Computer Algorithms I

Spring 2020

NP-completeness

efficient algorithm: has running time $O(n^k)$ for some integer k

polynomial time solvable problems

for many important problems no efficient algorithms are known

shortest paths / longest paths: find a shortest / longest path between two vertices

shortest paths: Bellman - Ford, Dijkstra

longest paths: no efficient algorithm known

can we prove that there is no efficient algorithm? NO

but it is generally believed that there are none

3-CNF satisfiability

3-CNF expression (3-conjunctive normal form)

$$(x_1 \vee x_2 \vee \bar{x}_3) \wedge (\bar{x}_1 \vee x_3 \vee x_4)$$

truth assignment

$$x_1 = 1,\ x_2 = 0,\ x_3 = 1,\ x_4 = 0$$

truth value of expression for this truth assignment

$$(1 \vee 0 \vee \bar{1}) \wedge (\bar{1} \vee 1 \vee 0) = 1$$

so this truth assignment satisfies the expression

the expression is satisfiable

3-CNF satisfiability: given a 3-CNF expression, is it satisfiable?

Clique problem

$G = (V, E)$ undirected graph

clique: $V' \subseteq V$ such that there is an edge between any two vertices in V'

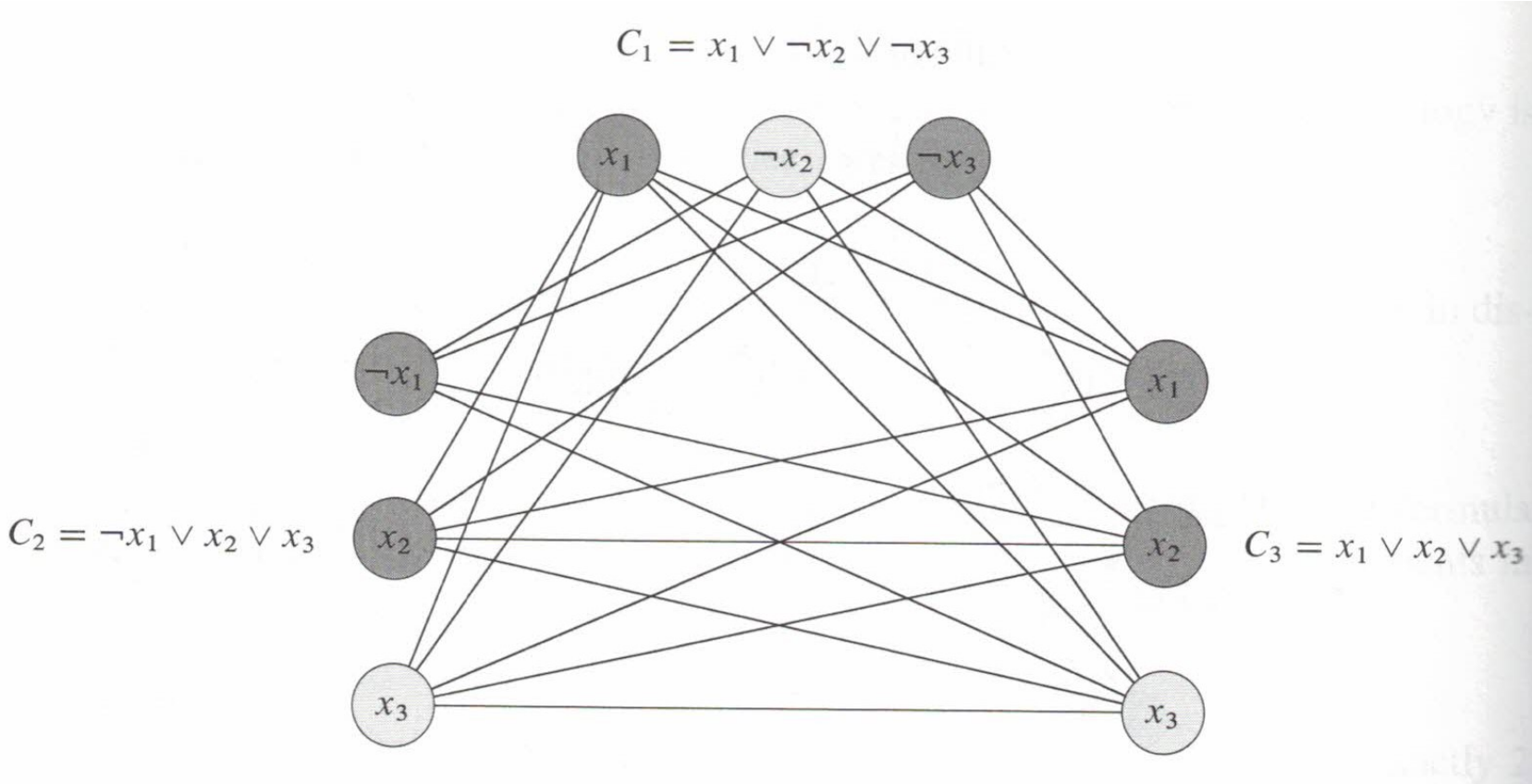
clique problem: given a graph G and a number k, does G have k vertices which form a clique?

