Lecture 10: NP-completeness: Subset-sum

Mika Göös

EPFL School of Computer and Communication Sciences

Lecture 10

Recap: NP-completeness reductions

NP-completeness

Definition: A language L is said to be **NP-complete** if

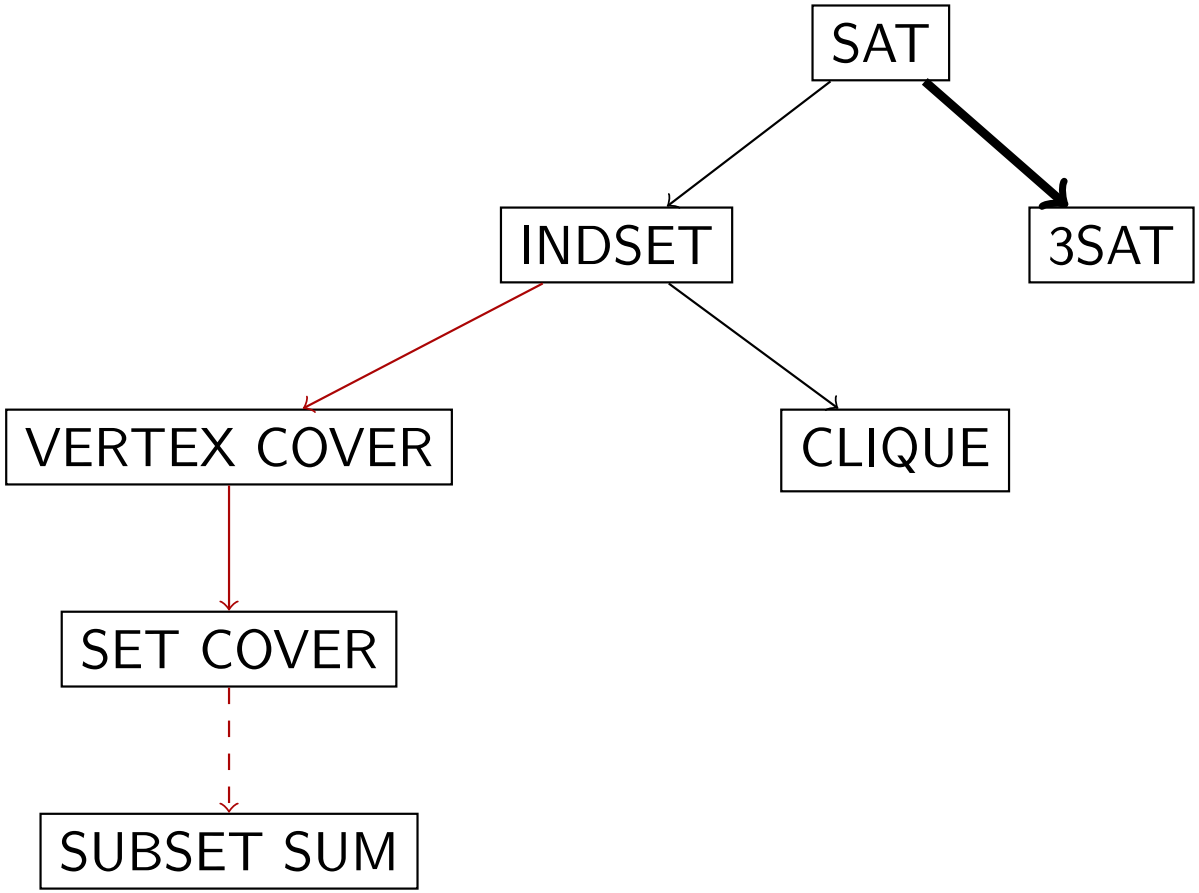
- L is in **NP**.
- For every language L' in **NP**, $L' \leq_P L$.

Observe: If **one NP**-complete language has a polynomial time decider, then **every** language in **NP** has a polynomial time decider, i.e. **P** = **NP**.

The Cook–Levin Theorem: SAT is **NP**-complete.

To show L is **NP**-complete:

- **[NP membership]** Give a poly-time verifier for L.
- **[NP hardness]** Show $C \leq_P L$ for some **NP**-complete language C (not the other way around)


SAT
INDSET
3SAT
VERTEX COVER
CLIQUE
SET COVER
SUBSET SUM

3SAT is NP-complete

$k\text{SAT} = \{\langle\varphi\rangle \ : \ \varphi \text{ is satisfiable and each clause of } \varphi \text{ contains } \leq k \text{ literals}\}$

Verifier for 3SAT: Just use the verifier for SAT.

Claim: SAT $\leq_P$ 3SAT

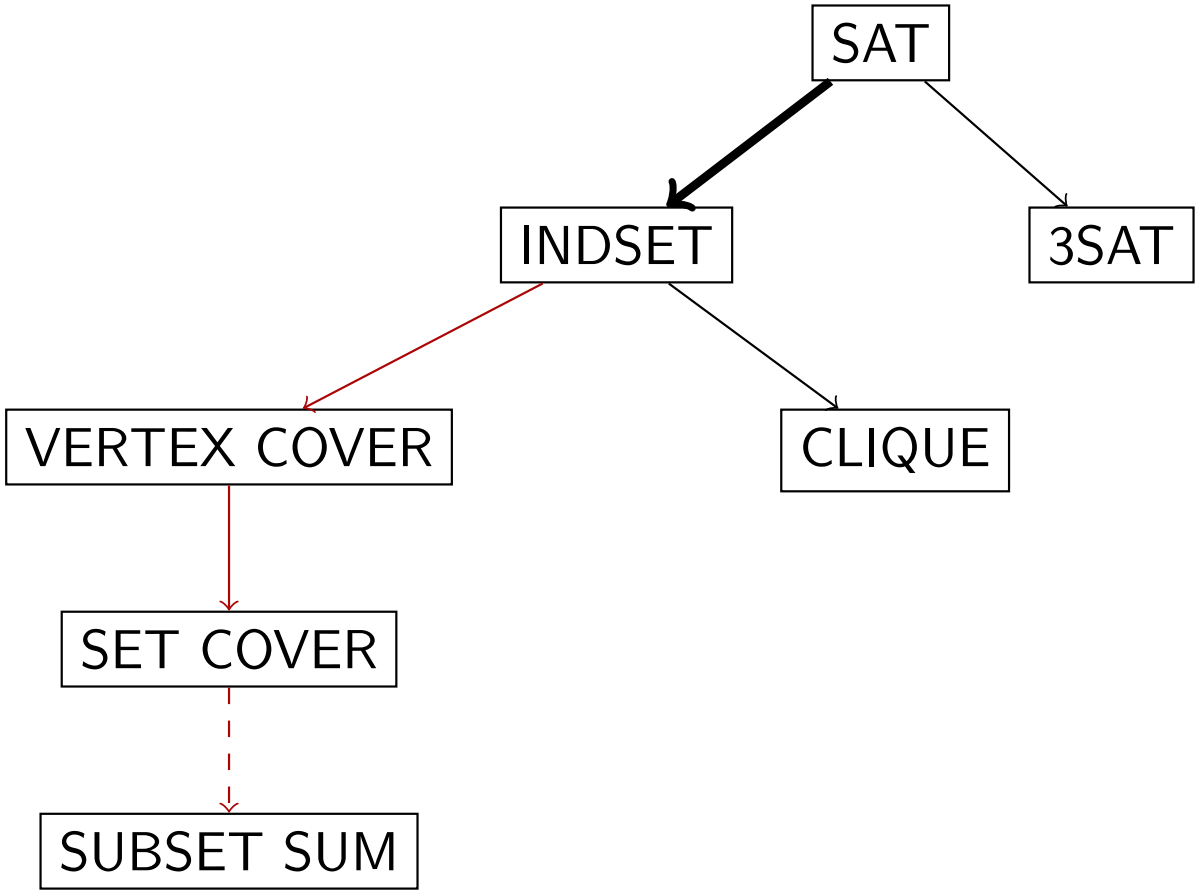
Reduction: Given φ,

- While φ contains a clause $K = (\ell_1 \vee \ell_2 \vee \ell_3 \vee \cdots \vee \ell_m)$ with > 3 literals

  Replace K with the following two clauses
  $$K_1 = (\ell_1 \vee \ell_2 \vee z)$$
  $$K_2 = (\overline{z} \vee \ell_3 \vee \cdots \vee \ell_m)$$
  } Preserves satisfiability (check!)

