SAT Solving : Introduction

Priyanka Golia

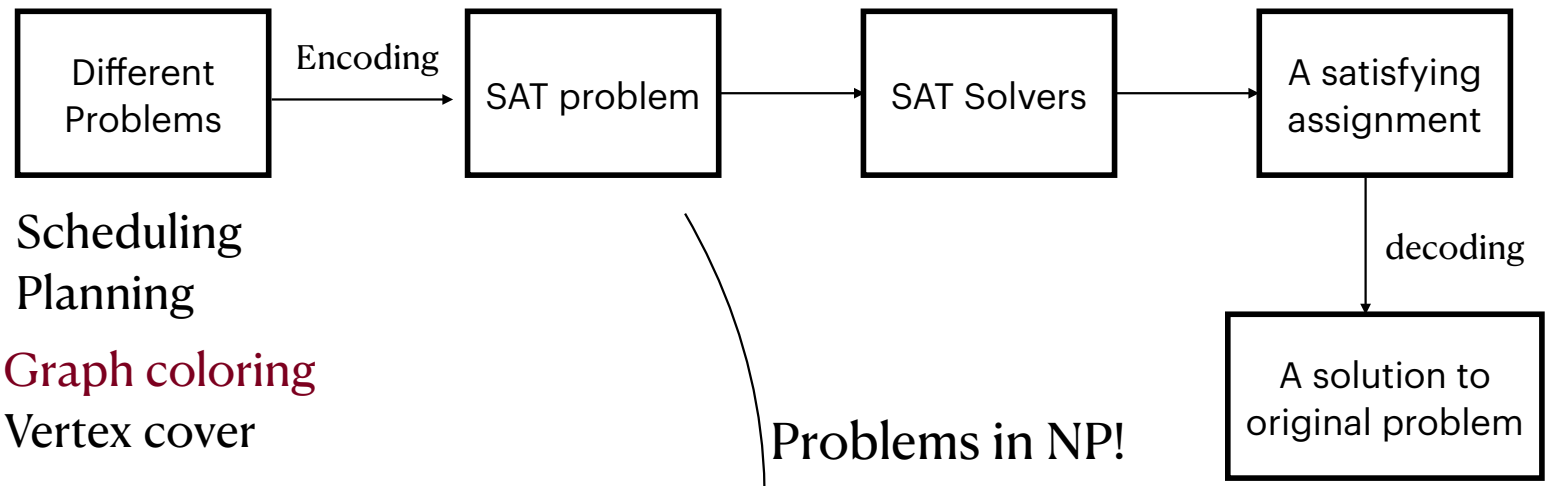
pgolia@cse.iitd.ac.in

SAT solvers

- Boolean formulas -> SAT Solvers
  - If formula is SAT, gives an satisfying assignment
  - Otherwise, UNSAT

Boolean Satisfiability (SAT) Simple to State, Rich in Structure

Despite its simplicity, it captures a vast range of real-world problems.



Fair Division

Goal: Divide items among agents fairly

Setup: n agents, m indivisible goods. Each agent has a utility function over goods (usually additive).

Outcome: Allocation of goods

Fairness notion:

EF (Envy-Free): No agent strictly prefers another agent's bundle over their own.

EF-1 (Envy-Free up to one good): Envy is removed by taking away one good from the envied bundle.

EFX (Envy-Free up to any good): Envy is removed by removing any good from the envied bundle

EF to SAT

Input:

Set of agents $\mathcal{A} = \{a_1, a_2, \ldots, a_n\}$ and a set of goods $\mathcal{G} = \{g_1, g_2, \ldots, g_m\}$

Valuation Matrix $V \in \mathbb{N}^{n \times m}$, such that $V[i][j]$ denotes the value of good g_j for agent a_i

We know,

Utility function, for any bundle $S \subseteq \mathcal{G}$, the additive utility of agent a_i is:

$$u_i(S) = \sum_{g_j \in S} V[i][j]$$

Encode to SAT problem such that the satisfying assignment leads to an allocation $A = (A_1, A_2, \ldots, A_n)$ of items to agents such that it is envy-free.

$$\forall i, k \in [n], u_i(A_i) \geq u_i(A_k)$$

$A_i \subseteq \mathcal{G}$ the set of goods assigned to agent a_i

EF to SAT

Propositional variables:

$x_{i,j}$ is True if good g_j is assigned to agent a_i

Item assignment constraints:

Each good is assigned to exactly one agent.

$$\bigwedge_{j=1}^{m} ExacltyOne(x_{1,j}, x_{2,j}, \ldots, x_{n,j})$$

$$\bigwedge_{j=1}^{m} AtLeastOne(x_{1,j}, x_{2,j}, \ldots, x_{n,j}) \wedge AtMostOne(x_{1,j}, x_{2,j}, \ldots, x_{n,j})$$

$$\bigwedge_{j=1}^{m} ((x_{1,j} \vee x_{2,j} \vee \ldots \vee x_{n,j}) \wedge \bigwedge_{1 \leq i < k \leq n} (\neg x_{i,j} \vee \neg x_{k,j}))$$

EF to SAT

Propositional variables:

$x_{i,j}$ is True if good g_j is assigned to agent a_i

EF constraints:

agent a_i not envying agent a_k is

$$\sum_{j=1}^{m} V[i][j] \,.\, x_{i,j} \geq \sum_{j=1}^{m} V[i][j] \,.\, x_{k,j}$$

For all pairs of agents:

$$\bigwedge_{1 \leq i < k \leq n} \left(\sum_{j=1}^{m} V[i][j] \,.\, x_{i,j} \geq \sum_{j=1}^{m} V[i][j] \,.\, x_{k,j} \right)$$

V[i][j]s are positive interger

Step Aside: Handling Sum-Based Constraints in SAT

Pseudo Boolean Constraints : linear inequality over Boolean variables.

$$\Sigma_i a_i x_i \geq b$$

x_i 's are Boolean variables

a_i 's are integer.

How do we convert Pseudo Boolean Constraints to CNF form?

Adder Circuit Encoding.

Sequential Counter Encoding.

Binary Decision Diagram Encoding.

Totalizer Encoding.

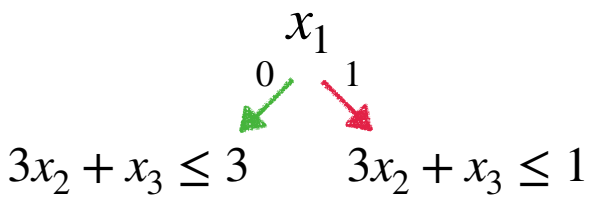
Sorting Network based Encoding.

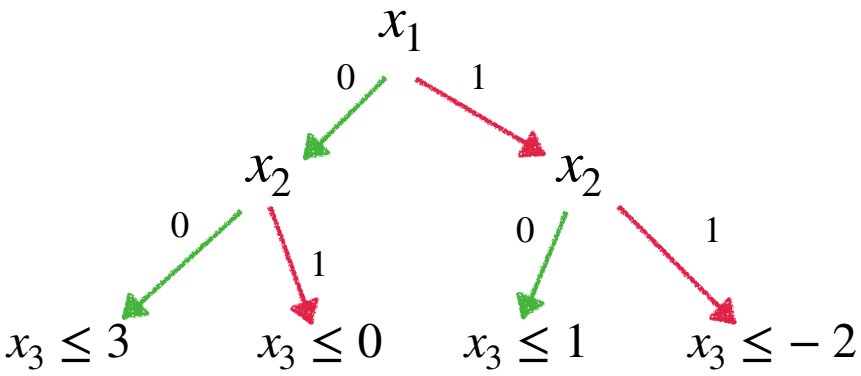
Scan me!

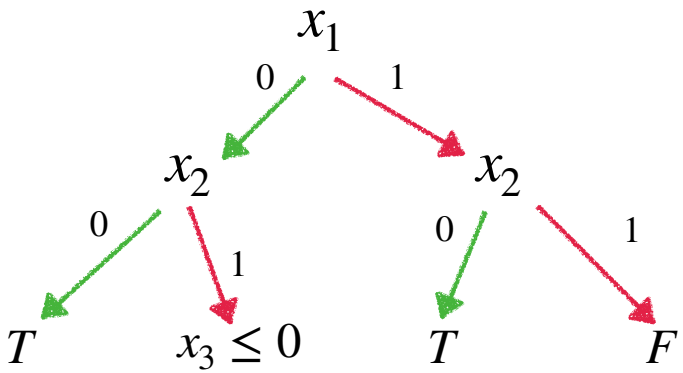
https://pysathq.github.io/docs/html/api/pb.html