do the satisfiability problem and the clique problem have anything to do with each other?

Reduction from 3-CNF-SAT to CLIQUE

given a 3-CNF expression ϕ, we construct a graph $G = (V, E)$ and a number k such that

$$\phi \text{ is satisfiable } \Leftrightarrow \ G \text{ has a clique of size } k$$



$k = 3$ (number of clauses in ϕ)

What is a reduction good for?

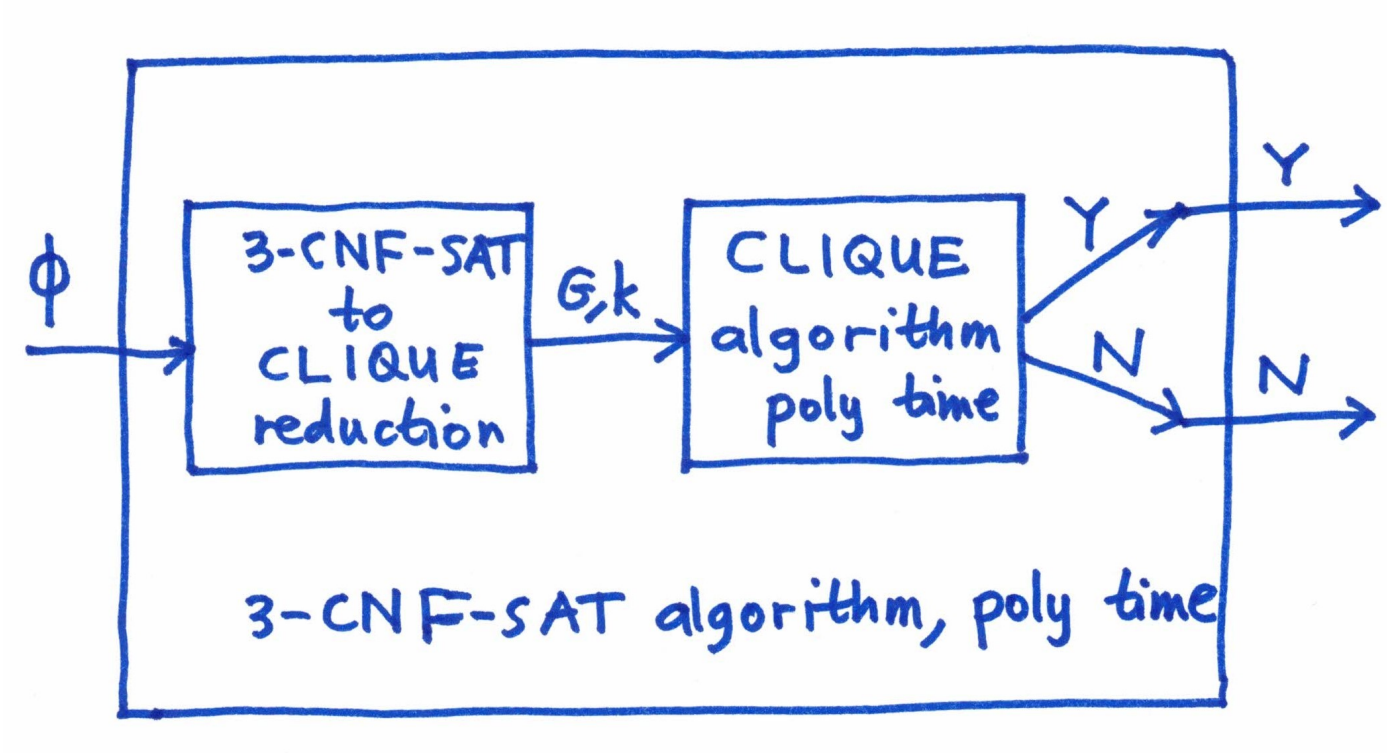
assume that we have an efficient algorithm for the clique problem

