Verifiability (reminder)

More Poly-Time Reductions

NP completeness

Cook-Levin Thm.: SAT is NP Complete

Bounded Halting is NPC

Beyond NP

Coping with computational intractability

Concluding Remarks

Sipser, Chapter 7, Sections 7.3, 7.4, 7.5, 10.1, 10.2
Verifiability (Reminder)

A verifier for a language $\mathcal{A}$ is an algorithm $\mathcal{V}$ where

$$\mathcal{A} = \{w \mid \mathcal{V} \text{ accepts } \langle w, c \rangle \text{ for some string } c\}$$

- The verifier uses the additional information $c$ to verify $w \in \mathcal{A}$.
- We measure verifier run time by length of $w$.
- The string $c$ is called a certificate (or proof) for $w$ if $\mathcal{V}$ accepts $\langle w, c \rangle$.
- A polynomial verifier runs in polynomial time in $|w|$ (so $|c| \leq |w|^{O(1)}$).
- A language $\mathcal{A}$ is polynomially verifiable if it has a polynomial verifier.
Verifiability

Not all problems are polynomially verifiable.

There is no known way to verify HAMPATH in polynomial time.

In fact, we will see many examples where $L$ is polynomially verifiable, but its complement, $\overline{L}$, is not known to be polynomially verifiable.
NP and Verifiability (Reminder)

Theorem: A language is in NP if and only if it has a polynomial time verifier.

Proof – Intuition:

- NTM simulates verifier by guessing the certificate.
- Verifier simulates NTM by using accepting branch as certificate.
Traveling Salesperson Problem (TSP)

Parameters:
- set of cities $C$
- set of inter-city distances $D$
- goal $k$
Traveling Salesman

Define \( \text{TRAVELING-SALESMAN} = \{ \langle C, D, k \rangle \mid (C, D) \text{ has a TS tour of total distance } \leq k \} \)

Remark: Can consider two versions – undirected and directed.

Recall \( \text{HAMCIRCUIT} = \{ \langle G \rangle \mid G \text{ has Hamiltonian circuit} \} \)

Theorem: Directed \( \text{HAMCIRCUIT} \) is polynomial-time reducible to directed \( \text{TRAVELING-SALESMAN} \),

\[ \text{HAMCIRCUIT} \leq_P \text{TRAVELING-SALESMAN} \]
HAMCIRCUIT $\leq_P$ TSP

The reduction: Given a directed graph $G = (V, E)$ we construct a directed traveling salesman instance.

- The cities are identical to the nodes of the original graph, $C = V$.
- The distance of going from $v_1$ to $v_2$ is 1 if $(v_1, v_2) \in E$, and 2 otherwise.
- The bound on the total distance of a tour is $k = |V|$.
HAMCIRCUIT $\leq_P$ TSP

Validity of Reduction

$\implies$ Suppose $G$ has a Hamiltonian circuit. The distance assigned by the reduction to all edges in this circuit is 1. Thus in $(C, D)$ there is a traveling salesman tour of total distance $|V| = k$, namely $(C, D, k) \in \text{TRAVELING-SALESMAN}$.

$\impliedby$ Suppose $(C, D)$ has a traveling salesman tour of total distance $|V| = k$. Tour cannot contain any edge of distance 2. Therefore it gives a Hamiltonian circuit in $G$.

Efficiency: Reduction in quadratic time (filling up distances for all edges of the complete graph). ♣
The Language SAT (reminder)

Definition: A Boolean formula is in conjunctive normal form (CNF) if it consists of clauses, connected with $\land$s. Each clause is a disjunction ($\lor$s) of literals.

For example $(x_1 \lor \overline{x_2} \lor \overline{x_3} \lor x_4) \land (x_3 \lor \overline{x_5} \lor x_6) \land (x_3 \lor \overline{x_6})$

Definition: $\text{SAT} = \{\langle \phi \rangle \mid \phi \text{ is satisfiable CNF formula}\}$
3SAT (reminder)

**Definition:** A Boolean formula is in 3CNF form if it is a CNF formula, and each clause has at most three literals.

\[(x_1 \lor \neg x_2 \lor \neg x_3) \land (x_3 \lor \neg x_5 \lor x_6) \land (x_3 \lor \neg x_6 \lor x_4)\]

Define

\[3\text{SAT} = \{\langle \phi \rangle \mid \phi \text{ is satisfiable 3CNF formula}\}\]

Clearly, if \(\phi\) is a satisfiable 3CNF formula, then for any satisfying assignment of \(\phi\), every clause must contain at least one literal assigned 1.
Recall

**SAT** = \{⟨φ⟩ | φ is a satisfiable CNF formula\}

**3SAT** = \{⟨φ⟩ | φ is satisfiable 3CNF formula\}

We will show a poly time reduction, which maps CNF formulae to 3CNF ones “clause by clause”. A clause with $ℓ$ literals is mapped to $ℓ$ clauses, built on the original literals together with $ℓ − 1$ new ones.