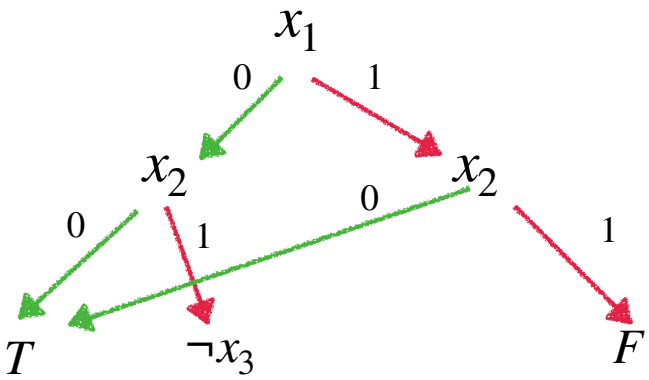
Pseudo Boolean Constraints Binary Decision Diagram Encoding.

$$2x_1 + 3x_2 + x_3 \leq 3 \qquad x_i \in \{0,1\}$$



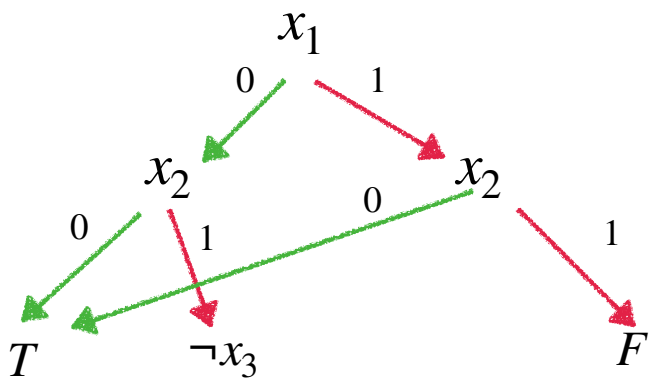






Pseudo Boolean Constraints Binary Decision Diagram Encoding.

$$2x_1 + 3x_2 + x_3 \leq 3 \qquad x_i \in \{0,1\}$$



Disjunction of all the paths that do not lead to a contradiction.

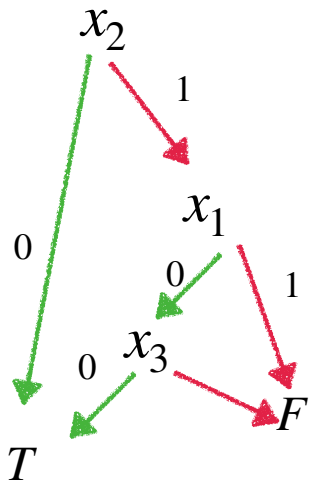
$$F = (\neg x_1 \wedge \neg x_2) \vee (x_1 \wedge \neg x_2) \vee (\neg x_1 \wedge x_2 \wedge \neg x_3)$$

$$\begin{aligned} F_{CNF} = {} & (\neg t_1 \vee \neg x_1) \wedge (\neg t_1 \vee \neg x_2) \wedge (x_1 \vee x_2 \vee t_1) \\ & \wedge (\neg t_2 \vee x_1) \wedge (\neg t_2 \vee \neg x_2) \wedge (\neg x_1 \vee x_2 \vee t_2) \\ & \wedge (\neg t_3 \vee \neg x_1) \wedge (\neg t_3 \vee x_2) \wedge (\neg t_3 \vee \neg x_3) \\ & \wedge (x_1 \vee \neg x_2 \vee x_3 \vee t_3) \wedge (t_1 \vee t_2 \vee t_3) \end{aligned}$$

Pseudo Boolean Constraints

Binary Decision Diagram Encoding.

$$2x_1 + 3x_2 + x_3 \leq 3 \qquad x_i \in \{0,1\}$$



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

$$F_{CNF} = (\neg t_1 \vee x_2) \wedge (\neg t_1 \vee \neg x_1) \wedge (\neg t_1 \wedge \neg x_3) \wedge (\neg x_2 \vee x_1 \vee x_3 \vee t_1) \wedge (t_1 \vee \neg x_2)$$

BDD based encoding is variable order sensitive.

Finding the optimal variable ordering for BDDs remains an open problem. However, several heuristics exist to obtain a 'good' ordering.

