



2 Theory of complexity

How difficult are the problems to solve? How
some machine models allow faster solution than the others?

Example: Given NEA A . Is it true that $L(A) = \Sigma^*$?



Terminology and Convention

n : denotes the size of the input

Unit is to be still specified. bit, tape symbols, RAM machine word.

(decision) "Problem" : One language which can be recognized.

We talk here only about decidable problems.

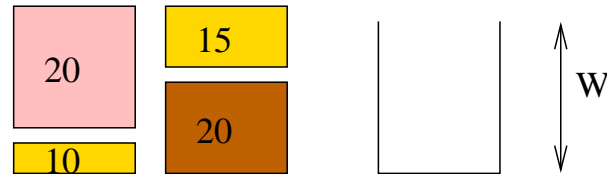


Complexity measures

- Time**. Here is the main thing
- Space
- Energy consumption
- Communication capacity
- Disk access
- Chip surface for Hardware implementing
- ...



Example Knapsacks problem

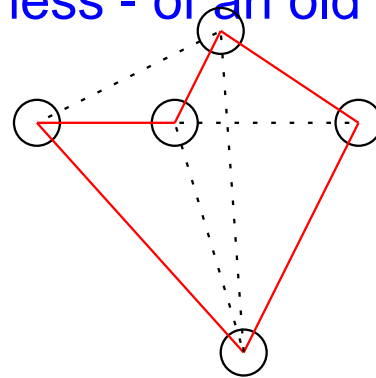


- Given n things with **weight** $w_i \in \mathbb{N}$ and **profit** p_i
- Is there a subset **x** of objects
- so that $\sum_{i \in \mathbf{x}} w_i \leq W$ and
- maximize the profit** $\sum_{i \in \mathbf{x}} p_i$



Example: Travelling Salesman Problem (TSP)

[The TSP- which it has to do, in order to receive orders and be certain a happy success in its business - of an old Commis Voyageur, 1832].



Given a graph $G = (V, V \times V)$, find a simple cycle

$C = (v_1, v_2, \dots, v_n, v_1)$ so that $n = |V|$ and $\sum_{(u,v) \in C} d(u, v)$ is minimal.

Formulate as decision problem: as we had



Hamilton cycle problem

[Hamilton, William Rowan, Memorandum respecting a new system of roots of unity. Philosophical Magazine, 12 1856]

$$M := \{G = (V, E) : \exists C \subseteq E : |C| = |V|, C \text{ is a simple cycle}\}$$

Coding a graph $G = (V, E)$ as a word of $\{0, 1, \#\}^*$:

$$V \subseteq \{0, 1\}^{\lceil \log |V| \rceil}$$

$$\text{Coding } w_G := \prod_{(u,v) \in E} u\#v\#$$



Minimal Steiner tree

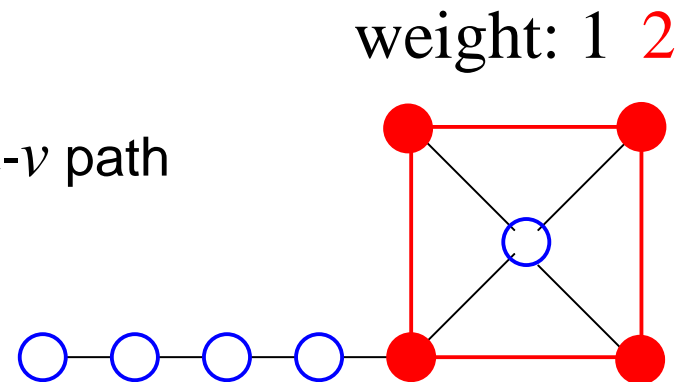
[C. F. Gauss 18??] Combinatorial optimization.

Given a graph $G = (V, E)$, with positive weighted edges $c : E \rightarrow \mathbb{R}_+$
 $V = R \cup F$, i.e., **Obligatory** nodes and **Steiner** nodes

find a tree $T \subseteq E$ with a minimal cost and all obligatory nodes are connected.

$$\forall u, v \in R : T \text{ exists } u\text{-}v \text{ path}$$

The network design problem





Upper bounds

Algorithms given and analysed.

Problem:

- Is the analysis exactly enough?
- Are there **better** algorithms?



Lower bounds

No algorithm could a better solution.

Trivial lower bounds :

$$T = \mathcal{O}(\text{inputSize} + \text{outputSize})$$

Problem: Hardly better bounds known!

We have to predicate over **all** algorithms!

Exceptions:

additional assumptions: for example $\Omega(n \log n)$ for **comparative sorting** of n elements

very limited models: for example $\Omega(n^2)$ for one tape TM-acceptor for $L_P := \{w : w = w^R\}$. **Communications complexity argument** collapses already by 2 tapes



Lower bounds solution

- Abstraction: Ignore smaller differences
- Classification: A set for problems that are "almost equally difficult".



A complexity class

$\text{time}_M(w)$ = Number of calculation steps of one TM M by an input w

$\text{TIME}(f(n)) =$

$$\{L : \exists \text{TM } M : L(M) = L \wedge \forall w \in \Sigma_M^* : \text{time}_M(w) \leq f(|w|)\}.$$

Here **many-tapes** TM.



Polynomial

A function $p : \mathbb{N} \rightarrow \mathbb{N}$ of the form

$$p(n) = a_k n^k + a_{k-1} n^{k-1} + \dots + a_1 n + a_0$$

with $a_i, k \in \mathbb{N}$



Complexity class P

$$\mathbf{P} := \bigcup_{\text{Polynomial } p} \text{TIME}(p(n))$$

By analogy we define in polynomial time computable **function**.



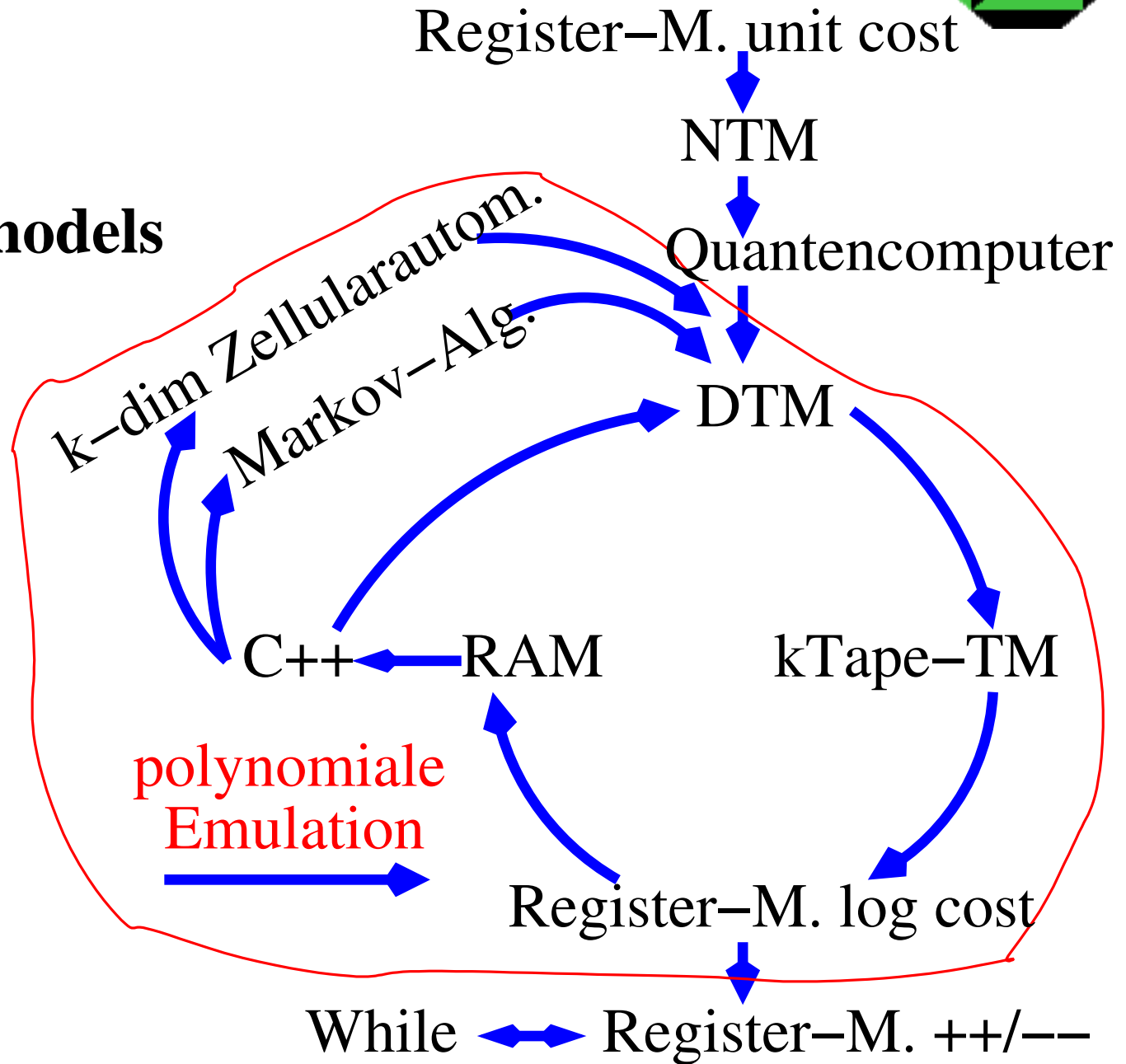
Complexity class **P**

$$\mathbf{P} := \bigcup_{\text{Polynomial } p} \text{TIME}(p(n))$$

Interpretation/Convention: Problem in **P** is **efficiently** solvable



P for different machine models





Transformation optimizing problem \rightarrow decision-making problem

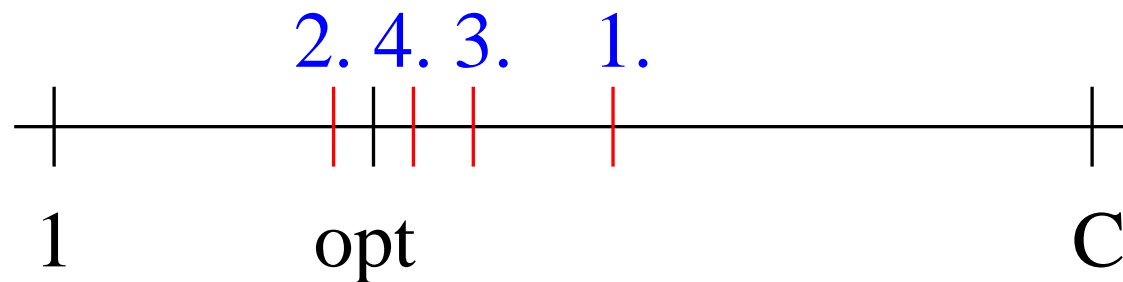
Assume: Target function value $\in 1..C$, whole-numbered.

Binary Search: $\lceil \log C \rceil$ Decision problem solved.

Polynomial in n , when $\log C$ polynomial in n .

Also if C has polynomial many bits.

This is the price of output complexity!





Another complexity class

Let M be a **non-deterministic** Turing machine

$$\text{ntime}_M(w) := \begin{cases} \min \{ |P| : P = (s)w \vdash^* u(f)v, f \in F \} & \text{if } w \in L(M) \\ 0 & \text{otherwise} \end{cases}$$

$$\text{NTIME}(f(n)) := \{ L : \exists \text{NTM } M : L(M) = L \wedge \forall w \in \Sigma_M^* : \text{ntime}_M(w) \leq f(|w|) \}$$



Complexity class NP

$$\mathbf{NP} := \bigcup_{\text{Polynomial } p} \mathbf{NTIME}(p(n))$$



Example: Knapsack problem

//Is there $x_1 \cdots x_n \in \{0, 1\}^n : \sum_i x_i w_i \leq W \wedge \sum_i x_i p_i \geq P$?

Procedure knapsack($\langle w_1, \dots, w_n \rangle, \langle p_1, \dots, p_n \rangle, W, P$)

for $i := 1$ **to** n **do** nondeterministically **guess** $x_i \in \{0, 1\}$

if $\sum_i x_i w_i > W$ **then** reject

if $\sum_i x_i p_i < P$ **then** reject

accept

Knapsack $\in NP$



Alternative definition for NTIME: oracle

A **DTM** M **oracle-accepted** $w \in \Sigma^*$ in time $t = \text{otime}_M(w)$ if

$\exists o \in \Gamma^* : M$ started by $o(s)w$

halts in t transitions

in a configuration $x(f)y$ with $f \in F$.