For example:

$$(x_1 \lor \overline{x}_2 \lor \overline{x}_3 \lor x_4 \lor x_8)$$

$$\mapsto$$

$$(x_1 \lor y_1) \land (\overline{y}_1 \lor \overline{x}_2 \lor y_2) \land (\overline{y}_2 \lor \overline{x}_3 \lor y_3) \land (\overline{y}_3 \lor x_4 \lor y_4) \land (\overline{y}_4 \lor x_8)$$
SAT $\leq_P$ 3SAT

Consider mapping $\phi \mapsto \phi_3$, e.g. $(x_1 \lor \overline{x_2} \lor \overline{x_3} \lor x_4 \lor x_8) \mapsto (x_1 \lor y_1) \land (\overline{y_1} \lor \overline{x_2} \lor y_2) \land (\overline{y_2} \lor \overline{x_3} \lor y_3) \land (\overline{y_3} \lor x_4 \lor y_4) \land (\overline{y_4} \lor x_8)$

Claim: $\phi$ has a satisfying assignment iff $\phi_3$ does.

Proof sketch: $\Leftarrow$ An assignment satisfying $\phi_3$ cannot “rely” on new literals alone – at least one original literal must be satisfied.

$\Rightarrow$ An assignment satisfying $\phi$ makes at least one literal per clause happy. In the “$\phi_3$ clause” of this literal the new variable is under no constraints. This enables propagation to a satisfying assignment that “relies” on new vars alone in rest of $\phi_3$ clauses.

This establishes validity of the reduction. Since it is in polynomial time (why?), we get SAT $\leq_P$ 3SAT. ♣.
Definition

A language $B$ is **NP-complete** if it satisfies

- $B \in NP$, and
- Every $A$ in NP is polynomial time reducible to $B$
Strategy

- Once we have one “structured” NP-complete problem, we can generate more by poly-time reduction.
- Getting the first one requires some work.
- This is what Steve Cook (then in Berkeley, now in Toronto) and Leonid Levin (then in Moscow, now in Boston) did in the early seventies.
Theorem: SAT is NP complete.

Prf.: Membership ($SAT \in NP$) is easy, using satisfying assignment as certificate.

- Must show that every NP problem reduces to SAT in poly-time.
- Idea: Suppose $L \in \mathcal{NP}$, and $M$ is an NTM that accepts $L$.
- On input $w$ of length $n$, $M$ runs in time $t(n) = n^c$.
- We consider the $n^c$-by-$n^c$ tableau that describes the computation of $M$ on input $w$. 
The Tableau

<table>
<thead>
<tr>
<th></th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>...</th>
<th>t(n)</th>
</tr>
</thead>
<tbody>
<tr>
<td>q₀</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td></td>
</tr>
</tbody>
</table>

*cell[1,1]*

*cell[1,t(n)]*
Row 1 in tableau represents initial configuration of $M$ on input $w$.

Row $i$ in tableau represents $i$-th configuration in a computation of $M$ on input $w$. 
A Formula Simulating the Tableau

- We construct a Boolean CNF formula $\phi_w$ that “mimics” the tableau.

- Given the string $w$ of length $n$, it takes $O(n^{2c})$ steps to construct $\phi_w$.

- The following property holds:
  \[ \phi_w \in SAT \text{ iff } M \text{ accepts } w. \]

- So the mapping $w \mapsto \phi_w$ is a poly time reduction from $\mathcal{L}$ to $SAT$, establishing $\mathcal{L} \leq_P SAT$.

- We still got a few small details to take care of...
Details of Formula (Partial List)

- We construct a Boolean CNF formula $\phi_w$ that “mimics” the tableau:

  - $\phi_w$ uses Boolean variables of three types.
  - $b_{i,j,\sigma}$ is true iff the $j$-th cell in $i$-th configuration contains the letter $\sigma \in \Gamma$.
  - $s_{i,q}$ is true iff in $i$-th configuration, $M$ is in state $q \in Q$.
  - $h_{i,j}$ is true iff in $i$-th configuration, $M$ has its head in cell $j$ on tape.

