Recitation 14 - Homework 7 and Time Complexity

John Chilton

December 6, 2006

Today

- Homework 7 Problems and Examples

TRUE or FALSE with explanation.

Big-O Problems

- $n^2 = O(n)$
- $3^n = 2^{O(n)}$
- $2^{2^n} = O(2^{2^n})$

Little-o Problems

- $n = o(2n)$
- $1 = o(1/n)$

Big-O Definition

$f(n) = O(g(n))$ if there exists *constants* n_0 and c such that:

$$\text{For all } n \geq n_0, f(n) \leq c * g(n)$$

Big-O Examples

Some other big-O examples.

- $n = O(n * log^3 n)$
- $n^2 = O(n * log^3 n)$
- $3^n = O(4^n)$
- $4^n = O(3^n)$
- $4^n = O(3^{2n})$

Little-o Definition

$f(n) = o(g(n))$ iff

$$\lim_{n \to \infty} \frac{f(n)}{g(n)} = 0$$

Little-o Examples

Some other little-o examples.

- $2n = o(n^2)$
- $\frac{1}{2}n^2 = o(n^2)$

Graph Isomorphism Examples

Some Examples also check out
`http://en.wikipedia.org/wiki/Graph_isomorphism`

Two graphs are isomorphic if the nodes of one graph can be relabelled in such a way that the resulting graph is equal to the second graph.

Two graphs are isomorphic if the nodes of one graph can be relabelled in such a way that the resulting graph is equal to the second graph.

$$ISO = \{(G, H) \mid G \text{ is isomorphic to } H\}$$

Show $ISO \in NP$.

NP

A set is in *NP* iff an instance can be verified in polynomial time with help of some certificate.

- Certificate for *COMPOSITE* is two factors
- Certificate for *HAMPATH* is potential path
- Certificate for *SUBSET* is a subset of elements

To show $ISO \in NP$ must figure out what certificate allows you to verify two graphs are isomorphic in polynomial time and describe how do this.

To show $ISO \in NP$ must figure out what certificate allows you to verify two graphs are isomorphic in polynomial time and describe how do this.

$V =$ On input $((G, H), c)$.
$\#$ *Use c to verify that G is isomorphic to H is polynomial time.*

$$LPATH = \{(G, a, b, k) \mid G \text{ contains a simple path from } a \text{ to } b \text{ of length at least } k\}$$

$$LPATH = \{(G, a, b, k) \mid G \text{ contains a simple path from } a \text{ to } b \text{ of length at least } k\}$$

- $(G, a, b, 4) \in LPATH$?

$LPATH = \{(G, a, b, k) \mid G$ contains a simple path from a to b of length at least $k\}$

- $(G, a, b, 4) \in LPATH$?
- $(G, a, b, 6) \in LPATH$?

$$LPATH = \{(G, a, b, k) \mid G \text{ contains a simple path from } a \text{ to } b \text{ of length at least } k\}$$

- $(G, a, b, 4) \in LPATH$?
- $(G, a, b, 6) \in LPATH$?
- $(G, a, g, 6) \in LPATH$?

$LPATH = \{(G, a, b, k) \mid G$ contains a simple path from a to b of length at least $k\}$

- $(G, a, b, 4) \in LPATH$?
- $(G, a, b, 6) \in LPATH$?
- $(G, a, g, 6) \in LPATH$?
- $(G, c, g, 7) \in LPATH$?

$$LPATH = \{(G, a, b, k) \mid G \text{ contains a simple path from } a \text{ to } b \text{ of length at least } k\}$$

Show $LPATH \in NP$.

$LPATH = \{(G, a, b, k) \mid G$ contains a simple path from a to b of length at least $k\}$

Show $LPATH \in NP$.

- What should the certificate be?
- How do you use to verify in polynomial time?

c is a simple path in G from a to b of length at least k.

$V =$ On input $((G, a, b, k), c)$.

1. Verify length of c at least k
2. Verify each pair of nodes is adjacent.
3. Verify no node visited twice along path.
4. Verify path starts with a and end with b.
5. If any condition not met, *reject*, else *accept*.

An aside

Why does this verifier make the problem a member of NP? The following TM T solves the problem in polynomial time on a nondeterministic TM.

$T =$ On input (G, a, b, k).

1. If k is larger than $|V|$, *reject*.
2. Nondeterministically pick path of length k to $|V|$.
3. Call this path c, and run V on $((G, a, b, k), c)$

$$MODEXP = \{(a, b, c, p) \mid (a^b \mod p) = (c \mod p)\}$$

Demonstrate $MODEXP \in P$. That is some deterministic TM decides $MODEXP$ in polynomial time.

The naive approach:

$F =$ On input a, b, c, p.

1. Compute a^b and store on tape
2. Compute a^b modulo p
3. Compute c modulo p.
4. Accept iff results of 2 and 3 are equal.

