Computational Models – Lecture 13[1]

Nachum Dershowitz & Yishay Mansour

Tel Aviv University

June 2017

[1] Based on foils by Iftach Haitner, Benny Chor, Maurice Herlihy.

Talk Outline

- CSAT, 3SAT $\in \mathcal{NPC}$
- 2SAT
- CLIQUE, IND-SET $\in \mathcal{NPC}$
- Integer-Programming is NP-Complete

- Sipser, 7.4–7.5

Section 1

Reminder

Polynomial-time reducibility, reminder

Definition 1 (poly-time computable functions)

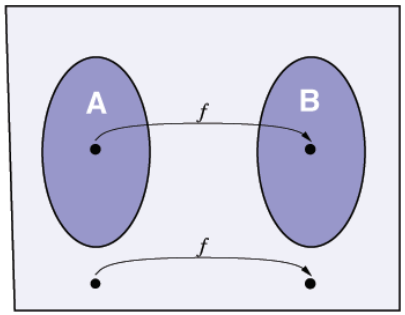
A function $f : \Sigma^* \to \Sigma^*$ is polynomial-time computable, if there is a poly-time deterministic TM that

- starts with input w, and
- halts with $f(w)$ on tape.

Definition 2 (polynomial-time reduction)

A polynomial-time computable $f : \Sigma^* \to \Sigma^*$ is a poly-time reduction from language A to B, if $x \in A \iff f(x) \in B$ for every $x \in \Sigma^*$.
Is such a reduction from A to B exists, we say that A is poly-time mapping reducible to B, denoted $A \leq_P B$.

The mapping f efficiently converts questions about membership in A to membership in B.



NP completeness, reminder

Definition 3 ($\mathcal{NP}$-complete)

A language B is NP-complete, if

- B $\in \mathcal{NP}$, and
- Every A $\in \mathcal{NP}$ is poly-time reducible to B (i.e., A $\leq_P$ B)

Let $\mathcal{NPC}$ denote the class of all $\mathcal{NP}$-complete languages.

Section 2

3SAT $\in \mathcal{NPC}$

CSAT

It is useful to consider a special version of SAT.

- A literal is a variable or negated variable: x or $\overline{x}$.
- A clause is several literals joined by $\vee$s: $(x_1 \vee \overline{x_2} \vee \overline{x_3})$
- A Boolean formula is in conjunctive normal form (CNF) if it consists of clauses, connected with $\wedge$s.
- For example: $(x_1 \vee \overline{x_2} \vee \overline{x_3} \vee x_4) \wedge (x_3 \vee \overline{x_5} \vee x_6) \wedge (x_3 \vee \overline{x_6})$
- The language of satisfied CNF formulas:

$$\text{CSAT} = \{\langle \phi \rangle : \ \phi \text{ is satisfiable CNF formula}\}$$

- Clearly, CSAT $\leq_P$ SAT (no need Cook-Levin for that).
- The *proof* of Cook-Levin can be modified to show that CSAT is NPC (take home exercise..)

3SAT

Definition 4

A Boolean formula is in k-CNF form, if it is a CNF formula, and all clauses have k literals.

Example of 3CNF: $(x_1 \vee \overline{x_2} \vee \overline{x_3}) \wedge (x_3 \vee \overline{x_5} \vee x_6) \wedge (x_3 \vee \overline{x_6} \vee x_4)$

The language of satisfied 3CNF formulas:

$$3SAT = \{\langle \phi \rangle : \ \phi \text{ is satisfiable 3CNF formula}\}$$

- Clearly $3SAT \leq_P SAT$.
- and $3SAT \in \mathcal{NP}$.
- We show next that $CSAT \leq_P 3SAT$, yielding that $3SAT \in \mathcal{NPC}$.
- Since 3SAT has more structure than CSAT, it is simpler to reduce it to other languages.

1SAT, 2SAT $\in \mathcal{P}$

Question 5

Is 1SAT $\in \mathcal{P}$?

Question 6

Is 2SAT $\in \mathcal{P}$?

Yes. . .

CSAT $\leq_P$ 3SAT

- The reduction maps CNF formulae to 3CNF ones "clause by clause".
- A clause with $d \leq 3$ literals is mapped to equivalent clause with 3 literals.