If $w \in \Sigma^*$ for M is **not** oracle-accepted, then $\text{otime}_M(w) := 0$.

$\text{OTIME}(f(n)) := \{L : \exists \text{DTM } M : L(M) = L \wedge \forall w \in \Sigma_M^* : \text{otime}_M(w) \leq f(|w|)\}$



Equivalence for NTIME and OTIME

NTM emulates DTM with oracle:

Nondeterministic computation „advises“ the oracle o

oracle-TM emulates NTM:

oracle gives the nondeterministic decisions.



The 1 000 000 \$ question

$$\mathbf{P = NP?}$$

One of 7 mathematical problems for which the
Clay Mathematics Institute
a price of
1 000 000 US\$
is announced.

Observation: $\mathbf{P \subseteq NP}$



Question for the public

100 researchers are questioned for **P = NP**

61: No

09: Yes

22: don't know

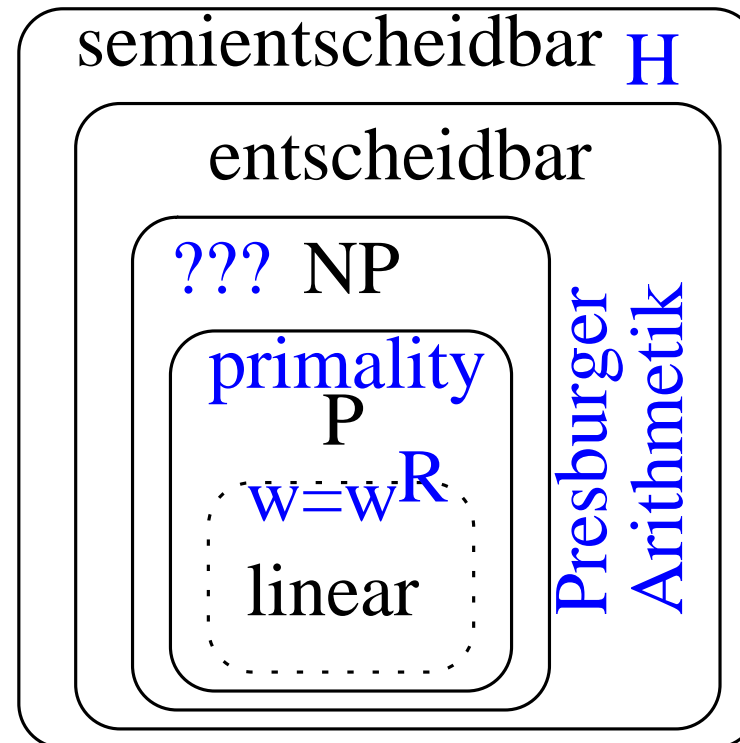
08: no answer

(independently of the accepted axioms.)

Why does this question so important?



A complexity hierarchy





Presburger Arithmetic

The decision problem for the first order calculus with the following restriction:

- Constants $0, 1$
- Variables $x_i \in \mathbb{Z}$
- Functions $+, -$
- Relations $<, =$
- Logical connectives \wedge, \vee, \neg
- Quantifiers \exists, \forall



Polynomial reductions

Let $A \subseteq \Sigma^*$ and $B \subseteq \Gamma^*$ be languages.

$A \leq_p B$ (A is polynomial reducible to B)

\Leftrightarrow

$\exists f : \Sigma^* \rightarrow \Gamma^* : \forall w \in \Sigma^* : w \in A \Leftrightarrow f(w) \in B$

where f is in **polynomial** time computable.



Example

Theorem: Hamilton Cycle \leq_p TSP

Proof:

Let $G = (V, E)$ be a non directed graph.

Define $d(u, v) = \begin{cases} 1 & \text{if } (u, v) \in E \\ 1 + \alpha & \text{else} \end{cases}$

Then, when G has a Hamilton cycle

\exists TSP tour with cost n

(otherwise optimal cost $\geq n + \alpha$)



Let $G = (V, E)$ be an arbitrary undirected graph.

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

\exists round tour C with weight n

—→ No edges in C have weight > 1

—→ All edges in C have weight 1

—→ all edges in C are edges in G

—→ C has a Hamilton cycle in G



Let $G = (V, E)$ be an arbitrary undirected graph.

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

G has Hamilton cycle C

→ C is a round tour with weight n .



Lemma: $A \leq_p B, B \in \mathbf{P} \rightarrow A \in \mathbf{P}$

Proof:

Let $A \leq_p B$ by function f .

Let M_f be a TM, such that f is computable by polynomial time bound p .

Further let $B \in \mathbf{P}$ by polynomial time bound q by TM M_B .

Consider **compositions** TM $M_A := (M_f; M_B)$.

M_A decides A .

The time calculation for an input w :

$$p(|w|) + q(|f(w)|) \leq p(|w|) + q(|w| + p(|w|))$$

It is polynomial in $|w| = n$



Lemma: $A \leq_p B, B \in \mathbf{NP} \rightarrow A \in \mathbf{NP}$

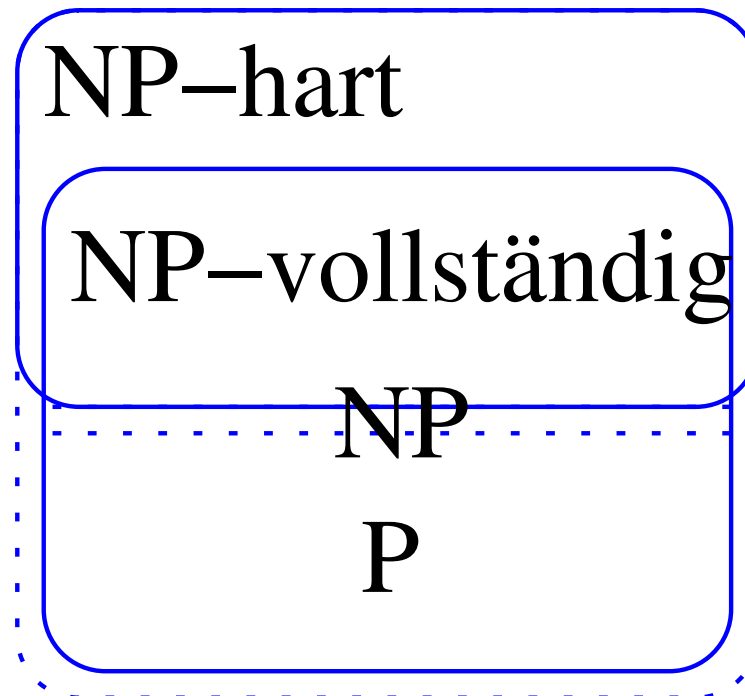
Proof: analog.



NP-hard and NP-complete problems

A is **NP-hard**: $\Leftrightarrow \forall L \in \mathbf{NP} : L \leq_p A$

A is **NP-complete**: $\Leftrightarrow A$ is **NP-hard** and $A \in \mathbf{NP}$.





A simple way to Glory and Richness?

Proposition: Let A be **NP-complete**. Then: $A \in \mathbf{P} \Leftrightarrow \mathbf{P} = \mathbf{NP}$

Proof:

Case $\mathbf{P} = \mathbf{NP}$:

$A \in \mathbf{NP} = \mathbf{P}$ also in particular $A \in \mathbf{P}$

Case $A \in \mathbf{P}$:

Let $L \in \mathbf{NP}$ be an arbitrary.

Since A is **NP-hard** then $L \leq_p A$

and because of $A \in \mathbf{P}$ it follows $L \in \mathbf{P}$

Also $\mathbf{P} = \mathbf{NP}$



SAT: The satisfiability problem

given: Formula F in the **Boolean logic**

$(\wedge \vee \neg \rightarrow ()),$ Variables)

question: is F satisfiable?, i.e.

\exists evaluation of the variables in $\{0, 1\}$ so that the truth value of $(F) = 1$?

Formally:

SAT

$:= \{\text{code}(F) \in \Sigma^* : F \text{ is satisfiable formula in the Boolean logic}\},$

code is an appropriate coding of the formulas as strings.



Theorem: [Cook 1971] and [Levin 1971] SAT is **NP**-complete.



Proof for $\text{SAT} \in \text{NP}$

Procedure satisfiable(F)

guess values $\in \{0, 1\}$ for all variables in F

substitute variables by their values

evaluate the resulting truth value v of F

if $v = 1$ **then** accept



Proof of that SAT is NP-hard.

Let

$L \in \mathbf{NP}$ arbitrary,

$M = (\{1, \dots, k\}, \Sigma, \{1, \dots, \ell\}, \delta, 1, H)$ NTM where $L(M) = L$,
($\sqcup = \ell$)

$p(n)$ polynomial time bound for acceptance of L by M for,

$w = w_1 w_2 \cdots w_n \in \Sigma^*$ arbitrary input.

Basic approach: We show that $L \leq_p \text{SAT}$,

by constructing a formula F of **polynomial length**, so that

$$w \in L \Leftrightarrow F \text{ is satisfiable}$$



Variables for F

- $\text{state}_{tz} = 1 \Leftrightarrow$ after t steps M is in state z
- $\text{pos}_{ti} = 1 \Leftrightarrow$ after t steps M is in a tape position i
- $\text{band}_{tia} = 1 \Leftrightarrow$ after t steps the tape position i is labelled by a

Notice that

$$0 \leq t \leq p(n) \text{ and}$$
$$-p(n) \leq i \leq p(n) + 1$$

There are only $\mathcal{O}(p(n)^2)$ variables



The architecture of $F =$

$R \wedge$ „The var. describe **always** one **configuration**“

$A \wedge$ „The var. describe the **initially** configuration $(1)w$ “

$U_1 \wedge$ „At the **head position** the correspond **modifications** as δ says“

$U_2 \wedge$ „At **non head positions** nothing is changing“

E „ M **accepts** w in $\leq p(n)$ steps“



Proof $w \in L \rightarrow F$ is satisfiable

$w \in L$

$\rightarrow \exists$ a sequence of configurations $P = (1)w \vdash^* u(h)v$ with $h \in H$ for M , $|P| = p(n)$

P defines a variable assignment so that F to get the truth value 1.

$\rightarrow F$ is satisfiable.



Proof F satisfiable $\rightarrow w \in L$

F satisfiable by assigned truth values of variables \rightarrow

□ R will be fulfilled \rightarrow

The Var. describe at each time a configuration

□ A will be fulfilled \rightarrow

The var. describe initially the configuration $(1)w$

□ U_1, U_2 will be fulfilled \rightarrow

from t to $t + 1$ there is a transition according δ

□ E will be fulfilled \rightarrow

the computation ends in one accepting state.

$\rightarrow w \in L$



There can be only one

$G(v_1, \dots, v_m) = 1 \Leftrightarrow$ Exactly one $v_i = 1$.

Implementation:

$G(v_1, \dots, v_m) =$

$v_1 \wedge \neg v_2 \wedge \neg v_3 \wedge \dots \wedge \neg v_m \vee$

$\neg v_1 \wedge v_2 \wedge \neg v_3 \wedge \dots \wedge \neg v_m \vee$

$\neg v_1 \wedge \neg v_2 \wedge v_3 \wedge \dots \wedge \neg v_m \vee$

...

