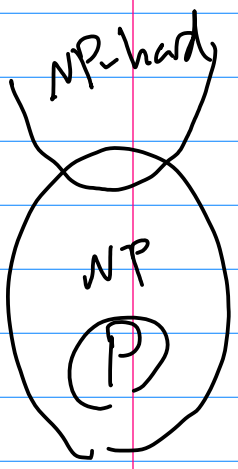
NP Hardness Reductions



P: $\exists$ polynomial time algorithm $A(x)$

NP: $\exists$ polynomial time <u>verifier</u> $A(x, y)$:

$\forall$ "true" x $\exists$ y s.t. $A(x, y) = YES$

$\forall$ "false" x $\nexists$ y s.t. $A(x, y) = YES$

Example: SAT
(satisfiability of Boolean Formula)

$$(x_1 \cup \bar{x}_2 \cup x_3 \cup x_4) \cap (x_5 \cup \overline{(x_6 \cap x_1)}) \cup \cdots$$

CNF-SAT: AND of ORs
"conjunctive normal form

$$(x_1 \cup \bar{x}_2 \cup x_3) \cap (x_1 \cup x_7) \cap (x_2 \cup \bar{x}_3 \cup x_8 \cup x_9) \cap \cdots$$

Cook's Theorem: CNF-SAT is NP-complete

(1) CNF-SAT $\in$ NP (easy)

For each input (formula),
"proof" is satisfying assignment
"verifier" checks the formula evaluates to True

(2) CNF-SAT is NP-hard

can reduce every NP problem q
to CNF-SAT
$\Rightarrow$ if CNF-SAT $\in$ P, then P = NP

$q \in NP \Rightarrow \exists$ verifier $A(x, y) =: A_x(y)$

x yes $\Leftrightarrow \exists y$ s.t. $A_x(y) =$ True

Idea: write A_x as CNF-SAT formula
see if $\exists y$ making it true

Turing Machine: tape, states, Location

Q_{tj} := 1 if machine in state j at time t
L_{tj} := 1 if " location j "

T_{tjk} := 1 if tape at position j & time t
is character k

but polynomial!

bunch ^ of rules for correct operation:

Machine in exactly 1 state at all times:

$$(Q_{t1} \cup Q_{t2} \cup \dots \cup Q_{ts}) \cap (\overline{Q_{t1}} \cup \overline{Q_{t2}}) \cap (\overline{Q_{s1}} \cup \overline{Q_{s3}}) \cap \dots$$

at least one — all pairs

Transition rule:

$$T_{tjk} \cap Q_{tl} \cap L_{tj}$$

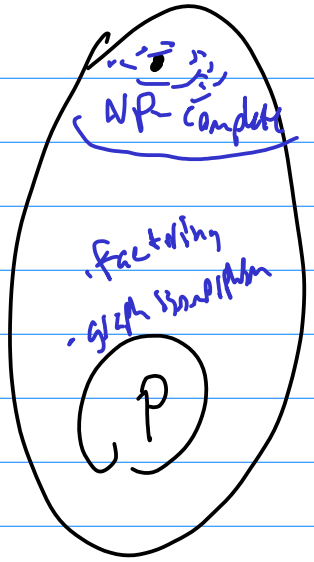
$$\Rightarrow T_{(t+1),j,f(k,l)}$$ ← what is written

$$Q_{t+1,\, g(k,l)}$$ ← new state

$$L_{t+1,\, j+h(k,l)}$$ ← move left/right

$(a,b,c) \Rightarrow d$ means $(a \cap b \cap c \cap \bar{d})$

$= (\bar{a} \cup \bar{b} \cup \bar{c} \cup d)$



Cook: CNF-SAT is NP-complete

Karp:

$\Rightarrow$ 3SAT is NP-complete

$\Rightarrow$ vertex cover
$\Rightarrow$ set covering
$\Rightarrow$ Steiner tree
$\Rightarrow$ ⋮

} 21 problems

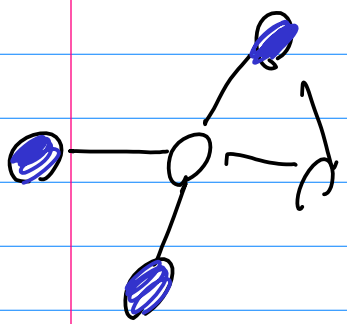
3-SAT: SAT w/ 3 vars/clause

$$(\ell_1 \vee \ell_2 \vee \bar{x}_3) \wedge (\bar{x}_1 \vee x_4 \vee x_5) \wedge (\ell_2 \vee \bar{\ell}_4 \vee \bar{x}_5)$$