- The formula $\phi_w$ consists of four parts:
  \[
  \phi_w = \phi_{\text{unique}}(M) \land \phi_{\text{start}}(w) \land \phi_{\text{accept}}(M) \land \\
  \phi_{\text{compute}}(M)
  \]
Details of Formula (cont.)

- \( \phi_{\text{unique}}(M) \) guarantees that the variables encode legal configurations. For example, at most one of \( b_{i,j,0} \) and \( b_{i,j,1} \) is true.

- \( \phi_{\text{start}}(w) \) guarantees that the variables corresponding to the first row \( (i = 1) \) encode the initial configuration of \( M \) on \( w \).

- \( \phi_{\text{accept}}(M) \) guarantees that \( M \) reached an accepting configuration.

- \( \phi_{\text{compute}}(M) \) guarantees that the configuration described by the \( i+1 \)-st row is a legal succession of the configuration described by the \( i \)-th row.
Details of Formula (cont.)

- $\phi_{\text{compute}}(M)$ is the “heart” of $\phi_w$. To construct it, employ locality of computations.

- To determine contents of tableau entry $(i, j)$ (cell $j$ in configuration $i$), only the contents of three tableau entries (from configuration $i - 1$), $(i - 1, j - 1), (i - 1, j), (i - 1, j + 1)$, and $M$’s table, are needed.

- If head not in area, nothing changes. And if it is, changes are local and determined using $M$.

![Tableau Diagram]

Slides modified by Benny Chor, based on original slides by Maurice Herlihy, Brown University.
The Tableau in Perspective

$\begin{array}{cccccc}
q_0 & 0 & 0 & 1 & 0 & \ldots & t(n) \\
\end{array}$

$cell[1,1]$  

$cell[1,t(n)]$
Correctness of Reduction

All four components of $\phi_w$ can be put in CNF, so $\phi_w$ itself ($\wedge$ of the four) is also in CNF.

The transformation $w \mapsto \phi_w$ is computable in time $O(n^{2c})$.

An assignment satisfying $\phi_{\text{unique}}(M) \wedge \phi_{\text{start}}(w) \wedge \phi_{\text{compute}}(M)$ corresponds to a valid computation of $M$ on $w$.

An assignment satisfying, in addition $\phi_{\text{accept}}(M)$, corresponds to an accepting computation of $M$ on $w$.

Therefore $M$ accepts $w$ iff $\phi_w \in SAT$.

For complete details, consult Sipser or take the Complexity course.
3SAT – Cousins and Cambrians

We now know that $\text{SAT} \leq_P \text{3SAT}$. Since $\text{SAT}$ is NP-complete and $\text{3SAT} \in \text{NP}$, this proves that $\text{3SAT}$ is itself NP-complete.

What about the $\text{3SAT} \leq_P \text{SAT}$ direction?

We now want to examine what happens if we further reduce the number of literals per clause in CNF formulae.

**Definition:** A Boolean formula is in 2CNF if it is a CNF formula, and all terms have at most two literals. For example

$$((x_1 \lor \overline{x}_2) \land (\overline{x}_5 \lor x_6) \land (\overline{x}_6 \lor \overline{x}_4))$$
3SAT – Cousins and Cambrians

**Definition:**

\[ 2\text{SAT} = \{ \langle \phi \rangle \mid \phi \text{ is satisfiable 2CNF formula} \} \]

- Betting time: Is \text{2SAT} NP-complete? Is it in \text{P}? Or maybe we do not know? . . .

- Well, turns out \text{2SAT} is in \text{P}. For details, though, you’ll have to refer to the algorithms course, or work it out yourselves.
Saw a Few Reductions

- \( \text{SAT} \leq_P \text{3SAT} \) \( \Rightarrow \) 3SAT is NP-complete
- \( \text{3SAT} \leq_P \text{Clique} \) \( \Rightarrow \) Clique is NP-complete
- \( \text{Clique} \leq_P \text{Independent Set} \) \( \Rightarrow \) IS is NP-complete
- \( \text{Clique} \leq_P \text{Vertex Cover} \) \( \Rightarrow \) VC is NP-complete
- \( \text{HamPath} \leq_P \text{HamCircuit} \)
- \( \text{HamCircuit} \leq_P \text{TSP} \)

Will now show 3SAT \( \leq_P \) HamPath, thus establishing NP-completeness of HamPath, HamCircuit, and TSP.
Hamiltonian Path

For any 3CNF formula $\phi$,

- we construct a graph $G$
- with vertices $s$ and $t$
- such that $\phi$ is satisfiable iff there is a Hamiltonian path from $s$ to $t$. 
Hamiltonian Path