$\neg v_1 \wedge \neg v_2 \wedge \neg v_3 \wedge \dots \wedge v_m$

Size: $\mathcal{O}(m^2)$



Boundary conditions

„The variables always describe a configuration“

$$R = \bigwedge_t G(\text{state}_{t1}, \dots, \text{state}_{tk}) \wedge \\ G(\text{pos}_{t,-p(n)}, \dots, \text{pos}_{tp(n)}) \wedge \\ \bigwedge_i G(\text{band}_{ti1}, \dots, \text{band}_{til})$$

Size $\mathcal{O}(p(n)(k^2 + p(n)^2 + p(n)\ell^2)) = \mathcal{O}(p(n)^3)$



Initial condition

„The var. describe the **initial** configuration $(1)w$ “

$$\begin{aligned}
 A = & \text{state}_{01} \wedge \text{pos}_{01} \wedge \\
 & \bigwedge_{j=1}^n \text{band}_{0j} w_j \wedge \\
 & \bigwedge_{j=-p(n)}^0 \text{band}_{0j} \sqcup \wedge \\
 & \bigwedge_{j=n+1}^{p(n)} \text{band}_{0j} \sqcup
 \end{aligned}$$

$$\text{Total } \mathcal{O}(1 + 1 + n + p(n) + p(n) - n) = \mathcal{O}(p(n))$$



Transition conditions $U_1, t \rightarrow t + 1$

„At the **head position** corresponded transition according δ “

$$U_1 = \bigwedge_{t,z,i,a} \left((\text{state}_{tz} \wedge \text{pos}_{ti} \wedge \text{band}_{tia}) \right. \\ \left. \rightarrow \bigvee_{\{(z',a',y) \in \delta(z,a)\}} (\text{state}_{t+1,z'} \wedge \text{pos}_{t+1,i+y} \wedge \text{band}_{t+1,ia'}) \right)$$

whereby the head movements are interpreted as numbers,
 $L = -1, N = 0, R = +1$.

$$\text{Size } \mathcal{O}((p(n) \cdot k \cdot p(n) \cdot \ell) \cdot (k \cdot \ell \cdot 3)) = \mathcal{O}(p(n)^2)$$



Transition conditions $U_2, t \rightarrow t + 1$

„At the **non head positions** nothing is changing“

$$U_2 = \bigwedge_{t,i,a} ((\neg \text{pos}_{ti} \wedge \text{band}_{tia}) \rightarrow (\text{band}_{t+1,i,a}))$$

$$\text{Size } \mathcal{O}((p(n) \cdot p(n) \cdot \ell)) = \mathcal{O}(p(n)^2) = \mathcal{O}(p(n)^2)$$



End condition E

„ M accepts w in $\leq p(n)$ steps“

$$E = \bigvee_{t \leq p(n)} \bigvee_{z \in H} \text{state}_{t,z}$$

Total $\mathcal{O}(p(n))$



Total size von F

R Boundary condition

$$\mathcal{O}(p(n)^3)$$

A Initial condition

$$\mathcal{O}(p(n))$$

U_1 Transition condition 1

$$\mathcal{O}(p(n)^2)$$

U_2 Transition condition 2

$$\mathcal{O}(p(n)^2)$$

E End condition „ M accepts w in $\leq p(n)$ steps“

$$\mathcal{O}(p(n))$$

Total $\mathcal{O}(p(n)^3)$.

This is again **polynomial**

and can in this time automatically written.



Proposition: $\mathbf{NP} \subseteq \bigcup_{p \text{ polynomial}} \text{TIME}(2^{\mathcal{O}(p(n))})$

Proof:

Let $T = (Q, \Sigma, \Gamma, \delta, s, F)$ be NTM, that recognized L with time bound $p(n)$.

One get a **deterministic algorithm**, such that all **combinations** $\leq p(n)$ of nondeterministic choice **systematically try to fulfill**.

There are maximum

$$(3 \cdot |Q| \cdot |\Gamma|)^{p(n)} = 2^{\log_2(3 \cdot |Q| \cdot |\Gamma|)p(n)} = 2^{\mathcal{O}(p(n))}$$

such combinations.



Another NP-complete problems

Observation: \leq_p is transitive, i.e.,

$$\forall L, L', L'' : L \leq_p L' \wedge L' \leq_p L'' \longrightarrow L \leq_p L''.$$

Lemma A: $L \in \mathbf{NP} \wedge \mathbf{SAT} \leq_p L \longrightarrow L$ is **NP**-complete.

Proof: We have to prove $\forall L' \in \mathbf{NP} : L' \leq_p L$.

SAT is **NP**-complete also $L' \leq_p \mathbf{SAT}$.

By the associativity of \leq_p it follows that

$$L' \leq_p \mathbf{SAT} \leq_p L.$$

qed



Another NP-complete problems

Lemma A: $L \in \mathbf{NP} \wedge \mathbf{SAT} \leq_p L \longrightarrow L$ is **NP**-complete.

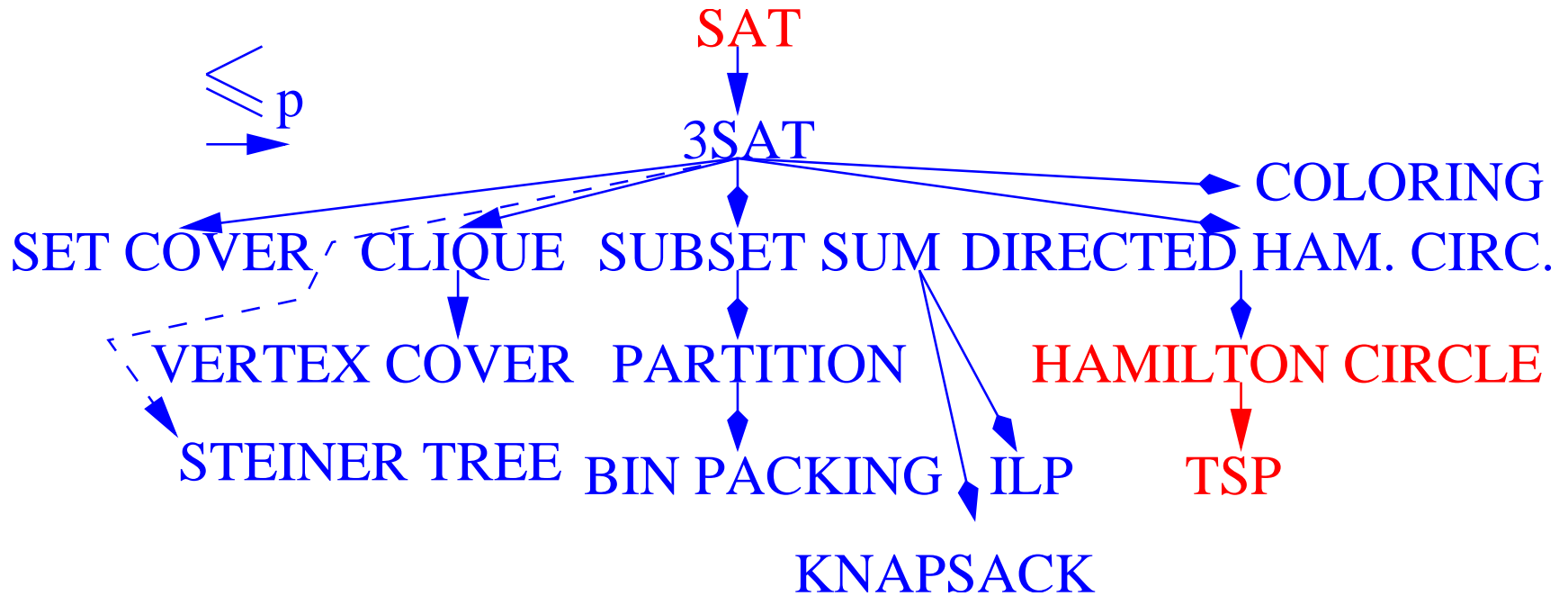
Lemma A':

$L \in \mathbf{NP} \wedge L'$ is NP complete $\wedge L' \leq_p L$
 $\longrightarrow L$ is **NP**-complete.

Proof: Analog of the proof of Lemma A.



Another NP-complete problems





3SAT (CNF)

Given: Boolean formula in **conjunctive normal form**, max. **3 Literals** per Clause

Question: Is F satisfiable?

Example: $(x \vee \neg y \vee z)(\neg u \vee x \vee z)(\neg x)$

Variable: x, y, z, \dots

Literal: Variable, \neg Variable

clause: Literal $\vee \dots \vee$ Literal

CNF formula: (clause) $\cdot \dots \cdot$ (clause)



Proposition: 3SAT is NP-complete

Proof:

Since SAT is in **NP**, it is enough to show that 3SAT is **NP-hard** and

$$\text{SAT} \leq_p \text{3SAT}$$

Case: It does not help to bring in CNF at all.

- CNF could be in exponential size
- One necessarily does not get 3CNF



Proof „3SAT is NP-hard“

We develop a polynomial Alg.,

such that it transforms a 3CNF-formula F in **satisfiable equivalent** formula F' , i.e.,

$$F \text{ satisfiable} \Leftrightarrow F' \text{ satisfiable}$$

Basic approach: a sequence of satisfiable equivalent **transformations**.

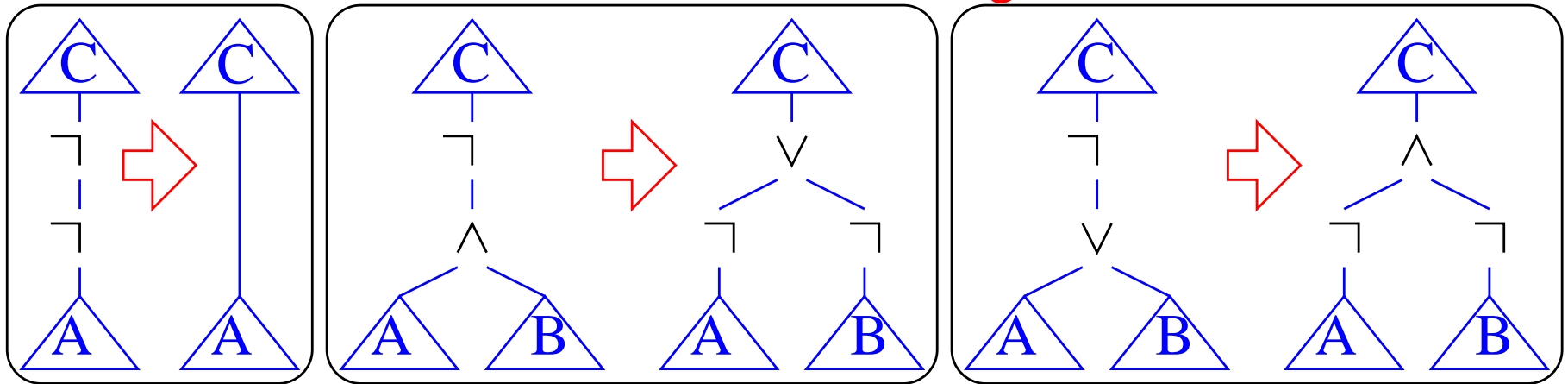
We consider the formula as a **tree** with max. grad two.

Leaves: variables x (or Literal $\bar{x} = \neg x$) Another Knots: \wedge, \vee, \neg



Putting the negations in the leaves

De Morgan

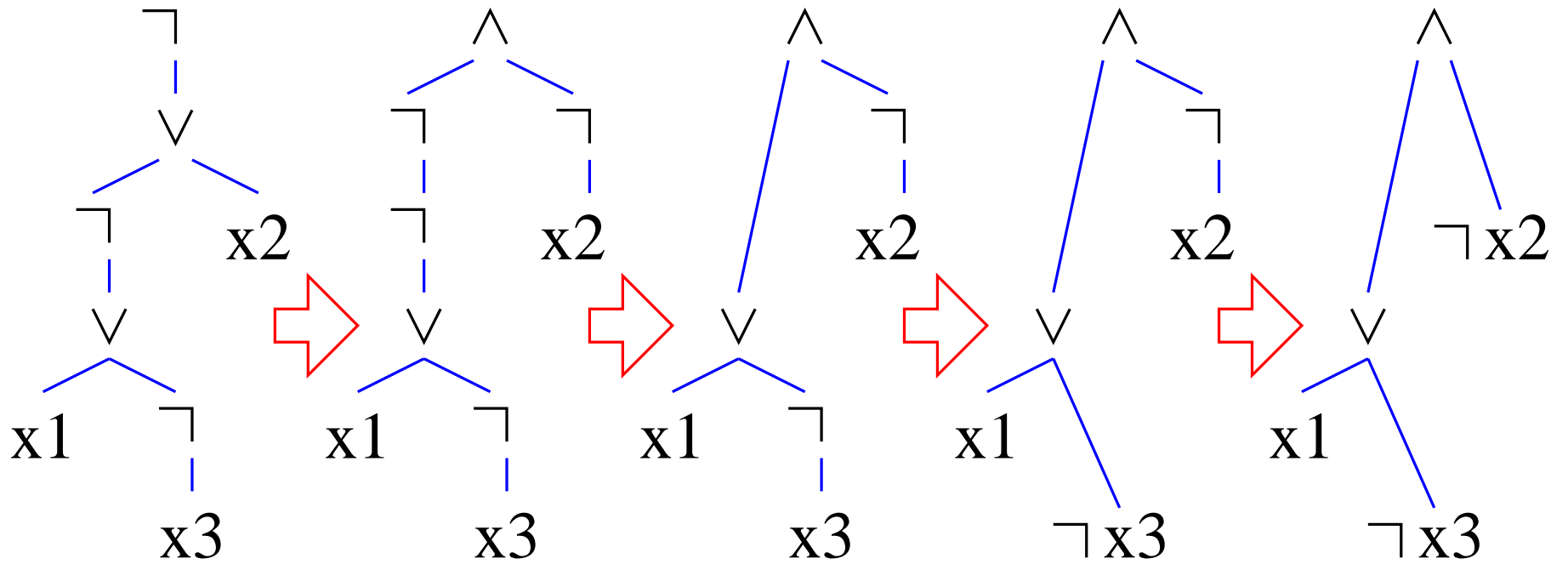


By traversing from up to down runs in linear time (RAM).

postcondition: formula letting as **binary tree** with inner knots \vee , \wedge and **Literals as leaves** interpreting.



