Complexity theory

Daniel Chatfield

May 10, 2015

2007 Paper 5 Question 12

1. (a) Give a precise definition of polynomial-time reductions. [2]

> Given two languages $L_1 \subseteq \Sigma_1^*$, and $L_2 \subseteq \Sigma_2^*$, L_1 is polynomial-time reducible to L_2 if there is a polynomial-time computable function f such that for every string $x \in \Sigma_1^*$, $f(x) \in L_2$ if, and only if, $x \in L_1$.

(b) Give a precise definition of NP-completeness. [3]

> A language is NP-HARD if every language in NP is polynomial-time reducible to it.
>
> A language is NP-COMPLETE if it is NP-HARD and is in NP.

(c) Let *Subset Sum* denote the following decision problem:

Given a set of positive integers $S = \{v_1, \ldots, v_n\}$ and a number t, determine whether there is a subset of S that sums to exactly t.

i. Explain why *Subset Sum* is in NP. [3]

> *Subset Sum* is in NP because it can be decided by a nondeterministic machine that non-deterministically guesses subsets and verifies whether they sum to t.
>
> The verification process is trivially polynomial time as it is simply the summation over the subset, which by virtue of being a subset is bounded by the size of the original set.

ii. Describe a polynomial-time reduction from the problem of 3-dimensional matching to *Subset Sum*. [9]

Let f be a function that takes an instance of 3DM (X, Y, Z, M) and returns an instance of Subset Sum.

Assign integer labels to each member of sets X, Y, and Z such that successive labels are successive powers of 2, starting with $2^0 = 1$ and ending with 2^{n-1} where $n = |X \cup Y \cup Z|$.

Let S be a new set such that for every triple in M, the sum of the integer labels in the triple is in S.

Return a tuple of S and t where $t = 2^n - 1$.

It is clear that f runs in polynomial time since it just assigns labels, one for each element in sets X, Y, and Z and then for each triple of M sums the 3 elements' labels.

To show that this is a reduction we show that an instance of 3DM has a matching iff S has a subset which sums to t.

t is one less than a power of 2, and thus since each label was a power of 2 the only summation of labels that will equal t is exactly one of each power of two up to 2^{n-1}. Therefore, if S has a subset which sums to t then there must be a subset whose summation is formed from every power of two up to 2^{n-1}. This means that the every element in X, Y, and Z must appear exactly once in the triples that correspond to the members of the subset and thus the instance has a matching.

If an instance has a matching then every element must appear exactly once in the triples within the matching. The summation over the elements in S that correspond to the triples in the matching can trivially be seen to equal $2^n - 1$ since it is the sum of successive powers of 2 up to 2^{n-1}.

iii. Explain why parts (i) and (ii) above imply that *Subset Sum* is NP-complete. [3]

3DM is NP-HARD. Since every language in NP is polynomial-time reducible to 3DM we know that for every langauge L in NP there exists a polynomial-time reduction to 3DM f_L. Let g be the reduction described in part (ii), the composition of f_L and g is a polynomial-time reduction from L to Subset Sum and thus Subset Sum is NP-HARD.

In part (i) we showed that Subset Sum was in NP and thus it is NP-COMPLETE.

2008 Paper 5 Question 12

2. (a) Give a precise definition of what it means for one decision problem to be polynomial-time reducible to another. [3]

> Given two languages $L_1 \subseteq \Sigma_1^*$, and $L_2 \subseteq \Sigma_2^*$, L_1 is polynomial-time reducible to L_2 if there is a polynomial-time computable function f such that for every string $x \in \Sigma_1^*$, $f(x) \in L_2$ if, and only if, $x \in L_1$.

(b) Consider the following two decision problems: [8]

HamCycle Given a graph $G = (V, E)$ does it contain a cycle that visits every vertex exactly once?

HamPath Given a graph $G = (V, E)$ and two distinguished vertices $s, t \in V$, is there a simple path in G that starts at s, ends at t and visits every other vertex exactly once?

Show that *HamCycle* is polynomial-time reducible to *HamPath*.

> Consider an instance of HamCycle $G = (V, E)$, the graph G is Hamiltonian if, and only if there exists a cycle that visits every vertex exactly once.
>
> Without loss of generality, we can fix the start/end vertex for the cycle since if a Hamiltonian cycle exists from one vertex then there must also be ones from ever other vertex as you can treat the cycle as a continuous loop and move the start position around it.
>
> Let's fix the start/end vertex as v. The penultimate vertex in a cycle has to have an edge between it and v since, without such an edge it would not be possible to return to v in one step.
>
> We can then construct a new graph $G' = (V', E')$ from G where $V' = V \cup \{s, t\}$ and E' contains all edges in E as well as an edge from s to v and edges from t to all vertices in V that share an edge with v.
>
> We now have an instance of HamPath (G', s, t). If there is a Hamiltonian Path then the first vertex after s must be v and the last one before t must be a vertex that shares an edge with v. It can therefore be seen that such a path over G' could be converted into a cycle over G by removing s and t from it and following the one remaining edge to complete the cycle.