$x_1, x_2, \bar{x}_3$ are "literals"

Independent Set (G):

Set $S \subseteq V$ s.t. no edge $(u,v) \in E$ w/ both $u, v \in S$



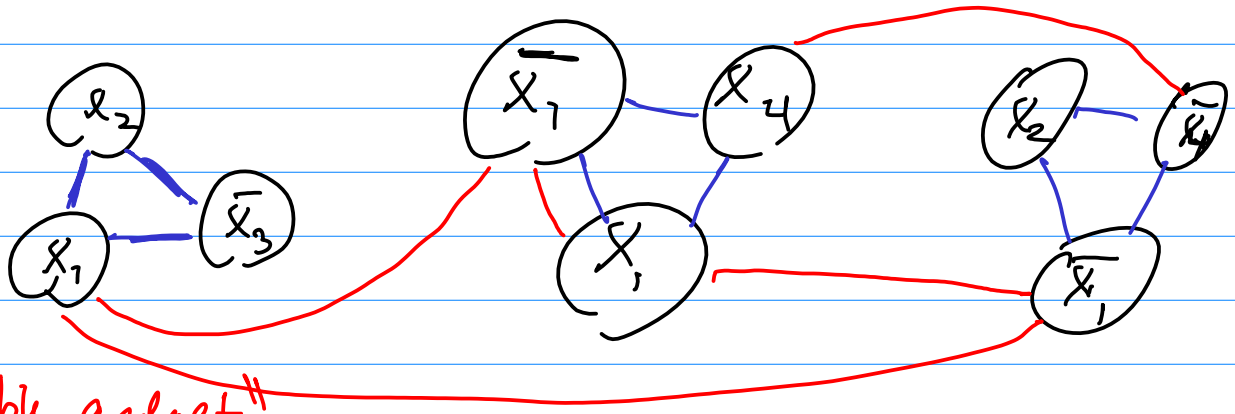
Max Ind Set (G, k): $\exists$ independent set of size $|S| \geq k$?

To show (new problem) hard,
reduce (old, hard) problem to (new, unknown) problem

Given 3SAT instance: (n vars, m clauses)

$(\ell_1 \vee \ell_2 \vee \bar{x}_3) \wedge (\bar{x}_1 \vee x_4 \vee x_1) \wedge (\ell_2 \vee \bar{\ell}_4 \vee \bar{x}_1)$

Produce a Max Ind Set instance
m clauses



draw edges between any two nodes with negated labels (x_i and $\bar{x}_i$ for some i)

Claim: Independent Set of size $\geq m$
$\Leftrightarrow$ formula satisfiable.

Proof, as is typical in NP-hardness, has two parts:

(1) size k indep set $\Rightarrow$ Satisfiable formula
(2) satisfiable formula $\Rightarrow$ size $\geq k$ ind. set