Here is a 3CNF formula $\phi$:

$$(a_1 \lor b_1 \lor c_1) \land (a_2 \lor b_2 \lor c_2) \land \cdots (a_k \lor b_k \lor c_k) \land$$

where

- each $a_i, b_i, c_i$ is $x_i$ or $\overline{x_i}$
- the $\ell$ clauses are $C_1, \ldots, C_\ell$
- the $k$ variables are $x_1, \ldots, x_k$. 
HamPath: NP Completeness Proof

Turn to a separate, postscript presentation
Integer Programming (IP)

- **Definition:** A linear inequality has the form
  \[ a_1 x_1 + a_2 x_2 + \ldots + a_n x_n \leq b \]
  where \( a_1, \ldots, a_n, b \) are real numbers, and \( x_1, \ldots, x_n \) are real variables.

- **The Integer Programming (IP) problem:**
  - **Input:** A set of \( m \) linear inequalities with integer coefficients \( (a_i, b) \) in \( n \) variables \( x_1, x_2, \ldots, x_n \).
  - The language **IP** is the collection of all systems of linear inequalities that have a solution where all \( x_i \) are integers.
Consider the following system of linear inequalities

\[ y \leq 2x \quad \text{green line} \]
\[ -2x + 1 \leq y \quad \text{red line} \]
\[ 4x - 2 \leq y \quad \text{purple line} \]
\[ 0 \leq x \leq 1 \]
\[ 0 \leq y \leq 2 \]
This set does have a unique solution: the right hand corner of the solid triangle, $(1, 2)$.

But if we change the constraint on $y$ to $0 \leq y \leq 1$, then we’d have no solution with integer coordinates, even though there are many solutions with rational, or real, coordinates.

Will now show IP is NP complete.

Membership in NP easy (why?)
\[ \text{SAT} \leq_p \text{IP} \]

\[ \text{SAT} = \{ \langle \phi \rangle \mid \phi \text{ is a satisfiable CNF formula} \} \]

For example, the following formula is in SAT:
\[
(x_1 \lor \overline{x}_2 \lor \overline{x}_3 \lor x_4) \land (x_3 \lor \overline{x}_5 \lor x_6) \land (x_3 \lor \overline{x}_6)
\]

Let \( \phi \) be a CNF formula with \( m \) clauses and \( n \) variables \( x_1, \ldots, x_n \) (either \( x_i, \overline{x}_i \), or both, can appear in \( \phi \)).

Will reduce \( \phi \) to an IP instance with \( 2n \) variables \( x_1, y_1, \ldots, x_n, y_n \) and \( m + 2n \) linear inequalities, and \( n \) linear equalities (why ?).
SAT $\leq_P$ IP

- Each $x_i$ in $\varphi$ corresponds to the variable $x_i$ in IP.
- Each $\overline{x_i}$ in $\varphi$ corresponds to the variable $y_i$ in IP.

For each $i$, we add the inequalities $x_i \geq 0$, $y_i \geq 0$, and the equality $x_i + y_i = 1$
(what do these three express?)

For each clause $k$, we add the inequality
$$\sum_{z_j \in \text{Clause}_k} z_j \geq 1$$
(what does this inequality express?)

For example, $(x_1 \lor \overline{x_2} \lor \overline{x_3} \lor x_4)$ is translated to
$$x_1 + y_2 + y_3 + x_4 \geq 1.$$
SAT \leq_p IP: Example

\varphi = (x_1 \lor \overline{x_2} \lor \overline{x_3} \lor x_4) \land (x_3 \lor \overline{x_5} \lor x_6) \land (x_3 \lor \overline{x_6})

translates to

\begin{align*}
x_1 + y_2 + y_3 + x_4 & \geq 1 \\
x_3 + y_5 + x_6 & \geq 1 \\
x_3 + x_6 & \geq 1 \\
x_1 & \geq 0, \ y_1 \geq 0, \ x_1 + y_1 = 1 \\
x_2 & \geq 0, \ y_2 \geq 0, \ x_2 + y_2 = 1 \\
x_3 & \geq 0, \ y_3 \geq 0, \ x_3 + y_3 = 1 \\
x_4 & \geq 0, \ y_4 \geq 0, \ x_4 + y_4 = 1 \\
x_5 & \geq 0, \ y_5 \geq 0, \ x_5 + y_5 = 1 \\
x_6 & \geq 0, \ y_6 \geq 0, \ x_6 + y_6 = 1
\end{align*}
SAT $\leq_P$ IP: Validity (sketch)