  $(x_1 \vee x_2) \longmapsto (x_1 \vee x_2 \vee x_2)$
- A clause with $d > 3$ literals is mapped to $d - 2$ clauses, built on the original literals together with $(d - 3)$ new variables.

  $\phi = (x_1 \vee \overline{x_2} \vee \overline{x_3} \vee x_4 \vee x_8) \longmapsto$
  $\phi_3 = (x_1 \vee \overline{x_2} \vee y_1) \wedge (\overline{y_1} \vee \overline{x_3} \vee y_2) \wedge (\overline{y_2} \vee x_4 \vee x_8)$

The reduction works!

Claim 7

ϕ has a satisfying assignment iff ϕ_3 does.

Proof's idea: (for case $d > 3$)
$\Leftarrow$ Given a satisfying assignment for ϕ_3, use the settings of the x_i's to get satisfying assignment for ϕ. Why does it work? For each set of clauses that came from a clause in ϕ, note that each y_i sets exactly one clause to true. There are $k-2$ clauses but only $k-3$ y_i's. So at least one clause needs to be satisfied by one of the x_j or $\overline{x_j}$ literals. Set this literal to true in ϕ.

$\Rightarrow$ An assignment satisfying ϕ, makes at least one literal per clause happy. In the "ϕ_3 clause" of this literal the new variables are under no constraints. Use the $d-2$ new variables to set the remaining $d-2$ clauses to "true".

CSAT $\leq_P$ 3SAT, cont.

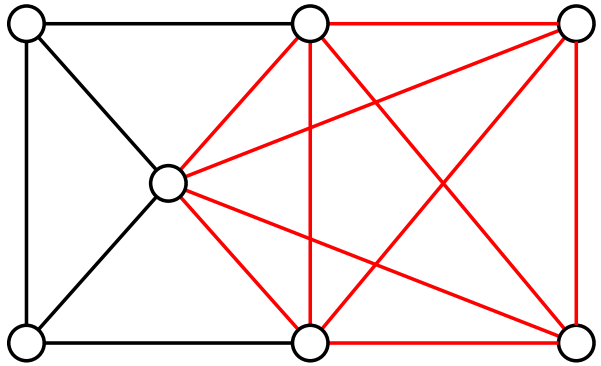
$\phi = (x_1 \vee \overline{x_2} \vee \overline{x_3} \vee x_4 \vee x_8) \longmapsto$
$\phi_3 = (x_1 \vee \overline{x_2} \vee y_1) \wedge (\overline{y_1} \vee \overline{x_3} \vee y_2) \wedge (\overline{y_2} \vee x_4 \vee x_8)$

- Doing the above mapping to each clause of a CNF formula, we get a 3CNF that is satisfied iff the original one is.
- Since this mapping is polynomial time (?), we get CSAT $\leq_P$ 3SAT. ♣.

Section 3

Clique is NPC

3SAT $\leq_P$ CLIQUE



CLIQUE $= \{\langle G, k\rangle : G$ is an undirected graph with a k-clique$\}$

Claim 8

3SAT $\leq_P$ CLIQUE.

Since CLIQUE $\in \mathcal{NP}$, it follows that CLIQUE $\in \mathcal{NPC}$.

Since IND-SET $\in \mathcal{NP}$, and CLIQUE $\leq_P$ IND-SET, it follows that IND-SET $\in \mathcal{NPC}$.

Proof's idea: We'll construct a poly-time reduction f that maps 3CNF formulae ϕ to pairs $\langle G, k\rangle$ of graphs and numbers.

The function f will have the property that ϕ is satisfiable, iff G has a clique of size k.

Proving 3SAT $\leq_P$ CLIQUE

On input ϕ, a 3CNF formula, the mapping reduction is defined as follows:

let k be the number of clauses in ϕ.

We construct a graph $G = G(\phi)$, see below, and output (G, k).

The graph G is defined as follows:

- Nodes in G are organized into triples $t_1, \ldots, t_k$.
- Each triple corresponds to a clause of ϕ
- Each node in a triple corresponds to a literal in corresponding clause.