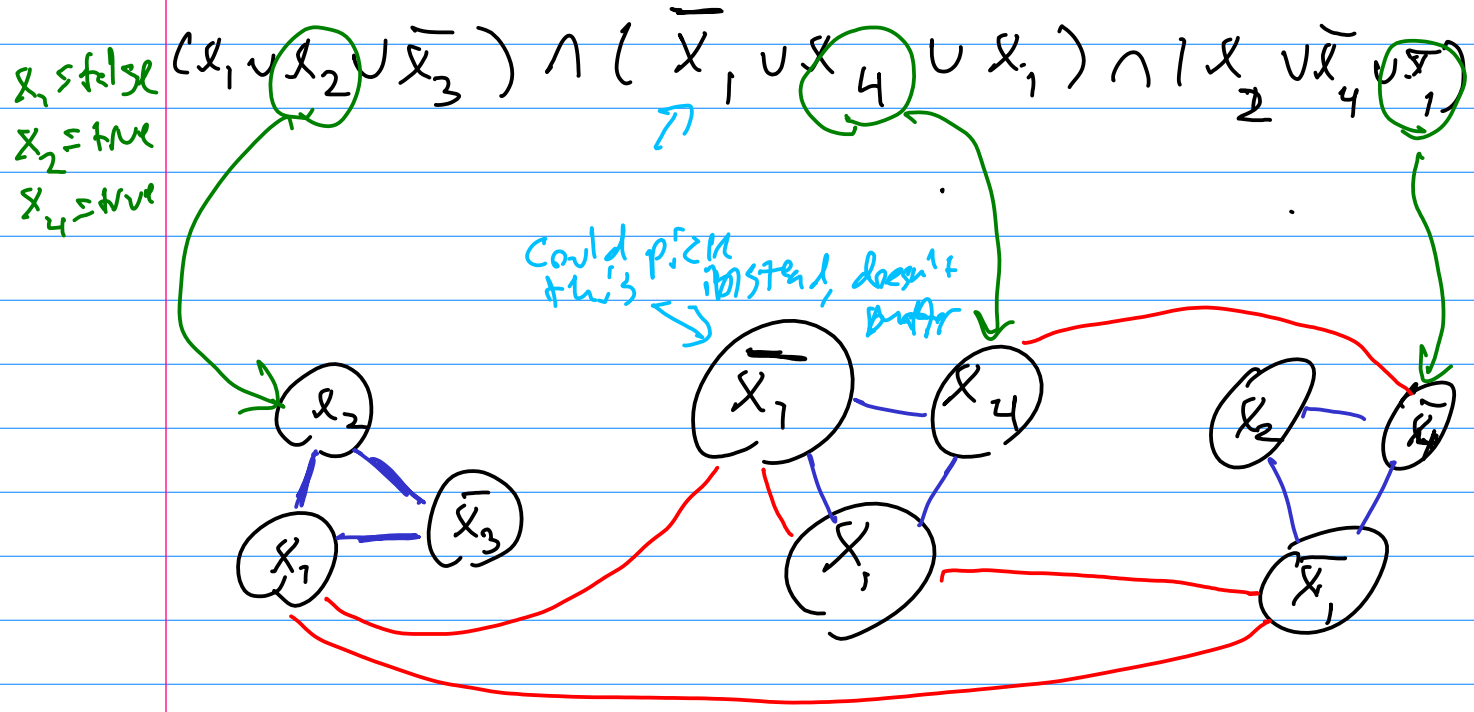
Both directions transform (any valid certificate for one) into (some certificate for other)

(satisfiable ⇒ large ind. set)

Pf ⇐. Suppose formula is satisfiable.

Then each clause has at least one satisfied literal.

Pick one such literal per clause (arbitrarily) and let S contain associated vertices.



Then: S has size M, since 1 per clause

Clause gadget edges are within a clause, S has 1 per clause, so we can't take both ends

Vertex gadget edges go x_i to $\bar{x}_i$ only one of which is satisfied, so S can't have both

⇒ S is independent set of size M.

(large independent set $\Rightarrow$ Satisfiable)

Pf $\Rightarrow$. Suppose $\exists$ independent set S of size n.

Define $\hat{x}_i = \begin{cases} 1 & \text{if any vertex labeled } x_i \text{ in } S \\ 0 & \text{otherwise} \end{cases}$

Claim! $\hat{x}_i$ is a satisfying assignment.

Clause gadget $\Rightarrow$ at most 1 vertex per clause lies in S.

S has size n & only n clauses
$\Rightarrow$ exactly 1 vertex per clause lies in S.

For each clause, if the vertex in S is labeled x_i, then we set $\hat{x}_i = 1$ and so $\hat{x}$ satisfies the clause.

But if the vertex is labeled $\bar{x}_i$, the vertex gadget edges imply no vertex labeled x_i lies in S

$\Rightarrow \hat{x}_i = 0 \Rightarrow$ clause satisfied by $\hat{x}$

$\Rightarrow$ all clauses satisfied by $\hat{x} \Rightarrow$ formula is satisfiable.

All this proves the claim:

formula satisfiable.

⇕

this graph has Independent Set of size $\geq m$

Therefore Max Ind Set is at least as hard as 3SAT

$\Rightarrow$ Max Ind Set is NP hard

turned

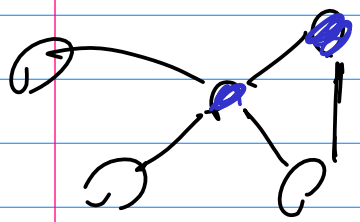
(arbitrary 3SAT instance)

into (very special graph for Ind Set)

& showed the solutions correspond.

To prove NP completeness: (1) NP hard, by reduction
(2) in NP, by checking answer.

Vertex cover (G, k): ∃ set S of $\leq k$ vertices s.t. all edges touch S



← size 2 ⇒ yes for $k \geq 2$.