Should show
(a) Reduction $g$ is poly-time computable
(b) $\varphi \in \text{SAT} \implies g(\varphi) \in \text{IP}$
(c) $g(\varphi) \in \text{IP} \implies \varphi \in \text{SAT}.$

- Poly time: easy (verify details!).
- Suppose $\varphi \in \text{SAT}.$ Take a satisfying assignment.
  If $x_i = 1$ assign $x_i = 1, y_i = 0$ in IP.
  If $x_i = 0$ assign $x_i = 0, y_i = 1$ in IP.
- So "sanity check" constraints satisfied. "Clause constraints" are satisfied due to at least one literal satisfied in each clause., implying $g(\varphi) \in \text{IP}.$
- $g(\varphi) \in \text{IP} \implies \varphi \in \text{SAT}$ is similar. ♣
Bounded $A_{TM}^A$

- Bounded $A_{TM}^A$: Given encoding $\langle M \rangle$ of non-deterministic TM, an input $w$, time bound $1^k$ in unary, does $M$ have an accepting computation of $w$ in $k$ steps or less?

- Bounded $A_{TM}^A$ is NP complete, via a “generic” reduction.

- Finding the reduction is easy (check!).

- Proving Bounded $A_{TM}^A$ is in NP seems less obvious.
Bounded $A^1_{TM}$ in NP

An instance of the problem has the form $\langle M, w, 1^k \rangle$. The universal NTM, $U$, decides this language:

- $U$ does not write anything on input tape.
- Copies $q_0w$ to second tape.
- Simulates $M$ step by step, keeping its configuration on second tape.
- Sets up a unary counter on third tape, initialize to zero and increments it for each simulated step of $M$.
- If counter reaches $k + 1$, $U$ rejects.
- How many steps of $U$ does it take to simulate $k$ steps of $M$?
Bounded $A_{\text{TM}}$ in NP

- How many steps $U$ takes to simulate $k$ steps of $M$?
- To simulate one step of $M$, NTM $U$ has to find an entry in $M$’s transition function that matches current simulated state and letter under head.
- This requires scanning all of $M$’s transition function (on input tape), which takes length of $\langle M \rangle$ steps of $U$.
- So to simulate $k$ steps of $M$, NTM $U$ takes $k \cdot |\langle M \rangle|$ steps.
- Denote $n = |\langle M, w, 1^k \rangle|$. What is $k \cdot |\langle M \rangle|$ in terms of $n$?
- $k \cdot |\langle M \rangle| = \theta(n^2)$, so $A_{\text{TM}} \in NTIME(n^2)$.
- **Question**: What would the NTIME complexity be if $k$ would be encoded in binary?
Yet More Intractable Problems

- Subgraph isomorphism is NP complete.
- Graph isomorphism is in NP, seems not to be in P, but we got many good reasons to believe it is not NP complete.
Chains of Reductions: NPC Problems

- SAT
- IntegerProg
- 3SAT
- Clique
- 3Color
- HamPath
- IndepSet
- Scheduling
- HamCircuit
- VertexCover
- TRAVELING-SALESMAN
- SetCover
- Knapsack

Slides modified by Benny Chor, based on original slides by Maurice Herlihy, Brown University.
NP Hardness

A language $B$ is NP-hard if it satisfies

- Every $A$ in NP is polynomial time reducible to $B$

We do not require $B \in NP$ (membership).

Example: The language

$$\{⟨Φ_1, Φ_2⟩ | Φ_1 \in SAT, Φ_2 \notin SAT\}$$

is NP-hard but apparently not NP-complete (why?).
Coping with NP-Completeness

- **Approximation** algorithms for hard optimization problems.
- **Randomized** (coin flipping) algorithms.
- **Fixed parameter** algorithms.
- **Heuristics**.

These stand in the forefront of current algorithmic research, and could easily fill up three or four advanced courses.

(figures from [http://wwwbrauer.in.tum.de/gruppen/theorie/hard/vc1.png](http://wwwbrauer.in.tum.de/gruppen/theorie/hard/vc1.png))
Coin Fipping TMs
The Dreaded Exam

- All material covered in class and recitations, from three parts of course (except the one class that was cancelled on Wednesday due to strike).
- Both multiple choice ("closed") and "open" questions.
- You can bring two double sided A4 pages (normal size).

- Piece of cake.
You have brains in your head.
You have feet in your shoes.
You can steer yourself
any direction you choose.
You’re on your own. And you know what you know.
And YOU are the guy who’ll decide where to go.

Hebrew translation (not given here, due to \LaTeX{} limitations, but highly recommended) by Leah Naor.