EF to SAT

Propositional variables:

$x_{i,j}$ is True if good g_j is assigned to agent a_i

EF constraints:

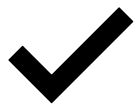
agent a_i not envying agent a_k is

$$\sum_{j=1}^{m} V[i][j] \,.\, x_{i,j} \geq \sum_{j=1}^{m} V[i][j] \,.\, x_{k,j}$$

For all pairs of agents:

$$\bigwedge_{1 \leq i < k \leq n} \left(\sum_{j=1}^{m} V[i][j] \,.\, x_{i,j} \geq \sum_{j=1}^{m} V[i][j] \,.\, x_{k,j} \right)$$

V[i][j]s are positive interger



EF to SAT

Set of agents $\mathcal{A} = \{a_1, a_2\}$

Set of goods $\mathcal{G} = \{g_1, g_2\}$

$V = \begin{bmatrix} 3 & 1 \\ 2 & 4 \end{bmatrix}$

$V = \begin{bmatrix} 10 & 0 \\ 10 & 0 \end{bmatrix}$

Does there exists an envy-free allocation?

$A_1 = \{o_1\}$

$A_2 = \{o_2\}$

EFX to SAT

Input:

Set of agents $\mathcal{A} = \{a_1, a_2, \ldots, a_n\}$ and a set of goods $\mathcal{G} = \{g_1, g_2, \ldots, g_m\}$

Valuation Matrix $V \in \mathbb{N}^{n \times m}$, such that $V[i][j]$ denotes the value of good g_j for agent a_i

We know,

Utility function, for any bundle $S \subseteq \mathcal{G}$, the additive utility of agent a_i is:

$$u_i(S) = \sum_{g_j \in S} V[i][j]$$

Encode to SAT problem such that the satisfying assignment leads to an allocation $A = (A_1, A_2, \ldots, A_n)$ of items to agents such that it is **envy-free up to any good**

$$\forall i, k \in [n] \wedge i \neq k \ \ , \forall g \in A_k : u_i(A_i) \geq u_i(A_k \backslash \{g\})$$

$A_i \subseteq (G)$ the set of goods assigned to agent a_i

EFX to SAT

Propositional variables:

$x_{i,j}$ is True if good g_j is assigned to agent a_i

Item assignment constraints:

Each good is assigned to exactly one agent.

$$\bigwedge_{j=1}^{m} ExacltyOne(x_{1,j}, x_{2,j}, \ldots, x_{n,j})$$

$$\bigwedge_{j=1}^{m} AtLeastOne(x_{1,j}, x_{2,j}, \ldots, x_{n,j}) \wedge AtMostOne(x_{1,j}, x_{2,j}, \ldots, x_{n,j})$$

$$\bigwedge_{j=1}^{m} ((x_{1,j} \vee x_{2,j} \vee \ldots \vee x_{n,j}) \wedge \bigwedge_{1 \leq i < k \leq n} (\neg x_{i,j} \vee \neg x_{k,j}))$$

EFX to SAT

Propositional variables:

$x_{i,j}$ is True if good g_j is assigned to agent a_i

EFX constraints:

agent a_i not envying agent a_k up to any good is:

$$\sum_{j=1}^{m} V[i][j] \,.\, x_{i,j} \geq \sum_{j=1}^{m} V[i][j] \,.\, x_{k,j}$$

$$\bigwedge_{l=1}^{m} (x_{k,l} \rightarrow (\sum_{j=1}^{m} V[i][j] \,.\, x_{i,j} \geq (\sum_{j=1}^{m} V[i][j] \,.\, x_{k,j} - V[i][l])))$$

For all pairs of agents:

$$\bigwedge_{1 \leq i < k \leq n} \bigwedge_{l=1}^{m} (x_{k,l} \rightarrow (\sum_{j=1}^{m} V[i][j] \,.\, x_{i,j} \geq (\sum_{j=1}^{m} V[i][j] \,.\, x_{k,j} - V[i][l])))$$

EFX to SAT

Propositional variables:

$x_{i,j}$ is True if good g_j is assigned to agent a_i

EFX constraints:

agent a_i not envying agent a_k up to any good is:

$$\sum_{j=1}^{m} V[i][j] \,.\, x_{i,j} \geq \sum_{j=1}^{m} V[i][j] \,.\, x_{k,j}$$

$$\bigwedge_{l=1}^{m} (x_{k,l} \rightarrow (\sum_{j=1}^{m} V[i][j] \,.\, x_{i,j} \geq (\sum_{j=1}^{m} V[i][j] \,.\, x_{k,j} - V[i][l])))$$