Example





Putting the negations on the leaves

$\text{normalize}(\neg x)$ **return** $\text{negNormalize}(x)$

$\text{normalize}(x \wedge y)$ **return** $\text{normalize}(x) \wedge \text{normalize}(y)$

$\text{normalize}(x \vee y)$ **return** $\text{normalize}(x) \vee \text{normalize}(y)$

$\text{normalize}(v)$ **return** v

$\text{negNormalize}(\neg x)$ **return** $\text{normalize}(x)$

$\text{negNormalize}(x \wedge y)$ **return** $\text{negNormalize}(x) \vee \text{negNormalize}(y)$

$\text{negNormalize}(x \vee y)$ **return** $\text{negNormalize}(x) \wedge \text{negNormalize}(y)$

$\text{negNormalize}(v)$ **return** $\neg v$



Non leaves knots \rightarrow New variables

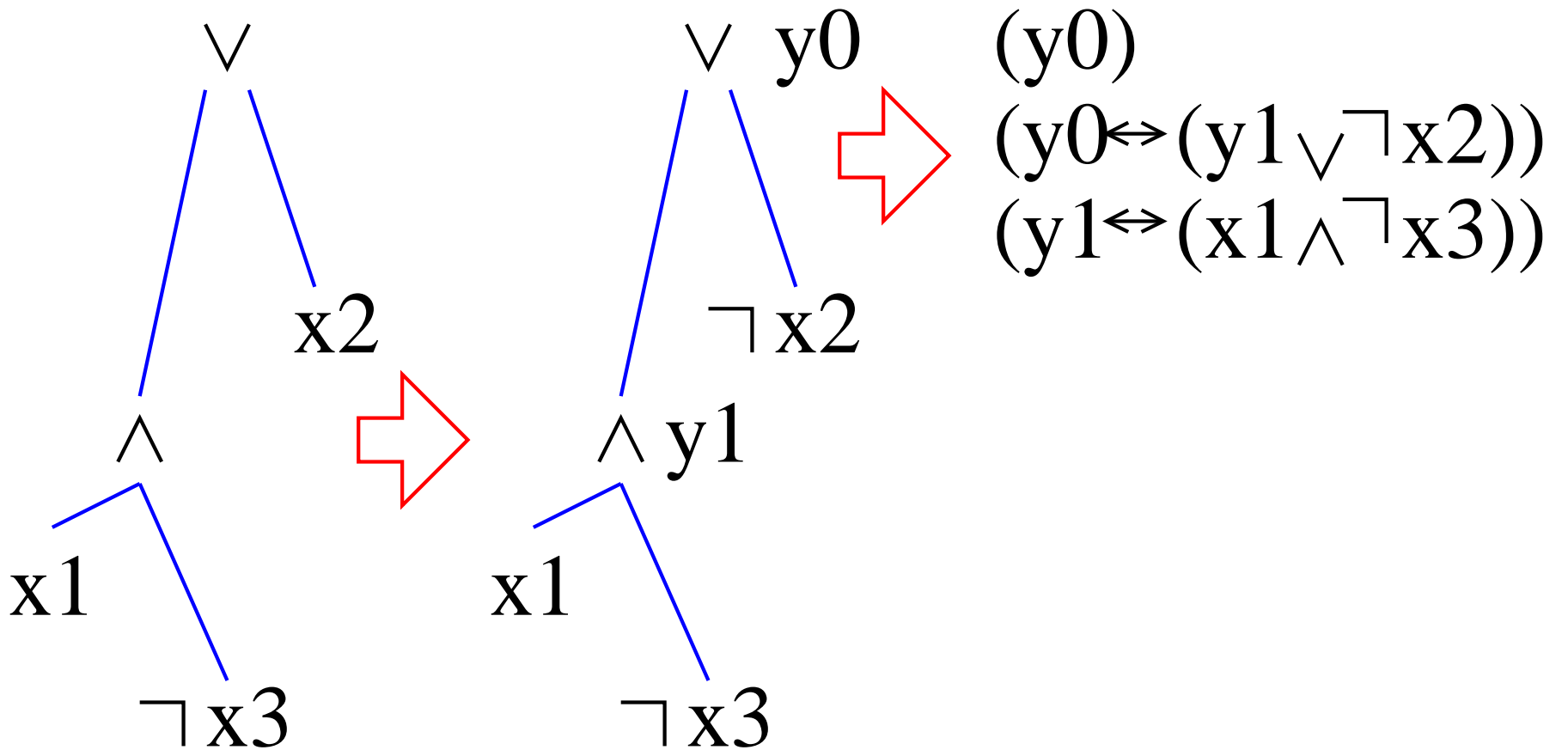
Arrange new variables for \wedge, \vee .

Let y_0 be the literal on the root.

$$F_1 := (y_0) \wedge \left(\bigwedge_{\substack{v \\ y \wedge z}} v \leftrightarrow (y \wedge z) \right) \wedge \left(\bigwedge_{\substack{v \\ y \vee z}} v \leftrightarrow (y \vee z) \right)$$



Example





Equivalence by satisfiability for F_1

Proof F is satisfiable $\rightarrow F_1$ is satisfiable:

Consider **satisfiable assignment** for F .

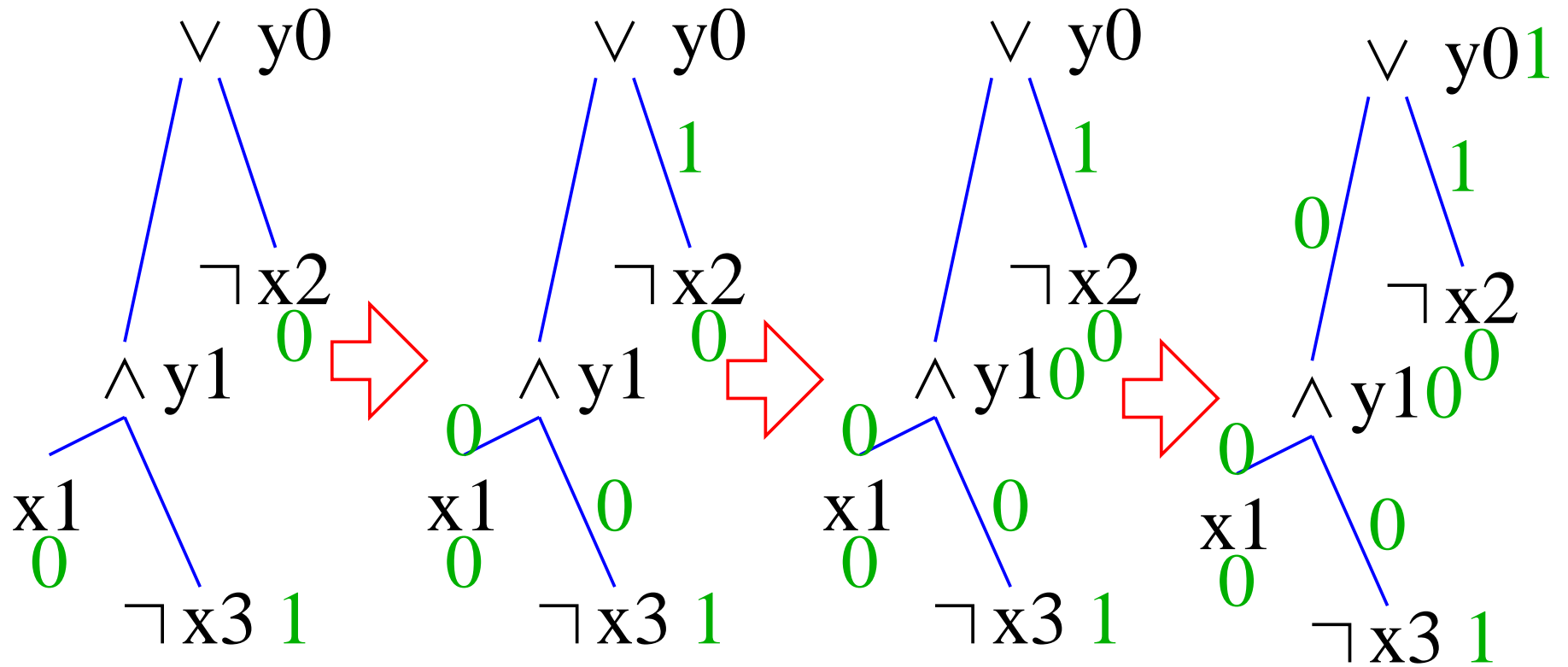
Take over these as **partial assignment for F_1** .

Evaluate the formula from bottom up.

Take over the resulting **truth values** of the subtree with the root variable v for the **assignment von v** .



Example





Equivalence by satisfiability for F_1

Proof F_1 is satisfiable $\rightarrow F$ is satisfiable:

Consider the assignment for F_1 .

Take over the truth values of the leaves for F .

Non leaves knots for F have the same truth value as the corresponding variables in F_1 .

y_0 is by 1 evaluated.

F is finally true.



$F_1 \rightsquigarrow 3\text{CNF}$

Build CNF for each subformula:

$$a \leftrightarrow (b \vee c) \rightsquigarrow (a \vee \neg b)(\neg a \vee b \vee c)(a \vee \neg c)$$

$$a \leftrightarrow (b \wedge c) \rightsquigarrow (\neg a \vee b)(\neg a \vee c)(a \vee \neg b \vee \neg c)$$

a	b	c	$b \vee c$	$b \wedge c$	$a \leftrightarrow (b \vee c)$	$a \leftrightarrow (b \wedge c)$
0	0	0	0	0	1	1
0	0	1	1	0	0	1
0	1	0	1	0	0	1
0	1	1	1	1	0	0
1	0	0	0	0	0	0
1	0	1	1	0	1	0
1	1	0	1	0	1	0
1	1	1	1	1	1	1



excursus: $2SAT \in P$

Given : Boolean formula in conjunctive normal form, max. 2 literal per clause.

Question: Is F satisfiable?

Define $\bar{x} := \neg x$, $\neg \bar{x} := x$.

graph $G = (V, E)$, $V := \{x, \neg x : x \in F\}$,
 $E := \{(\bar{A}, B), (\bar{B}, A) : (A \vee B) \in F\}$

Observation: F satisfiable

$\neg \exists$ cycle C in G , variable $x : x \in C \wedge \neg x \in C$.

These conditions we can verify in linear time.

(Strong connected component.)



SET COVER

Given:

Basic set M

Set system $\mathcal{T} \subseteq 2^M$

Parameter n

Question:

Are there elements $T_1 \in \mathcal{T}, \dots, T_n \in \mathcal{T}$?

$T_1 \cup \dots \cup T_n = M$?

