COMP 409, Lecture 8

February 11, 2003

1 NP Class

Definition 1 *Given a finite alphabet* Σ*, a* Decision Problem *is some language* $L \subseteq \Sigma^*$*. We assume that* $\emptyset \subset L \subset \Sigma^*$*.*

Definition 2 *A problem* $L \subseteq \Sigma^*$ *is in NP if there is a polynomial* P *and a polynomial time algorithm* A *such that* $x \in L$ *iff there exists* $y \in \Sigma^{\leq P(|x|)}$ *where* $A(x, y) = 1$*.*

A problem which is in PTIME is also in NP. NP lies somewhere between Polynomial and exponential time. So a problem L is in NP if we can somehow guess a solution and its really easy to check if it satisfies the problem.

1.1 Some Examples

1. **Diophantine equation** : *Does* $P(x_1, x_2) = 0$ *have integer solutions?* The solution is undecidable because we cannot bound the size of x_1 and x_2.. Hilbert's 10th problem asked if a technique for solving a general Diophantine existed. The impossibility of obtaining a general solution was proven by Matyasevich in the early 1970's. However, verifying the correctness of a particular solution (x_1', x_2') takes only polynomial time. This problem is not in NP, because short witnesses may not exist.

2. **Compositeness Problem** : *Given a number* $n \in N$*, is* n *composite ?* Given the factors n_1 and n_2, it is easy to verify if $n = n_1 * n_2$. However, the time taken to find suitable n_1 and n_2 is exponential of the length of the binary representation of n. This problem is in NP. It is in fact in PTIME.

3. **Eulerian Cycle** on graph $G(V, E)$: *Is there a bijective mapping* $P : E \to \{1, \ldots, |E|\}$ *such that if* $P(e_2) = (P(e_1) + 1) \bmod |E|$ *then there exist* $u, v, w \in V$ *such that* $e_1 = (u, v)$ *and* $e_2 = (v, w)$*?* An Eulerian Cycle on a graph G is a cycle that visits each edge exactly

once. An Eulerian Cycle exists iff the degree of each node in the graph is even. Thus, this problem is not only in NP, it is in fact in PTIME.

4. **Hamiltonian Cycle** on graph $G(V, E)$: *Is there a bijective mapping* $P : V \to \{1, \ldots, |v|\}$ *such that if* $P(v) = (P(u)+1) \bmod |V|$, *then* $(u, v) \in E$*?* A Hamiltonian cycle on a graph G is a cycle that visits each node exactly once and terminates at the initial node. This problem is in NP.

5. **K-Coloring Problem** : Consider a graph representation of a map. $G = (V, E)$, whose nodes are the partitions in the map and neighboring partitions (nodes) are connected by edges. A *k-coloring* is a mapping $C : V \mapsto \{1..k\}$, obeying the constraint $C(u) = C(v) \to (u, v) \notin E$. We say G is k-colorable if it has a k-coloring. *Graph k-colorability* is the problem if checking whether a given graph is k-colorable. Graph k-colorability is also in NP. The witness is the k-coloring C.

So there are problems that look very similar but one is in P and other in NP. A classical example is 2-coloring and 3-coloring of a graph. Finding if a graph can be colored by 2 colors is easy but doing the same with 3 colors is hard.

2 Complexity claases

We looked at complexity classes for deterministic algorithms in the previous lecture: *(Deterministic Hierarchy)*

$$LOGSPACE \subseteq PTIME \subseteq PSPACE \subseteq EXPTIME. \subseteq EXPSPACE$$

By introducing *nondeterminism*, which is the ability to *guess a solution*, we get nondeterministic algorithms and nondeterministic complexity classes *(Complexity Hierarchy)*:

$$NLOGSPACE \subseteq NPTIME \subseteq NPSPACE \subseteq NEXPTIME \subseteq NEXPSPACE$$

By combining the above two hierarchies, we get the following full hierarchy of complexity classes:

$$LOGSPACE \subseteq NLOGSPACE \subseteq PTIME \subseteq NPTIME \subseteq PSPACE$$

$$\subseteq NPSPACE \subseteq EXPTIME \subseteq NEXPTIME \subseteq EXPSPACE \subseteq NEXPSPACE$$

It is known that exponential gaps lead to strict containment:

$LOGSPACE \subset PSPACE \subset EXPSPACE.$
$NLOGSPACE \subset NPSPACE \subset NEXPSPACE.$
$PTIME \subset EXPTIME.$
$NPTIME \subset NEXPTIME.$

Theorem 1 ***Savitch's theorm:****It states that* $PSPACE = NPSPACE$ *and* $EXPSPACE = NEXPSPACE$

3 Reducibility

Definition 3 *Let $L_1, L_2 \subseteq \Sigma^*$. We say that L_1 is* polynomially reducible *to L_2, denoted $L_1 \leq_p L_2$ if there is a PTIME-computable function $f : \Sigma^* \to \Sigma^*$, such that $x \in L_1$ iff $f(x) \in L_2$.*

For example: Let L_+ be a language such that $L_+(a, b, c)$ is true iff $a + b = c$. Similarly for L_-, $L_-(a, b, c)$ is true only iff $a - b = c$. so now to reduce L_+ to L_- we have $f(a, b, c) = b, c, a$, where f is the function that reduces an instance of L_+ to L_-.So when we say $L_1 \leq_p L_2$. It basically means L_2 is at least as hard as L_1.