(c) The following decision problem is known to be solvable in polynomial

time:

EulerCycle Given a graph $G = (V, E)$ does it contain a cycle that visits every edge exactly once?

What can you conclude about the truth of the following statements? Justify your answers.

i. *EulerCycle* is polynomial-time reducible to *HamCycle*. [3]

> HamCycle is NP-complete and therefore every language in NP is reducible to it. EulerCycle is in P and is thus in NP since P is a subset of NP.
>
> The statement is therefore true.

ii. *EulerCycle* is polynomial-time reducible to *HamPath*. [3]

> This statement is true. From part (i) EulerCycle is reducible to HamCycle and from previous question HamCycle is reducible to HamPath and thus by the composition of these reductions EulerCycle is reducible to HamPath.

iii. *HamPath* is polynomial-time reducible to *EulerCycle*. [3]

> EulerCycle is in P, if HamPath is polynomial-time reducible to it then HamPath must also be in P.
>
> Since HamPath is NP-HARD this is only true if $P = NP$.

2010 Paper 6 Question 1

3. (a) Give precise definitions of polynomial-time reductions and NP-completeness. [4]

> Given two languages $L_1 \subseteq \Sigma_1^*$ and $L_2 \subseteq \Sigma_2^*$, L_1 is polynomial-time reducible to L_2 if there is a polynomial-time computable function f such that for every $x \in \Sigma_1^*$, $x \in L_1$ if, and only if $f(x) \in L_2$.
>
> A language is NP-HARD if every language in NP is polynomial-time reducible to it. A language is NP-COMPLETE if it is NP-HARD and also in NP.

(b) Prove that for any language L, L is polynomial-time reducible to some problem in NP if, and only if, L is in NP. [6]

> If a language is in NP it is reducible to itself.
>
> For the other direction, I'm going to assume that there is a language L_1 that is not in NP but is polynomial-time reducible to a language in NP L_2 and derive a contradiction.
>
> Let f be the polynomial-time reduction and M be the nondeterministic machine that decides L_2. Since the running time of f is polynomial, the length of $f(x)$ must be polynomial in the length of x. Since M, by definition, is polynomial in the length of its input by composition of polynomials is also polynomial in the length of x.

(c) In a simple graph $G = (V, E)$, a set of vertices $X \subseteq V$ is said to be a *vertex cover* of G if every edge $e \in E$ has one endpoint in X. A set $X \subseteq V$ is an independent set of G if there is no edge between any two vertices in X.

VERTEX COVER is defined as the decision problem where, given a graph $G = (V, E)$ and a positive integer k, we are to determine whether G contains a vertex cover with k or fewer vertices.

INDEPENDENT SET is defined as the decision problem where, given a graph $G = (V, E)$ and a positive integer k, we are to determine whether G contains an independent set with k or more vertices.

i. Show that a set X is a vertex cover of G if, and only if, its complement $V \backslash X$ is an independent set of G. [2]

> Let Y be the complement of X, if Y is an independent set then there is no edge between two vertices in Y. Since every edge has to vertices it is clear that each edge must either have one vertex in X and one in Y or they are both in X. X is therefore a vertex cover.
>
> The reverse implication is similar, if X is a vertex cover then at least one of the vertices of every edge must be in X and thus no edge can have both vertices in Y so Y must be an independent set.

ii. Use this to show that *VERTEX COVER* is polynomial-time reducible to *INDEPENDENT SET* and vice versa. [6]

Consider an instance of the VERTEX COVER decision problem, (V, E, k). From above if there is a vertex cover X then the complement must be an independent set. Since, in this problem, we are looking for a vertex cover with k or fewer vertices it is sufficient to show that there is an independent set with at least $n - k$ vertices (where n is the total number of vertices). The problem is therefore reduced to INDEPENDENT SET with $n - k$ as the target.

This function is clearly polynomial-time since it is just subtracting k from the number of vertices.

The reverse argument is identical.

iii. What can you conclude about the complexity of *VERTEX COVER*? [2]

VERTEX-COVER is reducible to INDEPENDENT-SET and is thus in NP as INDEPENDENT-SET is in NP.

INDEPENDENT-SET is NP-HARD, therefore every problem in NP is reducible to it. Since INDEPENDENT-SET is reducible to VERTEX-COVER, it follows by composition of reductions that every problem in NP is polynomial-time reducible to VERTEX-COVER and thus VERTEX-COVER is NP-HARD.

Since it is NP-HARD and in NP, it is NP-COMPLETE.