Solutions Exam MPRI

2020

Let $\mathcal{G}$ be a (two-player deterministic finite) reachability game. The underlying graph is $G = (V, E)$; we make the assumption that every vertex has at least one outgoing edge. We write $V = V_{\text{Eve}} \uplus V_{\text{Adam}}$ for the set of vertices controlled by Eve and Adam. The reachability objective is $\text{Reach}(F) = V^* F V^\omega$, *i.e.* the set of paths visiting $F \subseteq V$ at least once.

The goal of this problem is to construct efficient algorithms for computing $W_{\text{Eve}}(\text{Reach}(F))$, the set of vertices from which Eve has a winning strategy for the reachability objective. The important parameters here are n the number of vertices and m the number of edges.

For representation purposes, the game is given in the following way: for each vertex v, one bit describes whether it is controlled by Eve or Adam, and then we list all the successors of v.

The objective F is given as a boolean vector over V. To compute $W_{\text{Eve}}(\text{Reach}(F))$ we represent it as well using a boolean vector over V.

Question 1: We write $\mathcal{P}(V)$ for the set of subsets of V. Let us consider the operator $\mathbf{Pre}_F : \mathcal{P}(V) \to \mathcal{P}(V)$ defined by

$$\mathbf{Pre}_F(X) = F \cup \left\{v \in V_{\text{Eve}} : \exists (v, v') \in E, v' \in X\right\} \cup \left\{v \in V_{\text{Adam}} : \forall (v, v') \in E, v' \in X\right\}.$$

Prove that $\mathbf{Pre}_F$ is a monotone operator with respect to inclusion: if $X \subseteq X'$ then $\mathbf{Pre}_F(X) \subseteq \mathbf{Pre}_F(X')$.

Solution: “Oh come on!” is an acceptable answer.

Question 2: A prefixed point of $\mathbf{Pre}_F$ is a set $X \subseteq V$ such that $\mathbf{Pre}_F(X) \subseteq X$. Prove that:

(i) There exists a prefixed point of $\mathbf{Pre}_F$.

(ii) $W_{\text{Eve}}(\text{Reach}(F))$ is a prefixed point of $\mathbf{Pre}_F$.

(iii) The intersection of two prefixed points of $\mathbf{Pre}_F$ is another prefixed point of $\mathbf{Pre}_F$

(iv) There exists a least prefixed point of $\mathbf{Pre}_F$.

(v) $W_{\text{Eve}}(\text{Reach}(F))$ is the least prefixed point of $\mathbf{Pre}_F$.

Solution:

(i) V is a fixed point of $\mathbf{Pre}_F$.

(ii) Easy.

(iii) Easy.

(iv) Consequence of the previous point.

(v) It is a fixed point. To prove the converse, consider the complement (done in the lecture).

Question 3: Construct an algorithm for computing $W_{\text{Eve}}(\text{Reach}(F))$ based on Knaster - Tarski fixed point theorem, and show that it has complexity $O(n \cdot m)$.

Solution: Knaster - Tarski fixed point theorem gives an algorithm for computing the least prefixed point: it says that the sequence $(\mathbf{Pre}_F^k(\emptyset))_{k \geqslant 0}$ is eventually constant and its limit is the least prefixed point of $\mathbf{Pre}_F$. For each k we construct $\mathbf{Pre}_F^k(\emptyset)$ in a naive way, which has complexity $O(m)$. Since there are n iterations we get complexity $O(n \cdot m)$.

We want to improve the complexity to $O(n + m)$. Note that since every vertex has at least one outgoing edge, $n \leqslant m$, so this is actually $O(m)$.

Question 4: Prove that in time $O(m)$ we can get the following equivalent representation of the game: for each vertex v, one bit describes whether it is controlled by Eve or Adam, and then we list all the **predecessors** of v.

Solution: We go through all edges, and add them to the corresponding list.

Question 5: Prove that the algorithm given in pseudo-code below computes $W_{\text{Eve}}(\text{Reach}(F))$ and that its complexity is $O(m)$.

Algorithm 1: The linear time algorithm for reachability games.

```
Data: A reachability game.
Function Attractor():
    A ← F
    for v ∈ V_Adam do
        number-edges(v) ← number of outgoing edges of v
    k ← 1
    X_k ← F

    repeat
        for v ∈ X_k do
            Treat (v)
        k ← k + 1
    until X_k = X_{k+1}
    return A

Function Treat(v):
    for e = (u, v) ∈ E do
        if u ∈ V_Adam and u ∉ A then
            number-edges(u) ← number-edges(u) − 1
            if number-edges(u) = 0 then
                Add u to A
                Add u to X_{k+1}
        if u ∈ V_Eve and u ∉ A then
            Add u to A
            Add u to X_{k+1}
```

Solution: To prove correctness we need a suitable invariant: $X_k = \mathbf{Pre}_F^k(\emptyset))$.

A vertex can be added to A at most once, implying that an edge can be considered at most once, so the complexity is $O(m)$.