What is the runtime?

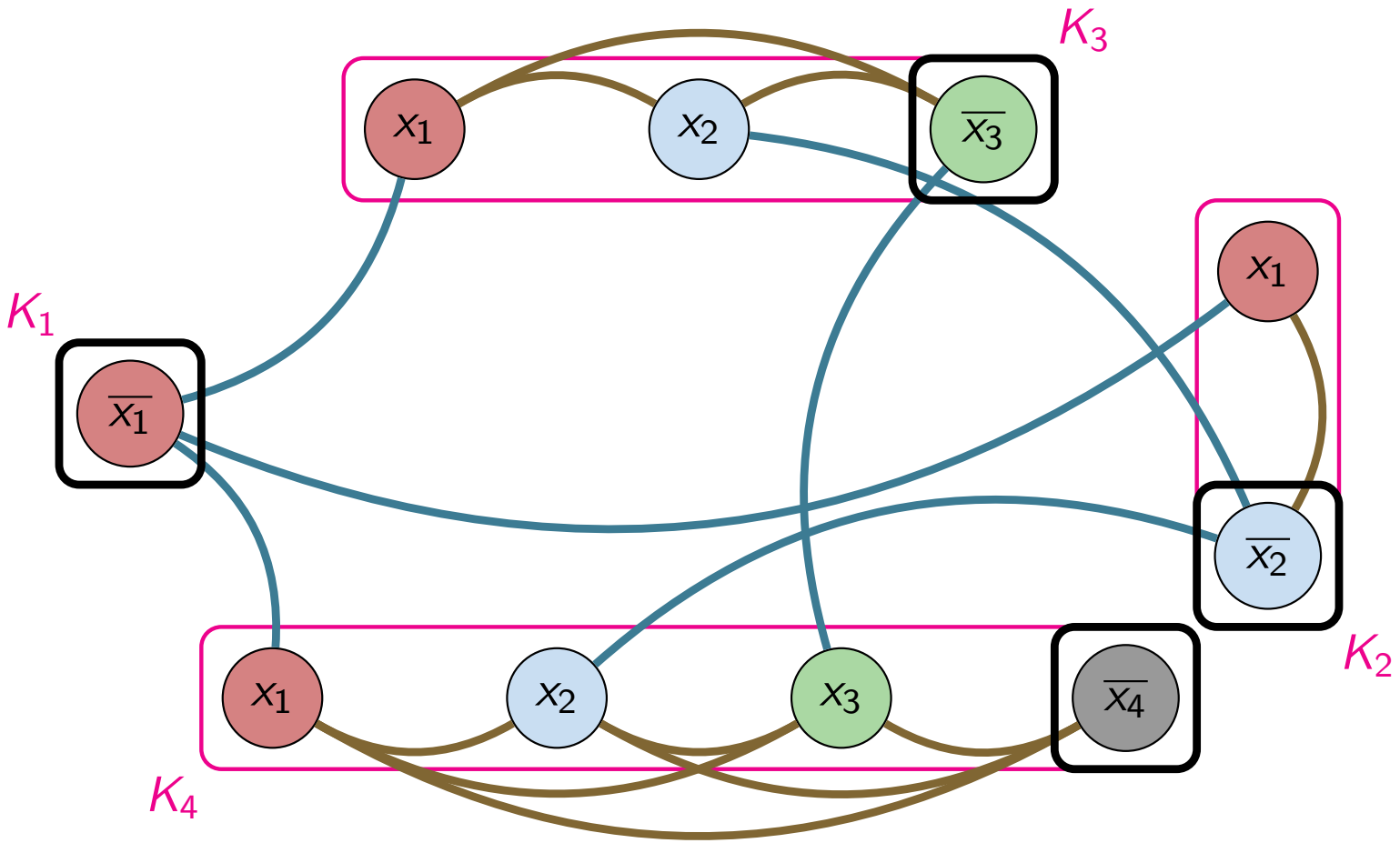
Does SAT $\leq_P$ 2SAT analogously?


SAT
INDSET
3SAT
VERTEX COVER
CLIQUE
SET COVER
SUBSET SUM

INDSET is NP-complete

Claim: SAT $\leq_P$ INDSET

$$\varphi = \underbrace{\overline{x_1}}_{K_1} \wedge \underbrace{(x_1 \vee \overline{x_2})}_{K_2} \wedge \underbrace{(x_1 \vee x_2 \vee \overline{x_3})}_{K_3} \wedge \underbrace{(x_1 \vee x_2 \vee x_3 \vee \overline{x_4})}_{K_4}$$



INDSET is NP-complete

Claim: SAT $\leq_P$ INDSET

Reduction f: On input φ,

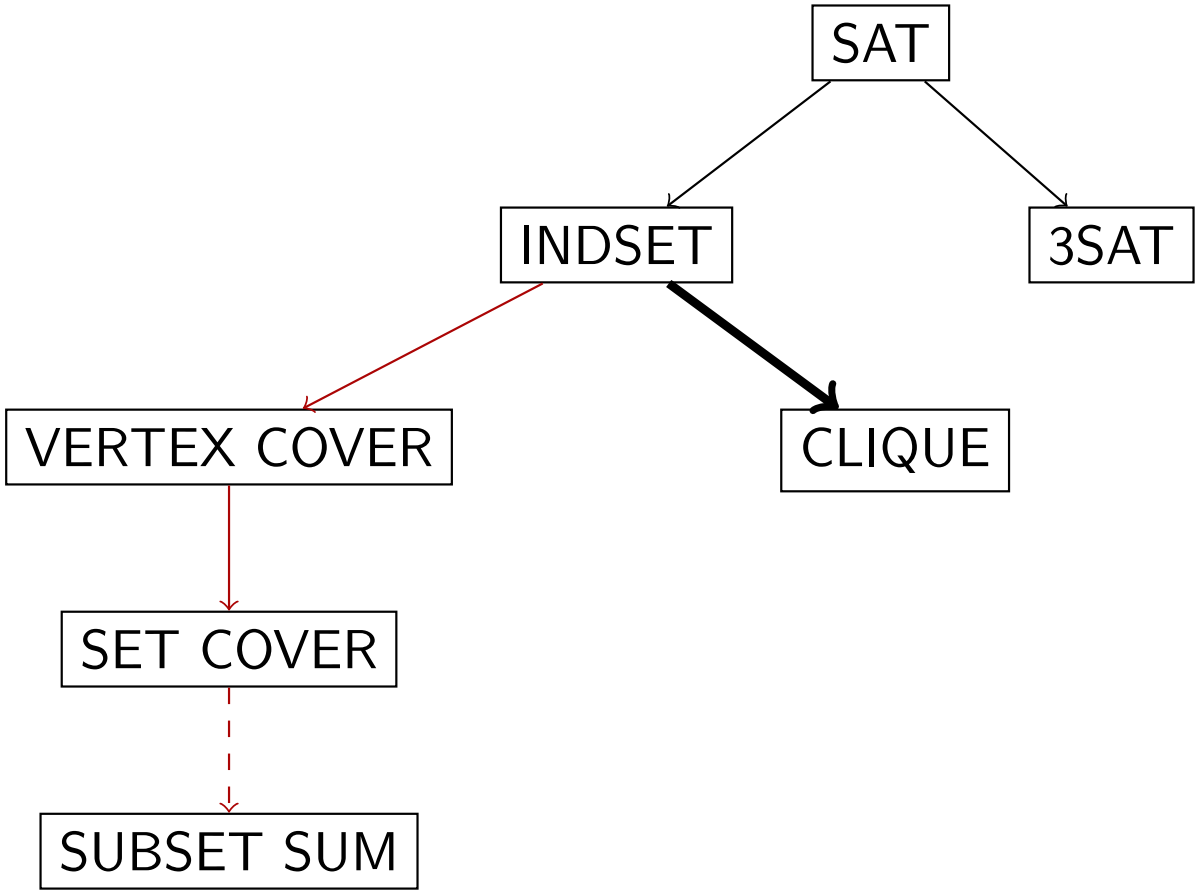
1. Let G be the graph generated as follows.
   1. Take a vertex for each literal of each clause.
   2. Add edges for pairs of conflicting literals.
   3. Add edges for pairs of literals from the same clause.
2. Let m be the number of clauses in φ.
3. Output (G, m).

