Code Jam 2022 - Round 3

Analysis: Revenge of GoroSort

In the original GoroSort problem from 2011, Goro could hold as many elements in place as he wanted; using the terminology of this problem, he could create arbitrary many box color groups with 1 ball (element) each. But unlike in this problem, he was only allowed to use at most one color group of size greater than 1. That made the optimal strategy relatively straightforward: he could repeatedly permute all list elements that were not in the correct places.

That strategy puts one additional element in the right place each turn, in expectation, which is clearly too slow for this problem. (Our version of that solution takes almost 100000 rounds!) What is more surprising is that the crux of the original problem — namely, thinking in terms of expected numbers of elements that become correctly placed after the bump — can even be *misleading* in this problem!

The perils of pairing

It is intutively clear that any elements that are already in the right place should be left alone (i.e., each should be put in its own color group). It also seems reasonable to not break up pairs of elements that are swapped (i.e., each is in the other's place). We can also reasonably guess that splitting such a pair would be useless, and that putting additional elements in that group would be worse than handling those other elements in their own group(s).

What about the other elements? One very tempting strategy is to try to pair them up into "trespasser pairs" such that one member of the group is in the other member's place. It may not be possible to do this with every element, but we can, e.g., find a way to tack leftovers onto existing trespasser pairs.

Consider a trespasser pair (x, y), where x belongs where y is and y belongs somewhere else entirely. The judge's "bumping" (permutation) process will put x in the right place with probability $\frac{1}{2}$. So if we somehow manage to create around $\frac{\mathbf{N}}{2}$ trespasser pairs, each bump will put around $\frac{\mathbf{N}}{4}$ more elements in the right places. This seems quite good, and it is good enough to pass Test Set 1, but not the other two. (Our trespasser-pair-based solutions take between 15000 and 16300 rounds, depending on the care taken with implementation details.)

Thinking in terms of cycles

Any permutation — and therefore any state in this problem — can be described as a multiset of cycles of particular lengths. For example, each element that is already in the correct place is a 1-cycle, and each pair of swapped elements is a 2-cycle.

If we try to implement the pairing idea above in a greedy way, we may miss some opportunities to form trespasser pairs. We can get as many as possible by first finding all the cycles, then going through each cycle and chopping it into trespasser pairs, plus perhaps one "trespasser trio" at the end. (A trespasser trio is a set of elements x, y, z, in some order, such that y is in x's place, z is in y's place, and z belongs outside of the group.) These improvements (especially creating trespasser trios where needed) help a lot, but not enough to pass Test Set 2; our trespasser pairs + one trio solution takes around 13100 rounds.

But what if we leave the cycles as they are, and make each cycle its own permutation group? Now we do much better, and pass Test Set 2 as well (but not Test 3; we take about 11800 rounds). Why is this cycle-based strategy so much better than the trespasser pair strategies?

When expectation doesn't satisfy our expectations

Here's the problem with chopping into trespasser pairs. Suppose that we have a 4-cycle like 2341. If we split it into two trespasser pairs 23 and 41, we will (in expectation) put one element in the correct place. But, as we will see later on, we get the same expectation of 1 if we designate the entire cycle as one group.

Does this necessarily mean these strategies are equally good? Let's set our expectations appropriately! We really care about the expected *total* number of rounds to finish sorting, so let's work toward calculating that instead. First we observe that:

- **If we split the 4-cycle into two trespasser pairs**:
  - With probability $\frac{1}{4}$, both elements end up in their correct places in the group, and we get two 1-cycles and one 2-cycle.
  - With probability $\frac{1}{4}$, neither element ends up in its correct place in the group, and we end up stuck at a 4-cycle.
  - Otherwise, with probability $\frac{1}{2}$, we end up with a 1-cycle and a 3-cycle.
- If we don't split the cycle before permuting:
  - With probability $\frac{1}{24}$, we get four 1-cycles.
  - With probability $\frac{1}{4}$, we get two 1-cycles and a 2-cycle.
  - With probability $\frac{1}{3}$, we get a 3-cycle and a 1-cycle.
  - With probability $\frac{1}{8}$, we get two 2-cycles.
  - With probability $\frac{1}{4}$, we are stuck at a 4-cycle.

