CSCI 1010
Models of Computation

Lecture 10
First NP-Complete Language
Overview

• Definitions of \textbf{NP} and \textbf{NP}-complete restated
• CIRCUIT SAT revisited
• Simulating NTM by a circuit
• Simulating the tape and control units
• Reduction of recognition of language in \textbf{NP} to computation by a circuit.
• Proof that this reduction can be done polynomial time.
• CIRCUIT SAT is \textbf{NP}-complete.
Polynomial-Time Reduction

• Definition: A polynomial-time reduction (P-time) from language $L_1 \subseteq \Gamma^*$ to language $L_2 \subseteq \Sigma^*$ is a reduction $f : \Gamma^* \rightarrow \Sigma^*$ $(x \in L_1 \iff f(x) \in L_2)$ computable by a DTM in time polynomial in the length of its input. ($f$ P-time translates $L_1$ to $L_2$)

The Recognizer for $L_1$ invokes the recognizer $R$ for $L_2$. 

© John E. Savage
NP-Complete Language

• A language \( L \subseteq \Gamma^* \) is **NP-complete** if
  1. it is in **NP** and
  2. for every language \( L_0 \subseteq \Gamma^* \) in **NP**, there is a P-time reduction \( f_0 \) from \( L_0 \) to \( L \).

• If only 2. holds, we say \( L \) is **NP-hard**.

Recognizer for \( L_0 \) in **NP** invokes recognizer \( R \) for \( L \).

Recognizer \( R \) for \( L \)
Is $P = NP$?

• Because an $NP$-complete language is in $NP$, it can be $P$-time reduced to another such language.

• Thus, an $NP$-complete language is a “hardest” language in $NP$ within polynomial bounds.

• If one $NP$-complete language recognizer requires exponential time, all do and $P \neq NP$.
  – Recall definition of a reduction.

• If one $NP$-complete language is in $P$, all are in $P$. That is, $P = NP$. 
Circuit Satisfiability

• A one-output circuit is **satisfiable** if its inputs can be chosen so that the output is 1.

• We generalize as follows:
  – A circuit is **satisfiable** if for fixed values of its *deterministic* inputs its *nondeterministic* inputs can be chosen so that the output is 1.
  – Previously all inputs were nondeterministic.

• CIRCUIT SAT is the set of satisfiable circuits.
A First \textbf{NP}-Complete Problem

- To show that language $L$ is \textbf{NP}-complete we must show that it is in \textbf{NP} and every language in \textbf{NP} can be reduced to $L$ in P-time.

- Our first \textbf{NP}-complete language is CIRCUIT SAT.
A First **NP**-Complete Problem

- A language $L$ in **NP** is specified by giving a polynomial $p(n)$ and an NTM $M_L$ such that $M_L$ non-deterministically recognizes $L$ in time $p(n)$ where $n$ is the length of the input string.

- We now give a P-time algorithm ALG that given an $L_0$ in **NP** produces a circuit satisfiable on input $w$ iff $w \in L_0$. The “Yes” circuit instances of these circuits are in CIRCUIT SAT.
Simulating NTM with a Circuit

• In Lecture 7 we simulated a DTM with a circuit!
• We now simulate NTM tape & control units with circuits.
• The combined circuit is satisfiable iff on input $w$ and with **proper choice inputs** the NTM can accept the input $w$. 
Simulation of P-time NTM

- NTM executes \( p(n) \) steps on inputs of length \( n \).
- Control unit is an NFSM
- \( m = p(n) + 1 \) tape cells are FSMs
- To simulate by circuits unwind each cell loop.
Recall - Simulating a DFSM by a Circuit

- FSM computes $f^{(T)}$ in $T$ cycles. **Unwind the loop!**
Simulating an NFSM by a Circuit
Modeling Tape Cells