Lemma 1 *$\leq_p$ is invariant of PTIME, that is, if L_2 $\in$PTIME and $L_1 \leq_p L_2$, then L_1 $\in$PTIME.*

Lemma 2 *$\leq_p$ is reflexive and transitive so it defines a pre-order.*

Definition 4 *A Decision problem $L \subseteq \Sigma^*$ is NP-*hard *if for every $L' \in NP$, we have $L' \leq_p L$.*

Definition 5 *A Decision problem $L \subseteq \Sigma^*$ is NP-*complete *if $L \in NP$ and L is NP-*hard.

Now it seems very difficult as how we prove that every problem in NP reduces to some problem. So we would like to use reducability to show that one problem is as hard as the other one. But even to do this we need to have a problem to start with, which is proved to be NP-*complete*.So here is a theorem that gives us such a problem.

Theorem 2 ***Cook-Levin Theorem:*** *SAT is NP-*complete.

Another useful result is:

Theorem 3 *If $P \neq NP$, then there are problem in NP that are neither NP-complete nor in P. This is known as Ladner's Theorem.*

Definition 6 *$3-CNF$ SAT: Given a formula in conjunctive normal form such that every clause has exactly 3 literals, find if its Satisfiable.The formula in CNF consists of:*

- *Atomic propositions: q_i*
- *Literals: q_i, $\neg q_i$, l_j*
- *Clause: k_i consists of disjunction of 3 literals, e.g., $(q_i \vee \neg p_i \vee \neg r_j)$*

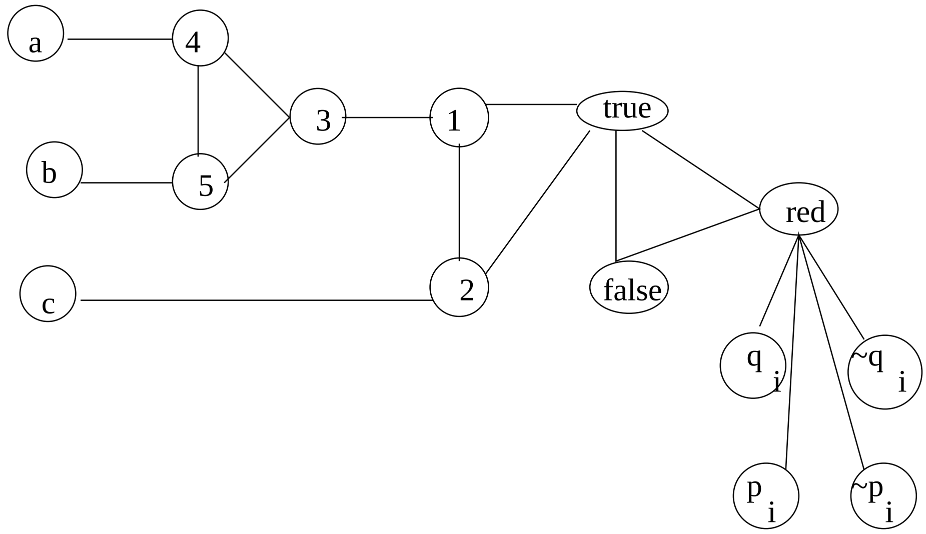

Figure 1: 3-SAT to 3-COLOR

- *Formula: conjunction of all $k_i^\prime s$*

Claim:$3-SAT$ is NP-*complete*
proof:$3-SAT \in NP$, because given a certificate that will be just a truth assignment for all the literals in the formula, it can be checked in linear time, by assigning the values to the variables and checking each clause, that the formula is satisfiable or not. So clearly $3-SAT \in NP$.
and now to show that its NP-*complete* we will show that $3-SAT \leq_p 3-colorability$.

We define a function $f : 3-CNF \to G(V,E)$ such that $\varphi \in 3-CNF$ is satisfiable iff $G(V,E)$ is 3-colorable.

Definition of $f(psi)$L

1. create 3 vertices true, false and red.
2. for every proposition $q_i \in AP(\psi)$ create 2 vertices q_i and $\neg q_i$ with an edge between them and connect both of them to red.
3. for every clause $k_i = a \vee b \vee c$ we have 3 vertices for the literals a, b, and c and five other vertices arranged in the manner shown in the diagram (for a, b, c we use the vertices created for the propositions in $AP(\psi)$).

Assume we have 3 colors - True, False and Red, coloring the nodes true, false, and red. Since every $q_i and \neg q_i$ are connected to a red vertex and to each other, none of them can have the color Red. So every literal has to have either the color True or the color False, and a literal and its negation cannot have the same color.

We now show that ψ is satisfiable iff $f(\psi)$ is 3-colorable. Suppose first that $f(\psi)$ is 3-colorable. We show that either a, b, or c is colored True. A truth assignment τ assigning 1 to p_i precisely when it is colored True satisfies ψ. This can be divided into two cases:

- if color(1) = Red then color(2)=False and color(c)=True.
- if color(1) = False then color(3) is either be Red or True, which means at least one of the 4 and 5 has to have the color False, ensuring that a are b have the color True.

Suppose now that ψ is satisfiable by a truth assignment τ. We then color p_i (resp., $\neg p_i$) with True (resp., False) if $\tau(p_i) = 1$, and with False (resp., True) if $\tau(p_i) = 0$. Consider a clause $k_i = a \vee b \vee c$. We have to consider the seven truth assignments that satisfy k_i. For each of these seven assignments we can assign proper colors to the vertices $1, 2, 3, 4, 5$. Thus, $f(\psi)$ is 3-colorable.