The naive approach:

$F=$ On input a, b, c, p.
1. Compute a^b and store on tape
2. Compute a^b modulo p
3. Compute c modulo p.
4. Accept iff results of 2 and 3 are equal.

Why not polynomial time?

The naive approach:

$F =$ On input a, b, c, p.

1. Compute a^b and store on tape
2. Compute a^b modulo p
3. Compute c modulo p.
4. Accept iff results of 2 and 3 are equal.

Why not polynomial time?

- a^b calculation
- Taking mod of such a large number

Show the following identity holds:

$$((m * n) \bmod p) = (((m \bmod p) * (n \bmod p)) \bmod p)$$

- Prove this result.
- Use it to describe a polynomial time algorithm for deciding *MODEXP*.

May also need the following identity, if n is even:

$$b^n = (b^2)^{\frac{n}{2}}$$

Part a.

$$SPATH = \{(G, a, b, k) \mid G \text{ contains a simple path from } a \text{ to } b \text{ of length at most } k\}$$

Show $SPATH \in P$.

- Show some polynomial time TM decides this set
- Think of how to utilize a breadth-first search (BFS)

Part b.

$$LPATH = \{(G, a, b, k) \mid G \text{ contains a simple (loopless) path from } a \text{ to } b \text{ of length at least } k\}$$

Show *LPATH* is NP-Complete.

- Show some polynomial time reduction from *UHAMPATH*
- This would show if you could solve this problem in *P* time, you could for *UHAMPATH* also

UHAMPATH requires there exist a path from a given a to b such that every node is visited exactly once, *LPATH* requires the path is of length at least k for a a given k. Relate these requirement to find a reduction from *UHAMPATH* to *LPATH*.

Problem 7.28.

$$SETSPLITTING = \{(S, C = \{C_i\}) \mid C_i \subseteq S \;\&\; (S, C) \text{ colorable}\}$$

S can be colored if each element can be chosen to be *red* or *blue* such that no C_i's elements are all the same color.

Coloring Examples

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

Coloring Examples

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

- C = {{1,2}, {3,5}}

Coloring Examples

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

- C = {{1,2}, {3,5}} $\in$ *SETSPLITTING*

Coloring Examples

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

- C = {{1,2}, {3,5}} $\in$ *SETSPLITTING*
- C = {{1,2,3}, {3,4,5}}

Coloring Examples

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

- C = {{1,2}, {3,5}} $\in$ *SETSPLITTING*
- C = {{1,2,3}, {3,4,5}} $\in$ *SETSPLITTING*

Coloring Examples

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

- C = {{1,2}, {3,5}} $\in$ *SETSPLITTING*
- C = {{1,2,3}, {3,4,5}} $\in$ *SETSPLITTING*
- C = {{1,2}, {1,3}, {1,4}, {2,4}}

Coloring Examples

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

- C = {{1,2}, {3,5}} $\in$ *SETSPLITTING*
- C = {{1,2,3}, {3,4,5}} $\in$ *SETSPLITTING*
- C = {{1,2}, {1,3}, {1,4}, {2,4}} $\notin$ *SETSPLITTING*

Coloring Examples

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

- C = {{1,2}, {3,5}} $\in$ *SETSPLITTING*
- C = {{1,2,3}, {3,4,5}} $\in$ *SETSPLITTING*
- C = {{1,2}, {1,3}, {1,4}, {2,4}} $\notin$ *SETSPLITTING*
- C = {{1,2,3}, {2,3}, {1}}

Coloring Examples

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

- C = {{1,2}, {3,5}} $\in$ *SETSPLITTING*
- C = {{1,2,3}, {3,4,5}} $\in$ *SETSPLITTING*
- C = {{1,2}, {1,3}, {1,4}, {2,4}} $\notin$ *SETSPLITTING*
- C = {{1,2,3}, {2,3}, {1}} $\notin$ *SETSPLITTING*

Problem 6. Show $SETSPLITTING$ is NP-complete, i.e. give a polynomial time reduction of some NP-complete problem to $SETSPLITTING$.

Some NP-complete problems SAT, $3SAT$, $HAMPATH$, $UHAMPATH$, $CLIQUE$, $VERTEX\,COVER$, $SUBSET\,SUM$, and $LPATH$.

