CSCE 411-502 Design and Analysis of Algorithms

Spring 2025

Instructor: Dr. Jianer Chen
Office: PETR 428
Phone: (979) 845-4259
Email: `chen@cse.tamu.edu`
Office Hours: MW 1:30 pm–3:00 pm

Senior Grader: William Kang
Phone: (979) 575-9987
Email: `rkdvlfah1018@tamu.edu`
Questions: via phone and email and by appointments

Assignment #7
(Due April 28)

1. Let Q_1, Q_2, and Q_3 be decision problems. Prove: if $Q_1 \leq_m^p Q_2$ and $Q_2 \leq_m^p Q_3$, then $Q_1 \leq_m^p Q_3$. You should explain why your reduction from Q_1 to Q_3 runs in time polynomial of the length of the instances of Q_1.

2. Prove: if an $\mathcal{NP}$-hard prolem is solvable in polynomial time, then $\mathcal{P} = \mathcal{NP}$.

3. Using the fact that the INDEPENDENT SET problem is $\mathcal{NP}$-complete, prove that the following problem is $\mathcal{NP}$-complete:

> CLIQUE: Given a graph G and an integer k, is there a set C of k vertices in G such that for every pair v and w of vertices in C, v and w are adjacent in G?

Hints for Question 3:

1. To prove that a problem is $\mathcal{NP}$-complete, you need to prove (1) the problem is in $\mathcal{NP}$, and (2) the problem is $\mathcal{NP}$-hard.
2. To reduce INDEPENDENT SET to CLIQUE, consider the "completment graph" co-G of a graph G, where co-G and G have the same set of vertices, and there is an edge between vertices v and w in co-G if and only if there is no edge between vertices v and w in G.

Definitions.

1. $\mathcal{P}$ is the collection of all (decision) problems that can be solved in polynomial time. Thus, $\mathcal{P}$ is the collection of all "easy" problems.

2. $\mathcal{NP}$ is the collection of all (decision) problems whose solutions, though perhaps not easy to construct, but can be checked in polynomial time. $\mathcal{NP}$ contains many problems that are not known to be in $\mathcal{P}$. Examples include TRAVELING SALESMAN, SATISFIABILITY, and INDEPENDENT SET. Huge amount of efforts has been paid trying to develop polynomial-time algorithms for these problems, but all failed. A common belief is that these problems are hard and do not belong to $\mathcal{P}$, i.e., $\mathcal{P} \neq \mathcal{NP}$.

3. $Q_1 \leq_m^p Q_2$ means that up to polynomial-time computation, Q_1 is not harder than Q_2.

4. $\mathcal{NP}$-hard problems are those that are not easier than any problems in $\mathcal{NP}$ (up to polynomial-time computation). Based on the common belief given in 2, an $\mathcal{NP}$-hard problem cannot be solved in polynomial time.

Some things you may want to remember.

1. To show $Q_1 \leq_m^p Q_2$, you need to construct a polynomial-time algorithm that computes a function f such that x is a yes-instance of Q_1 if and only if $f(x)$ is a yes-instance of Q_2.

2. To prove that a problem Q is in $\mathcal{NP}$, you need to construct a polynomial-time algorithm $A(x, y)$ such that for any yes-instance x_1 of Q, there is a y_1 such that $A(x_1, y_1) = 1$, and for any no-instance x_2 of Q, $A(x_2, y) = 0$ for all y.

3. To prove that a problem Q is $\mathcal{NP}$-hard, you need to pick a problem Q_0 that is known to be $\mathcal{NP}$-hard, and show $Q_0 \leq_m^p Q$.

4. To prove that a problem Q is $\mathcal{NP}$-complete, you need to prove both that Q is $\mathcal{NP}$-hard and that Q is in $\mathcal{NP}$.

5. You should remember of the definitions of at least the following $\mathcal{NP}$-complete problems: SATISFIABILITY, INDEPENDENT SET, VERTEX COVER, and PARTITION.