CS 535: Analysis of Algorithms

NP-completeness

(based in part on the CS 530 textbook, written by Sipser)

A **Hamilton path** in a directed graph G is a directed path that goes through each node exactly once. We consider the problem of testing whether a directed graph contains a Hamiltonian path connecting two specified nodes s and t, and call it $HAMPATH$. No one knows whether $HAMPATH$ is solvable in polynomial time.

Definition 1 *P is the class of decision problems that have polynomial time algorithms. NP is the class of decision problems that have polynomial time "verifiers".*

The *verifier* is a unrealistically powerful "nondeterministic" computer, provably no worse than supercomputers or quantum computers. The term NP comes from **nondeterministic polynomial time**.

NOTE: the degree of the polynomial does not matter (as long as it a constant)!

If G is an undirected graph, a **vertex cover** of G is a subset of the nodes where every edge of G touches one of those nodes. The vertex cover problem asks whether a graph contains a vertex cover of a specified (given, as part of the input) size.

Theorem 2 *One can find a hamiltonian path in polynomial time with access to a polynomial-time algorithm for the decision problem $HAMPATH$.*

One can find a minimum-size vertex cover in polynomial-time if one can decide $VERTEX-COVER$.

In the $SUBSET-SUM$ decision problem we have a collection of numbers, $x_1, ..., x_k$ and a target number t. We want to determine whether the collection contains a subcollection $\{y_1, ..., y_l\}$ that adds up to t. Notice that $\{y_1, ..., y_l\}$ and $\{x_1, ..., x_k\}$ are considered to be **multisets** and so allow repetition of elements.

A **clique** in an undirected graph is a subgraph, wherein every two nodes are connected by an edge. A **k-clique** is a clique that contains k nodes. The CLIQUE decision problem has as input a graph $G = (V, E)$ and an integer k, and the answer is YES if G has a k-clique (and NO, otherwise).

An **independent set** in in an undirected graph is a subgraph, wherein no two nodes are connected by an edge. The $INDEPENDENT - SET$ decision problem problem is to determine whether a graph contains an independent set of a specified size.

Theorem 3 *$HAMPATH$, $VERTEX-COVER$, $SUBSET-SUM$, $CLIQUE$, and $INDEPENDENT - SET$ are in NP. (See CS 530 for the proofs)*

One important advance on the P versus NP question came in the early 1970s with the work of Stephen Cook and Leonid Levin. They discovered certain problems in NP whose individual complexity is related to that of the entire class. If a polynomial time algorithm exists for any of these problems, all problems in NP would be polynomial time solvable. These problem are called **NP-complete**.

The first NP-complete problem that we present is called the **satisfiability problem**. Recall that variables that can take on the values TRUE and FALSE are called **Boolean variables**. Usually, we represent TRUE by 1 and FALSE by 0. The Boolean operations AND, OR, and NOT, represented by the symbols $\wedge$, $\vee$, $\neg$ and , respectively. We use the overbar as a shorthand for the $\neg$ symbol, so $\bar{x}$ means $\neg x$. A **Boolean formula** is an expression involving Boolean variables and operations. For example, $\phi = (\bar{x} \vee y) \wedge (x \vee \bar{z})$ is a Boolean formula. A boolean formula is **satisfiable** is some assignment of 0s and 1s to the variables makes the formula evaluate to 1. The preceding formula is satisfiable because the assignment $x = 0$, $y = 1$, and $z = 0$ makes ϕ evaluate to 1. We say the assignment *satisfies* ϕ. The **satisfiability problem** is to test whether a Boolean formula is satisfiable.

Theorem 4 ***Cook-Levin theorem*** *SAT $\in$ P iff P=NP. (See CS 530 for a proof)*

Definition 5 *Decision problem A is* ***polynomial time mapping reducible***[1]*, or simply* ***polynomial time reducible****, to decision problem B, written $A \leq_P B$, if a polynomial time algorithm M exists that takes as input an instance w of A and produces as output an instance $M(w)$ of B such that the A-answer for w is YES if and only if the B-answer for $M(w)$ is YES. The algorithm M is called the* ***polynomial time reduction*** *of A to B.*

Theorem 6 *For any two decision problems A, B, if $A \leq_P B$ and $B \in P$, then $A \in P$.*

Theorem 7 *For any three decision problems A, B, C, if $A \leq_P B$ and $B \leq_P C$, then $A \leq_P C$.*

A **literal** is a Boolean variable or a negated Boolean variable, as in x and $\bar{x}$. A **clause** is several literals connected with $\vee$s. A Boolean formula is in **conjunctive normal form**, called a **cnf-formula**, if it comprises several clauses connected with $\wedge$s. It is a **3cnf-formula** if all the clauses have three literals, as in

$$(x_1 \vee \overline{x_2} \vee \overline{x_3}) \wedge (x_3 \vee \overline{x_2} \vee \overline{x_6}) \wedge (x_3 \vee \overline{x_7} \vee x_1) \wedge (x_4 \vee x_5 \vee \overline{x_6}).$$

[1] Is is called **polynomial time many-one reducibility** in some other textbooks.

Let $CNF-SAT$ be the decision problem which takes as input a cnf-formula ϕ and asks for figuring out if the formula is satisfiable, which here means that each clause must contain at least one literal that is assigned 1. Let $3SAT$ be the variant of $CNF-SAT$ which takes as input 3cnf-formulas.

Theorem 8 *$3SAT$ is polynomial time reducible $CLIQUE$.*

Definition 9 *A decision problem B is* ***NP-complete*** *if it satisfies two conditions:*

1. *B is in NP, and*
2. *every A in NP is polynomial time reducible to B.*

Theorem 10 *If B is NP-complete and $B \in P$, then P=NP.*

Theorem 11 *If B is NP-complete and $B \leq_P C$ for C in NP, then C is NP-complete.*

Theorem 12 *$CNF-SAT$ is NP-complete. $3SAT$ is NP-complete.*

This theorem is Theorem 7.27, the Cook-Levin theorem, in a stronger form.

Corollary 13 *$CLIQUE$ is NP-complete.*

Theorem 14 INDEPENDENT-SET *is NP-complete.*

Theorem 15 VERTEX-COVER *is NP-complete.*

Theorem 16 *$HAMPATH$ is NP-complete.*

Theorem 17 *$SUBSET-SUM$ is NP-complete.*