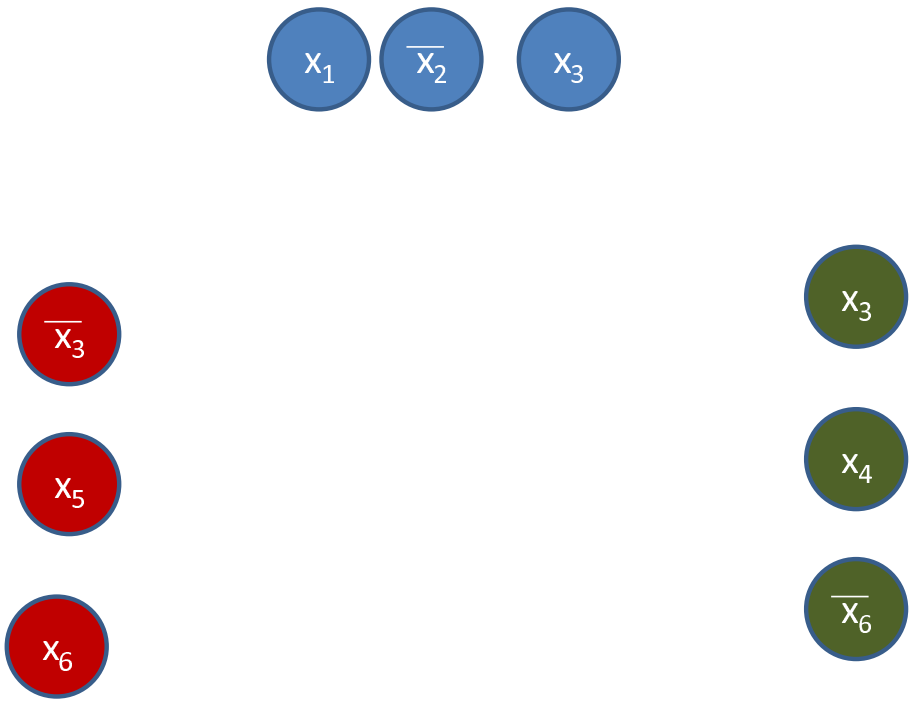
Ongoing example:

$$(x_1 \vee \overline{x_2} \vee x_3) \wedge (\overline{x_3} \vee x_5 \vee x_6) \wedge (x_3 \vee \overline{x_6} \vee x_4)$$

Nodes of G

$$\phi = (x_1 \vee \overline{x_2} \vee x_3) \wedge (\overline{x_3} \vee x_5 \vee x_6) \wedge (x_3 \vee x_4 \vee \overline{x_6})$$