Theorem: vertex-cover is NP-Complete

Pf Min Vertex Cover (G, k)

$\equiv$ Max Ind Set $(G, V - k)$

Max Clique $(G, k) \equiv$ Max Ind Set $(\bar{G}, k) \Rightarrow$ Max Clique NP hard

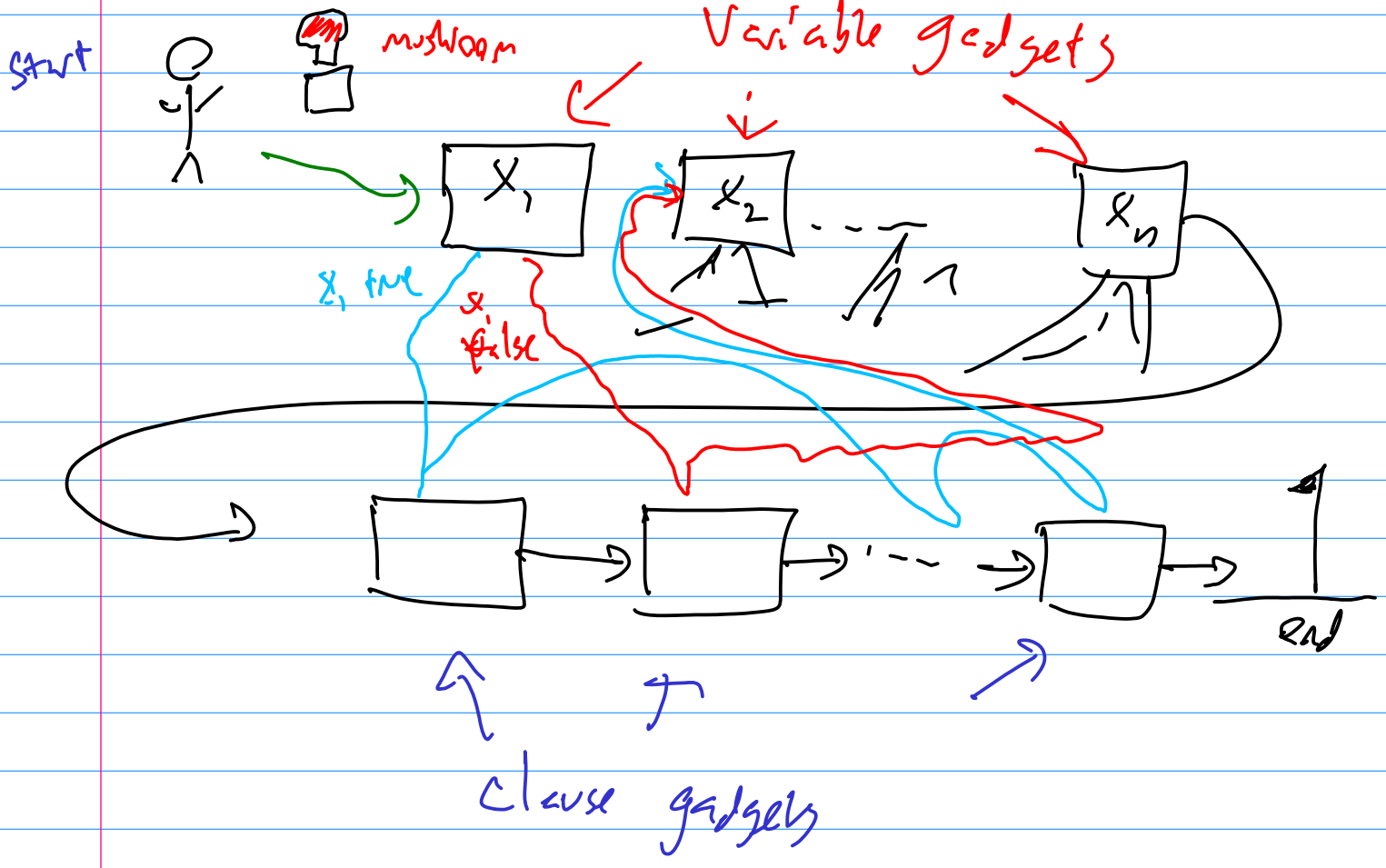
Mario is NP hard:

