The last of those quantities is the most difficult one to calculate. Here we need another observation that there are only 15 choose 5 = 3003 different possibilities for all U1, U2, U3. We can label these possibilities 1 through 3003.

This is a 3-dimensional query, but we can remove 1 dimension by processing the Us in increasing order of U1 and also preprocessing As in increasing order of A1. After that, we can create a 2-dimensional segment tree to store all (A2, A3)s that have A1 < U1. We can find all (A2, A3)s such that U2 > A2 and U3 > A3 in log 3003 × log 3003 time complexity. The other 3 quantities in the equation above can also be found in a similar way. The overall time complexity is on the order of 756756 × log 3003 × log 3003 times a small constant factor, which is fast enough to solve this dataset.