Add a node per literal

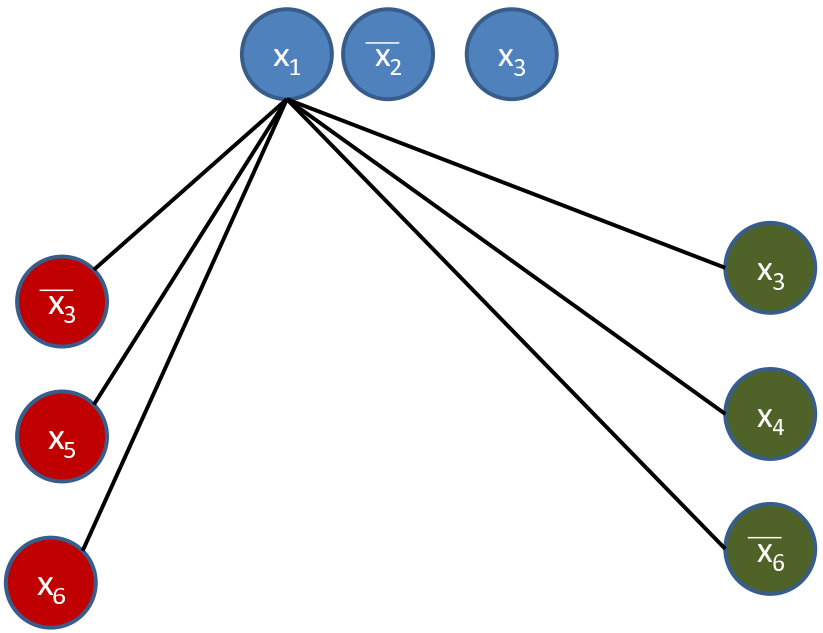


Edges of G

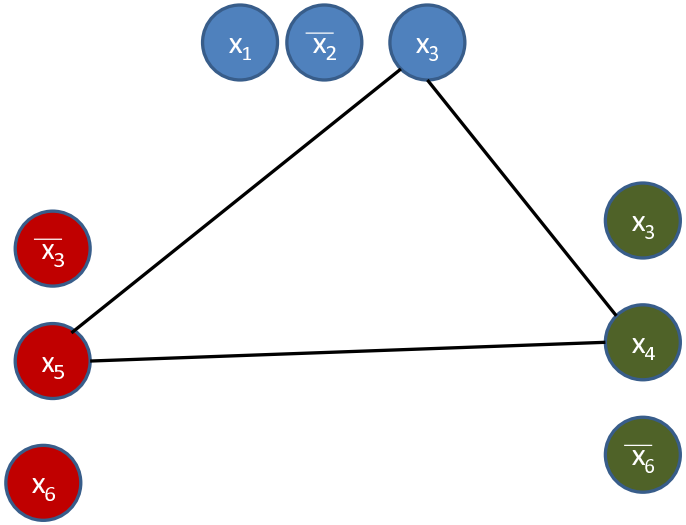
Add edges between all vertex pairs, except

- within same triple
- between contradictory literals (e.g., x_3 and $\overline{x_3}$)

$$\phi = (x_1 \vee \overline{x_2} \vee x_3) \wedge (\overline{x_3} \vee x_5 \vee x_6) \wedge (x_3 \vee x_4 \vee \overline{x_6})$$



$\phi \in 3SAT \implies \langle G, k \rangle \in CLIQUE$

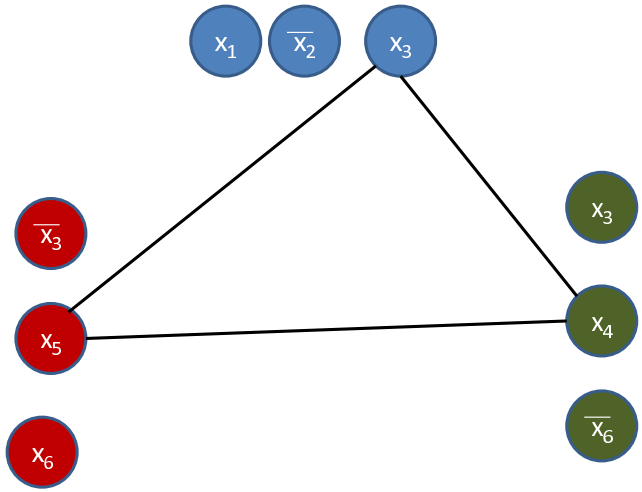


Proof: Suppose ϕ is satisfiable by an assignment ψ.
With respect to ψ:

- At least one literal is assigned to 1 in every clause (?)
- Select a 1-literal in every tuple;
- These literals can be joined by edges (?)

  Yielding a k-clique in G.

$\langle G, k \rangle \in \text{CLIQUE} \implies \phi \in \text{3SAT}$



Proof: Suppose G has a k-clique.

- No two of the cliques nodes are in the same triple. (?)
- G has $3k$ vertexes and k clauses, so each triple has exactly one clique node.
- Assign 1 to each node in clique
- Assignment has no contradictions (?)

  Yielding a satisfying assignment to ϕ.

