Announcements

- We have a midterm this Wednesday, October 14.

**PRIMES is in P**

Consider the following algorithm that determines if a given number is prime or not.

```python
isPrime(N):
    if (N < 2):
        return False
    for factor in {2,3,4,...,N-1}:
        if (N % factor == 0):
            return False
    return True
```

(a) Assume that the input to the function is encoded in binary. What is the length of the input, in terms of $N$, using this encoding scheme?

(b) What is the running time (in Big-Oh) of the long division algorithm you have learned in grade school?

(c) What is the running time of the above function isPrime in Big-Oh as a function of the input length?

(a) Approximately $\lg N$.

(b) It takes $O(mn)$ time to divide an $n$-bit number by an $m$-bit number.

(c) $O(n^2 \cdot 2^n)$ where $n$ is the length of the input. (Note that we are testing up to $N \approx 2^n$ factors.)

**Decisions**

Prove whether or not each of the following are decidable.

(a) $S = \{ M \mid \text{there exists a circuit family that decides } L(M) \}$

(b) $T = \{ M \mid L(M) \in P \}$

Fun fact: The set of winning positions for white in generalized $n \times n$ chess is known to be decidable but is also known not to be in $P$. 
(a) Recall that every boolean function is decidable by a circuit family. We can decide \( S \) by simply returning \( \text{True} \).
(b) If we had a decider for \( T \) (call it \( MP \)), we could construct the following decider for HALTS:

```python
def HALTS(<M,x>):
    def HELPER(b):
        run M(x)
        if b denotes a winning position for white in generalized chess, accept
        else reject
        return not(MP(<HELPER>))
```

Note that \( L(HELPER) \) is in \( P \) iff \( M(x) \) halts.

**Uncounting**

Are the following sets countable?

(a) The set of directed trees
(b) The set of circuit families
(c) Bonus: The set of circuit families that decide regular languages over the alphabet \( \{0, 1\} \)

(a) Yes, by the CS method. (What might be your choice of \( \Sigma \) here?)
(b) No, by injection from \( \{0, 1\}^\infty \). (Example: for an infinite binary string \( b \), create a circuit family \( C \) such that \( C_i \) is a circuit that always returns 1 if \( b_i = 1 \) and always returns 0 otherwise.)

**Sneaky structures (part 2)**

Prove by induction that a graph with maximum degree up to \( k \) must be \((k + 1)-colorable \).

Warning: induction on graphs can lead to subtle logical pitfalls if you aren’t careful. Contrast this proof with the bad induction from recitation 1 and understand why the latter doesn’t work.

First, we fix \( k \in \mathbb{N} \). Then we induct on \( n \), the number of vertices in the graph.

Base case \((n = 1)\): Duh. (We know \( k + 1 \geq 1 \).)
Induction hypothesis: Suppose for some \( n \in \mathbb{N}^+ \) that all graphs having up to \( n \) vertices and maximum degree up to \( k \) must be \((k + 1)-colorable \).
Induction step: Let \( G \) be a graph on \( n + 1 \) vertices and having maximum degree up to \( k \). Remove some vertex \( v \) and all of its incident edges from \( G \) to get \( G' \), a graph on \( n \) vertices. Note that \( G' \) still has maximum degree at most \( k \). By the IH, \( G' \) is \((k + 1)-colorable \). Consider some \( k \)-coloring of \( G' \), then add \( v \) and the removed edges back in. We know that \( v \) had at most \( k \) neighbors, so even if they all are different colors, there is a “free” color remaining that we can assign to \( v \). So we see that \( G \) is \((k + 1)-colorable \).

**Counting is hard**

Define a function smash where \( \text{smash}(n) \) returns the in-order concatenation of all the numbers \( 0 \rightarrow n \).

For example, \( \text{smash}(10) = \text{“012345678910“} \). Show that \( L = \{ \text{smash}(n) \mid n \in \mathbb{N} \} \) is irregular.
AFSOC that some DFA with \( s \) states accepts exactly \( L \). Now feed the strings smash(0), smash(1), ... smash(\( s \)) to the DFA. By PHP, some smash(\( i \)) and smash(\( j \)) where \( i \neq j \) must end in the same state. Now consider what happens when you concatenate \( i + 1 \) to both of these strings: the DFA will either accept or reject both, but it should accept only the former. Contradiction.