Claim: $\varphi \in \text{SAT} \implies f(\varphi) \in \text{INDSET}$

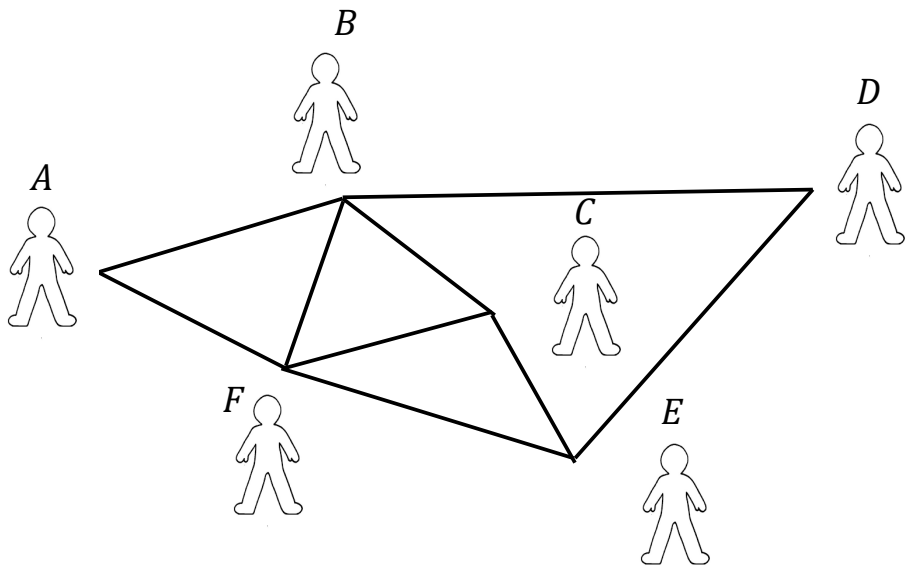
Proof: C: satisfying assignment of φ. Pick one `true` literal from each clause. The corresponding vertices form a independent set.

Claim: $f(\varphi) \in \text{INDSET} \implies \varphi \in \text{SAT}$

Proof: C: independent set in G, $|C| = m$. C contains one vertex from each group. Set the corresponding literals to `true` to get a satisfying assignment.


SAT
INDSET
3SAT
VERTEX COVER
CLIQUE
SET COVER
SUBSET SUM

CLIQUE



A,B,F is a 3-clique

Definition: A k-clique is a subset of k pairwise connected vertices

$$\text{CLIQUE} = \{\langle G, k\rangle \mid G \text{ has a clique of size } k\}$$

$\langle G, 3\rangle \in \mathit{CLIQUE}$? Yes

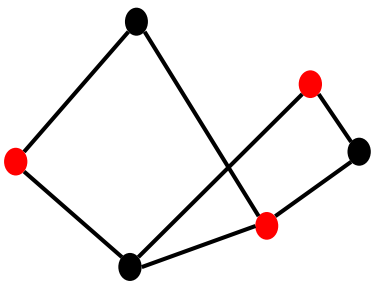
$\langle G, 4\rangle \in \mathit{CLIQUE}$? No

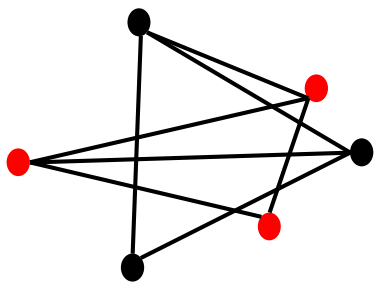
INDSET vs CLIQUE

For a graph G:

Independent set: A subset of pairwise **non-adjacent** vertices

Clique: A subset of pairwise **adjacent** vertices





Def (Complement): The *complement* of a graph $G = (V, E)$ is a graph $\bar{G} = (V, \bar{E})$ with same vertex set and edge set $\bar{E}$ s.t. $uv \in \bar{E}$ iff $uv \notin E$

Observation: If G is a graph and $\bar{G}$ its complement, then a subset S of the vertices of G is an independent set iff S is a clique of $\bar{G}$

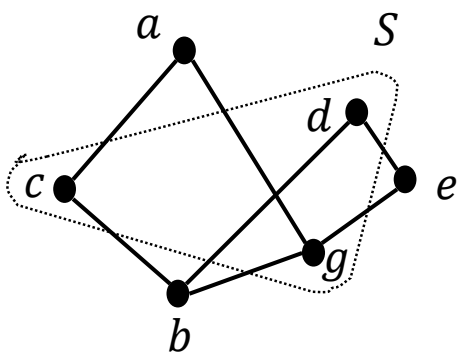
INDSET $\leq_p$ CLIQUE

Reduction: $f(\langle G = (V, E), k\rangle) := \langle \bar{G} = (V, \bar{E}), k\rangle$

Efficiency: The reduction is polynomial time

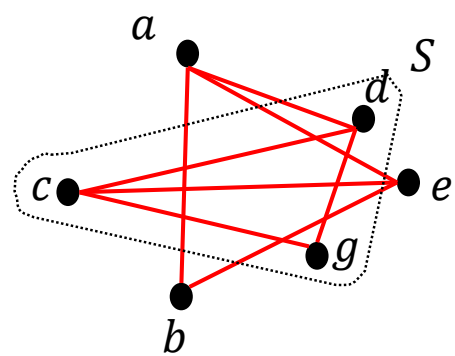
Correctness: $\langle G, k\rangle \in \text{INDSET} \Leftrightarrow \langle \bar{G}, k\rangle \in \text{CLIQUE}$

Proof of correctness:

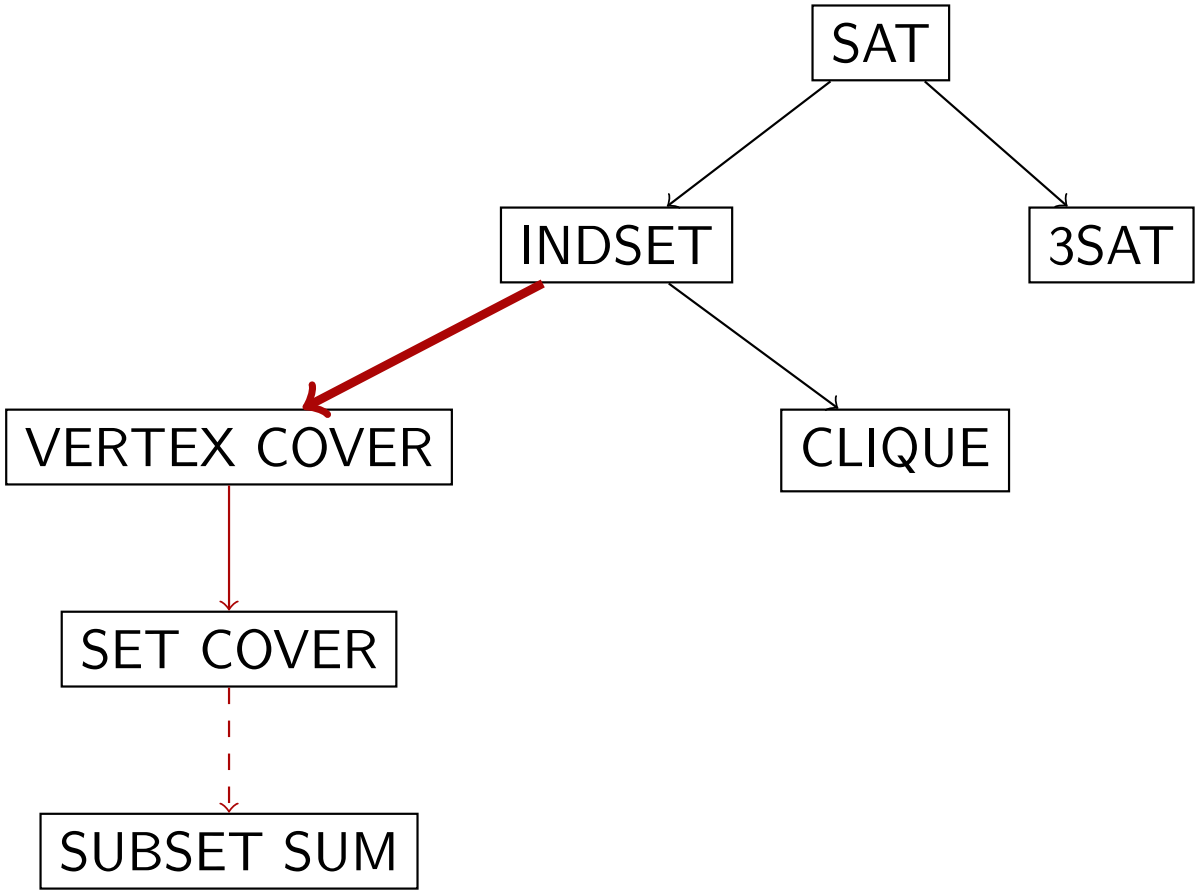


$\langle G, k\rangle \in INDSET, |S| = k$

$\Leftrightarrow$



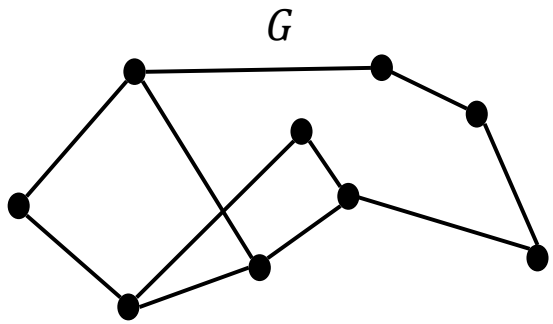
$\langle \bar{G}, k\rangle \in CLIQUE$


SAT
INDSET
3SAT
VERTEX COVER
CLIQUE
SET COVER
SUBSET SUM

Vertex Cover

Definition: For a graph $G = (V, E)$, a *vertex cover* is a subset S of V such that every edge of G is incident to at least one vertex in S

Example



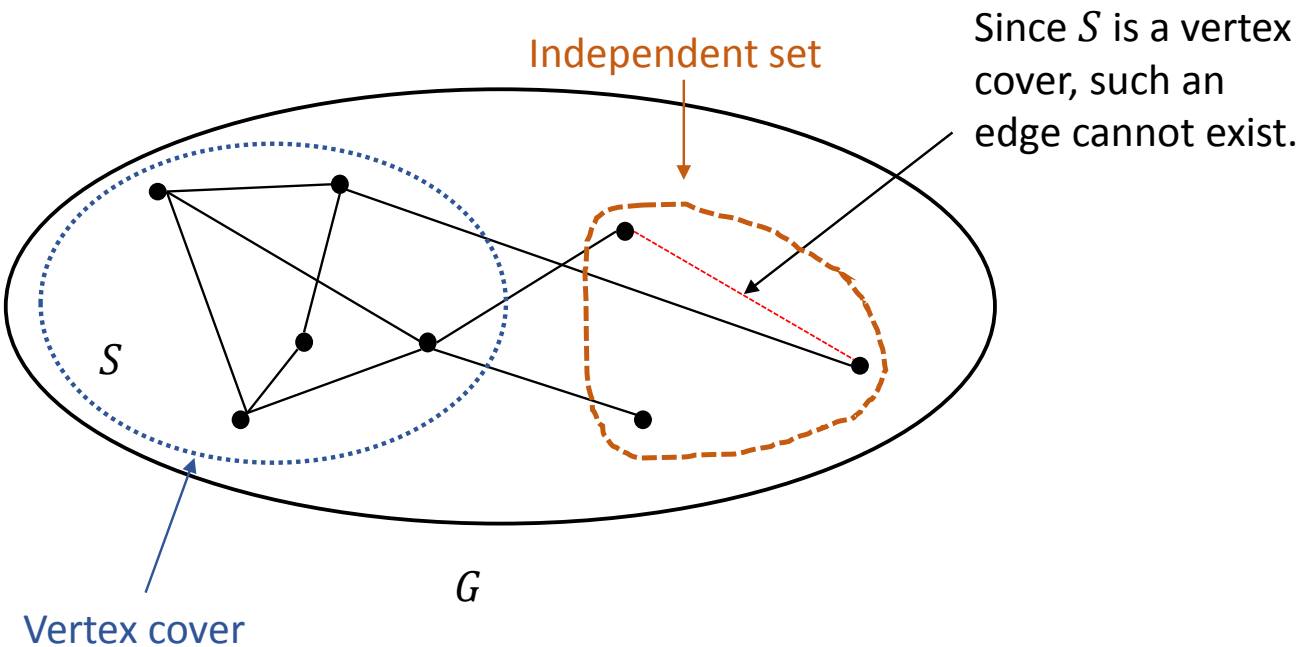
Definition:

VERTEX COVER $= \{\langle G, k\rangle : G$ is a graph that has a vertex cover of size $k\}$

- $\langle G, 4\rangle \in$ VERTEX COVER? Yes
- $\langle G, 3\rangle \in$ VERTEX COVER? No

INDSET vs VERTEX COVER

Lemma: For every graph G, a subset S of the vertices is a vertex cover if and only if $\bar{S}$ is an independent set where $\bar{S} = V \setminus S$



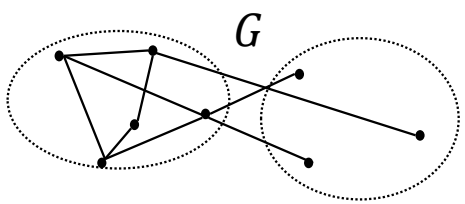
Corollary: For every graph G and a positive integer k, G has an independent set of size k iff G has a vertex cover of size $n - k$