I would pick $3SAT$.

$$Variables = \{x_1, x_2, x_3\}$$

$$\phi = \{x_1 \vee x_2 \vee x_3\} \wedge \{\bar{x_2} \vee x_2 \vee x_3\} \wedge \{\bar{x_1} \vee x_3 \vee \bar{x_3}\}$$

$$Variables = \{x_1, x_2, x_3\}$$

$$\phi = \{x_1 \vee x_2 \vee x_3\} \wedge \{\bar{x_2} \vee x_2 \vee x_3\} \wedge \{\bar{x_1} \vee x_3 \vee \bar{x_3}\}$$

$$\Downarrow\Downarrow\Downarrow$$

$$S = \{x_1, \bar{x_1}, x_2, \bar{x_2}, x_3, \bar{x_3}\}$$

$$C = \{\{x_1, x_2, x_3\}, \{\bar{x_2}, \bar{x_2}, \bar{x_3}\}, \{\bar{x_1}, x_3, \bar{x_3}\}\}$$

$$S = \{x_1, \bar{x_1}, x_2, \bar{x_2}, x_3, \bar{x_3}\}$$

$$C = \{\{x_1, x_2, x_3\}, \{\bar{x_2}, \bar{x_2}, \bar{x_3}\}, \{\bar{x_1}, x_3, \bar{x_3}\}\}$$

$$S = \{x_1, \bar{x_1}, x_2, \bar{x_2}, x_3, \bar{x_3}\}$$

$$C = \{\{x_1, x_2, x_3\}, \{\bar{x_2}, \bar{x_2}, \bar{x_3}\}, \{\bar{x_1}, x_3, \bar{x_3}\}\}$$

$$\Downarrow\Downarrow\Downarrow$$

$$S = \{{\color{red}x_1}, {\color{blue}\bar{x_1}}, {\color{blue}x_2}, {\color{red}\bar{x_2}}, {\color{red}x_3}, {\color{blue}\bar{x_3}}\}$$

$$C = \{\{{\color{red}x_1}, {\color{blue}x_2}, {\color{red}x_3}\}, \{{\color{red}\bar{x_2}}, {\color{red}\bar{x_2}}, {\color{blue}\bar{x_3}}\}, \{{\color{blue}\bar{x_1}}, {\color{red}x_3}, {\color{blue}\bar{x_3}}\}\}$$

$$\{x_1, x_2\}$$

$$\{x_1 \vee x_1 \vee x_2\} \wedge \{x_1 \vee x_1 \vee \bar{x_2}\} \wedge \{\bar{x_1} \vee \bar{x_1} \vee x_2\} \wedge \{\bar{x_1} \vee \bar{x_1} \vee \bar{x_2}\}\}$$

$\{x_1, x_2\}$

$$\{x_1 \vee x_1 \vee x_2\} \wedge \{x_1 \vee x_1 \vee \bar{x_2}\} \wedge \{\bar{x_1} \vee \bar{x_1} \vee x_2\} \wedge \{\bar{x_1} \vee \bar{x_1} \vee \bar{x_2}\}\}$$

$$\Downarrow\Downarrow\Downarrow$$

$S = \{x_1, \bar{x_1}, x_2, \bar{x_2}\}$

$$\{\{x_1, x_1, x_2\}, \{x_1, x_1, \bar{x_2}\}, \{\bar{x_1}, \bar{x_1}, x_2\}, \{\bar{x_1}, \bar{x_1}, \bar{x_2}\}\}$$

$S = \{x_1, \bar{x_1}, x_2, \bar{x_2}\}$

$$\{\{x_1, x_1, x_2\}, \{x_1, x_1, \bar{x_2}\}, \{\bar{x_1}, \bar{x_1}, x_2\}, \{\bar{x_1}, \bar{x_1}, \bar{x_2}\}\}$$

$$S = \{x_1, \bar{x_1}, x_2, \bar{x_2}\}$$

$$\{\{x_1, x_1, x_2\}, \{x_1, x_1, \bar{x_2}\}, \{\bar{x_1}, \bar{x_1}, x_2\}, \{\bar{x_1}, \bar{x_1}, \bar{x_2}\}\}$$

$$\Downarrow\Downarrow\Downarrow$$

$$S = \{{\color{red}x_1}, {\color{red}\bar{x_1}}, {\color{blue}x_2}, {\color{blue}\bar{x_2}}\}$$

$$\{\{{\color{red}x_1}, {\color{red}x_1}, {\color{blue}x_2}\}, \{{\color{red}x_1}, {\color{red}x_1}, {\color{blue}\bar{x_2}}\}, \{{\color{red}\bar{x_1}}, {\color{red}\bar{x_1}}, {\color{blue}x_2}\}, \{{\color{red}\bar{x_1}}, {\color{red}\bar{x_1}}, {\color{blue}\bar{x_2}}\}\}$$

If you can fix this, than you are half way to a solution, if the coloring ensures x_i and $\bar{x}_i$ have opposite colors, then colorable $\implies$ satisfiable.

There will still be satisfiable formula that are not colorable though. Consider for instance $\{x_1 \vee x_1 \vee x_1\}$. It is not colorable, though it is satisfiable.

Think about ways to modify (add, subtract, split, merge, etc.) the or clauses without changing the meaning to handle this situation.