For all pairs of agents:

$$\bigwedge_{1 \leq i < k \leq n} \bigwedge_{l=1}^{m} (x_{k,l} \rightarrow (\sum_{j=1}^{m} V[i][j] \,.\, x_{i,j} \geq (\sum_{j=1}^{m} V[i][j] \,.\, x_{k,j} - V[i][l])))$$

EFX to SAT

Set of agents $\mathcal{A} = \{a_1, a_2\}$

Set of goods $\mathcal{G} = \{g_1, g_2\}$

$V = [3 \;\; 1]$

$\quad\;\; [2 \;\; 4]$

Variables $x_{11}, x_{12}, x_{21}, x_{22}$

Each good is assigned to at least one agent

$$x_{11} \vee x_{21} \qquad x_{12} \vee x_{22}$$

Each good is assigned to at most one agent

$$\neg x_{11} \vee \neg x_{21} \qquad \neg x_{12} \vee \neg x_{22}$$

EFX constraints:

agent a_1 not envying agent a_2 up to any good is:

$$x_{21} \rightarrow (3x_{11} + x_{12} \geq (3x_{21} + x_{22} - 3))$$

$$x_{22} \rightarrow (3x_{11} + x_{12} \geq (3x_{21} + x_{22} - 1))$$

agent a_2 not envying agent a_1 up to any good is:

$$x_{11} \rightarrow (2x_{21} + 4x_{22} \geq (2x_{11} + 4x_{12} - 2))$$

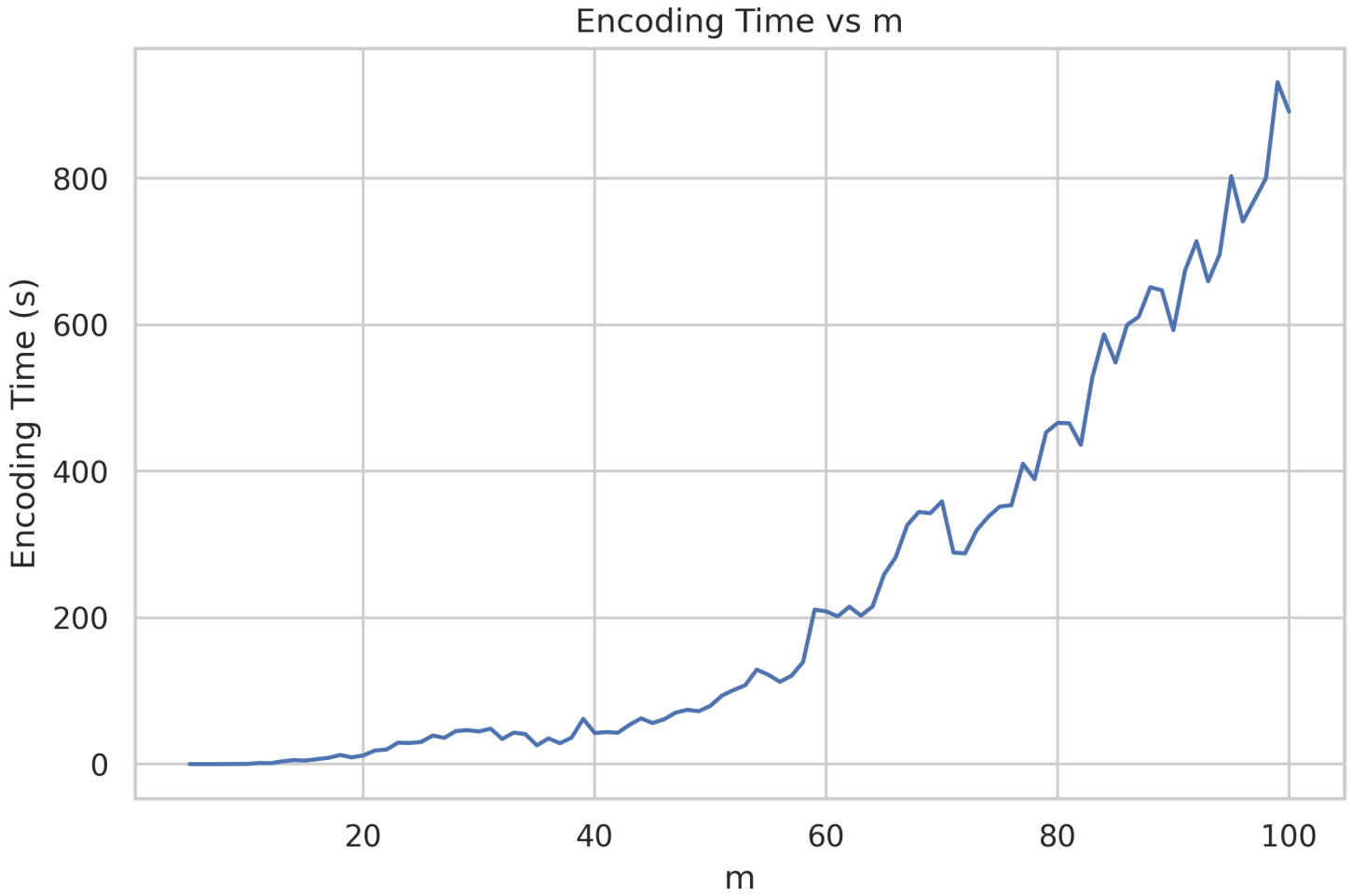
$$x_{12} \rightarrow (2x_{21} + 4x_{22} \geq (2x_{11} + 4x_{12} - 4))$$

