Today’s Topics

● Review: 3-SAT is NP-Complete
● Definition: PSPACE, NPSPACE
● Review: TQBF $\in$ PSPACE
● Review: LADDER $\in$ NPSPACE
● Review: LADDER $\in$ PSPACE
● 3-SAT $\leq_p$ NEQ-SAT
● 3-SAT $\leq_p$ DOMINATING-SET
Review: 3-SAT is NP-Complete

Assume: \( \phi = ((a \land b) \lor c) \land (\overline{a} \lor b) \)

Logical equivalence: \((A \rightarrow B)\) and \((\overline{A} \lor B)\) and \((\overline{A} \land B)\) and \((A \lor \overline{B})\)

\[ \phi’ = ((a \land b) \rightarrow z_1) \land ((\overline{a} \land b) \rightarrow \overline{z_1}) \land ((a \land \overline{b}) \rightarrow \overline{z_1}) \land ((\overline{a} \land \overline{b}) \rightarrow \overline{z_1}) \]
\[ \land ((z_1 \land c) \rightarrow z_2) \land ((\overline{z_1} \land c) \rightarrow z_2) \land ((z_1 \land \overline{c}) \rightarrow z_2) \land ((\overline{z_1} \land \overline{c}) \rightarrow \overline{z_2}) \]
\: repeat for each \(z_i\)
\[ \land (z_4) \]
Definition: PSPACE

$\text{SPACE}(f(n)) = \{ B \mid \text{some deterministic 1-tape TM decides } B \text{ in space } O(f(n)) \}$

So:

$\text{PSPACE} = \bigcup_k \text{SPACE}(n^k)$
**Definition: NPSPACE**

\[ \text{NSPACE}(f(n)) = \{ B \mid \text{some nondet. 1-tape TM decides } B \text{ in space } O(f(n)) \} \]

So:

\[ \text{NPSPACE} = \bigcup_k \text{NSPACE}(n^k) \]
Definition: TQBF (True Qualified Boolean Formula) is a Boolean formula where every variable has either an exists (\(\exists\)) or forall (\(\forall\)) qualifier.

ex) \(\varphi = \forall a \forall b \exists c. (a \lor c) \land ((b \land c) \lor \neg a)\)

See if formula is satisfiable?
Review: TQBF ∈ PSPACE

Proof: “On input $<\varphi>$

1. If $\varphi$ has no qualifiers, has no variables. So either $\varphi=$True or $\varphi=$False. Output accordingly
2. If $\varphi = \exists x. \psi$, then evaluate $\psi$ with $x$ set to True/False. Accept if either evaluates to True. Reject if not
3. If $\varphi = \forall x. \psi$, then evaluate $\psi$ with $x$ set to True/False. Accept if both evaluates to True. Reject if not”

- On every recursive level, space used is constant. --- Only has to remember setting for $x$
- The number of recursive levels is linear. --- Each level removes one variable and there are a linear number of variables.
Review: LADDER

A LADDER is a list of words such that:

- List starts with a word \textbf{U} and ending with target word \textbf{V}
- Every immediately adjacent pair of words differ in only one symbol

\[ \text{LADDER}_{\text{DFA}} = \{ <B,u,v> \mid B \text{ is a DFA and } L(B) \text{ contains a ladder } y_1,y_2,...,y_k \text{ where } y_1=\text{u and } y_k=\text{v} \} \]
Review: \textit{LADDER}_{\text{DFA}} \subseteq \text{NPSPACE}

Note:

- Cannot store sequence of guesses strings
- Cannot run indefinitely

Proof: “On input \langle B, u, v \rangle"

1. Let \( y = u \) and let \( m = |u| \)
2. Repeat at most \( t \) times where \( t = |\Sigma|^m \)
   a. Nondeterministically change one symbol of \( y \) at a time
   b. Reject if \( y \notin L(B) \)
   c. Accept if \( y = v \)
3. Reject if exceeds \( t \) steps”

Space usage: linear relative to the input
B/c only have to store one word at a time.
So, we are using polynomial space with nondeterminism, hence NPSPACE
Review: LADDER\textsubscript{DFA} $\in$ PSPACE

Idea: Use recursion and implicit memoization

Proof: “On input $<B,u,v,t>$

Let $|m| = |u| = |v|$

1. For $t=1$, accept if $u,v \in L(B)$ and differ by at most 1 character

2. For $t>1$, repeat for each $w$ of length $|u|$  
   a. Recursively test $u \rightarrow t/2 w$, $w \rightarrow t/2 v$  
   b. Accept if both accept

3. Reject if all fail”

Test $<B,u,v>$ by running $<B,u,v,t>$ where $t=|\Sigma|^m$
Review: \text{LADDER}_{DFA} \in \text{PSPACE} \ (\text{Space Analysis})

Recall: “On input \langle B, u, v, t \rangle \quad \text{Let } |m| = |u| = |v|:

1. For \( t=1 \), accept if \( u, v \in L(B) \) and differ by at most 1 character.
2. For \( t>1 \), repeat for each \( w \) of length \( |u| \)
   a. Recursively test \( u \rightarrow_{t/2} w, w \rightarrow_{t/2} v \)
   b. Accept if both accept
3. Reject if all fail”

Test \langle B, u, v \rangle \text{ by running } \langle B, u, v, t \rangle \text{ where } t = |\Sigma|^m

Space usage per recursive level is linear relative to input: \( O(m) \)
Number of recursive levels: \( \log(t) = \log(|\Sigma|^m) = O(m) \)

Total space usage: \( O(m) \times O(m) = O(m^2) \) which is polynomial relative to the input
Definition: NEQ-SAT

Let $\varphi$ be a 3cnf-formula and

The NEQ-SAT problem states that every clause in $\varphi$ has literals with unequal truth values. In other words, at least one True and one False value.

Firstly: NEQ-SAT $\in$ NP
3-SAT $\leq_p$ NEQ-SAT (cont.)

Part a)

Show that the negation of any satisfying NEQ-SAT formula is also a satisfying NEQ-SAT formula
3-SAT \leq_p \text{NEQ-SAT (cont.)}

Part b)

The reduction:

\[(y_1 \lor y_2 \lor y_3) \leftrightarrow (y_1 \lor y_2 \lor z_i) \land (\lnot z_i \lor y_3 \lor b)\]

where \(z_i\) is a new variable for each clause \(c_i\), and \(b\) is a single additional new variable.

The reduction:

\[(y_1 \lor y_2 \lor y_3) \rightarrow (y_1 \lor y_2 \lor z_i) \land (\lnot z_i \lor y_3 \lor b)\]

\[(y_1 \lor y_2 \lor z_i) \land (\lnot z_i \lor y_3 \lor b) \rightarrow (y_1 \lor y_2 \lor y_3)\]