NP-Completeness II

Design and Analysis of Algorithms
Andrei Bulatov

Traveling Salesperson Problem

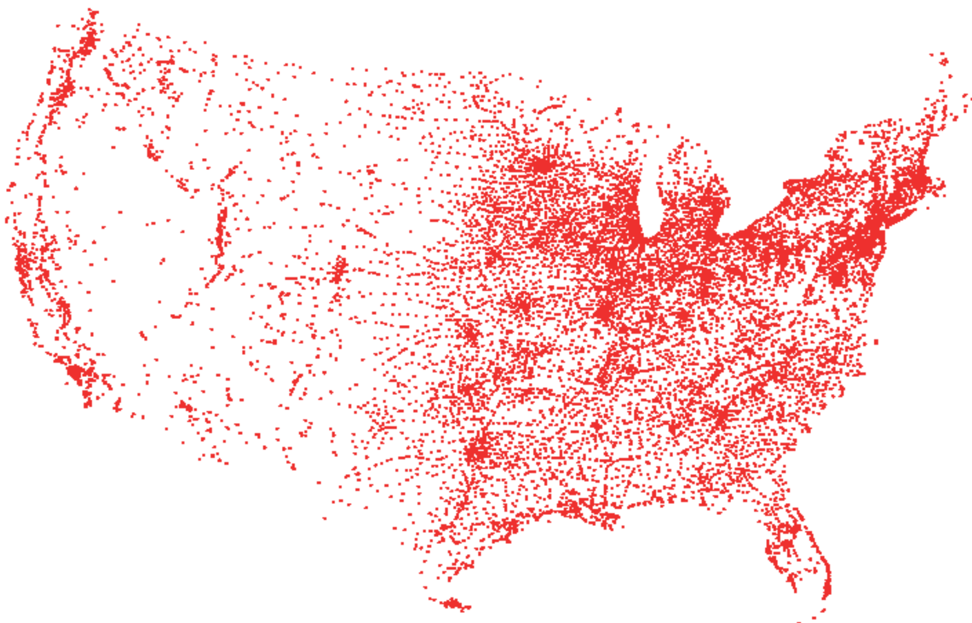
The Traveling Salesperson Problem

Instance:

A set of n cities, a pairwise distance function $d(u, v)$, number D

Objective:

Is there a tour of length $\leq D$?



All 13,509 cities in US with a population of at least 500
Reference: http://www.tsp.gatech.edu

Traveling Salesperson Problem

Theorem

Hamiltonian Cycle $\leq$ TSP

Proof

Given instance G = (V, E) of HAM-CYCLE , create n cities with distance function

$$d(u,v) = \begin{cases} 1 & \text{if } (u,v) \in E \\ 2 & \text{if } (u,v) \notin E \end{cases}$$

