APPENDIX TO PROBLEM 1

G RESEARCH

Abstract. A brief elaboration upon the submitted solution we published on the website, and some common mistakes made in other solutions.

1. On The Strategy Of Nim*

It should be noted that we got quite a number of solutions for this problem that were of the effect "keep the nim-sum 1 when the pass token is in the game" - more or less treating the pass token as a pile of size one. Unfortunately, this doesn't quite work. Let's sketch quite a number of the solutions we had:

- We can always go from something of Nim-sum bigger than 1 to something of nim-sum 1
- You can't go from nim-sum 1 to nim-sum 1 in one move
- Player 1's strategy is therefore:
  1) Force the piles to have nim-sum 1
  2a) Subsequently, if Player 2 changes the nim-sum to 0, take the pass token
  2b) Otherwise, force the piles to have nim-sum 1

Using normal nim-terminology, we define a P-state as a state when the previous player to play wins, and an N-state as when the next-player to play wins. The "proof" above is more or less saying that the P-states are when you have nim-sum 1 *and* the passing token, or nim-sum 0 and no passing token. However, examination of the states: {P, 1}, {P, 1, 2} and {P, 2, 3} show that this is clearly not correct. {P, 1} and {P, 2, 3} are N-states, and {P, 1, 2} is a P-state, and all of these examples contradict what the logic of the proof above would suggest!

So, what's going on? The issue lies with step 2a). In particular we assume that, if the opposing player makes the piles have nim-sum 0, we are able to make a valid move. But this is not always the case, as shown by the simplest state: {P, 1}. Here, the opponent makes the game have nim-sum 0, and we're unable to pass because the game is over.

The successful proofs that we got (congratulations!) all exploit the same strategy - force the game to be in a state so that, when the opponent makes it nim-sum 0, we must have tokens left on the board. The difference here is that we're not trying to entirely solve this game, but just show that we're in a class of this game that we can solve. Both proofs at the time of writing did this by forcing three groups of piles to have nim-sum 1 over the course of the game - which is a stronger condition that ensures that we will have tokens left in the other two groups when the nim-sum turns to 0 and we pass.

2. On Computing XORs

One step in the proof requires us to compute the XOR of 1, 3, 5, ..., 2019, which turns out to be 2. There's a number of ways of checking this - if one wanted to use a computer, we could write in Python:

```
reduce(lambda x, y: x^y, [2*i+1 for i in range(2020//2)])
```

Of course, this wouldn't be accepted in the IMO. Probably the simplest way to see what this is, is to notice that $(8k+1) \oplus (8k+3) \oplus (8k+5) \oplus (8k+7) = 0$ for any choice of integer k, as here only the last three bits matter (the higher ones get cancelled out). Thus we get that

$$1 \oplus 3 \oplus \cdots \oplus 2019 = 2017 \oplus 2019 = 1 \oplus 3 = 2$$