Claim then we also have an efficient algorithm for 3-CNF satisfiability!

how can we decide if ϕ is satisfiable?

construct G and k, run the clique algorithm on this input and return the answer!

Satisfiability algorithm using reduction to clique problem



NP-completeness

one can prove that many hard problems are "NP-complete"

any two NP-complete problems can be reduced to each other

if one of them can be solved efficiently then the others can be solved efficiently as well

so whether any of those problems have efficient algorithms is just one problem

Outline of the correctness of the reduction

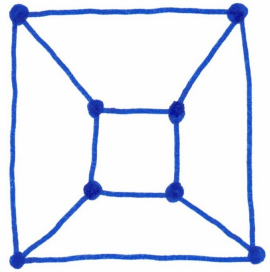
ϕ is satisfiable $\Rightarrow$ it has a satisfying truth assignment

this satisfying truth assignment can be used to find a clique in the graph

G has a clique of size k $\Rightarrow$ the labels of the nodes of the clique can be used to construct a satisfying truth assignment for ϕ, so ϕ is satisfiable

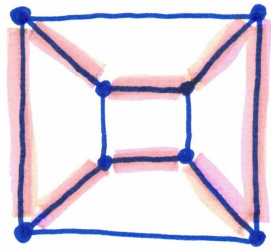
Hamilton cycle

Hamilton cycle: cycle going through every vertex exactly once



does this graph contain a Hamilton cycle?

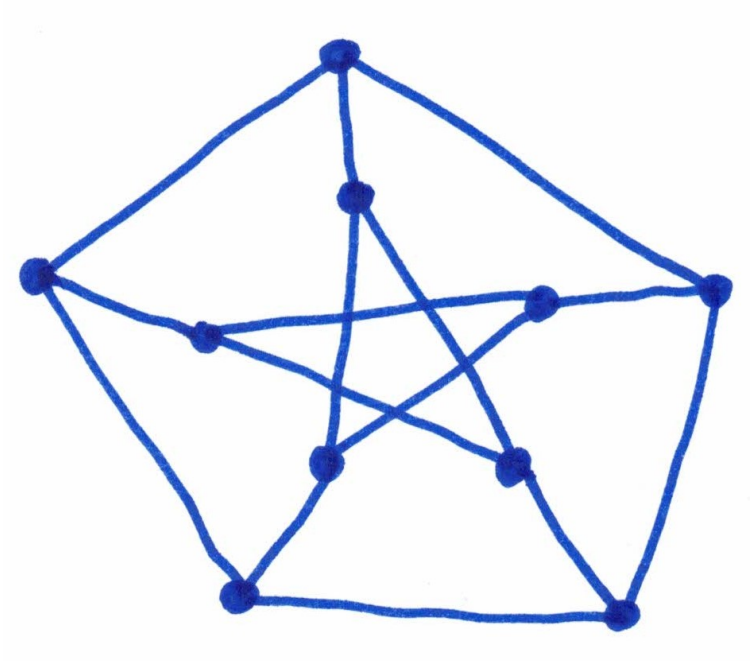
yes, here, we can certify that!



given a solution it is easy to verify that it is correct

how that solution was found does not matter

Another graph



does this graph contain a Hamilton cycle?

no... but how to verify that?

