Complexity Theory – Homework 5

Discussed on 26.05.2010.

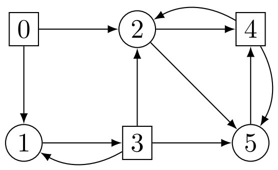
Exercise 5.1

A *two-person game* consists of a directed graph $G = (V_0, V_1, E)$ (called the *game graph*) whose nodes $V := V_0 \cup V_1$ are partitioned into two sets and a *winning condition*. We assume that every node $v \in V$ has a successor. The two players are called for simplicity player 0 and player 1. A *play* of the two is any finite or infinite path $v_1v_2\ldots$ in G where v_1 is the starting node. If the play is currently in node v_i and $v_i \in V_0$, then we assume that it is the turn of player 0 to choose v_{i+1} from the successors of v_i; if $v_i \in V_1$, player 1 determines the next move. The winning condition defines when a play is won by player 0. E.g.:

- In a *reachability game* the winning condition is simply defined by a subset $T \subseteq V_0 \cup V_1$ (*targets*) of the nodes of G, and a play is won by player 0 if it visits T within $n-1$ moves (where n is the total number of nodes of G). Hence, player 1 wins a play if he can avoid visiting T for at least $n-1$ moves.
- In a *revisiting game* player 0 wins a play $v_1v_2\ldots$ if the first node v_i which is visited a second time belongs to player 0, i.e., $v_i \in V_0$; otherwise player 1 wins the play.

We say that *player i wins node s* if he can choose his moves in such a way that he wins any play starting in s.

Example: Consider the following game graph where nodes of V_0 (V_1) are of circular (rectangular) shape:



In the reachability game with $T = \{5\}$ player 0 can win node 4: if player 1 moves from 4 to 5, player 0 immediately wins; if player 1 moves from 4 to 2, then player 0 can win again by moving from 2 to 5. On the other hand, player 1 can win node 0 by choosing to always play from 0 to 1 and from 3 to 1.

In the revisiting game played on the same game graph, player 0 can win node 2: he moves from 2 to 5 and then on to 4; no matter how player 1 then chooses to move, the play will end in an already visited node which belongs to player 0. Player 1 can e.g. win node 3 by simply moving to node 1.

(a) Consider a reachability game:

Show that one can decide in time polynomial in $\langle G, s, T\rangle$ if player 0 can win node s.

Hint: Starting in T compute the set of nodes from which player 0 can always reach T no matter how player 1 chooses his moves.

(b) Consider a revisiting game:

Show that it is **PSPACE**-complete to decide for a given game graph G and node s if player 0 can win s.

Remarks:

- A game is called *determined* if every node if won by one of the two players.

  Are reachability, resp. revisiting games determined?

- Assume that we change the definition of reachability game by dropping the restriction on the number of moves, i.e., player 0 wins a play if the play eventually reaches a state in T.

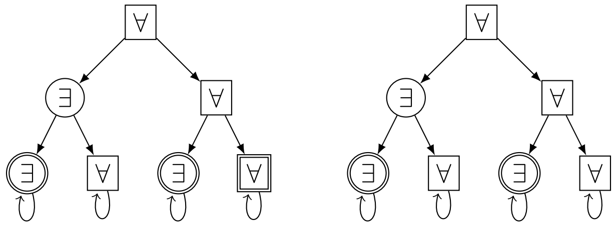
  Does this change the nodes player 0 can win for a given game graph?

Exercise 5.2

An *alternating Turing machine* (ATM) $M = (\Gamma, Q_\forall, Q_\exists, \delta_0, \delta_1)$ is an NDTM $(\Gamma, Q_\forall \cup Q_\exists, \delta_0, \delta_1)$ except that (i) the control states are partitioned into sets $Q_\forall$ and $Q_\exists$ and (ii) the acceptance condition is defined as follows:

> Consider the configuration graph $G(M, x)$. We extend the partition of the control states to the configurations (nodes) of $G_{M;x}$: a configuration is in V_0 if its control state is in $Q_\exists$; otherwise it is in V_1. We then can consider the reachability game played on $G(M, x)$ by the players 0 and 1 where the target set is the set of accepting configurations. M accepts x iff player 0 wins the initial configuration in this reachability game. (For the sake of completeness, assume that every halting/accepting configuration is its unique successor.)

Example: Consider the following configuration graphs where accepting configurations have a second circle/rectangle drawn around them. In the left graph the corresponding ATM accepts the input while it rejects the input in the right example:



A language is decided by an ATM M if M accepts every $x \in L$ and rejects any $x \notin L$. The time and space required by an ATM is the time and space required by the underlying NDTM.

The class **AP** consists of all languages L which are decided by an ATM M running in time $T(n) \in \mathcal{O}(n^k)$ for some $k \geq 1$.

(a) An *existential* (*universal*) ATM is an ATM with $Q_\forall = \emptyset$ ($Q_\exists = \emptyset$).

Show that any language $L \in \mathbf{AP}$ which is decided by an existential (universal) ATM is in **NP** (co**NP**).

(b) Define co**AP** as usual: $L \in \text{co}\mathbf{AP}$ iff $\overline{L} \in \mathbf{AP}$.

Show or disprove that $\mathbf{AP} = \text{co}\mathbf{AP}$.

(c) Show that QBF is in **AP**.

(d) Show that any $L \in \mathbf{AP}$ is in **PSPACE**.

Remark: Adapt the recursive decision procedure for QBF $\in$ **PSPACE** you have seen in the lecture.

Exercise 5.3

(a) Show that for any $L \in \mathbf{PSPACE}$ there is single-tape TM M (which may also write on its input tape) which decides L also in polynomial space.

(b) Show that it is **PSPACE**-complete to decide if a given word w can be derived by a given context-sensitive grammar G, i.e.,

$$\textsc{ConSens} := \{\langle G, w\rangle \mid \text{if } G \text{ is a context-sensitive grammar and } w \in L(G)\}.$$