Algorithmic Aspects of Game Theory

Tomasz Garbus

2019

Contents

1 Homework1

1.2 One-player game

Let $\mathcal{G} = \langle V, E\rangle$ be a legal graph. A move in a one-player game in $\mathcal{G}$ is a reversal of a single edge, such that the resulting graph is legal.
Consider the following problem: given a legal graph $\mathcal{G}$ and an edge $e \in E$, does there exists a sequence of moves that reverse edge e?
We proved that if we restrict the above problem to the case where every edge can be reversed at most once, then the problem is NP-complete (reduction from 3CNF)
Show that the above problem is PSpace-complete.
1. $\in$ PSpace
2. PSpace-hard
I will show a reduction from QBF in to one-player flow game. I will assume two things about the formula:

- It is in prenex normal form (i.e. all quantifiers preceed the portion containing an unquantified Boolean formula). Moreover, let's assume that the existental and universal quantifiers alternate – if it is not the case in the original input formula, we can introduce quantifiers with dummy variables, not used anywhere in the formula. For instance, $\exists_{x_1}\exists_{x_2}\phi(x_1, x_2) \mapsto \exists_{x_1}\forall_{y_1}\exists_{x_2}\phi(x_1, x_2)$ (y_1 is a "dummy" variable).
- The "body" of the formula is in conjunctive normal form.

5 Multi-reachability games

In a reachability games there is a set of vertices which the first player wants to reach. In multi-reachability games (MRG) there is a family of sets of vertices and the first player wins if every set in the family has been visited at least once.
1. Show that MRG game is PSpace-complete.
1. $\in$ PSpace
I will use the fact that a normal reachability game is solvable in polynomial time. I will also use Savitch theorem, in particular its corollary, stating that PSpace = NPSpace.
Let $G = \langle V, E, v_I, S\rangle$ be the MRG, where V is the set of vertices, E is the set of edges, v_I is the starting vertex and S is the family of sets the first player wants to visit.
The first player can nondeterministically choose an ordered sequence of vertices $v_1, v_2, ..., v_k$ such that $\forall_{T\in S}\exists_{1\leqslant i\leqslant k}(v_i \in T)$. Then we want to check in polynomial space whether the first player can visit selected vertices in the specified order, regardless of what the second player is doing. This can be done by considering around $4k$ normal reachability games. For each $i, 1 \leqslant i \leqslant k$, we create 4 reachability games:

- $\langle V', E', (v_{i-1}, 0), \{(v_i, 0)\}\rangle$
- $\langle V', E', (v_{i-1}, 0), \{(v_i, 1)\}\rangle$
- $\langle V', E', (v_{i-1}, 1), \{(v_i, 0)\}\rangle$
- $\langle V', E', (v_{i-1}, 1), \{(v_i, 1)\}\rangle$

where:
$V' = V \times \{0, 1\}$
$E' = \{((u, 0), (v, 1)), ((u, 1), (v, 0)) \mid (u, v) \in E\}$
$v_0 = v_I$
Player p starts, where $(v_{i-1}, p) = v_I$ (not necessarily the first player from our MRG)

Such extended set of vertices V' carries, together with the vertex, the information whose move it is. Having solved the single reachability games described above, we can perform an algorithm in a dynamic-programming-manner as follows:

```
wins: bool[k][2][2] = given      # wins[i][ps][pe] iff the first
                                 # player has a winning strategy in
                                 # a game <V', E', (v_{i-1}, ps), (v_i, pe)>

can_reach[][] = false * [k][2]   # At the end of the algorithm
                                 # can_reach[i][p] = true iff
                                 # first player can force the game
                                 # to reach vertex v_i and player
                                 # number p has the next move.

can_reach[0][0] = true           # First player starts.

for i := 1 to k:
  for ps := 0 to 1:
    for pe := 0 to 1:
      if can_reach[i-1][ps] and wins[i][ps][pe]:
        can_reach[i][pe] = true
```

If either `can_reach[k][0]` or `can_reach[k][1]` then the first player has a winning strategy.

2. PSPACE-hard
I will show a reduction of QBF problem to an MRG game.
An input to the QBF problem is a formula, about which I will make two assumptions:

- It is in prenex normal form (i.e. all quantifiers preceed the portion containing an unquantified Boolean formula). Moreover, let's assume that the existental and universal quantifiers alternate – if it is not the case in the original input formula, we can introduce quantifiers with dummy variables, not used anywhere in the formula. For instance, $\exists_{x_1}\exists_{x_2}\phi(x_1, x_2) \mapsto \exists_{x_1}\forall_{y_1}\exists_{x_2}\phi(x_1, x_2)$ (y_1 is a "dummy" variable).
- The "body" of the formula is in conjunctive normal form.

