Computational Complexity **Autumn, 2018**

Solution : Homework 2

Lecturer: Yuxi Fu *Homework taker: Xu Li 018033210002*

Problem 2.3.

Solution: Since each ration number can be represented by 2 integer. The matrix A can be represented using $O(mn)$ bits. Given a pair $< A, b >$, we can calculate all x satisfying $Ax = b$ in $O(n^3)$ time. And it takes $O(n)$ time to verify x, i.e., it takes $< A, b >$ can be verified in polynomial time and therefore LINEQ is in NP. ∎

Problem 2.6.

Solution:

(a) We construct the NDTM NU as follows. Similar as Universal Turing Machine, given a NDTM M_α, we use one work tape to simulate all work tapes of M_α. Since UTM can simulate a TM in $CT \log T$ steps, NU can also simulate a NDTM in $CT \log T$ steps.

(b) We construct the NDTM NU as follows. It guesses a sequence of snapshots and a sequence of choices by running the input machine on the input value without looking at the worktapes. It then verifies for each worktape of the input machine if the snapshots are legal. I To follow the content change of the tape being verified, V needs an additional tape. The length of a snapshot is a constant only relies on M_α. The verification takes the former snapshot for input and can be down also in constant time. Therefore NU runs for at most CT steps.

∎

Problem 2.13.

Solution:

(a) Modify the machine M so that it clears up its work tape before outputting a 1 and moves both heads to one. Then the final snapshot and head locations are unique. The proof of Cook-Levin Reduction gives a one-to-one and onto mapping between the set of certificates and the set of satisfying assignments. So the number of satisfying truth assignment equals to that of M accepting computation paths.

(b) To reduct SAT to 3SAT, we transform the CNF in SAT to clauses in 3SAT form as follows. We assume the CNF is $E = e_1 \vee e_2 \vee ... \vee e_k$ where each e_i is a disjunction of literals.

- If e_i is a single literal, say x, we introduce two new variables u and v. We replace x by the conjunction of four clauses as $(x \vee u \vee v) \wedge (x \vee u \vee \overline{v}) \wedge (x \vee \overline{u} \vee v) \wedge (x \vee \overline{u} \vee \overline{v})$. Since u and v appear in all combinations, the only way to satisfy all four clauses is to make x true.
- Suppose an e_i is the disjunction of two literals, $(x \vee y)$. We introduce a new variable z and replace e_i by the conjunction of two clauses $(x \vee y \vee \overline{z}) \wedge (x \vee y \vee z)$. As in case 1, the only way to satisfy both clauses is to satisfy $(x \vee y)$.
- If an e_i is the disjunction of three literals it is already in the form required for 3-CNF, so we take e_i.

- Suppose $e_i = (x_1 \vee x_2 \vee ... \vee x_m)$ for some $m \geq 4$. We introduce new variables $y_1, y_2, ..., y_{m3}$ and replace e_i by the conjunction of clauses

$$(x_1 \vee x_2 \vee y_1) \wedge (x_3 \vee \overline{y_1} \vee y_2) \wedge (x_4 \vee \overline{y_2} \vee y_3)...(x_{m2} \vee \overline{y_{m-4}} \vee y_{m-3}) \wedge (x_{m-1} \vee x_m \vee y_{m-3}). \quad (1.1)$$

■

Problem 4.10.

Solution: With perfect information of the game, we can construct the configuration graph. We can walk through the graph to find if these is a series of moves for player A that satisfies no matter what move that player B make, there is still a path leading to 'Player A wins'. If such move for player A exists, there is a winning strategy for player A (choose). Otherwise, there is a series of moves for player B satisfying no matter what moves took by player A, player B can also find a way to win. ■

Problem 4.11.

Solution: Let $L \in coNSPACE(s(n))$. Then there is a non-deterministic machine M using space $s(n)$ and with the following property: if $x \in L$ then $M(x)$ accepts on every computation path, while if $x \notin L$ then there is some computation path on which $M(x)$ rejects. Considering the configuration graph G_M of $M(x)$ for some input x of length n, we see that $x \in L$ iff there is no directed path in G_M from the starting configuration to the rejecting configuration. Since G_M has $V = 2^{O(s(n))}$ vertices, and the existence of an edge between two vertices i and j can be determined in $O(s(n)) = O(logV)$ space, we can decide L in space $O(logV) = O(s(n))$. ■