Example:

$\{1, 2, 3, 4, 5\}$

$\{\{1, 3\}, \{1, 2\}, \{2, 4, 5\}\}$

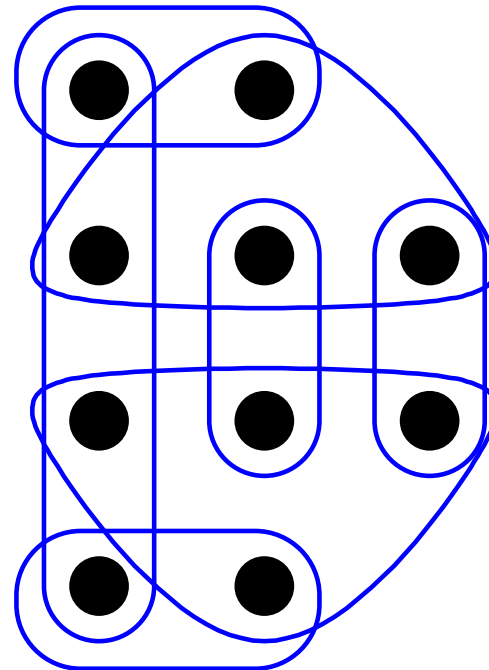
2

Yes:

$\{1, 3\}, \{2, 4, 5\}$



Set Cover Example



The minimal n : 3



Proposition: SET COVER is NP-complete

Proof of SET COVER \in NP:

Select nondet. n sets $T_1 \in \mathcal{T}, \dots, T_n \in \mathcal{T}$

Construct the union $M' = T_1 \cup \dots \cup T_n$

Check if $M = M'$



Proof „SET COVER is NP-hard“

We show $3SAT \leq_p SET\ COVER$.

Let $F = K_1 \wedge \dots \wedge K_m$ one CNF-formula with variables x_1, \dots, x_n .

Select $M = \{1, \dots, m+n\}$.

$T_i := \{j : x_i \text{ occurs in clause } K_j\} \cup \{m+i\}$

$T'_i := \{j : \neg x_i \text{ occurs in clause } K_j\} \cup \{m+i\}$

Set $\mathcal{T} := \{T_1, \dots, T_n, T'_1, \dots, T'_n\}$.

Parameter n .

To show: The Set Cover instance (M, \mathcal{T}, n) is solvable if F satisfiable.



Example

$$F = (x_1 \vee x_2 \vee x_3)(x_2 \vee \neg x_3 \vee x_4)(\neg x_3 \vee \neg x_4 \vee \neg x_1)$$

$$n = 4, m = 3.$$

$$T_1 = \{1, 4\},$$

$$\{3, 4\} = T'_1$$

$$T_2 = \{1, 2, 5\},$$

$$\{5\} = T'_2$$

$$T_3 = \{1, 6\},$$

$$\{2, 3, 6\} = T'_3$$

$$T_4 = \{2, 7\},$$

$$\{3, 7\} = T'_4$$

$$T_1 \cup T_2 \cup T'_3 \cup T_4 = \{1, 2, 3, 4, 5, 6, 7\}$$

$$\text{So } x_1 = x_2 = x_4 = 1, x_3 = 0$$



Proof F satisfiable $\longrightarrow (M, \mathcal{T}, n)$ is solvable.

Select $M = \{1, \dots, m + n\}$.

$T_i := \{j : x_i \text{ occurs in clause } K_j\} \cup \{m + i\}$

$T'_i := \{j : \neg x_i \text{ occurs in clause } K_j\} \cup \{m + i\}$

for $i = 1..n$

Select T_i in case of $x_i = 1$.

Select T'_i in case of $x_i = 0$.

$\{m + 1, \dots, m + n\}$ will be covered,

since for each variable x_i either T_i or T'_i will be selected.

$j \in \{1, \dots, m\}$ will be covered:

Let L be fulfilled literal in K_j

Case $L = x_i \longrightarrow x_i = 1 \longrightarrow j \in T_i$ and T_i will be selected.

Case $L = \neg x_i \longrightarrow x_i = 0 \longrightarrow j \in T'_i$ and T'_i will be selected.



Proof (M, \mathcal{T}, n) solvable $\longrightarrow F$ satisfiable

$$M = \{1, \dots, m + n\}.$$

$$T_i := \{j : x_i \text{ occurs in clause } K_j\} \cup \{m + i\}$$

$$T'_i := \{j : \neg x_i \text{ occurs in clause } K_j\} \cup \{m + i\}$$

$\{m + 1, \dots, m + n\}$ will be covered \longrightarrow

for every variable x_i either T_i or T'_i will be selected.

$$T_i \rightsquigarrow \text{set } x_i = 1.$$

$$T'_i \rightsquigarrow \text{set } x_i = 0.$$

$\forall j \in 1..m :$

Case 1, $j \in T_i$, T_i will be selected: $x_i \in K_j$ makes K_j true $(x_i = 1)$

Case 1, $j \in T'_i$, T'_i will be selected: $\neg x_i \in K_j$ makes K_j true $(x_i = 0)$



CLIQUE

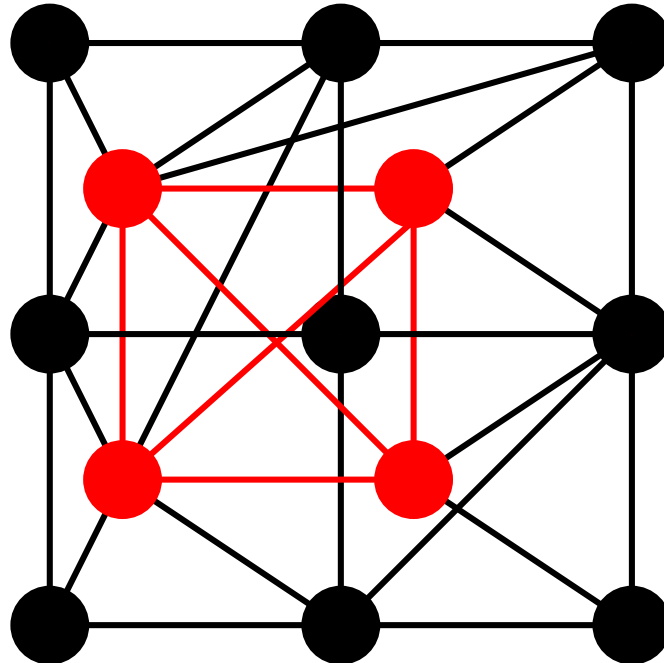
Given: A undirected graph $G = (V, E)$,

Parameter $k \in \mathbb{N}$.

Question:

Does G contain a **clique of size k** ?

i.e. $\exists V' \subseteq V : |V'| = k \wedge \forall u \neq v \in V' : \{u, v\} \in E$





Proof $\text{Clique} \in \text{NP}$

Guess k knots from V'

Check if they build a clique.

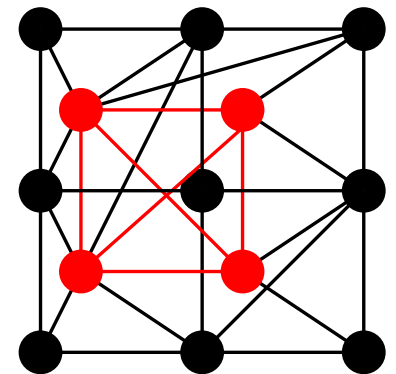
Given: Undirected graph $G = (V, E)$,

Parameter $k \in \mathbb{N}$.

Question:

Does G contain one **clique of size k** ?

i.e. $\exists V' \subseteq V : |V'| = k \wedge \forall u \neq v \in V' : \{u, v\} \in E$





Proof for „CLIQUE is NP-hard“

We show $3SAT \leq_p CLIQUE$.

Let $F = (z_{11} \vee z_{12} \vee z_{13}) \wedge \cdots \wedge (z_{m1} \vee z_{m2} \vee z_{m3})$ with literals $z_{ij} \in \{x_1, \dots, x_n\} \cup \{\neg x_1, \dots, \neg x_n\}$.

Construct a graph $G = (V, E)$:

$$V = \{(1, 1), (1, 2), (1, 3), \dots, (m, 1), (m, 2), (m, 3)\},$$

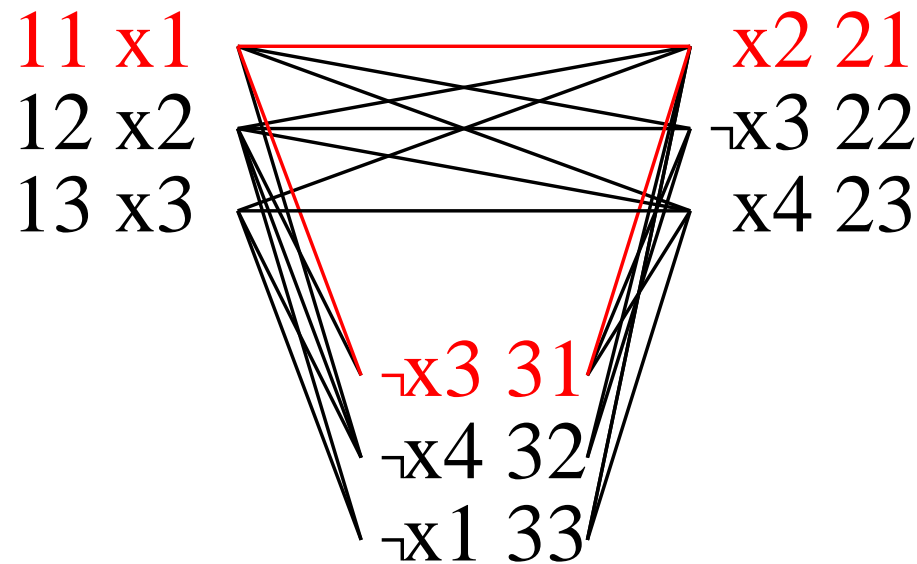
$$E = \{\{(i, j), (p, q)\} : i \neq p, z_{ij} \neq z_{pq}^-\}.$$

Parameter $k = m$.



Example:

$$F = (x_1 \vee x_2 \vee x_3)(x_2 \vee \neg x_3 \vee x_4)(\neg x_3 \vee \neg x_4 \vee \neg x_1)$$



Idea: since in each clause there is a fulfilled literal.

The edges connect „compatible“ literals



$$F = (z_{11} \vee z_{12} \vee z_{13}) \wedge \cdots \wedge (z_{m1} \vee z_{m2} \vee z_{m3}).$$

$$V = \{(1, 1), (1, 2), (1, 3), \dots, (m, 1), (m, 2), (m, 3)\},$$

$$E = \{\{(i, j), (p, q)\} : i \neq p, z_{ij} \neq z_{pq}^-\}.$$

F satisfiable $\longrightarrow G = (V, E)$ has m -clique

F satisfiable by assignment $B : \{x_1, \dots, x_n\} \rightarrow \{0, 1\}$

$\longrightarrow \forall j \in 1..m : \exists z_{j,k_j} : B$ makes z_{jk_j} true

These literals are pairwise **not complement each other**
and comes from **different clauses**.

$\longrightarrow (1, k_1), \dots, (m, k_m)$ are pairwise **connected**.

We build also one **m -clique**

qed



$$F = (z_{11} \vee z_{12} \vee z_{13}) \wedge \cdots \wedge (z_{m1} \vee z_{m2} \vee z_{m3}).$$

$$V = \{(1, 1), (1, 2), (1, 3), \dots, (m, 1), (m, 2), (m, 3)\},$$

$$E = \{\{(i, j), (p, q)\} : i \neq p, z_{ij} \neq z_{pq}^-\}.$$

$G = (V, E)$ has m -clique $\longrightarrow F$ satisfiable

$(j_1, k_1), \dots, (j_m, k_m)$ is m -clique with literals z_{j_i, k_i} .

Only edges between different j_i .

\longrightarrow these j_i are pairwise different.

$\longrightarrow j_1, \dots, j_m = 1, \dots, m$

Only edges between non complemented literals.

\longrightarrow these literals are pairwise non complemented.

$\longrightarrow \exists$ assignment $B : \forall j \in 1..m : z_{jk_j}[B] = 1$

$\longrightarrow F$ satisfiable by B

qed

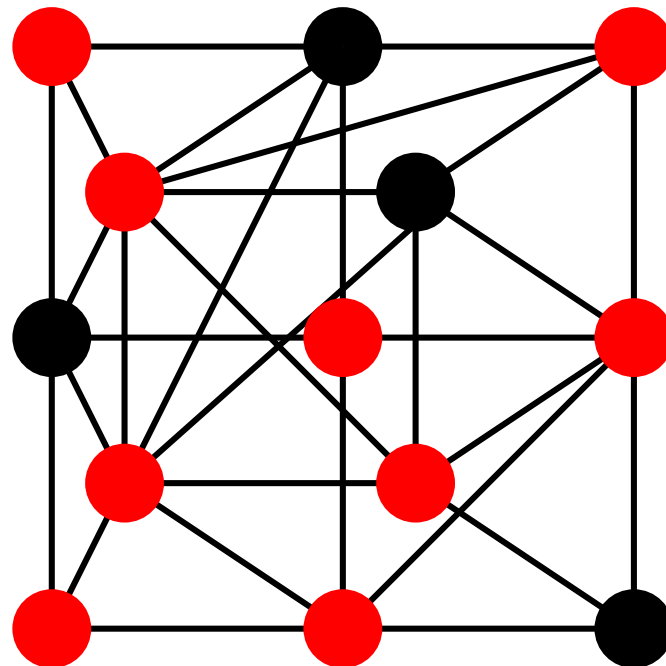


VERTEX COVER (knots covered)

Given: undirected graph $G = (V, E)$,

Parameter $k \in \mathbb{N}$.