Comparing these two probability distributions directly, and canceling out the parts that are the same, we need to know which is better:

- a $\frac{1}{6}$ probability of getting a 3-cycle and a 1-cycle, or
- a $\frac{1}{8}$ probability of getting two 2-cycles and a $\frac{1}{24}$ probability of getting four 1-cycles

Now we can assess each of those states in terms of the expected number of rounds to completion. A 3-cycle turns out to take 3 rounds, in expectation, to become all 1-cycles. (This comes from solving $\mathbf{E}[3] = 1 + \frac{1}{3} \cdot \mathbf{E}[3] + \frac{1}{2}\mathbf{E}[2] + \frac{1}{6}(0)$, and using the fact that $\mathbf{E}[2] = 2$.) By a similar analysis, two 2-cycles take $\frac{8}{3}$ rounds, in expectation, to become all 1-cycles. (And if we are lucky enough to reach four 1-cycles directly, 0 additional rounds are needed.)

Therefore, chopping up a 4-cycle into two trespasser pairs is strictly worse than leaving it as is! We wouldn't have known this if we had argued purely based on the expected number of correctly placed elements; it also matters what we leave behind. Intuitively, the trespassers can only be dealt with after their companions have been correctly placed, so the chopping strategy is less parallelizable. But we shouldn't assume this will always be true no matter how we chop...

Chopping is not always bad

It's tedious to perform the above analysis for cycles longer than 4. But we shouldn't give up hope. The expectations for the two strategies were the same for 4-cycles, but will that hold up in general?

Let's think about how many cycles we expect to see in a random permutation of length $\mathbf{N}$. You may have heard of the following problem: everyone loses their hats all at once, and each person puts on a random hat; in expectation, how many people get their own hats back? The probability that the each person gets their own hat is $\frac{1}{\mathbf{N}}$, and then by linearity of expectation, the total number of instances of someone getting their own hat is $\frac{1}{\mathbf{N}} \cdot \mathbf{N} = 1$.

So this gives us the expected number of 1-cycles. We can make a similar argument about 2-cycles: there are $\binom{\mathbf{N}}{2}$ distinct pairs of elements that could form a 2-cycle, and for each one, the probability that each has the other's element is $\frac{1}{\mathbf{N}} \cdot \frac{1}{\mathbf{N}-1}$. This all boils down to $\frac{1}{2}$. Indeed, the answer for general k-cycles is $\frac{1}{k}$.

Then how many total cycles should we expect? It's $\frac{1}{1} + \frac{1}{2} + ... + \frac{1}{\mathbf{N}}$. You may recognize this as the expression for the $\mathbf{N}$-th harmonic number. The harmonic numbers grow rather slowly; H_{100}, for example, is just over 5.

Therefore, if there are only around 5 cycles in our initial random permutation of length $\mathbf{N} = 100$, we expect to place around 5 elements. But if we chop the cycles into, say, 50 trespasser pairs, we should expect to place around 25 elements! How can it possibly be better to leave such large cycles alone? Does the argument about leaving more mess behind really still hold up?

One issue with chopping into pairs is that, intuitively, many roads to completion include forming one or more 4-cycles, and we have now seen that the pair-chopping strategy mishandles them. (And now we have reason to suspect that it mishandles larger cycles as well).

What if we chop a cycle into chunks larger than pairs? For example, if we chop a 6-cycle into two trespasser trios, each of them will produce $2 \cdot \frac{1}{3} = \frac{2}{3}$ correctly placed elements in expectation. That's $\frac{4}{3}$ overall, which is less than the $3 \cdot \frac{1}{2} = \frac{3}{2}$ we would have gotten from chopping into trespasser pairs. But these trespasser trios leave less of a mess; each one can only leave one element to clean up later, so there will be two such elements as compared to three.

It is hard to directly weigh the costs of cleaning up these leftover elements against the benefits of correctly placing more elements in expectation. The math for our 4-cycle example was already a bit cumbersome to carry out in a timed round, and there is not a strong incentive to do even more of it when the local testing tool can quickly tell us how well a strategy does, and when all three test sets are Visible. Indeed, we can find that leaving all cycles of length less than 6 intact, and breaking all cycles of length 6 or more into trespasser trios (or trespasser sets of four or five, when there are extra elements) does much better than leaving all cycles intact, and lets us pass Test Set 3, taking about 10950 rounds.

We can do even better by choosing different chunk lengths (e.g. break each cycle of length c into chunks of size roughly $\sqrt{c}$, to get down to around 10500 rounds), but that is not necessary for this problem.