1 Debate Evaluation Algorithms [40 points]

Based on Ahmed Abdelmeged's Dissertation.

Algorithm families play an important role in computer science. Consider the family P-ALGS of all polynomial-time algorithms. We would like to know whether P-ALGS contains an algorithm for the Satisfiability problem.

Here we deal with a more tractable family of algorithms. In this homework we study debate evaluation algorithms at a higher level. We consider all ranking algorithms that map a labeled graph representing a table about debate outcomes into a ranking. But we restrict ourselves to ranking algorithms which satisfy three properties (or axioms) that you are already familiar with:

- Non-Negative Effect for Winning (NNEW)
- Non-Positive Effect for Losing (NPEL)
- Collusion-Resilience (CR)

Can you say something interesting about any algorithm mapping labeled graphs to rankings and which satisfies the three properties: NNEW, NPEL and CR.

Let's restate NNEW and NPEL from homework 7. It is unacceptable for a ranking function to reward losing or to penalize winning. That is, a player's rank cannot be worsened by an extra win nor can it be improved by an extra loss.

1.1 NNEW

If $x \leq^T y$ for table T then $x \leq^{T'} y$ for table T' where T' contains an additional row where x won. We call this property Non-Negative Effect for Winning I (NNEW.I).

If $x \leq^T y$ for table T then $x \leq T''y$ for table T'' where T'' contains one fewer row where y won. We call this property Non-Negative Effect for Winning II (NNEW.II).

NNEW = NNEW.I and NNEW.II.

1.2 NPEL

If $x \leq^T y$ for table T then $x \leq T'y$ for table T' where T' contains an additional row where y lost. We call this property Non-Positive Effect for Losing (NPEL.I).

If $x \leq Ty$ for table T then $x \leq T''y$ for table T'' where T'' contains one fewer row where x lost. We call this property Non-Positive Effect for Losing (NPEL.II).

NPEL = NPEL.I and NPEL.II.

1.3 Background Recall

Here comes further background from hw6 and the midterm. You have been playing debates (semantic games) and you have recorded your debate results in tables such as the following one. You have column headers **PW**, **PL**, **Forced**. There is an additional column *Fault* which is computed from the first three. Each row of the table represents a debate g. **PW** and **PL** are the columns for the two participants. Because there is always a winner (there are no draws) we choose the **PW** column to show the winner. The **Forced** column shows which participant is forced. Remember that at most one is forced. We use entry `none` indicating that none of the two participants was forced. The *Fault* column depends on the columns **PL** and **Forced** and indicates whether the participant in column **PL** had a fault (lost in non-forced position).

We use the following Boolean functions for a debate g: g.participates(p) = p is a participant in debate g (either the winner or the loser). g.wins(p) = g.participates(p) and participant p wins in debate g. g.loses(p) = g.participates(p) and participant p loses in debate g. g.forced(p) = g.participates(p) and participant p is forced in debate g. g.fault(p) = g.participates(p) and participant p makes a fault in debate g. Note that !g.wins(p) = !g.participates(p) or g.loses(p).

Definition of fault: g.fault(p) = g.participates(p) and g.loses(p) and !g.forced(p).

Similar to the fault concept there is the control concept.

Definition of control: g.control(p) = g.participates(p) and (g.wins(p) or !g.forced(p)). Informally, a participant p is not in control in a debate if p is not involved in the debate or p loses while forced. Again informally we say that a participant p controls a debate if p either wins or p had a guaranteed opportunity to win (was non-forced). In a debate there is at least one of the two participants in control: the winner.

Consider algorithms that translate a debate table into a ranking of the participants. What is a ranking? A reflexive, transitive and complete binary relation.

Informally, our goal is to find the `best` participants based on the debates. We consider two ranking algorithms.

The first ranking algorithm, called $\leq_{WC}$, is based on win counting while the second, called $\leq_{FC}$ is based on a special kind of loss counting, called fault counting.

We fix a set of debates D. $x \leq^D y$ means that participant x is ranked weakly better than participant y for D. Sometimes we omit the superscript D when it is clear from context. Informally, weakly better means that either x is better then y or x and y have the same rank.

You might need some of the following in your answer: $wins(x)$ is the number of wins of participant x over all debates in D. Formally, $wins(x)$ is the number of rows $g \in D$ which satisfy $g.wins(x)$. $faults(x)$ is the number of faults that x makes over all debates in D. Formally, $faults(x)$ is the number of rows g in D which satisfy $g.fault(x)$.

In principle, all the functions defined above would require a superscript for D. For example, $wins^D(x)$ is the number of wins of participant x over all debates in D. However, we omit it in order not to clutter the definitions.

1.4 CR

Definition of collusion resilience: A ranking $\leq$ of participants is said to be collusion-resilient if for all debate tables D and for any two participants x and y, the property $x \leq^D y$ implies $x \leq^E y$, where E is D with added debates g that x cannot control, i.e., for which !g.control(x).

Informally, if $x \leq^D y$ and more debates that x cannot control are added to D resulting in E, then $x \leq^E y$ holds.

CR prevents debaters from gaming the debates by preventing another debater to be top ranked through intentional losses.

2 Example of Algorithms that satisfy NNEW and NPEL and CR

We show examples of ranking algorithms which have the desirable properties. The goal of the homework question is to figure out a property that is common to all of the algorithms.

2.1 Fault Counting

$\leq_{FC}$ is **defined** as: $x \leq_{FC} y$ if $faults(x) \leq faults(y)$.

NNEW and NPEL you have shown in hw 7. CR you have shown in the midterm.

2.2 Weighted Fault Counting

There are two kinds of faults that happen during a debate: (1) The debaters have chosen their prefered sides which are opposite to each other. (2) The debaters have chosen the same side.

If a fault happens with context (1), we count the fault with weight w_{opp}. With context (2) the weight is w_{same}.

$wfaults(x)$ is the number of weighted faults that x makes over all debates in D. Formally, $wfaults(x)$ is the number of weighted rows g in D which satisfy $g.fault(x)$.

$\leq_{WFC}$ is **defined** as: $x \leq_{WFC} y$ if $wfaults(x) \leq wfaults(y)$.

With the weights we can influence the competitiveness of the debaters. When w_{opp} is considerably less than w_{same} the debaters will avoid losing against a forced debater. On the other hand, the forced debater will be strongly motivated to try to find weaknesses in the opponent.

2.3 All Equal

$\leq_=$ is **defined** as: $x \leq_= y$ if true.

All debaters are ranked the same. (Everybody gets an A independent of performance.)

3 Make a claim and defend it

3.1 3 properties

Make a claim about the family of ranking algorithms that have all three properties NNEW, NPEL and CR.

3.2 NNEW and CR (2 properties)

Make a claim about the family of ranking algorithms that have two properties NNEW and CR.

Justify your claims by providing a proof. But the focus is on expressing the correct claims.

3.3 Hint

A ranking algorithm takes as input a table and maps it to a ranking relation for the debaters. We consider filters on the table which keep the ranking relation obtained invariant. The filter eliminates some rows from the table which we cannot use because of the CR and NNEW properties that we want to hold.

For any pair (x,y) of debaters, the relative rank that $\leq$ assigns to x with respect to y ($x \leq^D y$) for a table D satisfies: $x \leq^D y \iff x \leq^{f(D,x,y)} y$ where $f(D, x, y)$ is suitably defined.

f expresses which games are important to compute the ranking if we want CR and NNEW to hold.

Can you propose a suitable f?