Code Jam 2011 - Qualification Round

Analysis: GoroSort

The Solution

This problem is very mathematical in nature, requiring a lot of thought and only a little code to solve correctly. We put it as a problem on the Qualifying Round to give you something different to try without having to worry about time pressure or advancement. We hope you enjoyed it!

For an arbitrary array A, let n(A) be the number of elements that are not already in the correct position. It turns out that the answer you are looking for is simply n(A). Once you realize this, it is easy to calculate with a simple loop, but how do you prove it?

The Proof

Let's first show that the expected number of hits is never more than n(A). Suppose Goro always holds down only those elements that are already in position, and then he randomly permutes the rest. Let x(A) be the expected number of hits required for him to sort A using this strategy.

Lemma: x(A) = n(A) for all A.

We prove this by induction on n(A). If n(A) = 0, then the array is already sorted, and we're done. To set up the induction, let's suppose we have proven the lemma already for smaller values of n(A) and we are now trying to prove it for A. Let p_t be the probability that exactly t elements are still out of position after the first hit, and let x'_t be the expected number of hits required in this case. We make three observations:

- $p_0 * 0 + p_1 * 1 + p_2 * 2 + ... + p_N * N = N - 1$. In English, this is saying that the *expected* number of elements that are out of position after the first hit is exactly N - 1, or equivalently, the expected number of elements that are put into position by the first hit is exactly 1. This follows from "linearity of expectation": Goro is permuting N elements; each one has probability exactly 1/N of ending up in the correct position, and hence, the expected number of elements that end up in the correct position is N * 1/N = 1.
- $x'_t = t$ for $t \leq N - 1$. This is true by the inductive hypothesis.
- $x'_N = x(A)$. If no elements are put into the correct position by the first hit, then we will just randomly permute them all again in the next step, so nothing has changed, and hence $x'_N = x(A)$.

Now let's write down a formula for x(A):

$1 + p_0 * x'_0 + p_1 * x'_1 + ... + p_N * x'_N$
$= 1 + p_0 * 0 + p_1 * 1 + ... + p_N * N + p_N * (x(A) - N)$
$= N + p_N * (x(A) - N)$,

which simplifies to $(N - x(A)) * (1 - p_N) = 0$. Since $p_N < 1$, we must have x(A) = N, and the lemma is proven.

To complete the proof, we need to calculate y(A), the expected number of hits required for Goro to sort A if he uses the (still unknown) optimal strategy. Since $y(A) \le x(A)$ by definition, we have already proven $y(A) \le n(A)$.

To conversely prove $n(A) \le y(A)$, it would be nice to just extend the proof of the previous lemma. There is one big technical issue though: it is possible for n(A) to go up if Goro doesn't hold down enough elements, and so it is tricky to set up an induction on n(A). We'll resolve this by having a separate proof for this part, and this time use a slightly different induction hypothesis. We can then follow the previous argument very closely.

Lemma 2: Let K be a non-negative integer. Then for any $k \le K$, the statement $y(A) = k$ is equivalent to the statement $n(A) = k$.

We will prove this by induction on K. Both $y(A) = 0$ and $n(A) = 0$ are equivalent to the array already being sorted, so the K = 0 case is clear. Now, let's suppose we have proven the lemma for K already, and are trying to prove it for K+1. Choose A such that y(A) is the smallest possible value larger than K and consider the optimal strategy for Goro. Let T be the number of elements that are either (a) not in the correct position in A, or (b) permuted when Goro hits the table. Define p_i and x'_i as we did before. Note that $T \ge n(A) \ge K+1$ by the inductive hypothesis.

As in the previous lemma, we can now prove the following:

- $p_0 * 0 + p_1 * 1 + p_2 * 2 + ... p_T * T \ge T - 1$.
- $x'_i = i$ for $i \le K$. This follows directly from the inductive hypothesis.
- $x'_i \ge y(A)$ for $i > K$. By the inductive hypothesis, $n(A') > K$ implies $y(A') > K$, which then implies $y(A') \ge y(A)$.

As before, we can now write y(A) as

$$1 + p_0 * x'_0 + p_1 * x'_1 + ... p_T * x'_T$$
$$\ge 1 + (p_0 * 0 + p_1 * 1 + ... + p_T * T) + (p_{K+1} + p_{K+2} + ... + p_T) * (y(A) - T)$$
$$\ge T + (p_{K+1} + p_{K+2} + ... + p_T) * (y(A) - T)$$

which simplifies to $(y(A) - T) * (1 - p_{K+1} - ... - p_T) \ge 0$. The second term has to be positive (if not, then $y(A) \ge \min(x'_{K+1}, x'_{K+2}, ...) + 1$, which contradicts the third bullet point above is therefore impossible), so we must have $y(A) \ge T \ge n(A) \ge K+1$. Equality holds only if $n(A) = K+1$. The first lemma guarantees $y(A) \le x(A) \le K+1$ in this case, and the proof is complete!

Comments

- If you go over the proof carefully, you can see there are two things Goro needs to do in order to be optimal. (1) He needs to always hold down elements that are already in the correct position, and (2) he needs to ensure that for each element x that is permuted, the element in x's correct position is also permuted. This means he actually has some choice about what to do.
- On a programming contest, of course you do not need to work through a formal proof to implement a correct solution. The best contestants can solve this kind of problem by looking at small examples to see a pattern, and then using intuitive reasoning to see what's going on without formalizing everything. Mastering this kind of reasoning is a difficult art though!