EFX to SAT

Y axis — Size of encoding in 10^7 clauses. For $n = 4$

Thanks to Sharayu Deshmukh!

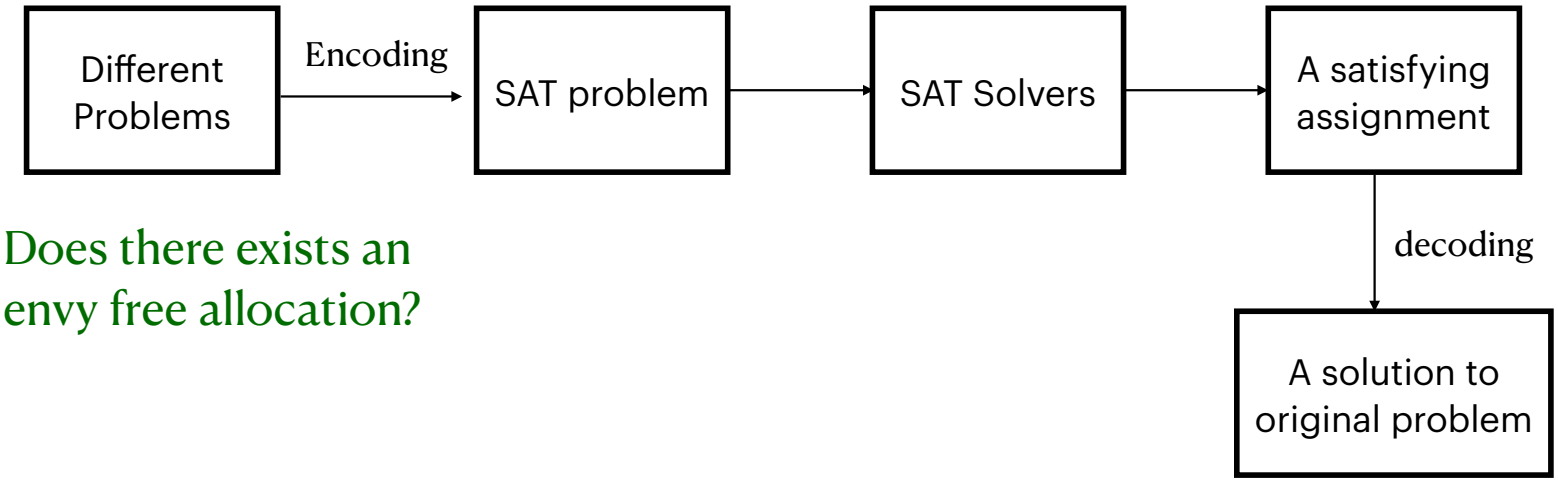
EFX to SAT



Thanks to Sharayu Deshmukh!

