CSCI 3130: Formal Languages and Automata Theory | Homework 6
The Chinese University of Hong Kong, Fall 2021 | due 11:59pm Thursday December 9

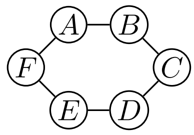
---

Collaborating on homework is encouraged, but you must write your own solutions in your own words and list your collaborators. Copying someone else's solution will be considered plagiarism and may result in failing the whole course.

Please answer clearly and concisely. Explain your answers. Unexplained answers will get lower scores or even no credits.

(1) (40 points) For each of the following problems, show that it is NP-complete: Namely, (1) it is in NP; and (2) some NP-hard language reduces to it. When showing NP-hardness, you can start from any language that was shown NP-hard in class or tutorial.

(a) $L_1 = \{\langle G, k\rangle \mid \text{Graph } G \text{ contains an induced matching of } k \text{ edges}\}$.
An edge subset $M = \{e_1, \ldots, e_k\}$ in $G$ is an induced matching if $u \in e_i$ and $v \in e_j$ for $i \neq j$ implies $u$ and $v$ are non-adjacent (i.e. vertices from different edges in $M$ are non-adjacent). In the graph on the right, $\{(A, B), (D, E)\}$ is an induced matching but $\{(A, B), (C, D)\}$ is not.




**Hint:** Reduce from INDEPENDENT SET. Attach new edges to vertices. Be specific about how to do so.

(b) $L_2 = \{\langle\varphi\rangle \mid \varphi \text{ is a satisfiable 3CNF formula where each literal appears in at most 3 places}\}$.

*Hint:* Reduce from 3SAT. Make several copies of each variable in the 3SAT formula, and write clauses that require all copies of the same variable to be equal.

(2) (20 points) For the following language IM, suppose some polynomial-time algorithm $A$ decides the corresponding *decision* problem. Using $A$, give a polynomial-time algorithm to *search* for a solution, whenever such a solution exists.

Briefly justify your algorithm, e.g. by giving an invariant. Insufficient explanation will get zero points.

$$\text{IM} = \{\langle G, k\rangle \mid \text{Graph } G \text{ contains an induced matching of } k \text{ edges}\}\ .$$

Induced matching is defined in Q1(a).

(3) (20 points) Throughout the semester, we looked at various models of computation and we came up with the following "hierarchy" of classes of languages:

$$\text{regular} \subseteq \text{context-free} \subseteq \text{P} \subseteq \text{NP} \qquad \text{decidable} \subseteq \text{recognizable}$$

We also gave examples showing that the containments are strict (e.g., a context-free language that is not regular), except for the containment P $\subseteq$ NP, which is not known to be strict.

There is one gap in this picture between NP languages and decidable languages. In this problem you will fill this gap.

(a) Show that 3SAT is decidable, and the decider has running time $2^{O(n)}$. (Unlike a *verifier* for 3SAT which is given a 3CNF formula $\varphi$ together with a potential satisfying assignment for $\varphi$, a *decider* for 3SAT is only given a 3CNF formula but not an assignment for it.)

(b) Argue that for every NP-language $L$ there is a constant $c$ such that $L$ is decidable in time $2^{O(n^c)}$. (Use the Cook–Levin Theorem.)

(c) Let $L'$ be the following language:

$$L' = \{\langle M, w\rangle \mid \text{Turing machine } M \text{ does not accept input } \langle M, w\rangle \text{ within } 2^{2^{|w|}} \text{ steps}\}.$$

It is not hard to see that $L'$ can be decided in time $O(2^{2^n})$.

Show that $L'$ cannot be decided in time $2^{O(n^c)}$ for any constant $c$, and therefore it is not in NP.

**Hint:** Assume that $L'$ can be decided by a Turing machine $D$ in time $2^{O(n^c)}$. What happens when $D$ is given input $\langle D, w\rangle$, where $w$ is a sufficiently long string?

(4) (20 points) A *heuristic* is an algorithm that often works well in practice, but it may not always produce the correct answer. In this problem, we will consider a heuristic for Clique.

Recall that the *degree* of a vertex is the number of edges incident to it. In the following, we assume the vertices in the input graph $G$ are labelled from 1 to $n$. Consider the following heuristic $A$ for Clique:

**Algorithm 1** GreedyClique$(G, k)$

**Require:** $G$ is a graph, $k$ is a nonnegative integer

Let $v$ be the vertex of maximum degree
    ▷ if there are more than one choice for $v$, break ties by choosing the smallest label
Set $S = \{v\}$
Let $N = \{u \in V \setminus S \mid (u, v') \in E \text{ for all } v' \in S\}$ be the set of vertices outside $S$
    that are adjacent to all vertices in $S$
**while** $N$ is not empty **do**
    Let $v$ be the vertex in $N$ of maximum degree
        ▷ breaking ties by choosing the smallest label
    Update $S$ as $S \cup \{v\}$
    Update $N$ as $\{u \in V \setminus S \mid (u, v') \in E \text{ for all } v' \in S\}$ the set of vertices outside $S$
        that are adjacent to all vertices in $S$
**end while**
**accept** if and only if $|S| \geqslant k$

(a) Show that $A$ runs in polynomial time.

(b) Show, using a loop invariant, that $S$ is guaranteed to be a clique of $G$ at the end of the while loop.
(Note: As a result, if $A$ accepts $\langle G, k\rangle$, then $\langle G, k\rangle \in$ Clique)

(c) Show that it is possible that $A$ rejects $\langle G, k\rangle$, even though $\langle G, k\rangle \in$ Clique. Give such an instance $\langle G, k\rangle$ where the graph $G$ contains at most 5 vertices.

Explain why your instance belongs to Clique (by giving a clique of size $k$ in $G$), and why heuristic $A$ rejects $\langle G, k\rangle$.