Political Patricians

Félix Moreno Peñarrubia

Setup

There is an unknown partition of $[0, \ldots, 999]$.

We can ask, for any partition, which elements do not have another element that appears in the same set in both partitions.

Only two sets

Suppose we query a partition made of pairs of elements.

If the elements of a pair are alone, they are in separate sets. If not, they are in the same set. Either way, we will know their relationship.

We can then ask a similar query using one element per pair, and repeat. After $\log_2 n$ queries, this gives us full knowledge of the unknown partition.

Sets of pairs

Suppose we split the full set into two parts, and ask a query in which each of the two parts are together. If an element ends up being alone, then its only friend is in the other half, otherwise it is in the same half.

We can ask $\log_2 n$ queries, splitting odd vs even, $0, 1$ and $2, 3 \bmod 4$, etc. This will let us figure out exactly where the friend of each element is.

$\sim n$ queries

Ask all pairs. For each pair, we can easily tell whether they are together or apart in the hidden set.

If we ask $\frac{n}{2}$ pairs at once, we ask all pairs in $\sim n$ queries.

$0.5 \log^2 n$ queries (~80 points)

Keep a bunch of sets $S_1, \ldots S_k$, each containing elements that we know to be internally distinct.

Initially we have $S_i = \{i\}$. We can use the solution of the "sets of pairs" subtask to merge each pair of consecutive S_is, figuring out which elements belong to the same set in the hidden partition.

We then create new S_i sets keeping a single representative from each merged pair.

$0.25 \log^2 n$ queries (~98 points)

Since we know that S_i and S_{i+1} are not in the same set, we can do better:

1. First find which elements of $S_i \cup S_{i+1}$ appear twice
2. We now have two sets A and B that we know to be separate but to hit the same sets in the hidden partition. Compare the first halves A_0, B_0 of A and B, and the second halves A_1, B_1. We can make eight more sets $A'[0,1,2,3]$ and $B'[0,1,2,3]$ as follows:
   - $A'[0], B'[0]$: $\text{alone}(A_0), \text{alone}(B_1)$
   - $A'[1], B'[1]$: $\text{non-alone}(A_0), \text{non-alone}(B_0)$
   - $A'[2], B'[2]$: $\text{alone}(A_1), \text{alone}(B_0)$
   - $A'[3], B'[3]$: $\text{non-alone}(A_1), \text{non-alone}(B_1)$
3. Repeat step 2, splitting all set pairs in parallel

On average, the size of sets reduces by a factor of 4 at every step.

Full score

We try to merge as many queries as possible.

We first do all 1. queries from the $0.25 \log^2 n$ solution; then, we keep a stack of all pairs of sets from 2. that we still have to disambiguate and greedily add them to queries where their elements are not used yet.

This does fewer queries. In particular, for the "sets of pairs" subtask this is guaranteed to do at most $\log_2 n$ queries.