Verification algorithm

problem: $HAM - CYCLE$, $3 - CNF - SAT$, $CLIQUE$

input: graph $G = (V, E)$, formula ϕ, graph $G = (V, E)$ and k

certificate: Hamilton cycle, satisfying truth assignment, clique of size k

verification algorithm: decide if a claimed certificate is indeed a certificate

easy in each case

Definitions

optimization problem - e.g., find minimum spanning tree

decision problem version: is there a spanning tree with weight $\leq w$? - yes/no

encoding: graphs, flow networks, sets of activities $\rightarrow$ bit sequences, 0-1 strings

set of strings: $\{0,1\}^* = \{\epsilon, 0, 1, 00, 01, 10, 11, 000, \ldots\}$

$e : I \rightarrow \{0,1\}^*$ (I: set of instances)

language: set of strings, $L \subseteq \{0,1\}^*$

problem $\rightarrow$ language: set of 'yes' instances

P and NP

$P = \{L \subseteq \{0,1\}^* : L \text{ computed by a polynomial time algorithm}\}$

verification algorithm: $A(x, y)$, x: input, y: certificate

example: for Hamilton cycle problem - x: graph, y: cycle

$$NP = \{L : L = \{x : \exists y(|y| = O(|x|^c)A(x, y) = 1\}\}$$

A is a polynomial time verification algorithm, and c is a constant

$HAM = \{\langle G \rangle : G \text{ is graph with a Hamilton cycle}\}$

$HAM = \{x : \exists y(|y| = O(|x|^c)\ A(x, y) = 1\}$

where A checks if y is indeed a Hamilton cycle in x

so $HAM \in NP$

Reduction, NP-completeness

languages $L_1, L_2 \subseteq \{0,1\}^*$

$L_1 \leq_P L_2$: L_1 polynomial time reducible to L_2:

there is a polynomial time computable function f such that for every x it holds that

$$x \in L_1 \quad \Leftrightarrow \quad f(x) \in L_2$$

L is NP-complete:

- $L \in NP$
- $L' \leq L$ for every $L' \in NP$

$P \subseteq NP$

million dollar question: $P \neq NP$?