TSP instance has tour of length $\leq$ n iff G is Hamiltonian

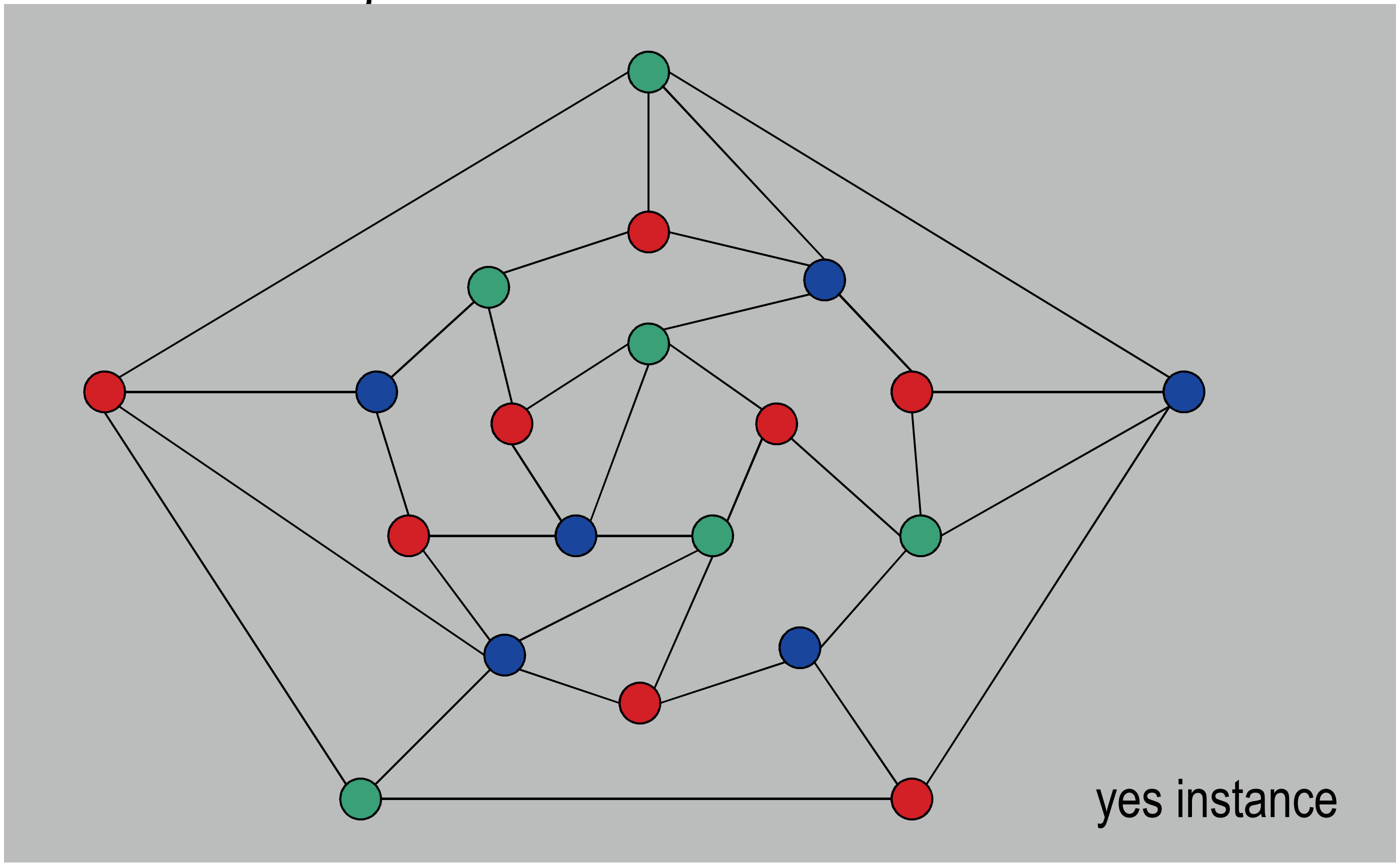
QED

Graph Coloring

The 3-Coloring Problem

Instance An undirected graph G

Objective: Does there exists a way to color the nodes red, green, and blue so that no adjacent nodes have the same color?



Applications of Graph Coloring

Register allocation: Assign program variables to machine registers so that no more than k registers are used and no two program variables that are needed at the same time are assigned to the same register.

Interference graph: Nodes are program variables names, edge between u and v if there exists an operation where both u and v are "live" at the same time.

Observation.

[Chaitin 1982] Can solve register allocation problem iff interference graph is k-colorable.

Fact.

3-Coloring $\leq$ k-Register Allocation for any constant $k \geq 3$.

3-Colorability: NP-Completeness

Theorem

3-SAT $\leq$ 3-Coloring

Proof

Given 3-SAT instance Φ, we construct an instance of 3-COLOR that is 3-colorable iff Φ is satisfiable.

Construction:

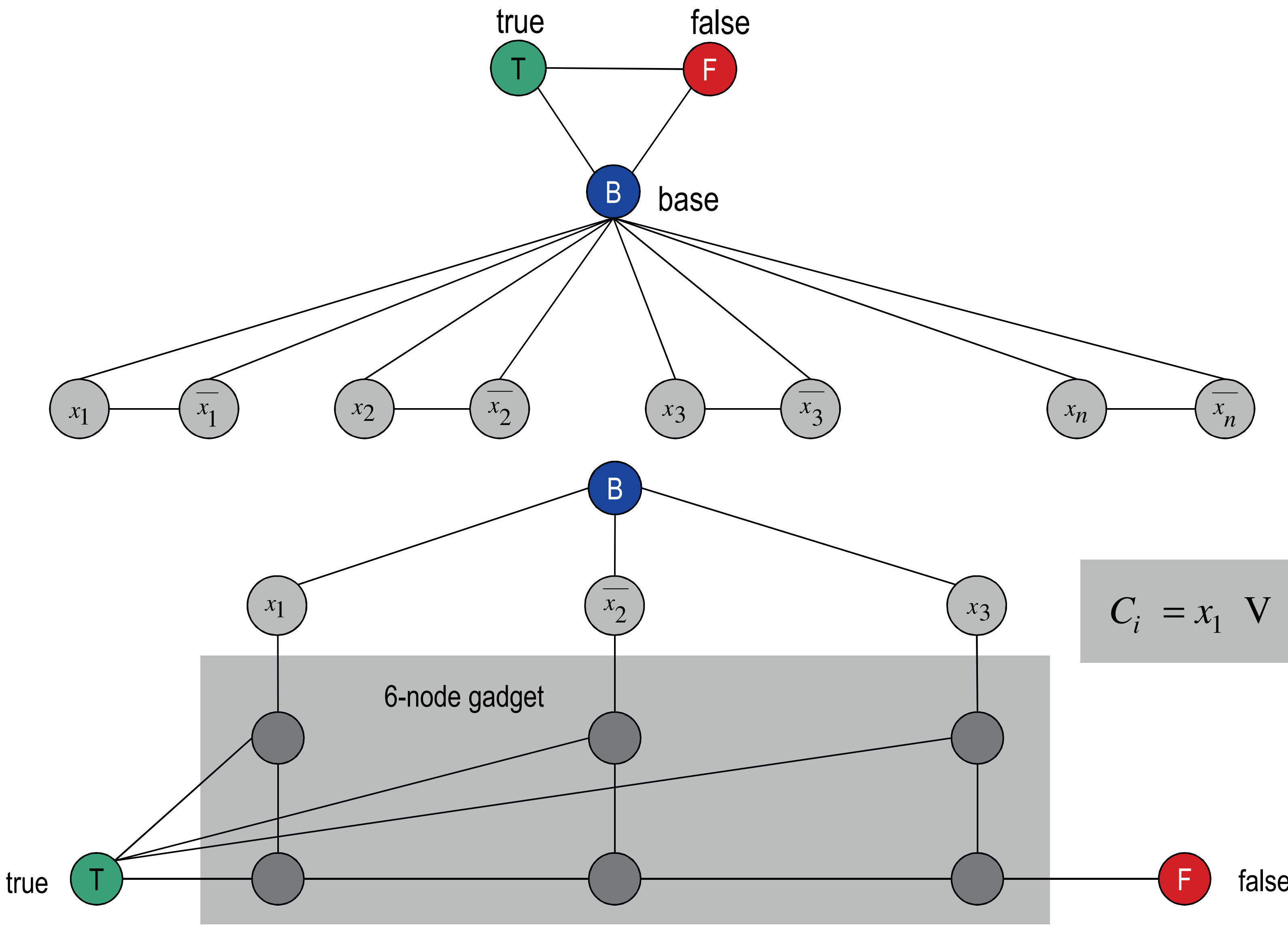
For each literal, create a node.

(i) Create 3 new nodes T, F, B; connect them in a triangle, and connect each literal to B.

(ii) Connect each literal to its negation.

(iii) For each clause, add gadget of 6 nodes and 13 edges. (to be described next)

3-Colorability: NP-Completeness



$$C_i = x_1 \vee \overline{x_2} \vee x_3$$

3-Colorability: NP-Completeness

Claim.

Graph is 3-colorable iff Φ is satisfiable.

$\Rightarrow$

Suppose graph is 3-colorable.

Consider assignment that sets all T literals to true.

(ii) ensures each literal is T or F.

(iii) ensures a literal and its negation are opposites.