INDSET $\leq_p$ VERTEX COVER

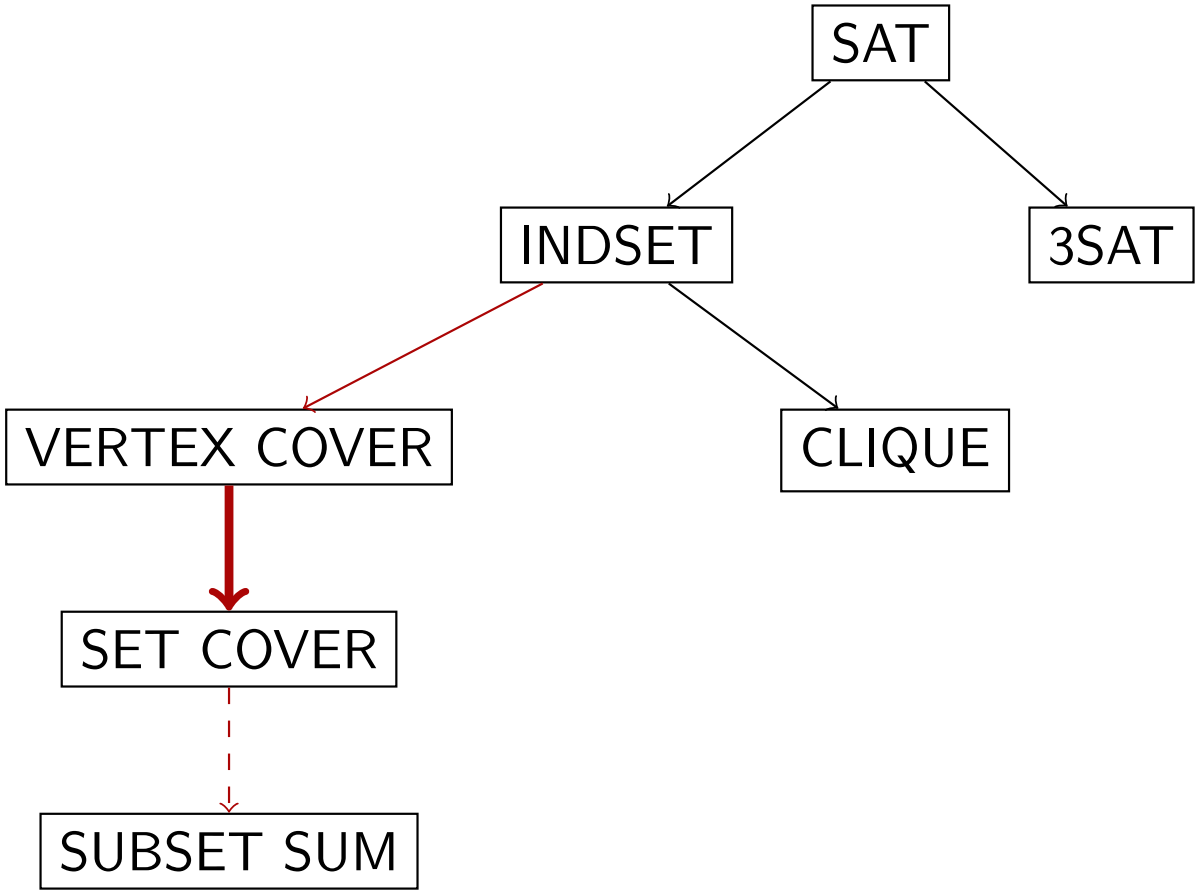
Reduction: $f(\langle G, k\rangle) := \langle G, n-k\rangle$, where n is the number of vertices of G

Efficiency: The reduction f is polynomial time

Correctness: Lemma/Corollary on previous slide!

$\langle G, k\rangle \in INDSET \iff$ 

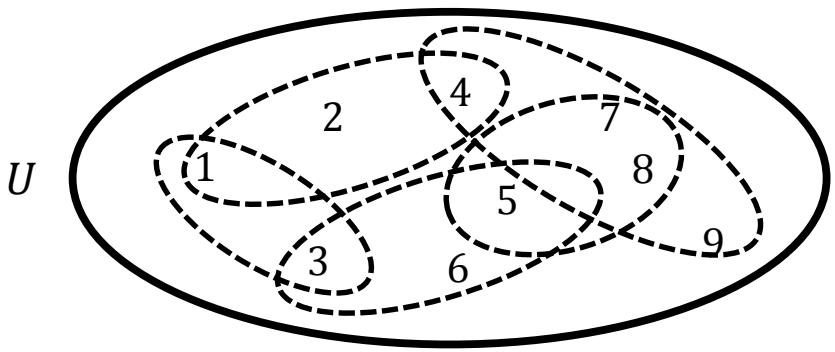
$\iff \langle G, n-k\rangle \in VERTEX\ COVER$


SAT
INDSET
3SAT
VERTEX COVER
CLIQUE
SET COVER
SUBSET SUM

Set Cover

Definition: Let $U = \{1, \ldots, n\}$ and $\mathcal{F} = \{T_1, \ldots, T_m\}$ be a family of subsets $\forall i, T_i \subseteq U$

A subset $\{T_{i_1}, \ldots, T_{i_k}\} \subseteq \mathcal{F}$ is called a set cover of size k if $\bigcup_{j=1}^{k} T_{i_j} = U$



$$\text{SET COVER} = \{\langle U, \mathcal{F}, k\rangle \mid \mathcal{F} \text{ contains a set cover of size } k\}$$

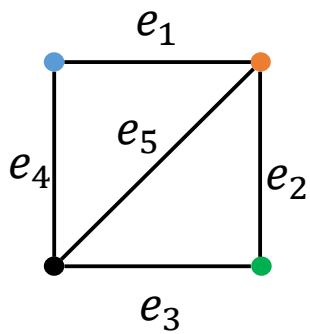
Example:

- $\langle U, \mathcal{F}, 3\rangle \in$ SET COVER? Yes $\{\{1, 2, 4\}, \{3, 6, 4\}, \{4, 7, 8, 9\}\}$
- $\langle U, \mathcal{F}, 2\rangle \in$ SET COVER? No

VERTEX COVER $\leq_p$ SET COVER

Key idea: VERTEX COVER is a *special case* of SET COVER

Example:



$k = 2$

$$U = \{e_1, e_2, e_3, e_4, e_5\}$$

$$\mathcal{F} = \{\{e_1, e_4\}, \{e_1, e_2, e_5\}, \{e_2, e_3\}, \{e_3, e_4, e_5\}\}$$

$$k' = k = 2$$

Reduction:

- Let $\langle G = (V, E), k \rangle$ be an instance of VERTEX COVER
- Set $U := E$
- For every vertex $v \in V$ create a set S_v

$$S_v := \{e \in E \mid e \text{ is incident to } v\}$$

- Let $\mathcal{F} := \{S_v \mid v \in V\}$ and set $k' = k$

Efficiency: Obvious from construction

Correctness ($\Rightarrow$): If G has a vertex cover of cardinality k, then U can be covered by k sets

Proof:

- Suppose $C \subseteq V$ is a vertex cover of G and $|C| = k$
- Every edge e_i is adjacent to at least one vertex in C

