BU CLA CS 530: Analysis of Algorithms
Spring 1995

Problem Set 11 (due Friday 95.04.14 at 4:30 p.m.)

---

1. (a) Consider the following algorithm for CLIQUE (decide whether an undirected graph $G = (V, E)$ has a clique of size k, see [CLR] page 947). First, generate all k-subsets of V. There are $\mathcal{O}(n^k)$ such subsets. Second, check whether any of the subgraphs induced by these subsets is complete. Why is this not a polynomial-time algorithm for the clique problem (which would then imply P = NP)?

   (b) Relative to the decision problem CLIQUE, we gave 3 optimization problems in class; these were called CLIQUE-1 (find size of a maximum clique), CLIQUE-2 (find a maxi<u>mum</u> clique) and CLIQUE-3 (find a maxi<u>mal</u> clique). We showed CLIQUE $\equiv_{\mathrm{P}}$ CLIQUE-1 $\equiv_{\mathrm{P}}$ CLIQUE-2.[1] Show that CLIQUE-3 is solvable in polynomial time. Is it the case that CLIQUE $\leq_{\mathrm{P}}$ CLIQUE-3? How about CLIQUE-3 $\leq_{\mathrm{P}}$ CLIQUE? Justify briefly.

2. Define the following decision problem HAM-PATH′: Given an undirected graph G, determine whether or not G has a hamiltonian path. HAM-PATH′ is different from HAM-PATH in [CLR] (Exercise 36.2-6, page 929); the latter specifies two distinct vertices as part of an input instance. DIR-HAM-PATH is the directed version of HAM-PATH: Given a directed graph G and two distinct vertices v and w, determine whether or not G has a hamiltonian path from v to w. Likewise, DIR-HAM-PATH′ is the directed version of HAM-PATH′ (no end vertices specificied as part of input instances), and DIR-HAM-CYCLE the directed version of HAM-CYCLE.

   (a) Exercise 36.2-6, page 929, in [CLR]. (Hint: This is easy. See the argument on page 926 showing that HAM-CYCLE is polynomial-time verifiable.)

   (b) Show that DIR-HAM-CYCLE is NP-complete. (Hint: HAM-CYCLE $\leq_{\mathrm{P}}$ DIR-HAM-CYCLE.)

   (c) Show that DIR-HAM-PATH′ is NP-complete. (Hint: DIR-HAM-CYCLE $\leq_{\mathrm{P}}$ DIR-HAM-PATH′. Split an arbitrary vertex $v \in V$ into v^{in} and v^{out}.)

   (d) Show that HAM-PATH′ is NP-complete. (Hint: DIR-HAM-PATH′ $\leq_{\mathrm{P}}$ HAM-PATH′. Split every vertex $v \in V$ into v^{in}, v^{mid} and v^{out}, and add edges $\{v^{\mathrm{in}}, v^{\mathrm{mid}}\}$ and $\{v^{\mathrm{mid}}, v^{\mathrm{out}}\}$.)

   (e) Exercise 36.5-5, page 961, in [CLR]. (Hint: HAM-PATH′ $\leq_{\mathrm{P}}$ HAM-PATH.)

---

[1] The reduction relation $\leq_{\mathrm{P}}$ is used in [CLR], page 930, to relate decision problems only. But we can use it to relate computational problems in general, as done in class. Moreover, our $\leq_{\mathrm{P}}$ in class was more liberal than the $\leq_{\mathrm{P}}$ in [CLR]: We wrote $\mathcal{P}_1 \leq_{\mathrm{P}} \mathcal{P}_2$ in case we have a polynomial-time algorithm $\mathbf{A}_1$ for $\mathcal{P}_1$ which is allowed, on each one of its runs, to make "finitely many" calls (by contrast to "at most one" call in [CLR]) to an algorithm $\mathbf{A}_2$ for $\mathcal{P}_2$ — where the time for $\mathbf{A}_1$ is polynomial provided every call to $\mathbf{A}_2$ is assumed to take one time unit. Other textbooks make the distinction between these two versions of $\leq_{\mathrm{P}}$ and study their relationship; for the presentation of Chapter 36 in [CLR], it suffices to define the more restrictive version only, although you will find that the more liberal version is often more convenient to use.

3. Show that VERTEX-COVER (page 949 in [CLR]) remains NP-complete even if all vertices in the input graph are restricted to have even degrees. (Hint: The sum of the degrees of the vertices in an undirected graph is always an even number.)

4. This is a continuation of Problem 2 in Problem Set 8. We say that a Monte Carlo algorithm is *consistent* if it never gives two different correct solutions to the same input instance. If the Monte Carlo algorithm is for a decision problem (i.e. output is always *yes* or *no*), then the algorithm is necessarily consistent (why?). But if the Monte Carlo algorithm is for a more general computational problem, then it may or may not be consistent. For example, if the computational problem is that of finding a maximal independent set (MIS) in an undirected graph G, then a correct answer is not unique generally: The Monte Carlo algorithm, if it is not consistent, may return different correct solutions (i.e. different MIS's) in different runs on the same input instance G.

   Let $\mathbf{A}$ be a Monte Carlo algorithm for some computational problem $\mathcal{P}$. Denote by $\mathbf{A}(I)$ the outcome of running $\mathbf{A}$ on input instance I. Let the error probability of $\mathbf{A}$ be $1/4$, i.e. $\mathbf{A}(I)$ is a correct output with probability $3/4$. Consider the following algorithm $\mathbf{B}$:

```
B(I)
    x ← A(I)
    y ← A(I)
    z ← A(I)
    if x = y or x = z
       then return x
       else return z
```

   (a) Show that if $\mathbf{A}$ is consistent, then $\mathbf{B}$ is also a consistent Monte Carlo algorithm but with a smaller error probability. (Hint: See solutions for Problem Set 8.)

   (b) Show that if $\mathbf{A}$ is not consistent, then neither is $\mathbf{B}$ consistent and, moreover, its error probability can exceed $1/4$. (Hint: Define an appropriate input-output behavior for $\mathbf{A}$. For example, suppose that $\mathbf{A}$ on an arbitrary input I always returns one of 6 possible answers $a_1, \ldots, a_6$, such that each of a_1, a_2, a_3, a_4, and a_5 is correct and returned with probability $3/20$, and a_6 is wrong and returned with probability $1/4$.)

Created: 95.04.05