Question: $\exists V' \subseteq V : |V'| = k \wedge \forall \{u, v\} \in E : u \in V' \vee v \in V'$





Proof VERTEX COVER \in NP

... as the previous...



Proof von „VERTEX COVER is NP-hard“

We show $\text{CLIQUE} \leq_p \text{VERTEX COVER}$.

Consider the **complemented graph**

$$\bar{G} = (V, \bar{E}) \text{ with } \bar{E} = \{u \neq v \in V : \{u, v\} \notin E\}.$$

G has **Vertex Cover** V' of size k

\Leftrightarrow

$$\exists V' \subseteq V : |V'| = k \wedge \forall u \neq v : \{u, v\} \in E \rightarrow (u \in V' \vee v \in V')$$

\Leftrightarrow

$$\exists V' \subseteq V : |V'| = k \wedge \forall u \neq v : \{u, v\} \notin E \vee (u \in V' \vee v \in V')$$

\Leftrightarrow

$$\exists V' \subseteq V : |V'| = k \wedge \forall u \neq v : \{u, v\} \notin E \vee \neg(u \notin V' \wedge v \notin V')$$

\Leftrightarrow

$$\exists V' \subseteq V : |V'| = k \wedge \forall u \neq v : (u \notin V' \wedge v \notin V') \rightarrow \{u, v\} \notin E$$

\Leftrightarrow

$$\exists V' \subseteq V : |V'| = k \wedge \forall u \neq v : \{u, v\} \in V \setminus V' \rightarrow \{u, v\} \in \bar{E}$$

\Leftrightarrow

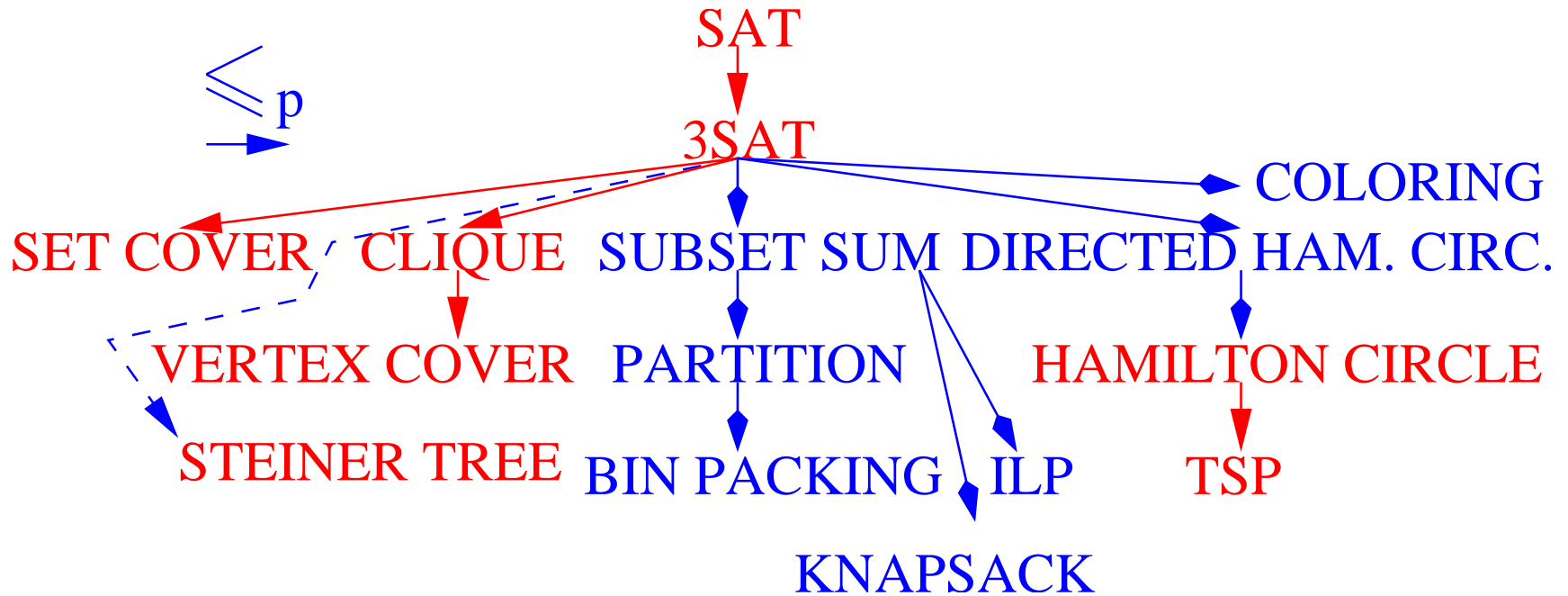
$$\exists \bar{V}' \subseteq V : |\bar{V}'| = |V| - k \wedge \forall u \neq v : \{u, v\} \in \bar{V}' \rightarrow \{u, v\} \in \bar{E}$$

\Leftrightarrow

\bar{G} has one **clique** of size $|V| - k$



Viewed Reductions





Proof technics for $A \leq_p B$:

Special instance

Show, that A is a special instance (in a wider sense) for B .

Every elem. object in the formulated problem for A corresponds to one elementary object in the formulated problem in B .

Example: HAMILTON CYCLE \leq_p TSP.

Example: SUBSET SUM \leq_p KNAPSACK.



Proof technics for $A \leq_p B$:

Another interpretation

Show, that A is a special instance (in a wider sense) for B .

Every elementary object in the formulated problem for A corresponds to one elementary object in the formulated problem in B .

Example: VERTEX COVER \leq_p CLIQUE.

Example: CLIQUE \leq_p VERTEX COVER.



Proof technics for $A \leq_p B$:

Gadgets

Every elementary object in the formulated problem for A will construct **more** connected objects in B .

Example: $3\text{SAT} \leq_p \text{SET COVER}$; $x_i \rightsquigarrow T_i, T'_i$.

Example: $\text{SAT} \leq_p 3\text{SAT}$; knots for $F \rightsquigarrow$ three clauses.

Example: $L \leq_p \text{SAT}$;

- States, head position, tape symbols \rightsquigarrow many variables.
- Initial configuration \rightsquigarrow formula A
- $\delta \rightsquigarrow$ transition fit U_1
- Final states \rightsquigarrow End condition E



Proof technics for $A \leq_p B$:

Border conditions

Force certain characteristics by additional construction.

Example: $L \leq_p \text{SAT}$; R, U_2 .

Example: $\text{SAT} \leq_p \text{3SAT}$; (y_0) .

Example: $\text{3SAT} \leq_p \text{COLORING}$



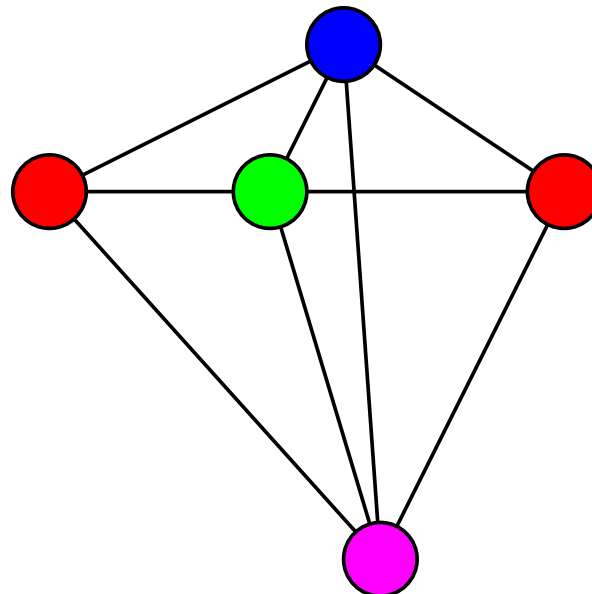
COLORING

Given: undirected graph $G = (V, E)$,

Parameter $k \in \mathbb{N}$.

Question:

Is there one knots coloring $c : V \rightarrow \{1, \dots, k\}$
with $\forall \{u, v\} \in E : c(u) \neq c(v)$.





Applications of coloring

- Maps coloring (4 are enough !)
- The set of knots in **independent sets** factoring.
For example for **parallelization**.
- Register allocation.
knots=In one procedure compute the value,
edges=The value could not be in the same register.
- ...

Related problem: Frequency of a radio sender



Proof COLORING \in NP

... the same ...



Proof von „COLORING is NP-hard“

We show $3\text{SAT} \leq_p \mathbf{3}\text{-COLORING}$ (i.e. $k = 3$).

Let $F = K_1 \wedge \dots \wedge K_m$ with literals from

$\{x_1, \dots, x_n\} \cup \{\neg x_1, \dots, \neg x_n\}$.

Exactly 3 literals per clause.

Idea:

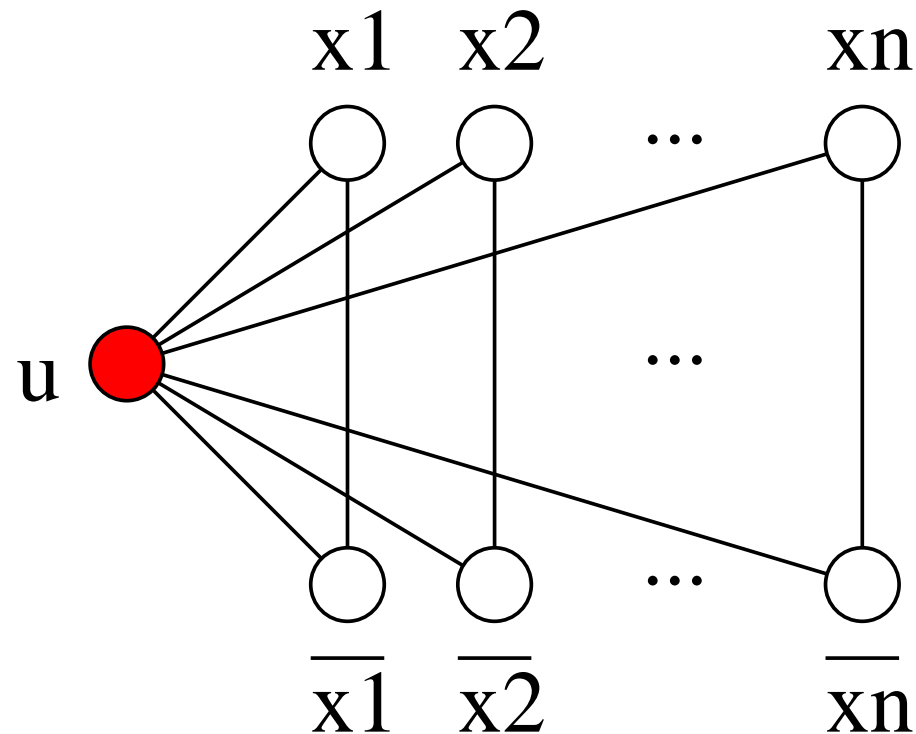
Construct a graph $G = (V, E)$ with $2n + 5m + 2$ knots, so that

G 3-coloring $\Leftrightarrow F$ satisfiable.



Boundary condition forces consistent truth values for literals

One part of G :



Colors $0 = \text{true}$, $1 = \text{false}$, $2 = \text{rot} = c(u)$.