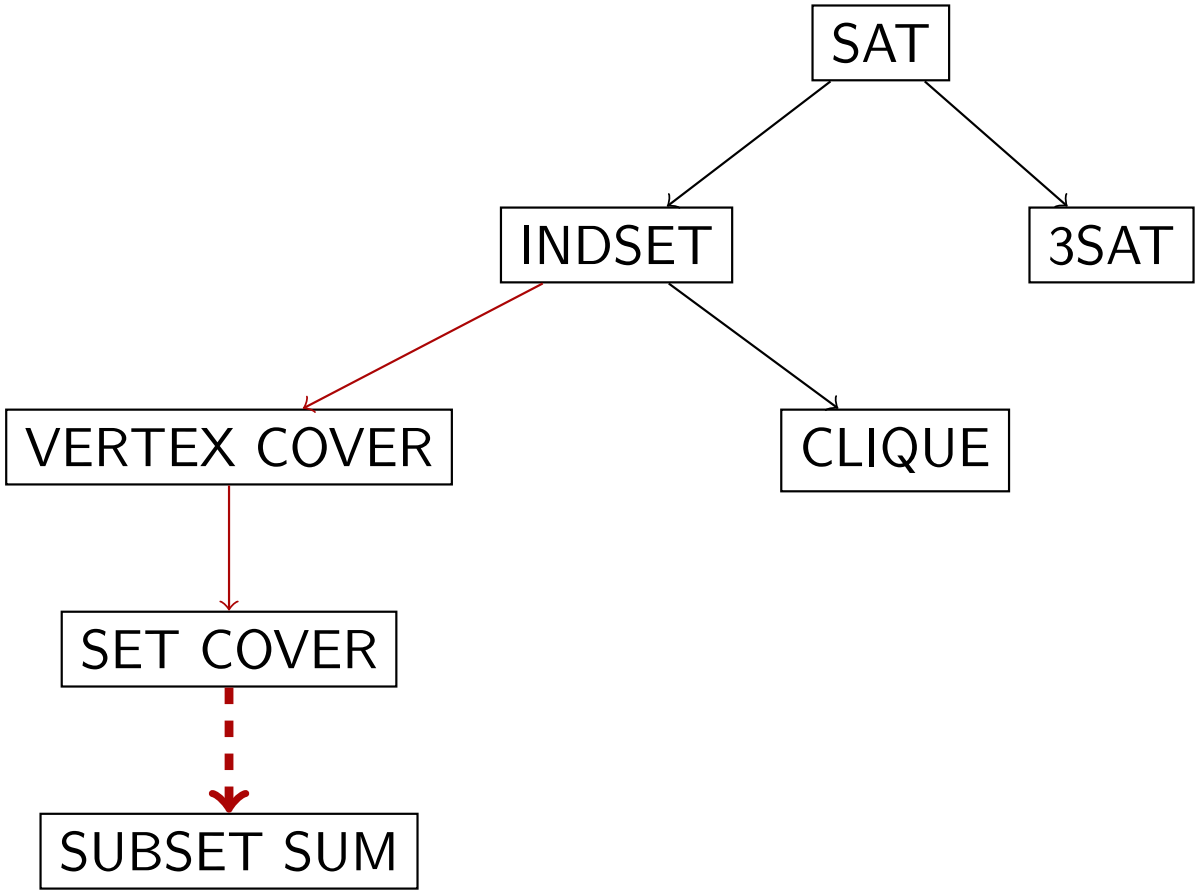
$$\bigcup_{v \in C} S_v = E$$

- Hence U can be covered by k sets

Correctness ($\Leftarrow$): If U can be covered by k sets, then G has a vertex cover of cardinality k

Proof:

- Let $S_{v_1}, S_{v_2}, \ldots, S_{v_k}$ be a collection of sets which cover $U = E$
- We claim that $C = \{v_1, v_2, \ldots, v_k\}$ is a vertex cover of G
- Indeed, every edge e in G belongs to S_{v_i} for some $i \in \{1, 2, \ldots, k\}$
- Hence, every edge e in G is incident to some vertex $v_i \in C$


SAT
INDSET
3SAT
VERTEX COVER
CLIQUE
SET COVER
SUBSET SUM

SUBSET-SUM

Let X denote a (multi) set of positive integers

Definition: SUBSET-SUM
$= \{\langle X, s\rangle : X \text{ contains a subset whose elements sum to } s\}$

Example: $X = \{1, 3, 4, 6, 13, 13\}$

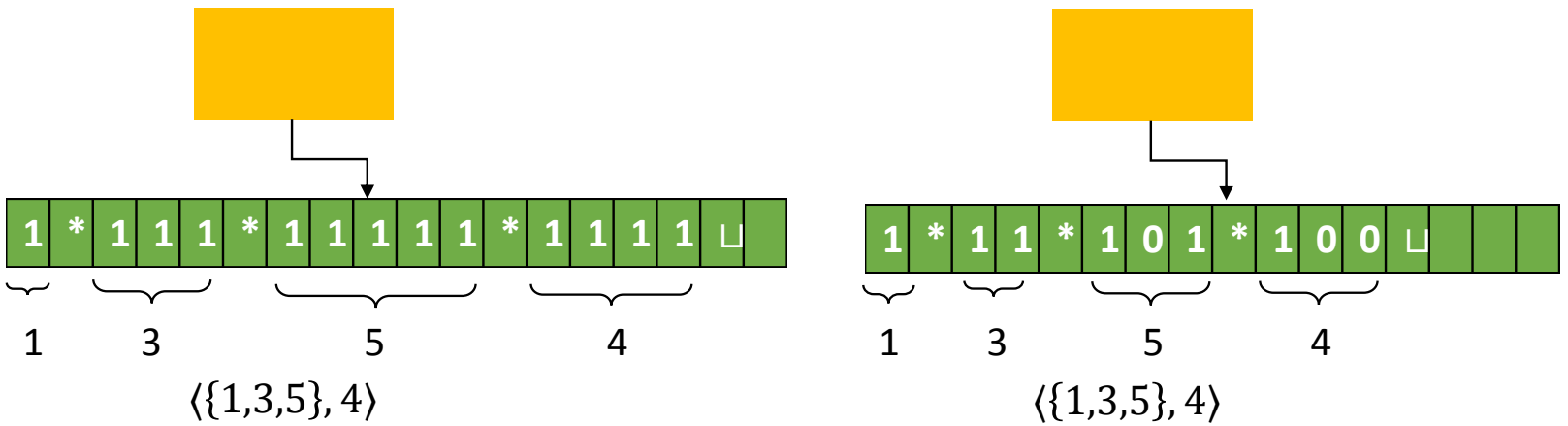
- $\langle X, 8\rangle \in$ SUBSET-SUM? Yes $T = \{1, 3, 4\}$
- $\langle X, 12\rangle \in$ SUBSET-SUM? No

Question: SUBSET-SUM $\in P$?

It depends on the input length

In $SUBSET\text{-}SUM$ we use binary encoding

Unary vs Binary



Input length: $x_1 + x_2 + \cdots + x_m + s$ $\gg$ Input length: $\lceil \log(x_1) \rceil + \cdots + \lceil \log(x_m) \rceil + \lceil \log(s) \rceil$

SUBSET-SUM: An Algorithm

- Let $X = \{x_1, x_2, \dots, x_m\}$, $sum := \sum_{j=1}^{m} x_j$
- FOR $i = 1 \,..\, m$ DO
- FOR $j = 0 \,..\, sum$ DO
- $A[i, j] := False$
- FOR $i = 0$ to m DO
- $A[i, 0] := True$
- FOR $i = 1 \,..\, m$ DO
- FOR $j = 1 \,..\, sum$ DO
- IF $A[i-1, j-x_i] = True$ OR $A[i-1, j] = True$ THEN $A[i, j] := True$
- **IF $A[m, s] = True$, ACCEPT; ELSE, REJECT**

Running time: For the input $\langle \{x_1, \dots, x_m\}, s \rangle$, the above algorithm takes $O\left(m \cdot \left(\sum_{i=1}^{m} x_i\right)\right)$ steps to output whether or not $\langle X, s \rangle \in SUBSET\text{-}SUM$

Unary representation: The length of the input $x_1 + x_2 + \cdots + x_m + s$. In this case the running time is **polynomial** in the input length

Binary representation: The length of the input $\lceil \log(x_1) \rceil + \cdots + \lceil \log(x_m) \rceil + \lceil \log(s) \rceil$. In this case, the running time is **exponential** in the input length

SUBSET-SUM is **NP**-Complete

Theorem: SET COVER $\leq_p$ SUBSET-SUM

Corollary: SUBSET-SUM is **NP**-Complete

Indeed, SUBSET-SUM $\in$ **NP** as it is easy to verify in polynomial time that $\sum_{x \in S} x = s$ for a given subset $S \subseteq X$.

SET COVER vs SUBSET-SUM

The goal: SET COVER $\leq_p$ SUBSET-SUM

	SET COVER	**SUBSET SUM**
INPUT:	▶ $U = \{1, 2, \ldots, n\}$ ▶ A family $\mathcal{F} = \{T_1, \ldots, T_m\}$ of subsets of U ▶ A number k	▶ A set of positive integers $X = \{x_1, \ldots, x_\ell\}$ ▶ A positive integer s
GOAL:	Select k elements of $\mathcal{F}$ whose union is U	Select a subset of X whose sum of elements is s

First Question: **How to map subsets to numbers?**

Encoding subsets as numbers

Idea 1 (Vector representation): Think of sets as 0-1 vectors!

U $=$

1	2	3	4	...	$n-1$	n

Encoding subsets as numbers

Idea 1 (Vector representation): Think of sets as 0-1 vectors!

U	$=$	1	2	3	4	...	$n-1$	n

T = $\{2, 3, n\}$ is then encoded by

T	$\rightarrow$	0	1	1	0	...	0	1

How to assign a number to this length-n vector?