Recap

- We've constructed a poly-time computable function f.
- We saw that f has the property that $\phi \in$ 3SAT iff $f(\phi) \in$ CLIQUE.
- Therefore, f is a poly-time reduction from 3SAT to CLIQUE

  $\Longrightarrow$ 3SAT $\leq_P$ CLIQUE. ♣

Section 4

Integer Programming (IP) is NPC

Integer Programming (IP)

Definition 9

A linear inequality has the form

$$a_1 x_1 + a_2 x_2 + \ldots + a_n x_n \leq b$$

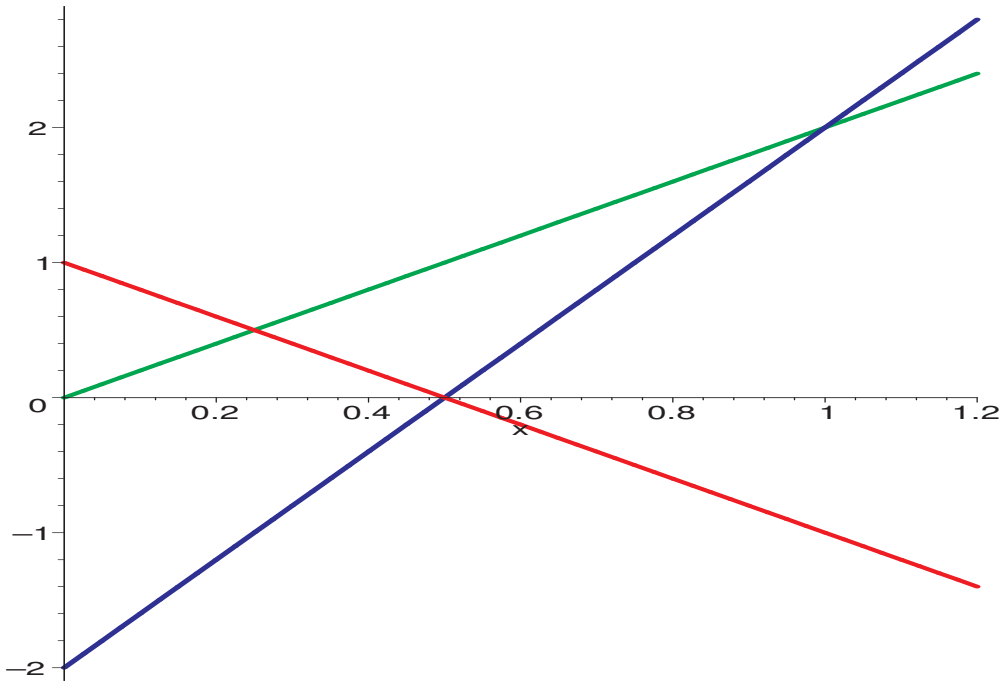
where $a_1, \ldots, a_n, b$ are real numbers, and $x_1, \ldots, x_n$ are real variables.

The Integer Programming (IP) problem]:
Input: A set of linear inequalities with integer coefficients in variables $x_1, x_2, \ldots, x_n$.
Question: Does this set has an integer solution – all x_i are integers?

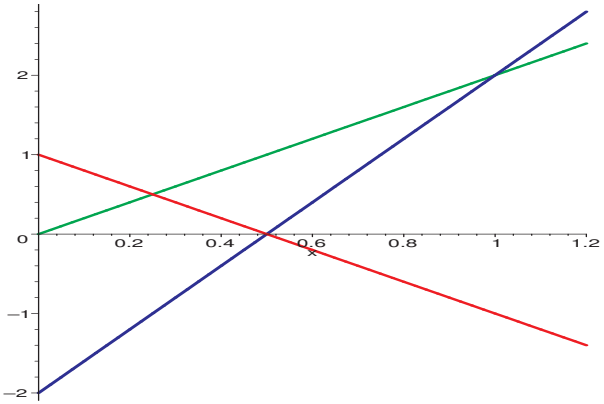
Integer Programming: Example

Consider the following system of linear inequalities

$$
\begin{aligned}
y &\leq 2x && \text{below green line} \\
-2x + 1 &\leq y && \text{above red line} \\
4x - 2 &\leq y && \text{above purple line} \\
0 \leq x &\leq 1 \\
0 \leq y &\leq 2
\end{aligned}
$$



Integer Programming: Example, 2



This set does have a unique (integer) solution: the right hand corner of the solid triangle, $(1, 2)$.

But if we change the constraint on y to $0 \leq y \leq 1$, then we'd have no solution with integer coordinates, even though there are many solutions with rational, or real, coordinates.

