CSCI 510 **Models of Computation**

HW4: Brainpower

Due: Oct 6, 2010

Include your *full name*, *CS login*, and the problem number(s) on each piece of paper you hand in, and please staple your pages together before handing in.

While collaboration is encouraged in this class, please remember not to take away notes from collaboration sessions other than your scheduled lab section.

Problem 1

Consider the following two languages:

$$\begin{aligned}\textsc{VertexCover} = \{\langle G, k\rangle \mid\ & G \text{ is an undirected graph, } k \text{ is a number of verticies,} \\ & \text{and there exists a set } S \text{ of } k \text{ or fewer vertices of } G \text{ such that} \\ & \text{every edge of } G \text{ is connected to at least one vertex in } S.\}\end{aligned}$$

$$\begin{aligned}\textsc{SubsetUnion} = \{\langle S, Q, k\rangle \mid\ & S \text{ is a set of integers, } Q \text{ is a set of sets of integers,} \\ & \text{and } k \text{ is a positive integer such that there is a collection} \\ & \text{of } k \text{ elements } Q \text{ where the union of those } k \text{ elements is } S.\}\end{aligned}$$

For example, if

$$S = \{1, 2, 3, 5, 7, 9\}, Q = \{\{1, 2, 5\}, \{1, 3, 7\}, \{2, 3, 5, 9\}\}, k = 3,$$

then $\langle S, Q, k\rangle \in$ SubsetUnion since $\{\{1, 2, 5\}, \{1, 3, 7\}, \{2, 3, 5, 9\}\}$ contains 3 sets from Q and the union of those 3 sets is equal to S.

Alternatively, if

$$S = \{1, 2, 3, 4, 5\}, Q = \{\{1\}, \{2, 3\}, \{4, 5\}\}, k = 2,$$

then $\langle S, Q, k\rangle \notin$ SubsetUnion since no two elements of Q union to form S.

a. Explain in a few sentences how one can reduce the language VertexCover to the language SubsetUnion in polynomial time.

b. Prove that your reduction works.

c. Explain why your reduction has polynomial running time.

d. Prove that if VertexCover is NP-Complete, then SubsetUnion is also NP-Complete.

Problem 2

Define the following two languages:

- SUBSTRING = $\{\langle x, s\rangle \mid x \text{ is a substring of } s\}$
- PALINDROME = $\{\langle w\rangle \mid w = w^R$, where w^R is the reverse of $w\}$

Give an algorithm to reduce SUBSTRING to PALINDROME. That is, assuming you have a machine M that can decide whether a string w is in PALINDROME, construct a machine, using M as a subroutine, that, on input $\langle x, s\rangle$ decides if x is a substring of s. Your algorithm must run in polynomial time.

Hint: If necessary, you may run M multiple times during your computation. Note that if your reduction is polynomial time, you can run M at most a polynomial number of times.

The following questions are lab problems. Please remember to prepare a solution for your assigned problem before going to your lab section.

Lab Problem 1

Consider the following language:

MOD-EXP = $\{\langle a, b, c, p\rangle \mid a^b \equiv c \bmod p$ where a, b, c, p are **binary** integers$\}$

a. Consider the following algorithm to decide MOD-EXP:

```
On input ⟨a, b, c, p⟩:
    let r ← 1
    repeat b times:
        r ← a * r mod p
    if r = c mod p, accept; else reject
```

This creates the result $r = a^b \bmod p$ by multiplying a by itself b times. Then it checks whether it is equal to $c \bmod p$.

While this algorithm is correct, it does not run in polynomial time. Explain why it does not run in polynomial time.

b. Give a polynomial-time algorithm for this problem. Explain why it takes polynomial time. You can assume multiplication and modulo operations can be performed in polynomial time.

Lab Problem 2

2SAT is the language of *satisfiable* boolean formulas consisting of the conjunction of a number of boolean clauses, each containing exactly 2 boolean literals. That is, 2SAT consists of satisfiable boolean formulas in which each clause is of the form $(x_a \vee x_b)$, possibly negating either literal, and all clauses are "ANDed" together. Example:

$$(x_a \vee x_b) \wedge (x_a \vee \overline{x_c}) \wedge (\overline{x_b} \vee \overline{x_d}) \wedge (\overline{x_b} \vee x_d)$$

ConSAT is the language of *satisfiable* conditional clauses in which one boolean literal implies another boolean literal. In other words, ConSAT is the set of all satisfiable boolean formulas in which conditionals of the form $(x_a \Rightarrow x_b)$ are "ANDed" together. Example:

$$(\overline{x_a} \Rightarrow \overline{x_b}) \wedge (x_c \Rightarrow x_b) \wedge (\overline{x_b} \Rightarrow x_d) \wedge (x_d \Rightarrow \overline{x_c})$$

Note that $(x \Rightarrow y)$ means "if x is true, then y must also be true." The statement constrains y only when x is true, and it never constrains x.

PATH is the language containing triplets of the form (G, p, q), where G is a directed graph, and there exists a directed path from vertex p to vertex q in G.

a. Give a polynomial time reduction from 2SAT to ConSAT.

b. Give a polynomial time reduction from ConSAT to PATH.
Hint: Use a directed graph to model the implications.

c. Given the reductions above, is 2SAT in P? (No proof required.)

Lab Problem 3

Journey is going on tour, and they need to pack up their stuff to go in the truck. Like any good band, they have a whole bunch of guitars that need to go into the van, each of which is in its own case. Every case has the same length and width, but there are many different heights; after all, some guitars need fancier cases than others. The heights of the guitars are defined in the set $S = \{x_1, x_2, \ldots, x_n\}$, where x_i is the height of the ith guitar case. Because of federal regulations, all guitars need to be packed into specially-made bins for transport. All of the bins have the same length and width

as the guitar cases, and are of height b. This means that you can only put guitars into a bin as long as the total sum of the heights of the cases in that bin is less than or equal to b. The band members want to know if all of their guitars can fit into at most k bins.

Prove that the Guitar Packing Problem is NP-Complete by a reduction from SubsetSum. You can assume without loss of generality that in the original subset sum problem, the target has a value that is at least half of the total sum of the values in the set (it may help to figure out why you can assume this).