	2^{n-1}	2^{n-2}	2^{n-3}	...	2^1	2^0	
Binary:	b_1	b_2	b_3		b_{n-1}	b_n	$\sum_{j=1}^{n} b_j 2^{n-j}$

Encoding subsets as numbers

Idea 1 (Vector representation): Think of sets as 0-1 vectors!

U	$=$	1	2	3	4	...	$n-1$	n

$T = \{2, 3, n\}$ is then encoded by

T	$\rightarrow$	0	1	1	0	...	0	1

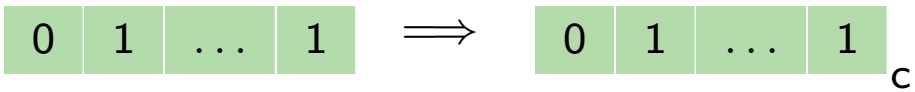
How to assign a number to this length-n vector?

	2^{n-1}	2^{n-2}	2^{n-3}	...	2^1	2^0	
Binary:	b_1	b_2	b_3		b_{n-1}	b_n	$\sum_{j=1}^{n} b_j 2^{n-j}$

	c^{n-1}	c^{n-2}	c^{n-3}	...	c^1	c^0	
c-ary: (for $c \in \mathbb{N}$)	b_1	b_2	b_3		b_{n-1}	b_n	$\sum_{j=1}^{n} b_j c^{n-j}$

Reduction: Part 1 (encoding sets as numbers)

$\langle U = \{1, \ldots, n\}, \mathcal{F} = \{T_1, T_2, \ldots, T_m\}, k \rangle \quad \Longrightarrow \quad X = \{a_1, \ldots, a_m\}$ in which

$T_i \quad \Longrightarrow$ 

$\Longrightarrow \quad a_i = \sum_{j=1}^{n} b_j c^{n-j}$

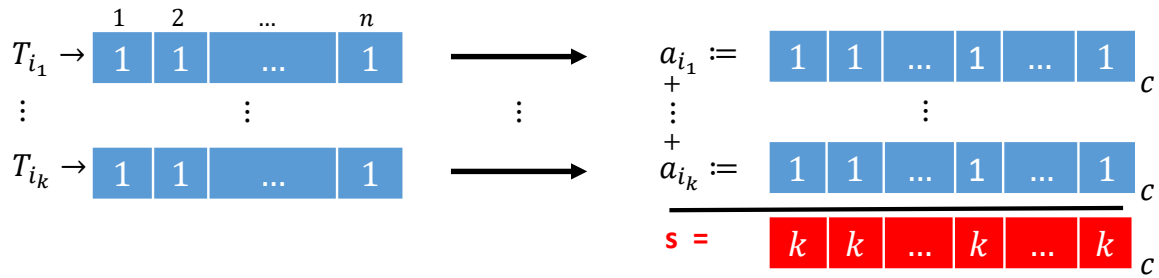
- We set the value of c later

How to pick s?

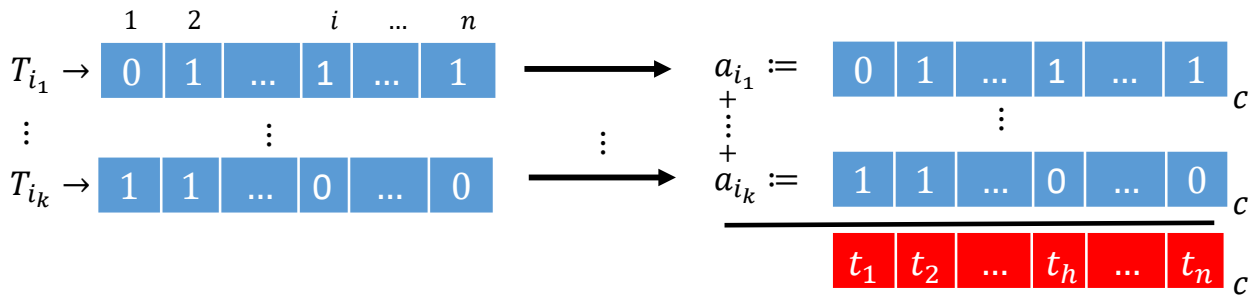
The reduction should ensure that:

$$\langle U, \mathcal{F}, k \rangle \in \text{SET COVER} \Rightarrow \langle X, s \rangle \in \text{SUBSET-SUM}$$

One example: suppose there exist k sets $T_{i_1} = \cdots = T_{i_k} = U$

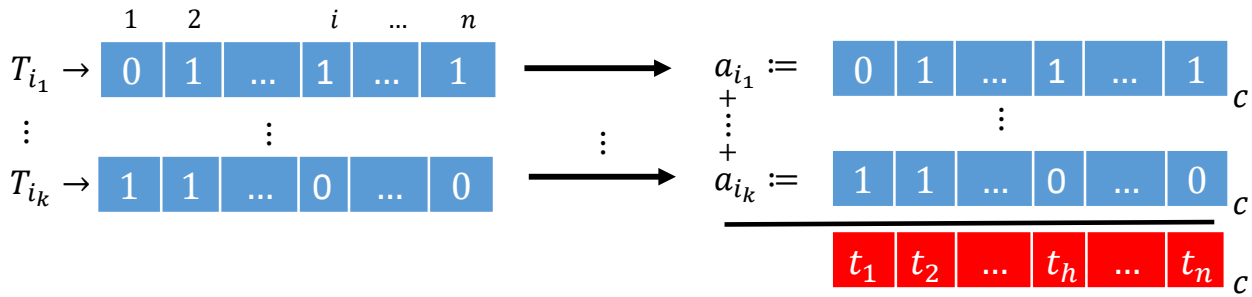


More generally: $\bigcup_{j=1}^{k} T_{i_j} = U$ *but sets not identical*



$\Sigma_{j=1}^{k} a_{i_j} =$ t_1 t_2 ... t_n $_c$ $\leq$ k k ... k $_c$

Reduction: Part 2 (selecting s)



$$\Sigma_{j=1}^{k} a_{i_j} = \boxed{t_1 \; t_2 \; \ldots \; t_n}_c \leq \boxed{k \; k \; \ldots \; k}_c$$

Observation: Since $\bigcup_{j=1}^{k} T_{i_j} = U, \forall h, 1 \leq t_h \leq k$

Idea 2 (additional numbers) The reduction adds to X for every $h \in \{1, 2, \ldots, n\}$, $k - 1$ integers

$$b_h = \boxed{0 \; 0 \; \ldots \; 1 \; \ldots \; 0}_c$$

h-th pos

By adding $k - t_h$ of each b_h's, we ensure:

$$\text{if } \bigcup_{j=1}^{k} T_{i_j} = U \text{ then } \sum_{j=1}^{k} a_{i_j} + \sum_{h=1}^{n} (k - t_h) b_h = s$$

What about the other direction?

Suppose: $\langle X, s\rangle \in$ SUBSET-SUM Thus $\exists a_{i_1}, \ldots, a_{i_p}, \underbrace{b_1, \ldots, b_1}_{r_1 \text{ times}}, \ldots, \underbrace{b_n \ldots, b_n}_{r_n \text{ times}} \in X$

s.t.

$$\sum_{j=1}^{p} a_{i_j} + r_1 \cdot b_1 + \cdots r_n \cdot b_n = s = \boxed{k}\boxed{\ldots}\boxed{k}_c$$

Claim: $\forall h \in \{1, \ldots, n\}, \exists q \in \{1, \ldots, p\}$ s.t. the h:th coordinate of a_{i_q} is 1

Proof:

- Crucial idea: force no carry over by selecting c large enough ($c = m + kn + 1$ larger than the number of summands)
- The h-th digit of $\sum_{j=1}^{p} a_{i_j} + r_1 \cdots b_1 + \cdots + r_n \cdot b_n$ is k
- Since $r_h \leq k - 1$ this means that the h-th digit of some a_{i_q} must be 1