Proving NP-Completeness

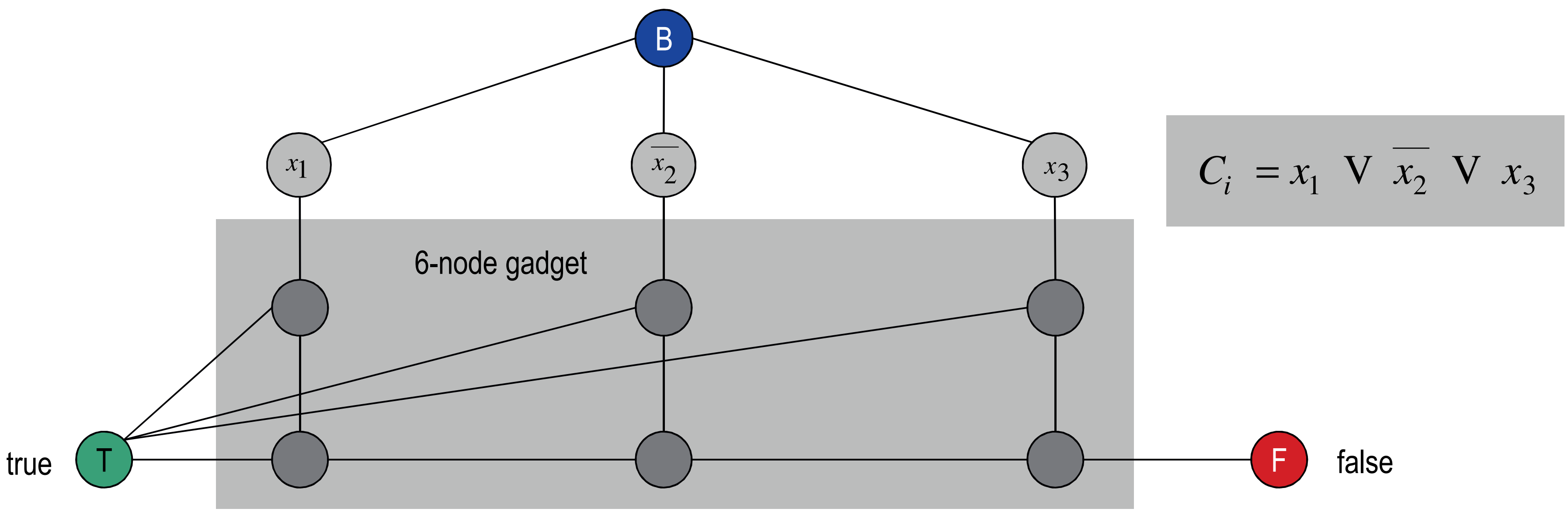
$\Rightarrow$ Suppose graph is 3-colorable.

Consider assignment that sets all T literals to true.

(ii) ensures each literal is T or F.

(iii) ensures a literal and its negation are opposites.

(iv) ensures at least one literal in each clause is T.

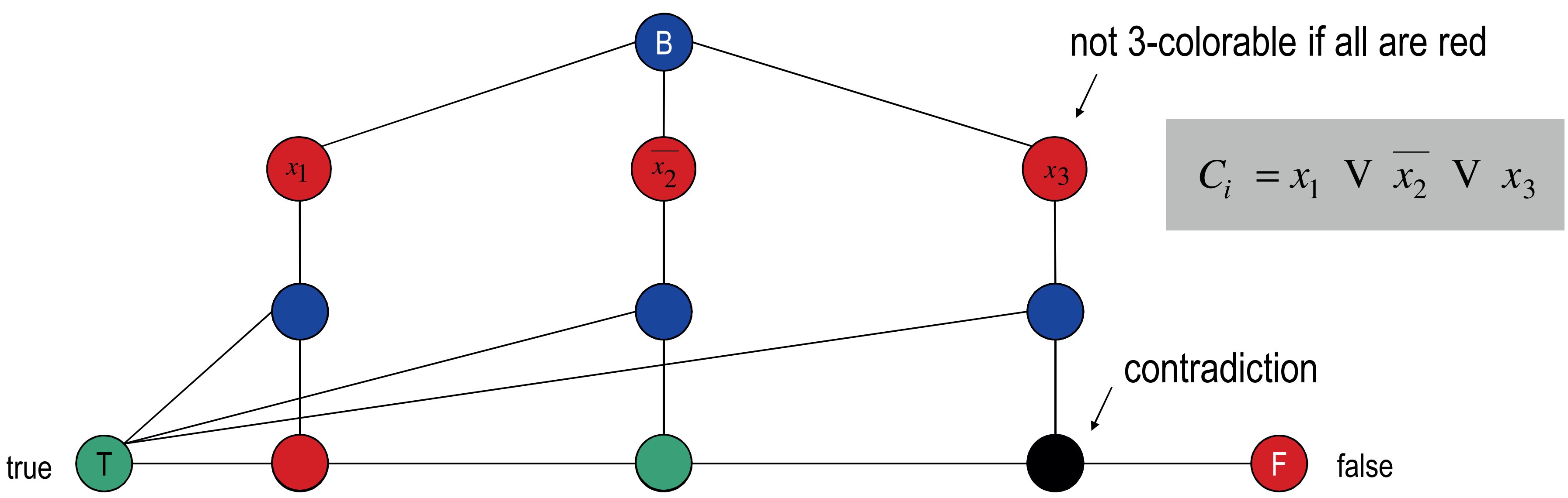


Proving NP-Completeness

⇒ Suppose graph is 3-colorable.

Consider assignment that sets all T literals to true.

- (ii) ensures each literal is T or F.
- (iii) ensures a literal and its negation are opposites.
- (iv) ensures at least one literal in each clause is T.