The Language IP

The language of integer linear inequalities with integer solution:

$$\text{IP} = \{\langle e_1, \ldots, e_m \rangle : \\ e_1, \ldots, e_m \text{ are integer linear inequalities with (joint) integer solution}\}$$

Theorem 10

$\text{IP} \in \mathcal{NPC}$.

IP $\in \mathcal{NP}$

Question 11

Do we always have a small enough certificate?

Consider the following equations

$$\begin{aligned} x_0 &= 1 \\ x_1 &= 2x_0 \\ &\vdots \\ x_m &= 2x_{m-1} \end{aligned}$$

The solution $x_m = 2^m$!

Question 12

Solving a linear set of m equations with m unknowns: $Ax = b$, where $|a_{i,j}|, |b_i| < M$. How large can $|x_i|$ get?

Answer: Recall Cramer's Rule, using determinants. $|A| < m! \cdot M^m$. (Same holds for inequalities.)
Hence, solution size is polynomial bounded.

CSAT $\leq_P$ IP

$$\text{CSAT} = \{\langle\phi\rangle : \ \phi \text{ is a satisfiable CNF Boolean formula}\}$$

Let ϕ be a CNF formula with ℓ clauses and k variables $x_1, \ldots, x_k$.

We reduce ϕ to an IP instance E with $2k$ variables $x_1, y_1, \ldots, x_k, y_k$, $\ell + 2k$ linear inequalities, and k linear equalities.

- Each x_i in ϕ corresponds to the variable x_i in E.
- Each $\overline{x_i}$ in ϕ corresponds to the variable y_i in E.
- For each i, we add E the inequalities $x_i \geq 0$, $y_i \geq 0$, and the equality $x_i + y_i = 1$ (what do these three express?)
- For each clause C of ϕ, we add E the inequality $\sum_{z \in C} z \geq 1$ (what does this inequality express?)

  For example, $(x_1 \vee \overline{x_2} \vee \overline{x_3} \vee x_4)$ is translated to $x_1 + y_2 + y_3 + x_4 \geq 1$.

Reduction Example

$\phi = (x_1 \vee \overline{x_2} \vee \overline{x_3} \vee x_4) \wedge (x_3 \vee \overline{x_5} \vee x_6) \wedge (x_3 \vee \overline{x_6})$ is translates to

$$
\begin{aligned}
x_1 + y_2 + y_3 + x_4 &\geq 1 \\
x_3 + y_5 + x_6 &\geq 1 \\
x_3 + y_6 &\geq 1 \\
x_1 \geq 0,\ y_1 \geq 0,\ x_1 + y_1 &= 1 \\
x_2 \geq 0,\ y_2 \geq 0,\ x_2 + y_2 &= 1 \\
x_3 \geq 0,\ y_3 \geq 0,\ x_3 + y_3 &= 1 \\
x_4 \geq 0,\ y_4 \geq 0,\ x_4 + y_4 &= 1 \\
x_5 \geq 0,\ y_5 \geq 0,\ x_5 + y_5 &= 1 \\
x_6 \geq 0,\ y_6 \geq 0,\ x_6 + y_6 &= 1
\end{aligned}
$$

CSAT $\leq_P$ IP: Validity (sketch)