Corollary: $T_{i_1} \cup T_{i_2} \cup \cdots \cup T_{i_p} = U$

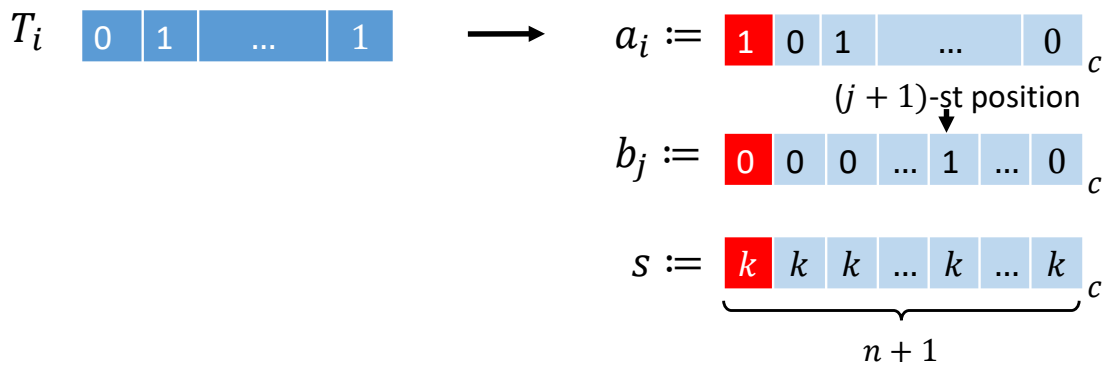
Proof:

- $\forall h \in \{1, \ldots, n\}$, the h-th coordinate of a_{i_q} is 1 iff $h \in T_{i_q}$ (by the reduction)
- Thus, at least one of the subsets $T_{i_1}, \ldots, T_{i_p}$ contains h

Final question: $T_{i_1} \cup \cdots \cup T_{i_p} = U \Rightarrow \langle U, \mathcal{F}, k\rangle \in$ SET COVER? only if $p = k$

Reduction part 3

Idea 3 (enforce $p = k$): Introduce a new coordinate in the leftmost position of a_i's, b_i's and s



Claim: If $\sum_{j=1}^{p} a_{i_j} + r_1 \cdot b_1 + \cdots r_n b_n = s$, then $p = k$

Proof

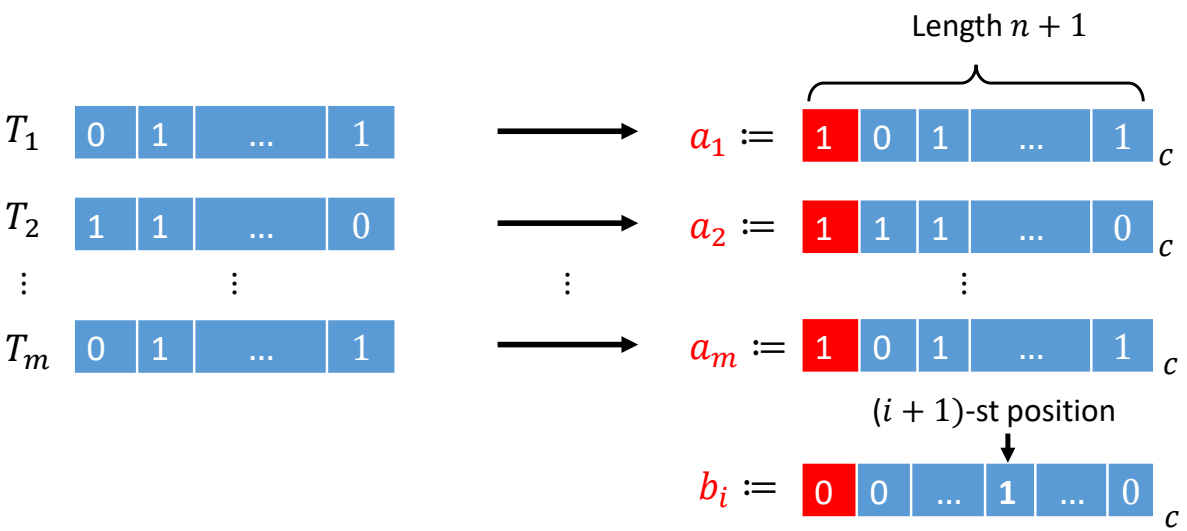
- The leftmost digit of $\sum_{j=1}^{p} a_{i_j} + r_1 \cdot b_1 + \cdots r_n b_n = s$ is k (base c)
- The leftmost digit of b_i is 0
- The leftmost digit of a_{i_j} is 1
- No carry over (since the base $c = m + kn + 1$ is large enough)
- Hence, it must be that $p = k$

The Complete Reduction

Given: $\langle U = \{1, \dots, n\}, \mathcal{F} = \{T_1, \dots, T_m\}, k \rangle$

Define: $X = \{a_1, \dots, a_m, \underbrace{b_1, \dots, b_1}_{k-1 \text{ times}}, \dots, \underbrace{b_n, \dots, b_n}_{k-1 \text{ times}}\}$,

$s := \overbrace{k\ k\ k\ \dots\ k}^{\text{Length } n+1}{}_c$ where:



Theorem: SET COVER $\leq_p$ SUBSET-SUM

Proof: Efficiency of reduction:

- the numbers in X we construct have length $\leq (n+1) \log c$ in binary
- We can compute them in polynomial time

Correctness ($\Rightarrow$)

Suppose: $\langle U, \mathcal{F}, k \rangle \rangle \in$ SET COVER

- Therefore, $\exists T_{i_1}, \ldots, T_{i_k}$ s.t. $\cup_{j=1}^{k} T_{i_j} = U$
- First coordinate of $\sum_{j=1}^{k} a_{i_j}$ is k
- Let t_h be the $(h+1)$st coordinate of $\sum_{j=1}^{k} a_{i_j}$
- Since each element is covered we have $1 \leq t_h \leq k$
- By adding $(k - t_h)$ copies of b_h we can make the $(h+1)$st digit equal to k without affecting other digits
- Thus, $\sum_{j=1}^{k} a_{i_j} + (k - t_1)b_1 + \cdots + (k - t_n) \cdot b_n = s$

- Hence, $\langle X, s \rangle \in$ SUBSET-SUM

Correctness ($\Leftarrow$)

Suppose: $\langle X, s\rangle \in$ SUBSET-SUM Thus $\exists a_{i_1}, \ldots, a_{i_p}, \underbrace{b_1, \ldots, b_1}_{r_1 \text{ times}}, \ldots, \underbrace{b_n \ldots, b_n}_{r_n \text{ times}} \in X$

s.t.

$$\sum_{j=1}^{p} a_{i_j} + r_1 \cdot b_1 + \cdots r_n \cdot b_n = s = \boxed{k}\boxed{\ldots}\boxed{k}_c$$

Let $T_{i_1}, \ldots, T_{i_k} \in \mathcal{F}$ be the sets corresponding to $a_{i_1}, \ldots, a_{i_k}$

Step 1: $p = k$

- There is no carry over when computing the sum
- The first digit in the sum = number of a_{i_j}'s = p
- The first digit in $s = k$ and so $p = k$

Step 2: for every $h \in \{1, 2, \ldots, n\}$ some set T_{i_q} contains h

- The $(h+1)$st digit of $\sum_{j=1}^{k} a_{i_j} + r_1 \cdot b_1 + \cdots r_n b_n$ is k
- The $(h+1)$st digit of b_i is 1 f $i = h$ and 0 otherwise
- Since $r_h \leq k = 1$ the $(h+1)$st digit of $\sum_{j=1}^{k} a_{i_j}$ is at least 1
- Therefore the $(h+1)$st digit of at least one of $a_{i_1}, \ldots, a_{i_k}$ is 1
- So some set T_{i_q} contains h

Conclusion: $\bigcup_{j=1}^{k} T_{i_j} = U$ and so $\langle U, \mathcal{F}, k\rangle \in$ SET COVER

NP-complete problems you can use:

CLIQUE, SAT, 3SAT, INDSET

VERTEXCOVER, SETCOVER, SUBSET-SUM,

PERFECT-3-MATCHING