Reduce 3SAT to Mario

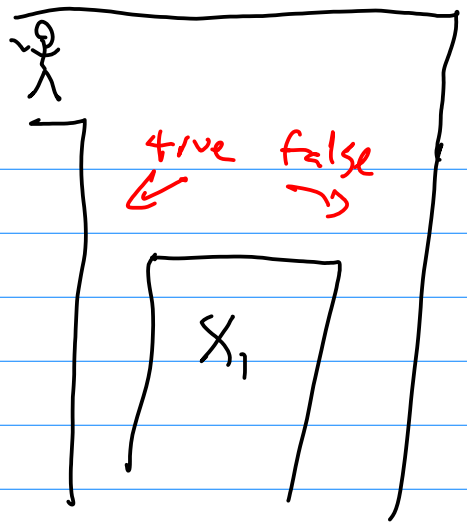
given 3 SAT instance,
can construct (polynomial size) mario level
so any solution to level lets you learn
the solution to your problem.

(=) any true master of mario must be able to solve 3SAT)

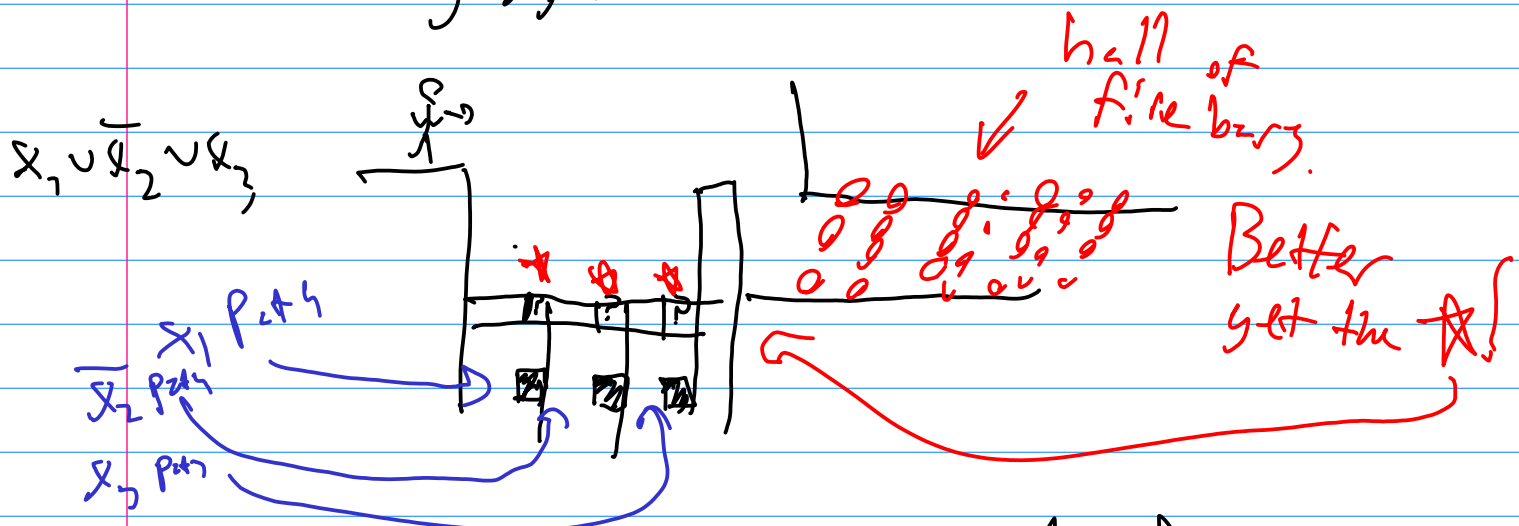
turns (any 3SAT) into (bizarre Mario level)



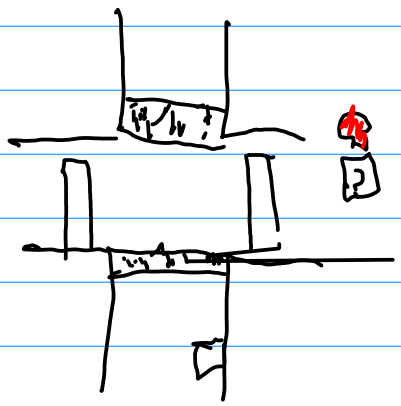
Variable gadget:

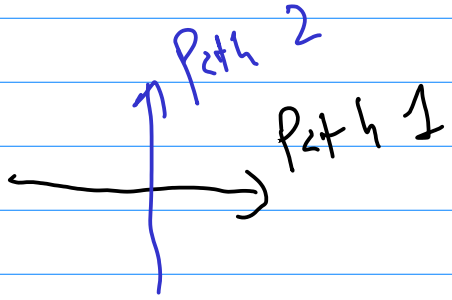


clause gadget



crossover gadget (because planar)





(many, many other ways to write 3SAT into Mario)

Subset Sum

Given set X of integers,
target T
does $\exists\ S \subseteq X$ with sum T?

