More NP-completeness

Recall how to prove a problem $Z$ is NP-complete (after 1st proof):

1. show $Z \in \text{NP}$
2. show $X \leq_p Z$ for some known NP-complete $X$.

and recall our plan:

\[
\begin{align*}
\text{Circuit-SAT} & \leq_p \text{3-SAT} \leq_p \text{HAM.CYCLE} \leq_p \text{TSP} \\
\text{INDSET} & \leq_p \text{VERTEX COVER} \leq_p \text{SET COVER} \\
\text{SUBSET SUM} & \leq_p \text{today}
\end{align*}
\]
Subset Sum

Input: Numbers $w_1, \ldots, w_n$ and $W$  

$Q$: Is there a subset $S \subseteq \{1, \ldots, n\}$ st. $\sum_{i \in S} w_i = W$  

Summary:
- dyn. prog. alg. $O(n \cdot W)$ "pseudo-poly."
- branch and bound alg. $O(2^n)$

Thm. Subset Sum is NP-complete.

PF $(\subseteq \Pi)$  

$\exists$ 3-SAT $\leq_P$ Subset Sum  

Assume we have a poly. time alg. for Subset Sum.  

Give a poly. time alg. for 3-SAT.

We've seen how to turn 3-SAT into a packing problem (ind. set) and into a sequencing problem (ham. cycle) and now we must turn it into a number problem.

Idea: specify the bits of the numbers

Given 3-SAT formula $F$ with clauses $c_1, \ldots, c_m$ variables $x_1, \ldots, x_n$
Create a 0-1 matrix

\[
\begin{array}{cccc}
C_1 & C_2 & \cdots & C_m \\
\hline
x_1 & 1 & 0 & \\
-x_1 & 0 & 1 & \\
x_2 & 0 & 0 & \\
-x_2 & 1 & 0 & \\
x_3 & 1 & 0 & \\
-x_3 & 0 & 0 & \\
\vdots & \vdots & \vdots & \\
\end{array}
\]

\[\text{e.g.,}\]
\[C_1 = (x_1 \vee -x_2 \vee x_3)\]
\[C_2 = (-x_1 \vee x_4 \vee x_5)\]

\[M[x_i, C_j] = 1 \text{ if } x_i \text{ appears in } C_j\]
\[M[-x_i, C_j] = 1 \text{ if } -x_i \text{ appears in } C_j\]

We assume that no clause contains the same variable twice.

We will regard the rows as binary (or other base) numbers

\[C_1, C_2, C_3, \ldots\]

Target sum \[\geq 1\] \[\geq 1\] \[\geq 1\] \[\ldots\]

Issues:

1. we need some way to ensure we pick row \( x_i \) or row \( -x_i \) but not both

2. we need to handle target \( \geq 1 \)

What can the sum down a column be? 1 or 2 or 3

Add slack of 1 or 2
Finally:
\[ W = \text{interpret bottom row in base 10} \]
\[ S = \text{one number for each row, interpreting the row in base 10} \]
so we have \( 2n + 2m \) numbers of \( n \times m \) digits.
\text{Claim: poly-time \( (\text{and poly-size}) \)
Claim: F is satisfiable iff S has a subset with sum W.

Pf. \( \Rightarrow \) if \( x_i \) is True, pick row \( x_i \);
if \( x_i \) is False, pick row \( \neg x_i \).

Then column \( c_j \) adds to 1, 2, or 3.

Use slack rows \( s_j^1, s_j^2 \) to increase sum to 4.

\[ 1 + s_j^1 + s_j^2 = 4 \]
\[ 2 + s_j^2 = 4 \]
\[ 3 + s_j^1 = 4 \]

This gives a set of rows (i.e., elements of S) with sum W.

\( \Leftarrow \) Suppose S has subset S with sum W.

Note: any column sum is \( \leq 6 \) so no carries occur
and column sums really must give target digit.

Because \( x_i \) column sum is 1, we choose row \( x_i \)
\( x_i \) or row \( \neg x_i \) (not both) — set variable \( x_i \) accordingly.

Because column \( c_j \) sum is 4 and slacks sum to \( \leq 3 \), we must have chosen a literal to
satisfy the clause \( c_j \).

For NP-completeness
Note: all our reductions use the black-box subroutine
only once and return its YES/NO answer.
You should always use this stronger reduction.

OPEN: Are the 2 kinds of reduction equivalent inside NP?
The first NP-completeness proofs.

Circuit Satisfiability

\[(x_1 \land x_2) \lor (\neg x_1 \land \neg x_2)\]

i.e. \(x_1\) same as \(x_2\)

\(x_1 \equiv x_2\)

A circuit is a directed acyclic graph. Sources (no edge entering) are labelled with variables or 0 or 1 — these are inputs.

Sink (no edge leaving) — there is one sink — output.

Internal nodes

A circuit computes an output in an obvious way when values are given to input variables.

e.g. above \(x_1 = 0, x_2 = 1\) outputs 0

(compute values at internal nodes from sources to sink)
Circuit Satisfiability

Input: A circuit C.

Q: Is there an assignment of values to inputs s.t. output is 1? (Is C satisfiable?)

Thm Circuit SAT is NP-complete.

Def. 1 Circuit SAT ∈ NP - easy
(2) (High-level idea) We must prove
for every problem \( X \in \text{NP} \), \( X \leq_p \text{Circuit SAT} \)
i.e., for every problem \( X \in \text{NP} \), there is a poly-time
alg. to transform any input \( I \) for \( X \) into a circuit
\( C \) s.t. \( C \) is satisfiable \( \iff \) \( I \) is a \text{YES} input for \( X \).
(Thus a poly-time alg. for Circuit SAT yields a
poly-time alg. for \( X \)).

What can we use? Just that \( X \in \text{NP} \),
i.e., there is a poly-time verification alg. \( A \) for \( X \)
that takes 2 inputs \( I, R \) and outputs \text{YES}/\text{NO} s.t.
\( I \) is a \text{YES} input for \( X \) \iff
\( \exists R \text { size}(R) \leq \text{poly. in size}(I) \)
s.t. \( A(I, R) \) outputs \text{YES}

Idea: convert alg. \( A \) with known input \( I \) and
unknown input \( R \) to a circuit \( C \) with input
variables = bits of \( R \) s.t. \( C \) is satisfiable \iff
\( \exists R \text { s.t. } A(I, R) \) outputs \text{YES},

program alg. \( A \). Compile, assemble ..., 
at hardware level, this is implemented by \( \land, \lor, \neg \)
gates. We get a circuit.

Inputs to circuit: bits of \( I \) (known)
bits of \( R \) (variables)
Internal nodes of circuit – memory locations after each time step of alg. A.

Because size(R) is poly. and A runs in poly. time, the circuit has poly. size.

Is there an alg. to convert A, to c? Yes, compiler, assembler etc. and poly. time.