**CSCI 510** **Models of Computation**

# Midterm 1: Panic Attack

*Due: Oct 20, 2010*

## Problem 1

Design an FSM that adds two $k$-bit binary numbers when the two least significant bits of both numbers are given first, followed by the two next least significant numbers, etc. Using this FSM, design a logic circuit to add two $k$-bit numbers. Determine the size and depth of this circuit. The size is the number of gates in the circuit and the depth is the length of the longest path from an input to an output.

Do other circuits exist that use about the same number of gates but have much smaller depth? If you answer this question by pointing to an existing solution, please give a high-level explanation of the method used to design the circuit with smaller depth.

**Hint:** Have you read the book?

## Problem 2

Design a Turing machine that recognizes the language $L \subseteq \{0,1\}^*$ that consists of strings containing at least twice as many 1s as 0s.

## Problem 3

You are a rock singer, holding tryouts for your new band. You have $n$ guitarists at the tryouts. Your goal is to select $k$ guitarists for your group. Since all of these guitarists are very good, it doesn't matter which $k$ of them you choose. However, in order to maximize collaboration between the band members, you and your manager decide that these $k$ guitarists should know each other. We call such a set of people a $k$-complete set.

The information about the guitarists is captured in an $n$-vertex graph $G = (V, E)$. $V$ has one vertex per guitarist and $E$ has an edge between two vertices if the corresponding guitarists know each other.

Your task is to find a $k$-complete band of $k$ guitarists from the graph $G = (V, E)$ if one exists. If none exists, report this fact. If possible, you seek an algorithm for this problem that runs in polynomial time in $|V| + |E|$, the number of vertices plus edges in the graph.

Remembering how CS51 inspired you to become a rocker, you formalize this as a function $f_{band}$ mapping a graph $G$ and a binary integer $k$ to either a $k$-tuple if a solution exists or $\perp$ indicating that there is no $k$-complete set of vertices in $G$. That is, you write this as a mapping $f_{band} : \{(V, E), k\} \mapsto \{V^k \cup \{\perp\}\}$.

Because you have heard so much about languages, you also choose to consider the decision version of this problem specified by the language $\mathcal{L}$ defined below where $(G, n, k)$ is an instance of the problem. The Yes instances are the instances $(G, n, k)$ that contain a $k$-complete graph.

$$\mathcal{L} = \{(G, n, k) \mid G \text{ is an } n\text{-vertex graph containing a } k\text{-complete subgraph}\}$$

a. Your manager, who has also taken CS51, tells you that the language $\mathcal{L}$ is **NP**-complete. Show that this implies that $f_{band}$ cannot be computed in time polynomial in the size of the input $\{(V, E), k\}$, unless $\mathbf{P} = \mathbf{NP}$.

b. Suppose your manager can decide whether an instance is a Yes instance in one unit of time. Your manager expects that you can devise a polynomial-time algorithm to compute $f_{band}$ by giving him instances $(G, n, k)$. Show that your manager is correct by exhibiting an algorithm and showing that its running time is polynomial in the length of the input.

c. Suppose that you have discovered a correct polynomial-time algorithm for deciding membership in $\mathcal{L}$, then can we say that computing $f_{band}$ is in **P**? **Hint:** How is **P** defined?

## Problem 4

Recall the IndependentSet problem you saw in lecture. On input $\langle G, k\rangle$, IndependentSet is **NP**-complete. Now consider the problem

$$k\text{-IndependentSet} = \{\langle G\rangle \mid G \text{ has an independent set of } k \text{ veritces}\}.$$

In other words, an instance of $k$-IndependentSet is a graph $G = (V, E)$, and a Yes instance is a graph $G$ such that $\langle G, k\rangle$ is a Yes instance of IndependentSet.

a. Prove that 5-IndependentSet $\in$ **P**.

b. Prove that $k$-IndependentSet $\in$ **P** for any fixed $k$.

c. Given that IndependentSet is **NP**-complete, why does part (b) not show that **P** = **NP**?

   **Hint:** Remember the reduction you saw in class when you proved that IndependentSet is **NP**-complete. What was $k$?

## Problem 5

An collector is selling off a set of $n$ pieces of rock memorabilia. Rather than selling off each piece individually, he thinks he could get a higher price if he sold items by organizing them into sets (for example, one set could be only classic drum sets, or one set could contain only instruments used at Woodstock). Some items could potentially be part of more than one set, but the collector has to decide which sets contain which items before he starts selling. This problem is formulated as follows.

We are given a set of items $S = \{1, 2, 3, \ldots, n\}$ and $m$ subsets $S_1, S_2, \ldots, S_m$ where for each $i$, $S_i \subseteq S$. Attached to the $i$th subset is a price of $x_i$ dollars. The auctioneer would like to bring in at least $k$ dollars. Thus, the question the auctioneer asks is if there are disjoint subsets $S_{i_1}, S_{i_2}, \ldots, S_{i_l}$ such that

$$\sum_{j=1}^{l} x_{i_j} \geq k$$

Prove this task is **NP**-complete by reduction from 3SAT.