Algorithmic Game Theory and Applications

Coursework 2: Solution of Question 2

Question 2:

Consider the following restricted kind of Muller game on a graph. The winner is determined as follows: we are given a set $F \subseteq V$. Every infinite play π where $\mathit{inf}(\pi) \cap F \neq \emptyset$ is winning for player 1. All other plays are losing for player 1. Describe an efficient (polynomial time, and as efficient as you can get it) algorithm for determining the winner and computing a memoryless winning strategy in such a game.

Note: This winning condition above is called a Büchi-condition.

Solution:

Given a node x, let $Pl_1(x)$ be true iff Player 1 controls x. Similarly, $Pl_2(x)$ is true iff Player 2 controls x. Let $Post(x)$ be the set of immediate one-step successors of node x.

Given a set of states S, let $Force_1(S)$ be the set of states from which Player 1 can force the game into S. (I.e., this is the solution of a basic reachability game.) Note that Player 1 might not be able to force the game into any *particular* state in S, but only into some state in S (i.e., Player 2 cannot avoid visiting S, but he may get to choose which state in S is visited).

Now we compute Player 1's winning set Win_1 in the game above. Just forcing the game into F is not enough; we need to visit F infinitely often.

One first needs to compute the unique largest subset $F' \subseteq F$ s.t. $F' \subseteq Win_1$. In other words, $F' = F \cap Win_1$. Thus we get the following condition:

$$\begin{aligned} \forall x \in F'. \quad & (Pl_1(x) \wedge Post(x) \cap Force_1(F') \neq \emptyset) \vee \\ & (Pl_2(x) \wedge \emptyset \neq Post(x) \subseteq Force_1(F')) \end{aligned}$$

In particular, the largest such set F' is unique, since it is the largest fixpoint of a suitable monotone decreasing function f on a complete lattice (the powerset of the set of states V).

Let $f(S) := \{x \in S \cap F \mid (Pl_1(x) \wedge Post(x) \cap Force_1(S) \neq \emptyset) \;\vee\; (Pl_2(x) \wedge \emptyset \neq Post(x) \subseteq Force_1(S))\}$.

F' can be computed by starting with $F' = F$ applying function f (i.e., removing states that do not satisfy the condition) repeatedly until a fixpoint is reached. Note that several rounds of such refinement may be needed, since the condition itself depends on the current set.

At the end one obtains the largest fixpoint F' of f, i.e., $F' = f(F')$ and F' is the largest set satisfying this condition. Trivially, $F' \subseteq F$. Moreover, $F' \subseteq \mathit{Win_1}$, and thus $F' \subseteq F \cap Win_1$. For the reverse inclusion observe that $f(F \cap Win_1) = F \cap Win_1$, i.e., $F \cap Win_1$ is a fixpoint of f. Since F' is the *largest* fixpoint of f, we obtain $F \cap Win_1 \subseteq F'$. To altogether we have $F' = F \cap Win_1$ as required.

Finally, we get $Win_1 = Force_1(F')$.

Note that even in systems where Player 1 can win from some state, he might not be able to enforce any particular loop or the recurrence of any particular state in F. Consider the system:

$$\begin{array}{lcl} X & --> & Y \\ X & --> & Z \\ Y & --> & X \\ Z & --> & X \end{array}$$

$F = \{Y, Z\}$. Player 1 owns Y, Z and Player 2 owns X. In this case $F' = \{Y, Z\}$ and $Win_1 = \{X, Y, Z\}$. However, Player 1 can neither force the infinite recurrence of Y, nor the infinite recurrence of Z. It is for Player 2 to decide whether X or Y (or both) appear infinitely often. I.e., Player 2 loses, but has some influence about how he loses.