Note that the above assumptions do not reduce the expressive power of input formulas. Every possible formula can be represented in the described format. QBF problem for such normalized formulas is still PSPACE-complete.
Let $\forall_{x_1}\exists_{x_2}\forall_{x_3}...\exists_{x_n}(y_{1,1}\vee...\vee y_{1,k_1})\wedge(y_{2,1}\vee...\vee y_{2,k_2})\wedge...(y_{m,1}\vee...\vee y_{m,k_m})$ be the input QBF formula, where $y_{...} \in \{x_1, ..., x_n, \neg x_1, ...\neg x_n\}$.
The created multi-reachability game is $G = \langle V, E, v_I, S\rangle$, where:

- V is the set of vertices. Vertices are indexed by all variables bound by quantifiers and their negations, plus there is the initial vertex. $V = \bigcup_{1\leqslant i\leqslant n} \{v_{x_i}, v_{\neg x_i}\} \cup \{v_I\}$. Note that the vertices are positions and:
  $Pos = V$
  $v_x \in Pos_\exists$ iff x_i is bound by an existential quantifier
  $v_x \in Pos_\forall$ iff x_i is bound by an universal quantifier
- E is the set of edges. $E = \{(v_I, v_{x_1}), (v_I, v_{\neg x_1})\} \cup \bigcup_{2\leqslant i\leqslant n} \{(v_{x_{i-1}}, v_{x_i}), (v_{x_{i-1}}, v_{\neg x_i}), (v_{\neg x_{i-1}}, v_{x_i}), (v_{\neg x_{i-1}}, v_{\neg x_i})\}$
- v_I is a starting vertex.
- S is the family of sets of vertices that the first player wants to reach. It is created directly from the CNF formula, i.e. $S = \bigcup_{1\leqslant i\leqslant m} \{\{v_{y_{i,j}} \mid 1 \leqslant j \leqslant k_i\}\}$

The first player is the existential player, their opponent is the universal player (and they start the game). The QBF formula is satisfiable iff the first player has a winning strategy.

2. Show that one-player MRG is NP-complete.
1. $\in$ NP

The algorithm is as follows:

1 We make a nondeterministic selection of one element of every set of vertices to be visited and we also choose nondeterministically a permutation of those verices – the order in which we want to visit them. If some vertex was chosen more than once, we only leave one copy of it in the chosen order.

2 It is possible to check in polynomial time whether the chosen vertices can be visited in the chosen order. Let $v_1, v_2, ..., v_k$ be the chosen sequence of vertices. For any pair of vertices u, w, it can be decided in polynomial time whether w is reachable from u (e.g. a simple depth-first search algorithm). Let $v_0 = v_I$, the starting vertex of the game. The player has a winning strategy if $\forall_{1 \leqslant i \leqslant k}(v_i$ is reachable from $v_{i-1})$.

2. <u>NP-hard</u>
I will show the reduction of 3SAT problem (satisfiability of CNF formula where each clause has at most 3 literals) to one-player MRG. The resulting game will be very similar as in previous subproblem.
Let $(y_{1,1} \vee y_{1,2} \vee y_{1,3}) \wedge ... \wedge (y_{m,1} \vee y_{m,2} \vee y_{m,3})$ be the input formula, where $\underset{\substack{1 \leqslant i \leqslant m \\ 1 \leqslant j \leqslant 3}}{\forall} y_{i,j} \in \bigcup_{1 \leqslant k \leqslant n} \{x_k, \neg x_k\}$.
We create a one-player MRG as follows:
$G = \langle V, E, v_I, S \rangle$, where:

- $V = \bigcup_{1 \leqslant i \leqslant n} \{v_x, v_{\neg x}\} \cup \{v_I\}$ is the set of vertices. Each vertex (except for the initial one) corresponds to some variable or its negation. Visiting a vertex is synonymous to assigning a value to the variable.
- E is the set of edges. $E = \{(v_I, v_{x_1}), (v_I, v_{\neg x_1})\} \cup \bigcup_{2 \leqslant i \leqslant n} \{(v_{x_{i-1}}, v_{x_i}), (v_{x_{i-1}}, v_{\neg x_i}), (v_{\neg x_{i-1}}, v_{x_i}), (v_{\neg x_{i-1}}, v_{\neg x_i})\}$.
- v_I is the initial vertex.
- S is the family of sets of vertices that the player wants to reach. It is constructed directly from the CNF formula, i.e. $S = \bigcup_{1 \leqslant i \leqslant m} \{\{v_{y_{i,j}} \mid 1 \leqslant j \leqslant 3\}\}$

The formula is satisfiable iff the player can find a path in the graph, starting from v_I and visiting at least one vertex from each set in S. Moreover, the indices of visited vertices form a satisfying valuation of the variables occuring in the input formula.