DP: $O(|X| \cdot T)$ time (like Knapsack)

Problem: T might be exponentially large in the length of the input
(1000-bit nums are easy to write, 2^{1000} time is slow)

It's in NP (can check the subset)
We now show NP-hardness ($\Rightarrow$ NP-complete)
by reduction from vertex cover.

Given $G = (V, E)$, target K

Construct Subset Sum instance as follows:

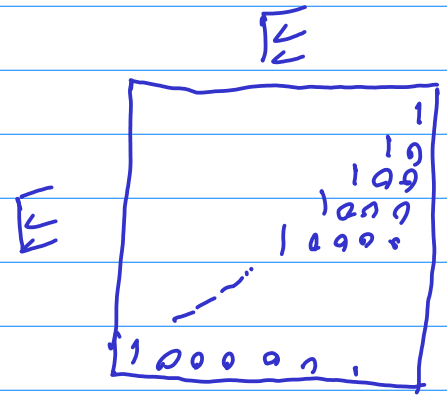
Number the edges $0, 1, \ldots, |E|-1$ arbitrarily

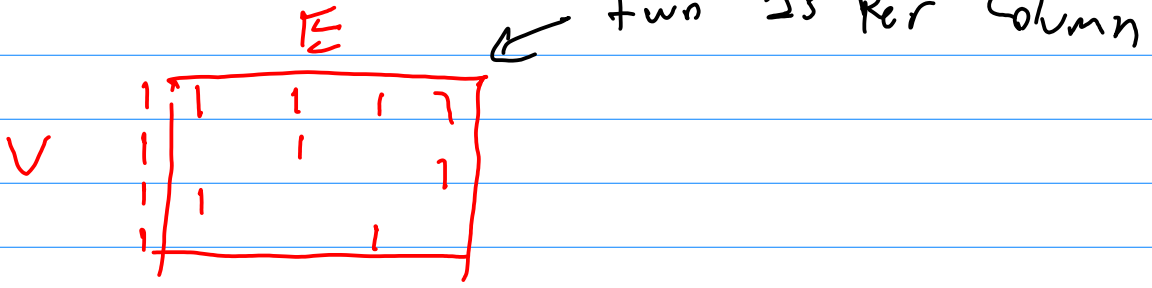
X contains:

Edge gadget: $b_e := 10^e \quad \forall e \in 0, \ldots, |E|-1$

Vertex gadget: $a_v = 10^{|E|} + \sum_{e \ni v} 10^e \quad \forall v \in V$

Target $T = K \cdot 10^{|E|} + \sum_{e=0}^{|E|-1} 2 \cdot 10^{|E|}$





T $K\,2\,2\,2\,2\,2\ldots2$

* technical note, for $k < |V|$, $\leq$ and $=$ are equiv.

Will show $\exists$ size $\leq k$ vertex cover $\Leftrightarrow$ $\exists$ subset summing to T

Note: since each column has only 3 nonzero entries, all 1, no subset has carries in last E digits.

Pf $\Rightarrow$ Let $A \subseteq V$ be the vertex cover.

Let $B \subseteq E$ contain the edges incident to exactly 1 $v \in A$.
($E \setminus B$ all incident to 2, since cover)

Then $|A| = k$, so

$$\sum_{v \in A} a_v + \sum_{e \in B} b_e = k \cdot 10^E + \underbrace{(22\ldots2)}_{E}$$
$$= T.$$

Pf $\Leftarrow$. Suppose a subset sums to T.
Let
A', B' be the subsets of a_v, b_e respectively, so

$$\sum_{v \in A'} a_v + \sum_{e \in B'} b_e = T,$$ ← 2 in last E digits

$\sum_{v \in A'} a_v$: 0 or 1 in each of last E digits

$\sum_{e \in B'} b_e$: must be 1 or 2 in last E digits

$\Rightarrow$ A' is a valid vertex cover.

How large is A'? each $a_v > 10^E$, so

$$|A'| \cdot 10^E < \left(\sum_{v \in A'} a_v\right) \leq T < (K+1) \cdot 10^E$$

or $|A'| < K+1$

or $|A'| \leq K$.

$\Rightarrow$ Subset Sum is NP-hard

($\Rightarrow$ NP complete)