Proving NP-Completeness

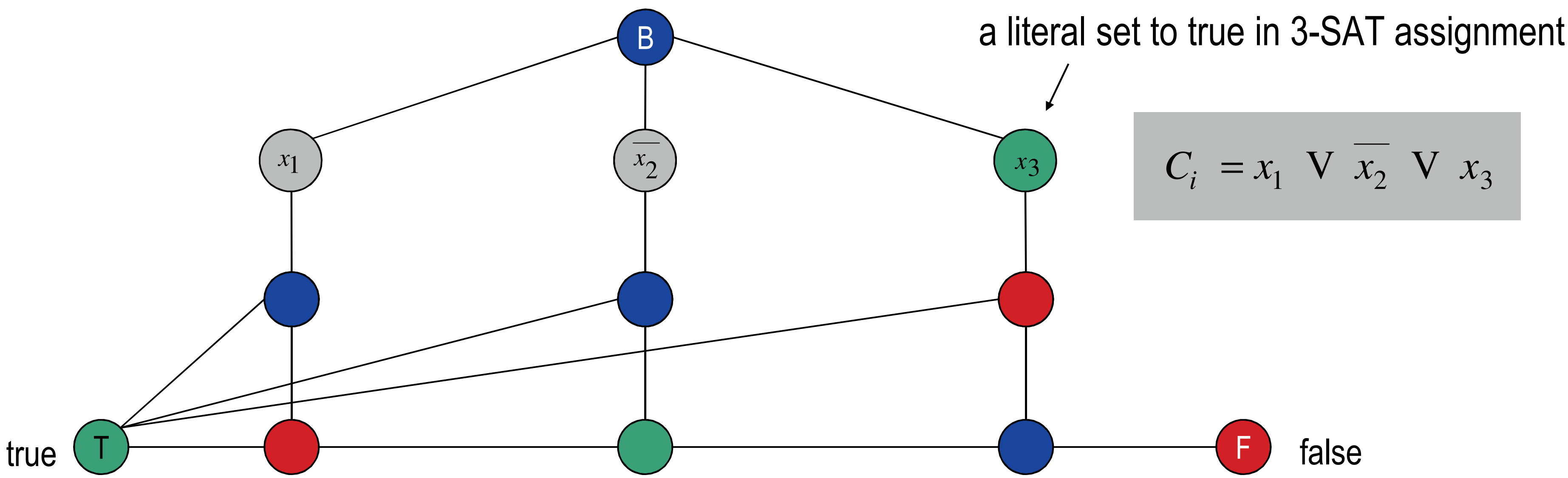
$\Leftarrow$ Suppose 3-SAT formula Φ is satisfiable.

Color all true literals T.

Color node below green node F, and node below that B.

Color remaining middle row nodes B.

Color remaining bottom nodes T or F as forced.



Subset Sum

The Subset Sum Problem

Instance

Given natural numbers $w_1, \ldots, w_n$ and an integer W

Objective:

Is there a subset that adds up to exactly W?

Example:

{ 1, 4, 16, 64, 256, 1040, 1041, 1093, 1284, 1344 }, W = 3754.

Yes. 1 + 16 + 64 + 256 + 1040 + 1093 + 1284 = 3754.

Remark.

With arithmetic problems, input integers are encoded in binary.

Polynomial reduction must be polynomial in binary encoding.

Subset Sum: NP-Completeness

Theorem

3-SAT $\leq$ Subset Sum

Proof

Given an instance Φ of 3-SAT, we construct an instance of SUBSET-SUM that has solution iff Φ is satisfiable.

Subset Sum: NP-Completeness

Construction.

Given 3-SAT instance Φ with n variables and k clauses, form 2n + 2k decimal integers, each of n+k digits, as illustrated below.

$$C_1 = \bar{x} \vee y \vee z$$
$$C_2 = x \vee \bar{y} \vee z$$
$$C_3 = \bar{x} \vee \bar{y} \vee \bar{z}$$

	x	y	z	C_1	C_2	C_3	
x	1	0	0	0	1	0	100,110
¬ x	1	0	0	1	0	1	100,001
y	0	1	0	1	0	0	10,000
¬ y	0	1	0	0	1	1	10,111
z	0	0	1	1	1	0	1,010
¬ z	0	0	1	0	0	1	1,101
	0	0	0	1	0	0	100
	0	0	0	2	0	0	200
	0	0	0	0	1	0	10
	0	0	0	0	2	0	20
	0	0	0	0	0	1	1
	0	0	0	0	0	2	2
W	1	1	1	4	4	4	111,444

dummies to get clause columns to sum to 4 (applies to the six unlabeled rows)