Should show

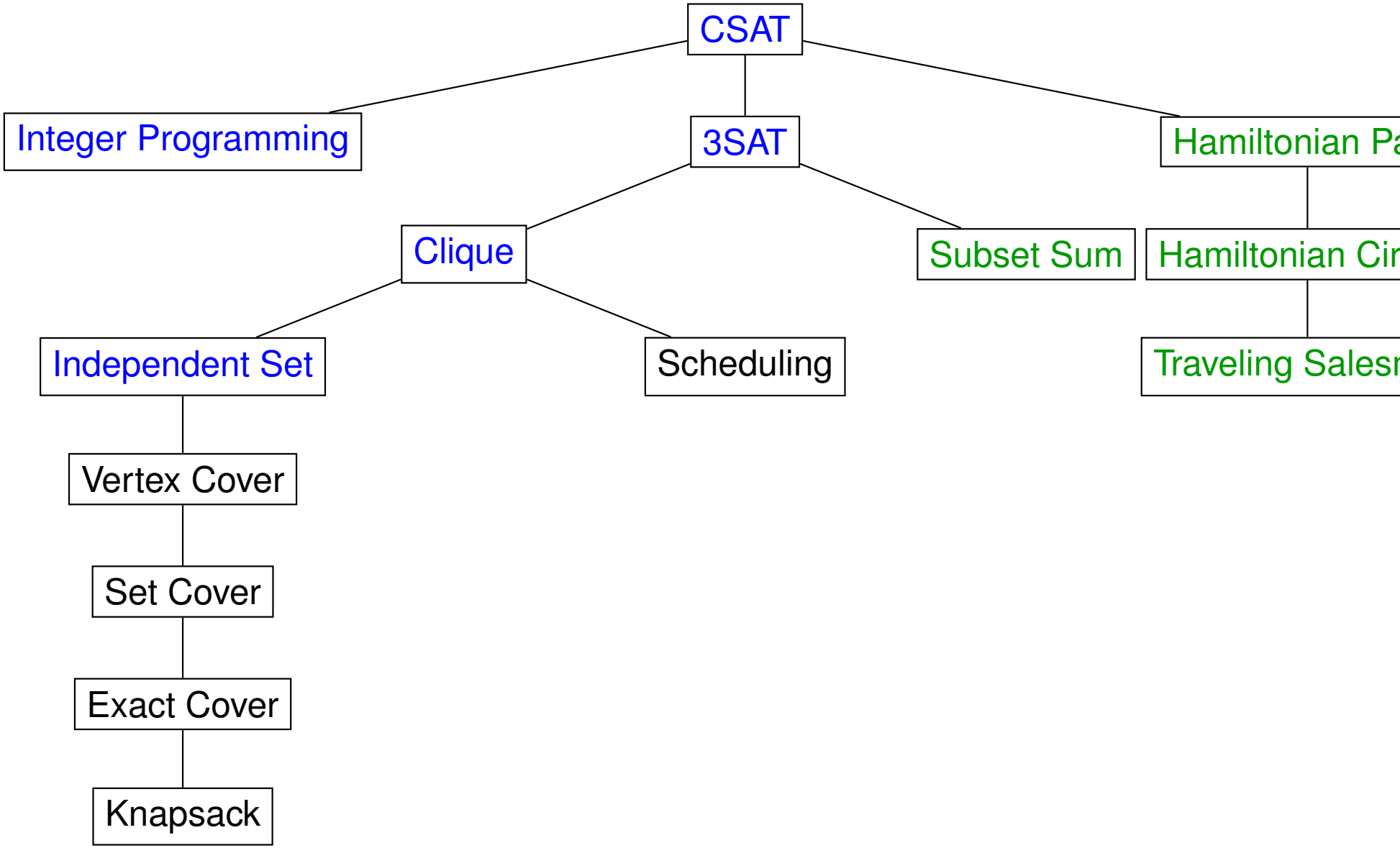
1. Reduction, g, is poly-time computable
2. $\phi \in \text{CSAT} \implies g(\phi) \in \text{IP}$
3. $g(\phi) \in \text{IP} \implies \phi \in \text{CSAT}$.

1. Poly time: easy (verify details!).
2. Suppose $\phi \in \text{CSAT}$. Take a satisfying assignment.
   If $x_i = 1$, in $g(\phi)$ assign $x_i = 1, y_i = 0$.
   If $x_i = 0$, in $g(\phi)$ assign $x_i = 0, y_i = 1$.
   - "Sanity check" constraints (i.e., $x_i \geq 0$, $y_i \geq 0$, $x_i + y_i = 1$) are satisfied.
   - "Clause constraints" (i.e., $\sum_{z \in C} z \geq 1$) are satisfied due to at least one literal satisfied in each clause.
   - Implying $g(\phi) \in \text{IP}$.
3. $g(\phi) \in \text{IP} \implies \phi \in \text{CSAT}$, is similar.

Section 5

Summary

Chains of reductions: NPC languages



List of NP-complete languages contains thousands of problems from combinatorics, operation research, VLSI design, computational geometry, bioinformatics, ...

The end