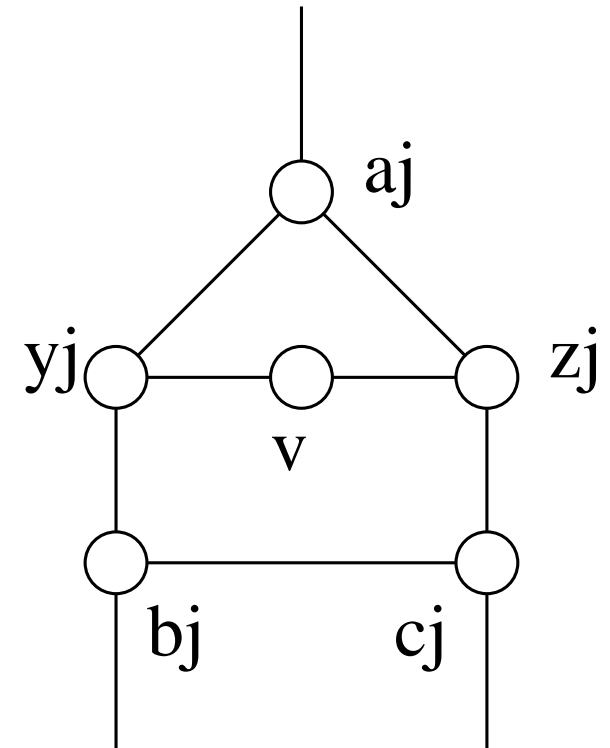
Gadget for clause $K_j = (a_j \vee b_j \vee c_j)$

v : All over equal $(u, v) \in E$

a_j, b_j, c_j : Own knots for every clause.

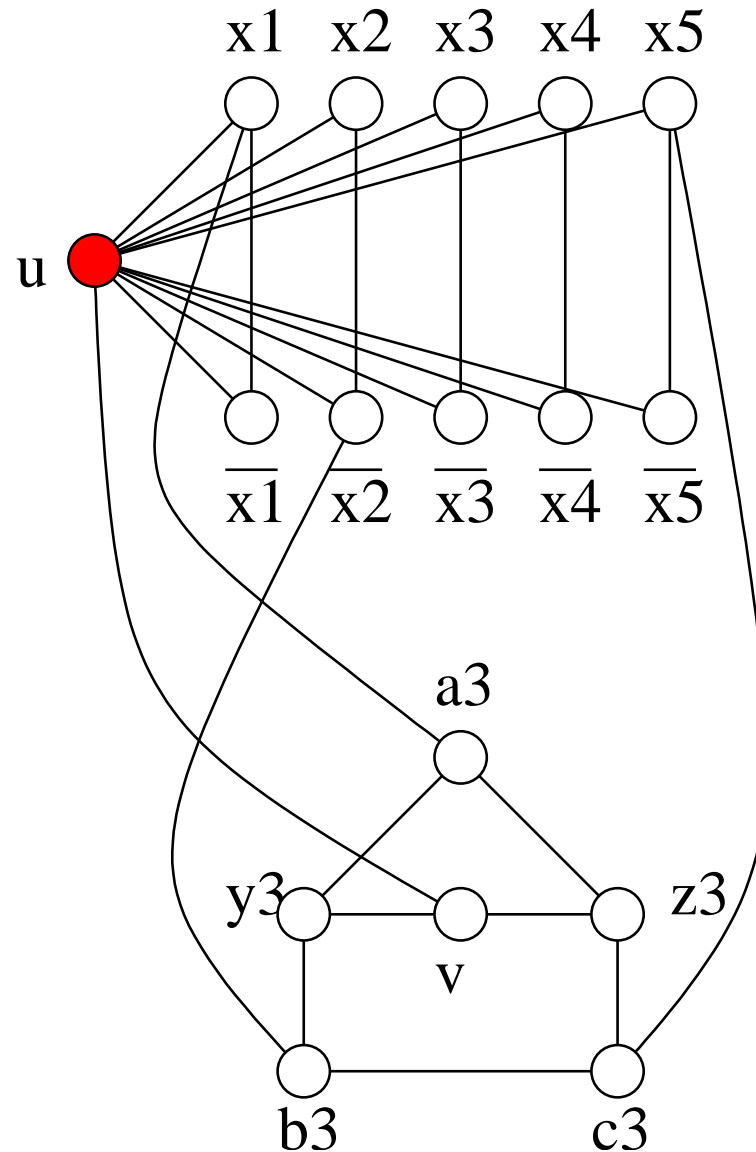
Edges correspond to literal knots.

Attention: Color $a_j \neq$ Truth value $a_j, \dots!$





Example: Gadget for clause $K_3 = (x_1 \vee \bar{x}_2 \vee x_5)$



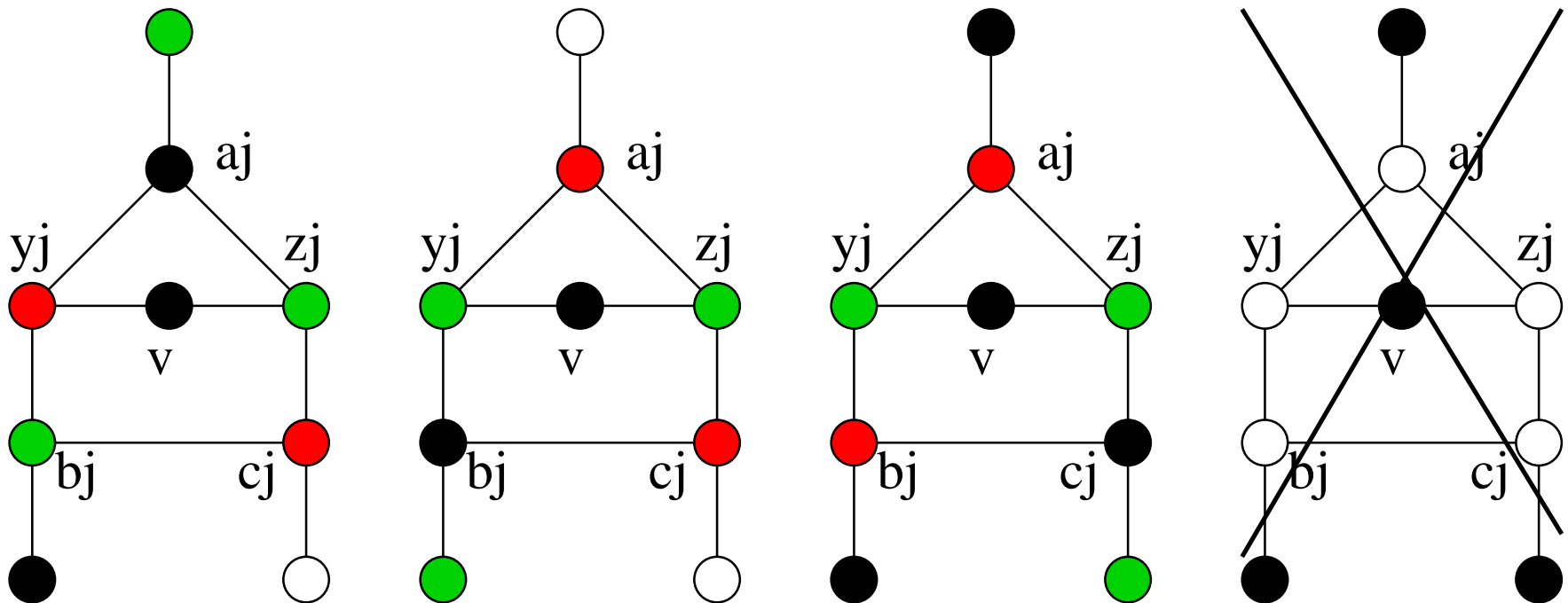


Proof: F satisfiable $\longrightarrow G$ 3-coloring

$c(u) := \text{rot}$, $c(v) := \text{false}$,

select $c(x_i) = \neg c(\bar{x}_i)$ corresponded to one satisfiable assignment.

Plainly case distinction for clause-Gadget j :





Proof: G 3-coloring $\longrightarrow F$ satisfiable

Name the color von u

rot.

Name the color $c(v) \neq c(u)$

false.

Name the 3. color

true.

The graph forces

consistent truth values for literal knots

$$x_i, \bar{x}_i \in \{\text{true}, \text{false}\}.$$

To show:

The assignment B makes F true—

$$F[B] = \text{true}$$

Proof by contradiction.

Assume

$$F[B] = \text{false?}$$

$\longrightarrow \exists j :$

$$K_j[B] = \text{false.}$$

\longrightarrow All literals from K_j are false.



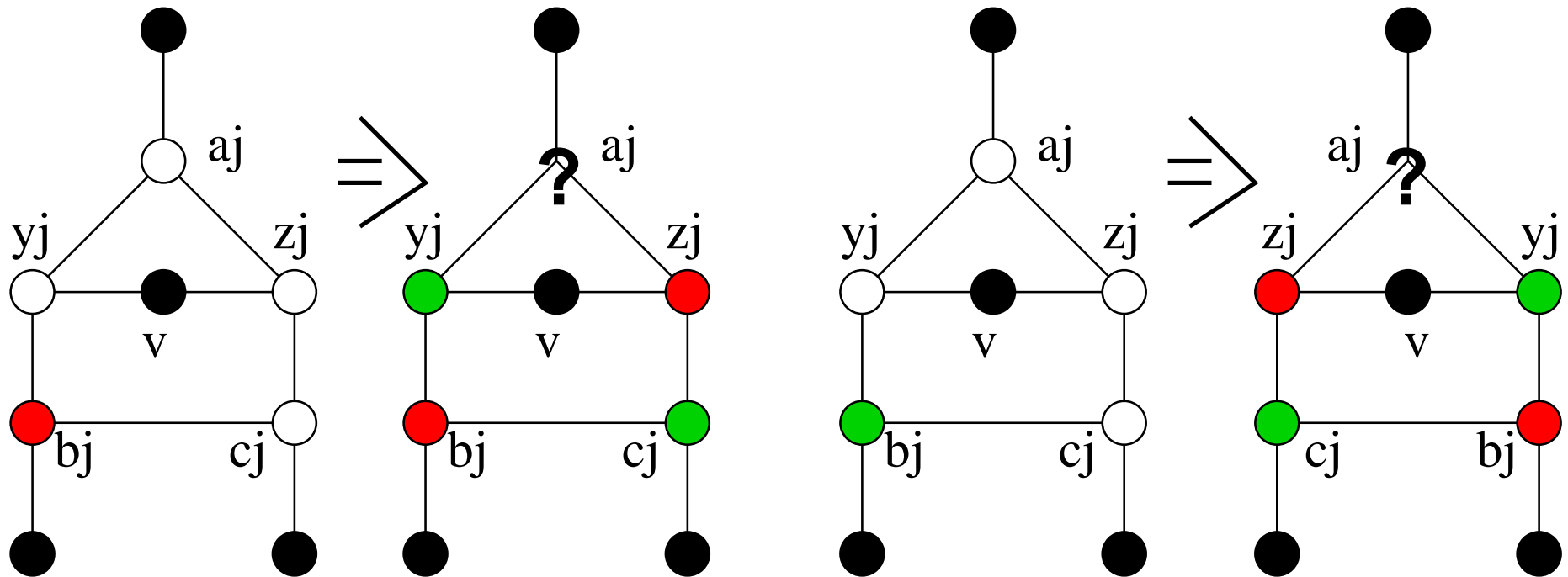
Proof: G 3-coloring $\longrightarrow F$ satisfiable

[...] All literals from K_j are false

$\longrightarrow c(b_j) = \text{rot}$

oder

$c(b_j) = \text{true}$



Evidently $F[B]$ is also true.

qed



Excursus: 2-coloring \in P

2-coloring graphs are called **bipartit**.

Observation: **tree** are 2-coloring.

Observation: **The odd cycles** are not 2-coloring. The same is true for graphs, which have an odd cycle as a subgraph.



Excursus: 2-coloring \in P

Consider an arbitrary **strong connected component** $C \subseteq V$.

Consider an arbitrary **spanning tree** T for C .

Consider **2-coloring** c von T .

$\exists \{u, v\} \in E : c(u) = c(v)$?

→ $\{u, v\} \cup \{ \text{Path from } u \text{ to } v \text{ in } T \}$ build **odd cycle** (*).

→ G is **not a 2-coloring**.

Then: c is **legal 2-coloring** for G .

(*)-Exercise: Show by induction on the path size, that two knots with equal colors in T has even distance.



SUBSET SUM

Given: n Materials with weight $w_i \in \mathbb{N}$

Parameter $W \in \mathbb{N}$.

Question:

Is there a subset M von $\{1, \dots, n\}$,

so that $\sum_{i \in M} w_i = W$

\approx Special case of knapsack with equal profits.