- **Tape state** at time t:
  \[ A_t = (a_{0,t}, a_{1,t}, \ldots, a_{m-1,t}) \]
  - \( a_{i,t} \) is contents of ith cell at time t.
  - Each cell holds b bits, i.e.
    \[ a_{i,t} = (a_{0,i,t}, a_{1,i,t}, \ldots, a_{b-1,i,t}) \]

- **Head position at time t**:
  \[ s_t = (s_{0,t}, s_{1,t}, \ldots, s_{m-1,t}) \]
  - only one 1 in \( s_t \)
Encoding of Tape Parameters

- Head movement command: $h = (h_{+1}, h_0, h_{-1})$ – only one 1
- Input to memory cell: $w = (w_0, w_1, ..., w_{b-1})$
- Output of $j$th cell, time $t$: $v_{j,t} = (v_{0,j,t}, v_{1,j,t}, ..., v_{b-1,j,t})$
  $v_{j,t} = 0$ b-tuple if head not over $j$th cell at time $t$. 

© John E. Savage
• Circuit $C_{j,t}$ receives input from adjacent cells $C_{j,t-1}, C_{j-1,t-1}, C_{j+1,t-1}$ as well as $w_t$, $s_{t-1}$ and $h_t$. It produces $a_{j,t}$, $v_{j,t}$, and $s_t$. 
Details of Cell Circuit Design

- $C_1$ changes cell contents if the head is over the cell.
- $C_2$ moves head if head over cell
- $C_3$ outputs cell contents if head over cell else 0.
- $O(S)$ gates used in all cells, $S = mb$. 
Output of Tape Unit

• Output of tape unit is vector OR of cell outputs. O(S) gates used here.
• All but one cell output is vector 0.
Circuit Simulating T-step NTM

- One column/time step
- Inputs to first column are initial values.
- $c_j$ is choice input to control unit on jth step.
Size of Circuit Simulating T-step NTM

- Tape Unit: \(O(S)\) gates per time step, \(O(ST)\) total
- Control Unit: \(O(1)\) gates/time step, \(O(T)\) total
Translation to CIRCUIT SAT

• Let L be in \textbf{NP}. There is an NTM \( M_L \) that can accept every \( w \in L \) in \( p(n) \) time steps and accepts no \( w \notin L \).

• \( M_L \) description is used to construct a circuit with choice inputs that simulates \( T = p(n) \) steps of \( M_L \) on input \( w \), \( n = |w| \).

• Build circuit with output = 1 if \( q_T \) is an accepting halt state and 0 otherwise. Circuit size is \( O(1) \).

• Circuit is satisfied if and only if \( w \in L \).
Final Steps

• Given an L in NP that runs in $p(n)$ steps on accepted inputs $w$, $n = |w|$, we have shown that a circuit exists that is satisfiable iff $w \in L$.

• The circuit has size $O(ST)$. Since $S \leq T$, the size is $O(T^2) = O(p^2(n))$, a polynomial in $n$.

• Is there a program ALG running in polynomial time that prints out the circuit description?
P-Time Program to Print Circuit

A program that writes circuit simulating $T$ steps of NTM.

First $n$ inputs specified.

Set rest to blank.

Write one copy of CU/time step

One copy of $m$-cell tape/step.

One $m$-input OR/step.

Running time = $O(p(n))$ on RAM
CIRCUIT SAT is **NP**-Complete

- We have shown that recognition of an arbitrary language $L$ in **NP** can be reduced in polynomial time to recognition of CIRCUIT SAT.

- Since CIRCUIT SAT is in **NP**, as shown earlier, we conclude that CIRCUIT SAT is **NP**-complete.

- Later we use this result to prove that other languages are **NP**-complete.
Review

• Definitions of $\text{NP}$ and $\text{NP}$-complete restated
• CIRCUIT SAT revisited
• Simulating NTM by a circuit
• Simulating the tape and control units
• Reduction of recognition of language in $\text{NP}$ to computation by a circuit.
• Proof that this reduction can be done polynomial time.
• CIRCUIT SAT is $\text{NP}$-complete.