Three basic properties

1. if $L_1 \leq_P L_2$ and $L_2 \in P$ then $L_1 \in P$

   proof: diagram

2. if L is NP-complete and $L \in P$ then $P = NP$

   proof: diagram

3. if $L' \leq_P L$, L' is NP-complete and $L \in NP$ then L is NP-complete

   proof: combine reductions

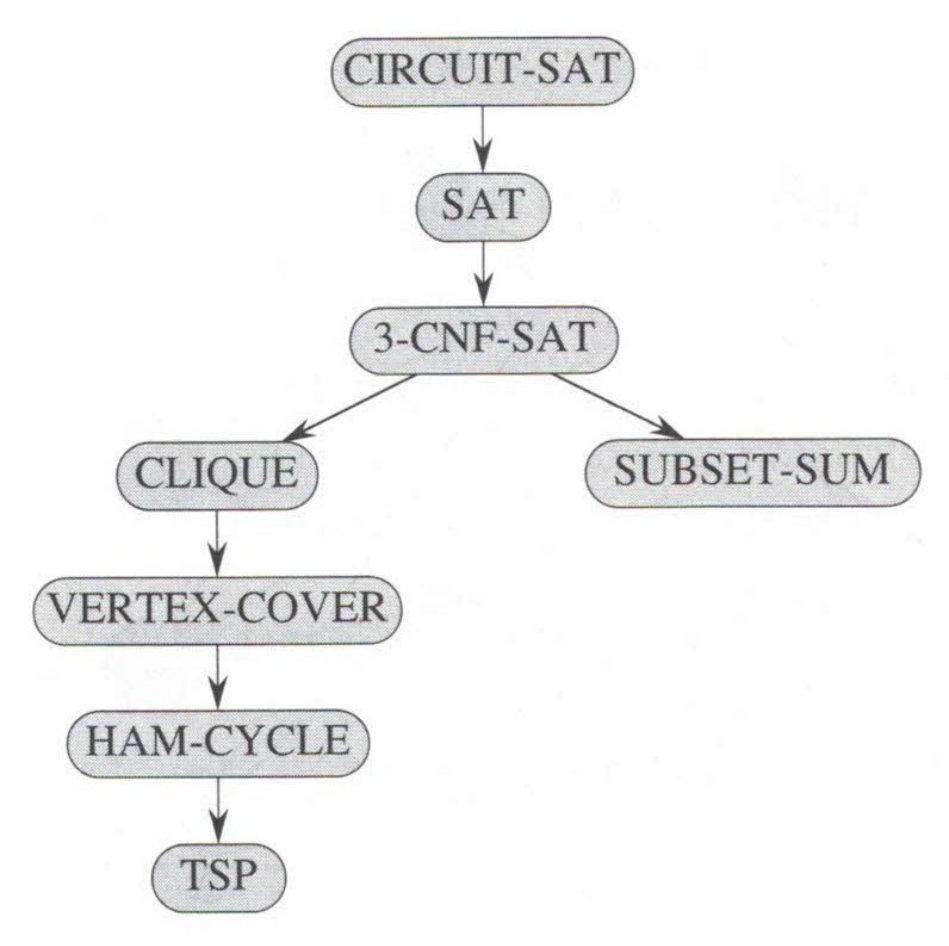
NP-complete problems

Theorem
$3 - CNF - SAT$ *is NP-complete*

Theorem
$3 - CNF - SAT \leq_P CLIQUE$

Theorem
CLIQUE is NP-complete

NP-complete problems



Subset sum problem

given: set S of positive integers, target t

is there a subset of S adding up to t?

example: $S = \{2, 3, 5, 6, 8, 9\}$, $t = 17$

$SUBSET - SUM = \{\langle S, t\rangle : \exists S' \subseteq S, \sum_{s \in S'} s = t\}$

$SUBSET - SUM \in NP$: certificate is (encoding of) S' (e.g., $\langle\{2, 6, 9\}\rangle$

$SUBSET - SUM$ is NP-complete: in NP, reduction from $3 - CNF - SAT$

$3 - CNF - SAT \leq_P SUBSET - SUM$

$\phi \to S, t$: ϕ satisfiable $\Leftrightarrow$ S has subset adding up to t

		x_1	x_2	x_3	C_1	C_2	C_3	C_4
v_1	=	1	0	0	1	0	0	1
v'_1	=	1	0	0	0	1	1	0
v_2	=	0	1	0	0	0	0	1
v'_2	=	0	1	0	1	1	1	0
v_3	=	0	0	1	0	0	1	1
v'_3	=	0	0	1	1	1	0	0
s_1	=	0	0	0	1	0	0	0
s'_1	=	0	0	0	2	0	0	0
s_2	=	0	0	0	0	1	0	0
s'_2	=	0	0	0	0	2	0	0
s_3	=	0	0	0	0	0	1	0
s'_3	=	0	0	0	0	0	2	0
s_4	=	0	0	0	0	0	0	1
s'_4	=	0	0	0	0	0	0	2
t	=	1	1	1	4	4	4	4

Figure 34.19 The reduction of 3-CNF-SAT to SUBSET-SUM. The formula in 3-CNF is $\phi = C_1 \wedge C_2 \wedge C_3 \wedge C_4$, where $C_1 = (x_1 \vee \neg x_2 \vee \neg x_3)$, $C_2 = (\neg x_1 \vee \neg x_2 \vee \neg x_3)$, $C_3 = (\neg x_1 \vee \neg x_2 \vee x_3)$, and $C_4 = (x_1 \vee x_2 \vee x_3)$. A satisfying assignment of ϕ is $\langle x_1 = 0, x_2 = 0, x_3 = 1 \rangle$. The set S produced by the reduction consists of the base-10 numbers shown; reading from top to bottom, $S = \{1001001, 1000110, 100001, 101110, 10011, 11100, 1000, 2000, 100, 200, 10, 20, 1, 2\}$. The target t is 1114444. The subset $S' \subseteq S$ is lightly shaded, and it contains v'_1, v'_2, and v_3, corresponding to the satisfying assignment. It also contains slack variables s_1, s'_1, s'_2, s_3, s_4, and s'_4 to achieve the target value of 4 in the digits labeled by C_1 through C_4.