Roberto Perissa 859143   15/04/2020

Security 2 – ARBAC lab challenge report

Implementation:

I tried two different approaches to solve this challenge. The second one proved to be successful.

First idea – All possible permutations:

The first approach I took was to compute all possible permutations of the assigning rules; try to assign the target role considering a permutation a possible path. This approach wasn't correct, one over all because it has a fixed path length. It also had terrible performances.

Second idea – Try to assign:

The second approach is to search the possibility to assign the target role in a smart way. Starting with the target role, find all the assigning rules that has that role as target. Find all the users that can apply those rules and begin the search. For every rule compute the preconditions and try to make any user satisfy them.

With this logic, 6 out of 8 problems were solved quickly, but the remaining problems had a critical point not yet considered. They couldn't be solved since it was impossible to find the possibility to assign the role: a precondition role needed an assigning role that no user had. To solve this, the assigning roles had to be checked before searching the assignment of precondition roles, so this step has been added to the logic.

The approach can be summed up as:

1. try to assign, if possible, return success;
2. search for what you need to assign, do sub changing if possible;
3. try to assign, if possible, return success;
4. redo from point 1 until nothing can be done;
5. check if assign is possible, if possible, return success, failure otherwise.

This logic proved to be good enough to solve correctly all the challenge problems, but I want to underline that it is not sure that every problem can be solved: more complex problems could need considering the removing rules in the searching paths. With this logic addition then every problem would be successfully solved.

Note: no time-out was needed in this solution as suggested in the task specifications.

Performance discussion:

By a performance viewpoint, the forward or backward optimizations make the difference in time of execution not appreciable on human scale. But, if this is true for the problems specific of this challenge, it's possible that these optimizations can give a critical difference in general/generic problems.

Mean time standard solver: 1 seconds

Mean time optimizations solvers: 0.7 seconds

It's a difference of 0.3, so a 30% in time efficiency. On larger problems 30% can drastically change situations and with high probability this percentage can only grow in larger and larger problems as the slicing "cuts" more and more possibilities.

My conclusion is that the optimizations should be always used.