Attention! [\[Schöning\]](#) named SUBSET SUM RUCKSACK.



Schöning	Sanders
3CNF-SAT	3SAT
MENGENUBERDECKUNG	SET COVERING
KNOTENUBERDECKUNG	VERTEX COVER
RUCKSACK	SUBSET SUM
—	KNAPSACK (RUCKSACK)
FÄRBBARKEIT	COLORING



Proof SUBSET SUM \in NP

... we know...



Proof „SUBSET SUM is NP-hard“

We show $3SAT \leq_p SUBSET\ SUM$.

Let $F = K_1 \wedge \dots \wedge K_m = (z_{11} \vee z_{12} \vee z_{13}) \wedge \dots \wedge (z_{m1} \vee z_{m2} \vee z_{m3})$

with

literals $z_{ij} \in \{x_1, \dots, x_n\} \cup \{\neg x_1, \dots, \neg x_n\}$.

We define a SUBSET SUM instance with weights

$(v_1, \dots, v_n, v'_1, \dots, v'_n, c_1, \dots, c_m, d_1, \dots, d_m)$

and sum W



The SUBSET SUM instance

$(v_1, \dots, v_n, v'_1, \dots, v'_n, c_1, \dots, c_m, d_1, \dots, d_m), W$

$m + n$ decimal numbers (words in $\{0, \dots, 9\}$)

Let $v_{ij} :=$ the number that occurs for x_i in clause j . (Decimal digit)

Let $v'_{ij} :=$ the number occurs for $\neg x_i$ in clause j . (Decimal digit)

$$v_i = v_{i1}v_{i2} \cdots v_{im} 0^{i-1} 10^{n-i},$$

$$v'_i = v'_{i1}v'_{i2} \cdots v'_{im} 0^{i-1} 10^{n-i}$$

$$c_j := 0^{j-1} 10^{m-j} 0^n$$

$$d_j := 0^{j-1} 20^{m-j} 0^n$$

$$W := \underbrace{4 \cdots 4}_m \underbrace{1 \cdots 1}_n$$



Example

$$F = (x_1 \vee \neg x_3 \vee x_5) \wedge (\neg x_1 \vee x_5 \vee x_4) \wedge (\neg x_2 \vee \neg x_2 \vee \neg x_5)$$

$$m = 3, n = 5$$

$v_1 = 100$	10000	$v'_1 = 010$	10000
$v_2 = 000$	01000	$v'_2 = 002$	01000
$v_3 = 000$	00100	$v'_3 = 100$	00100
$v_4 = 010$	00010	$v'_4 = 000$	00010
$v_5 = 110$	00001	$v'_5 = 001$	00001
$c_1 = 100$	00000	$d_1 = 200$	00000
$c_2 = 010$	00000	$d_1 = 020$	00000
$c_3 = 001$	00000	$d_1 = 002$	00000

$$W = 444 \ 11111$$



Proof idea

Observation:

by decimal addition the weights are tread **no overflow**.

$$W = 444 \mathbf{11111}$$

The right **block** could only rise, so for each variable x_i either $x_i \equiv v_i$ or $\neg x_i \equiv v'_i$ is selected.

Position j of the left block stays for **clause j** .

The selected $v_i, v'_i =$ numbers the **fulfilled literals**:

1,2 or 3 for fulfilled clauses.

The c_j, d_j are **Slack-variables**, the position j of fulfilled clauses of the value could **fulfil 4** .



F satisfiable \longrightarrow Instance solution

Consider assignment B with $F[B] = 1$.

Select for M , v_i when $x_i[B] = 1$ and v'_i when $x_i[B] = 0$.

Up to know it holds $\sum_{i \in M} w_i = t_1 t_2 \cdots t_m 1^n$

with $t_j \in \{1, 2, 3\}$ (each clause has 1–3 fulfilled literals).

for $j := 1$ **to** m **do**

if $t_j = 1$ **then** select c_j and d_j // $1 + 1 + 2 = 4$

if $t_j = 2$ **then** select d_j // $2 + 2 = 4$

if $t_j = 3$ **then** select c_j // $3 + 1 = 4$

Now is $\sum_{i \in M} w_i = 4^m 1^n = W$.

qed.



Instance solution $\longrightarrow F$ satisfiable

Consider M with $\sum_{i \in M} w_i = W = 4^m 1^n$.

For each variable i will either v_i or v'_i fulfilled.

or else would be position i in **right** block $\neq 1$.

Select an assignment B with $x_i = 1$ if v_i fulfilled, $x_i = 0$ otherwise.

Statement: $F[B] = 1$ (By contradiction)

Assume: $F[B] = 0$ also $\exists j : K_j[B] = 0$.

$\longrightarrow \sum$ selected v_i, v'_i is 0

on position j

$\longrightarrow \sum$ selected c_i, d_i is $\leq 3 \neq 4$

on position j

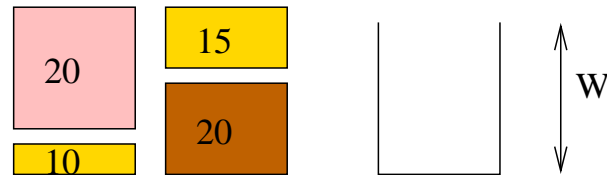
A contradiction.

So $F[B] = 1$

qed.



Example Knapsack problem



- n objects with **weights** $w_i \in \mathbb{N}$ and **profit** p_i
- Select a subset **x** from the objects
- so that $\sum_{i \in \mathbf{x}} w_i \leq W$ and
- maximize the profit** $\sum_{i \in \mathbf{x}} p_i$



Knapsack problem

∈ **NP**: we know

NP-hard: We show $\text{SUBSET SUM} \leq_p \text{KNAPSACK}$

Obviously the SUBSET SUM is an instance

(w_1, \dots, w_n) with $\text{sum} = W$

Equivalent for KNAPSACK-Instance

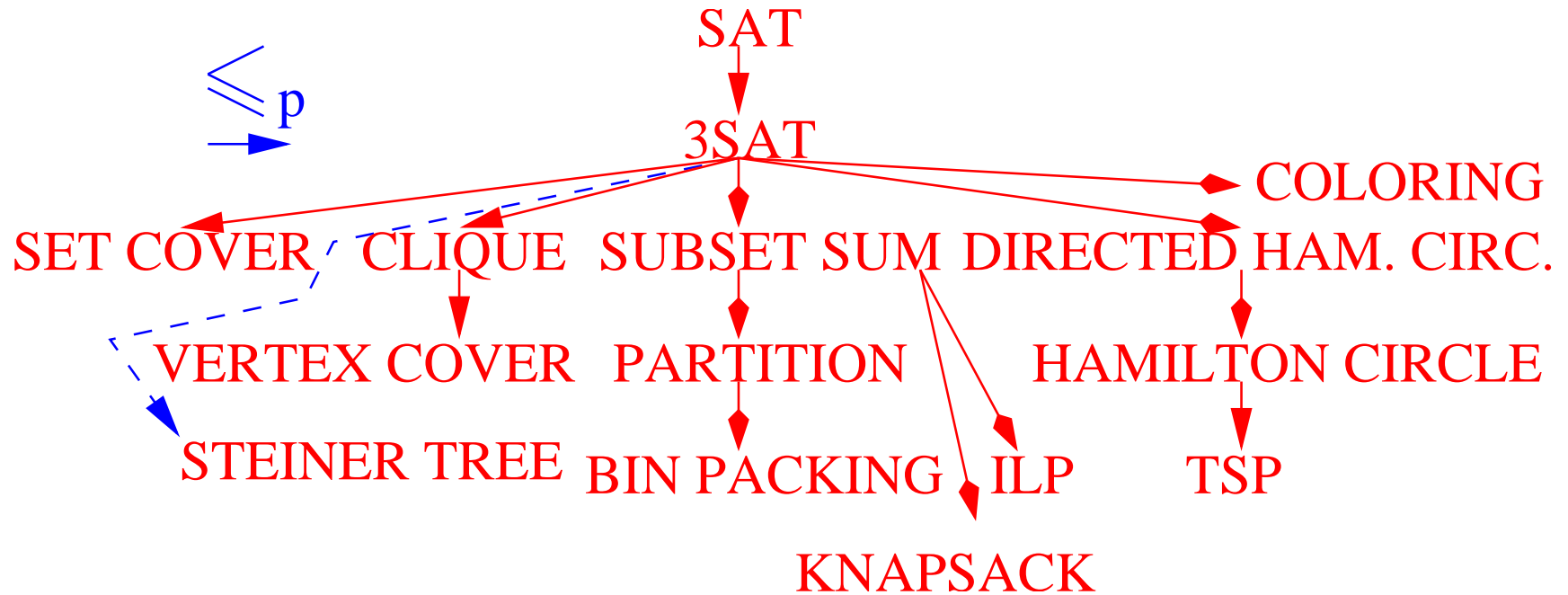
$\underbrace{(w_1, \dots, w_n)}_{\text{weights}}, \underbrace{(w_1, \dots, w_n)}_{\text{profit}}$ with

weight bound $\leq W$ and

profit bound $\geq W$.



Reductions





PARTITION

Given: n Objects with weights $w_i \in \mathbb{N}$

Question:

Is there a subset M von $\{1, \dots, n\}$,

so that $\sum_{i \in M} w_i = \sum_{i \notin M} w_i$?

in **NP**, since ...



Proof $\text{SUBSET SUM}_{\leq p}$ PARTITION

Given one **SUBSET SUM Instance** $S = (w_1, \dots, w_k), W$.

Let $M := \sum_{i=1}^k w_i$.

Consider PARTITION Instance $P = (w_1, \dots, w_k, M - W + 1, W + 1)$.

Total weight: $M + M - W + 1 + W + 1 = 2(M + 1)$.

to show P has solution $\Leftrightarrow S$ has solution.



SUBSET SUM Instance $S = (w_1, \dots, w_k), W$. $M := \sum_{i=1}^k w_i$.

PARTITION Instance $P = (w_1, \dots, w_k, M - W + 1, W + 1)$. ($\Sigma = 2(M + 1)$).

Case P has solution $\longrightarrow S$ has solution:

Let J be a solution for P , i.e., $\sum_{j \in J} P[j] = M + 1$.

Either $M - W + 1$ or $W + 1$ will be selected:

$$M - W + 1 + W + 1 = M + 2 > M + 1, \sum_{i=1}^k w_i = M < M + 1$$

Case $M - W + 1$ will be selected:

Then it is $\sum_{j \in J \cap \{1, \dots, k\}} P[j] = M + 1 - (M - W + 1) = W$.

Also it is $J \cap \{1, \dots, k\}$ is solution for S .

Case $W + 1$ will be selected:

Then $\sum_{j \in J \cap \{1, \dots, k\}} P[j] = M + 1 - (W + 1) = M - W$.

Also $\{1, \dots, k\} \setminus J$ is solution for S .



SUBSET SUM Instance $S = (w_1, \dots, w_k), W$.

$$M := \sum_{i=1}^k w_i.$$

PARTITION Instance $P = (w_1, \dots, w_k, M - W + 1, W + 1)$.

Total weight: $2(M + 1)$.

Case S has solution $\longrightarrow P$ has solution:

Let I be a solution for S , i.e., $\sum_{i \in I} w_i = W$.

Then $J := I \cup \{k + 1\}$ is a solution for P , then

$$\sum_{j \in J} P[j] = \sum_{i \in I} w_i + M - W + 1 = W + M - W + 1 = M + 1$$



BIN PACKING

Given:

Dozen of cookies $b \in \mathbb{N}$.

Cookies of size $w_1, \dots, w_n \in \mathbb{N}$

Number of cookies k .

Question:

Could we put all cookies somehow in the dozen?, i.e.,



$$\exists f : 1..n \longrightarrow 1..k : \forall j \in 1..k : \sum \{w_i : f(i) = j\} \leq b$$



BIN PACKING is NP-complete

Obviously BIN PACKING is in **NP**.

On the other hand $\text{PARTITION}_{\leq p}$ BIN PACKING (Special instance):

$$(w_1, \dots, w_k) \mapsto \begin{cases} \text{dozen size:} & b = \frac{1}{2} \sum_{i=1}^k w_i \\ \text{number:} & k = 2 \\ \text{weights:} & w_1, \dots, w_k \end{cases}$$