3. Show that if the sets are singletons then MRG is P-complete.

1. <u>$\in$ P</u>
I will again use the fact that a single reachability game is decidable in polynomial time. Let $G = \langle V, E, v_I, S = \{\{v_1\}, \{v_2\}, ..., \{v_k\}\} \rangle$.
Just like in the first subproblem, let's create some single reachability games:
For each $i, 1 \leqslant i \leqslant k$, we create 4 reachability games:

- $\langle V', E', (v_{i-1}, 0), \{(v_i, 0)\} \rangle$
- $\langle V', E', (v_{i-1}, 0), \{(v_i, 1)\} \rangle$
- $\langle V', E', (v_{i-1}, 1), \{(v_i, 0)\} \rangle$
- $\langle V', E', (v_{i-1}, 1), \{(v_i, 1)\} \rangle$

where:
$V' = V \times \{0, 1\}$
$E' = \{((u, 0), (v, 1)), ((u, 1), (v, 0)) \mid (u, v) \in E\}$
$v_0 = v_I$
Player p starts, where $(v_{i-1}, p) = v_I$ (not necessarily the first player from our MRG)

If (and only if) the first player wins a game $\langle V', E', (u, p), \{(v, q)\} \rangle$, it means that, starting from node u in the original game, if it's turn of player p, the first player has a strategy for the game to always reach the node v in a way that player q moves next.
Having solved the above games, we can create a relation R such that $((u, p), (v, q)) \in R$ iff first player has a winning strategy in game $\langle V', E', (u, p), \{(v, q)\} \rangle$. Note that this relation is transitive. We turn this relation into a graph, such that the nodes are pairs (v_i, p) $(0 \leqslant k, p \in \{0, 1\})$ and the directed edge between two nodes iff they are in relation R. Then we turn this graph into a DAG of strongly connected components. Since the relation is transitive, every strongly connected component will also be a clique, so finding a Hamiltonian cycle is trivial inside each component. There exists a Hamiltonian path in the graph if and only if the DAG of strongly connected components is a single path. The first player has the winning strategy in the original MRG if and only if there exists a

Hamiltonian path in the constructed DAG. Every step of this method is polynomial in time.

2. P-hard
I will show a reduction of circuit value problem (CVP) to an MRG game where the sets are singletons. The input to the problem is a boolean circuit and an input to the circuit. The first layer consists of inputs and their negations (every boolean circuit can be transformed to a form where the only negations are the once in the input layer). Every node in further layers is either a conjunction or disjunction of its inputs.

The general idea is that we will treat the circuit as a graph and reverse the edges. In $\vee$ nodes the first player moves, in $\wedge$ nodes, the second player moves. The first player wins iff they have a strategy for reaching any input set to `true`, which can be expressed by creating a new vertex w such that all inputs set to true have an outgoing edge to w. The first player therefore wants to reach w.

In order to transform this problem to an MRG game with singleton sets, we first treat the given circuit as a graph $G_{cir} = \langle V_{cir}, E_{cir}, r\rangle$, where V_{cir} is the set of vertices, E_{cir} the set of edges and $r : V_{cir} \to \{\wedge, \vee\} \cup \{in, \neg in\} \times \mathbb{N}$ is a function describing a vertex's role. $\wedge$ and $\vee$ of course mean that the vertex is a conjunction/disjunction of its inputs, (in, k) means that it is the k-th input value, $(\neg in, k)$ means that it is the negation of k-th input value. Let s be the input sequence and $n = |s|$.

The game we construct is $\mathcal{G} = \langle Pos = Pos_\exists \cup Pos_\forall, Moves, win\rangle$, where:

- win is the set of singletons the first player wants to reach. $win = \{\{w\}\}$, where w is a new vertice such that $\forall_{p \in Pos}((r(p) = (in, i) \wedge s[i] = true) \vee (r(p) = (\neg in, i) \wedge s[i] = false) \to (p, w) \in Moves)$.
  In other words, w can be reached from any input vertex from the original circuit, which evaluates to `true`.
- $Pos = V_{cir} \cup \{w\}$ is the set of positions.
  Vertices with $\wedge$ symbol belong to the opponent, all others belong to the first player:
  $Pos_\forall = \{p \mid r(p) = \wedge\}, Pos_\exists = \{p \mid p \notin Pos_\forall\}$
- $Moves$ is the set of edges, i.e. moves on the graph.
  $Moves = \{(v, u) \mid (u, v) \in E_{cir}\} \cup \{(p, w) \mid (r(p) = (in, i) \wedge s[i] = true) \vee (r(p) = (\neg in, i) \wedge s[i] = false)\}$
- The starting position of the game is the top vertex of the circuit (the one to be evaluated in circuit value problem).

The first player can reach w in this game if and only if the original circuits evaluates to 1. For each $\vee$ node, the first player can choose which compound of the alternative they want to be true. For each $\wedge$ node, we let the second player choose any compound of the conjunction. Once the game reaches a node corresponding to an input in the original circuit (which of course happens in finite time since this is a DAG), the first player wins if and only if this input evaluates to `true`.