Code Jam 2018 - World Finals

Analysis: Swordmaster

The only way we can fail to become the Swordmaster is to get stuck in a situation where for all remaining duelists (duelists that we have not beaten yet) we will not defeat them without learning additional skills. At any point in time, the remaining duelists can be divided into four groups:

1. D1: Duelists we certainly can defeat.
2. D2: Duelists that have an attack we cannot defend against and a defense for every one of our attacks.
3. D3: Duelists that do not have a defense for one of our attacks, but have an attack we cannot defend against.
4. D4: Duelists that do not have an attack we cannot defend against, but have a defense for every one of our attacks.

If there is any remaining duelist in D1, we should just defeat them. If there is someone in D3 and there is someone in D4, then we can make some progress (i.e. reducing the number of remaining duelists, or moving a remaining duelist to D1) by doing the following:

1. We choose any duelist in D3 (call them A) and any duelist in D4 (call them B).
2. We ask A for a duel. Since A is in D3, A has an attack that we don't defend against. Either A will use an attack that we don't defend against (thus we will learn the attack), or we can defend and win the duel (thus learning all of A's attacks). At the end of the duel, we will learn an attack which we can't defend against. Let us call this attack is *a*.
3. If B cannot defend against *a*, then B has moved to D1 and we have made some progress. Otherwise, we ask B for a duel and use attack *a*. If B cannot defend, then we will win the duel and make some progress. Otherwise, we will learn defense *a*.
4. Now that we have learned defense *a*, A might have moved to D1, or might still be in D3 (by having another attack that we can't defend against). If A moved to D1, then we have made progress. Otherwise, we repeat step 2 until A has no attack that we can't defend against.

Therefore, the only possibilities in which we might fail to become the Swordmaster are when either every remaining duelist is in D2 or D3, or every remaining duelist is in D2 or D4.

If every remaining duelist is in D2 or D3, then every remaining duelist has an attack we can't defend. From this point, every remaining duelist can just attack us with an attack we can't defend against and choose to not defend against, which means that we will not learn any new defenses. Therefore, we can't defeat any more duelists and are certain to fail to become the Swordmaster. Notice that we can actually detect this situation up front — if there exists a group of duelists G_1 (not containing us), such that every duelist in G_1 has at least one attack against which nobody outside G_1 can defend, then we are doomed. The strategy for the duelists is that everybody in G_1 can just successfully attack us and choose to not defend (thus we are not learning any new defense known by duelists in G_1).

We can check the existence of G_1 by starting with G_1', the set of all duelists except us. We can iteratively remove from G_1' any duelist we can defend against regardless of their attack, and get all their defenses. This can be done in O(**N** × **P**) time. This complexity is also bounded by the sum of the number of attacks and defenses known to all duelists, which is linear in the size of the input.

Test set 1

For test set 1, the situation when every remaining duelist is in D2 or D4 is similar. This means every remaining duelist can defend against all of our attacks. Therefore, there exists a group of duelists G_2 (not containing us), such that every duelist in G_2 has a defense for every attack available outside G_2. Therefore, we are also doomed, since everybody in G_2 can just defend against our attack and always use attack 1 (thus we are not learning any new attack known by duelists in G_2). We can check the existence of such G_2 up front with an algorithm analogous to the one presented above to check for G_1.

Test set 2

For test set 2, the situation when every remaining duelist is in D2 or D4 is complex. The difference is that our opponents **have to** attack, so even when dueling with some duelist who can defend against all of our attacks, we can learn a new attack. This new attack potentially causes some duelist to move from D4 to D1, or from D2 to D3. Therefore, the fact that every remaining duelist is in D2 or D4 is not a sufficient condition for us to fail to become the Swordmaster.

Now imagine we are in a situation where we are actually doomed — we are in a position where, finally, every remaining duelist is in D2 or D4 and we will never defeat any duelist, nor will learn a new attack or defense. Such a situation has to be reached at some point since there is only a finite number of steps of learning (attack or defense) to be made. In this situation, we could fight a duel with every remaining duelist and learn the attacks they use. By definition, we already knew all these attacks, and so, also by definition, all our opponents can defend against all these attacks. So, for us to be doomed through this path, there must exist a group G_2 of duelists and an assignment of an attack for each of them $a : G_2 \rightarrow$ Attacks, such that every duelist d in G_2 knows attack a(d) and knows how to defend against every attack known outside G_2 and every attack a(d') for d' in G_2.

This condition is harder to check. Let us assume there exists such G_2. Consider any two duelists X and Y. If X has an attack that Y cannot defend against, then Y being in G_2 implies X must be in G_2 as well. This defines a directed graph on the duelists, with the edge going from Y to X.

We find the strongly connected components (SCCs) of this graph. If any duelist of a SCC is in G_2, then the whole SCC is in G_2. Therefore, G_2 will be an upper subset (a subset with no edges going out of the subset) of the directed acyclic graph (DAG) of the strongly connected components of the graph. Notice that if we can choose some upper subset of duelists to be a valid set G_2, then any subset of G_2 which is also an upper subset will also be a valid G_2 with the same attack selections. Therefore, it is enough to check only leaf SCCs in the DAG. If any valid G_2 exists, then a G_2 which is a leaf in the DAG also exists.

Therefore, for each leaf SCC in the DAG, we can check whether each duelist in the SCC can choose an attack that every duelist in the SCC can defend against. To do this, we can first find the set *D* which is an intersection of the defenses known by every duelist in the SCC, and see whether every duelist in the SCC has an attack in *D*.

To construct the graph in O(**N** × **P**) time, we can introduce nodes for attacks as well. We add an edge from a duelist to an attack if the duelist cannot defend against the attack, and an edge from an attack to a duelist if the duelist knows that attack. The overall solution runs in O(**N** × **P**) time.