For $n = 4$

EFX to SAT

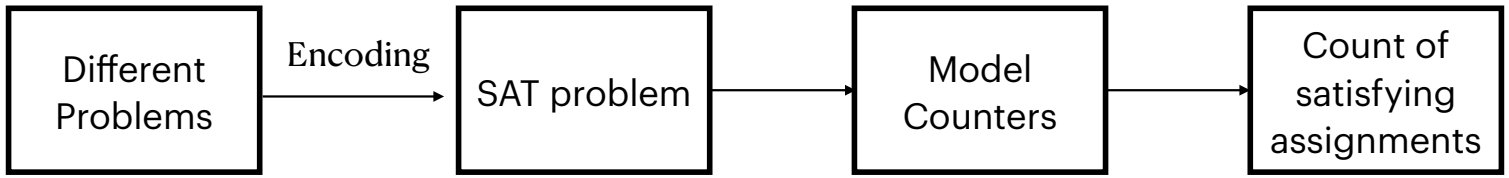


Does there exists an envy free allocation?

EFX to SAT

Does there exists an envy free allocation?

How many allocation exists?



EFX to SAT How many allocations exists?

$$
\begin{aligned}
&ModelCounter(F, count)\{ \\
&\quad Result, \sigma = CheckSAT(F) \\
&\quad if\ (Result == SAT)\{ \\
&\qquad count++\ \} \\
&\quad else\ \ Return\ count \\
&\quad ModelCounter(F \wedge \neg\sigma, count)\}
\end{aligned}
$$

Assuming access to a NP oracle !

EFX to SAT How many allocations exists?

$ModelCounter(F)\{$

$If\ F\ is\ 0\ then\ Return\ 0$

$If\ F\ is\ 1\ then\ Return\ 1$

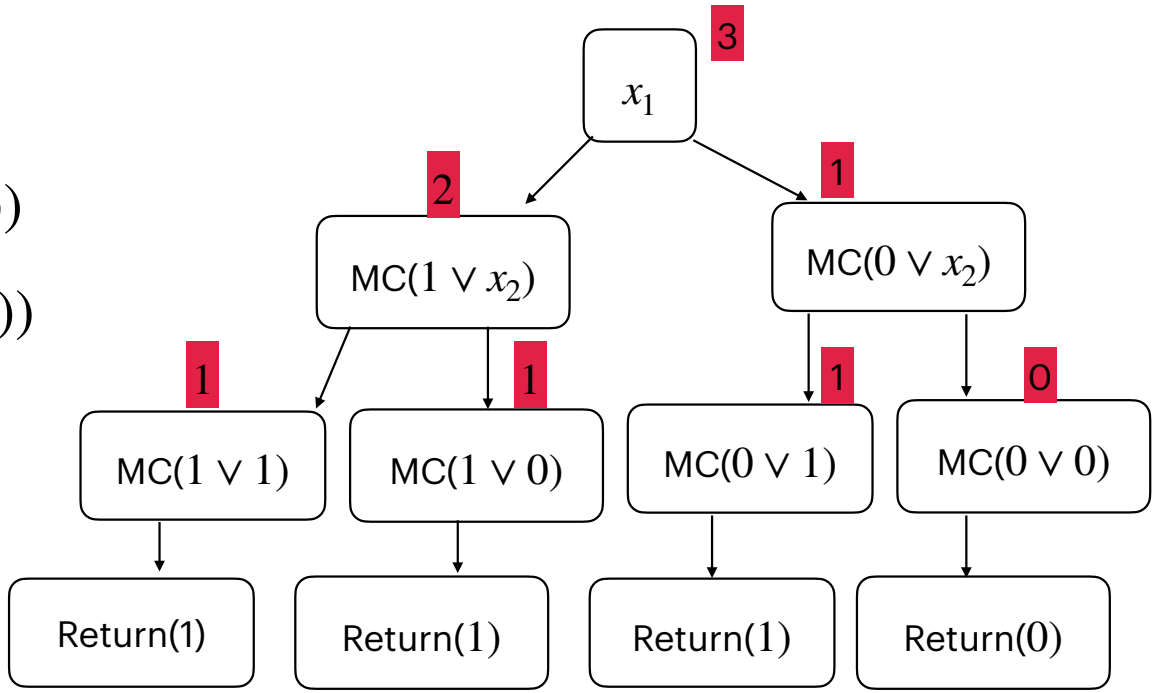
$pick\ \ x \leftarrow VARs(F)$

$C_o = ModelCounter(F(x \mapsto 0))$

$C_1 = ModelCounter(F(x \mapsto 1))$

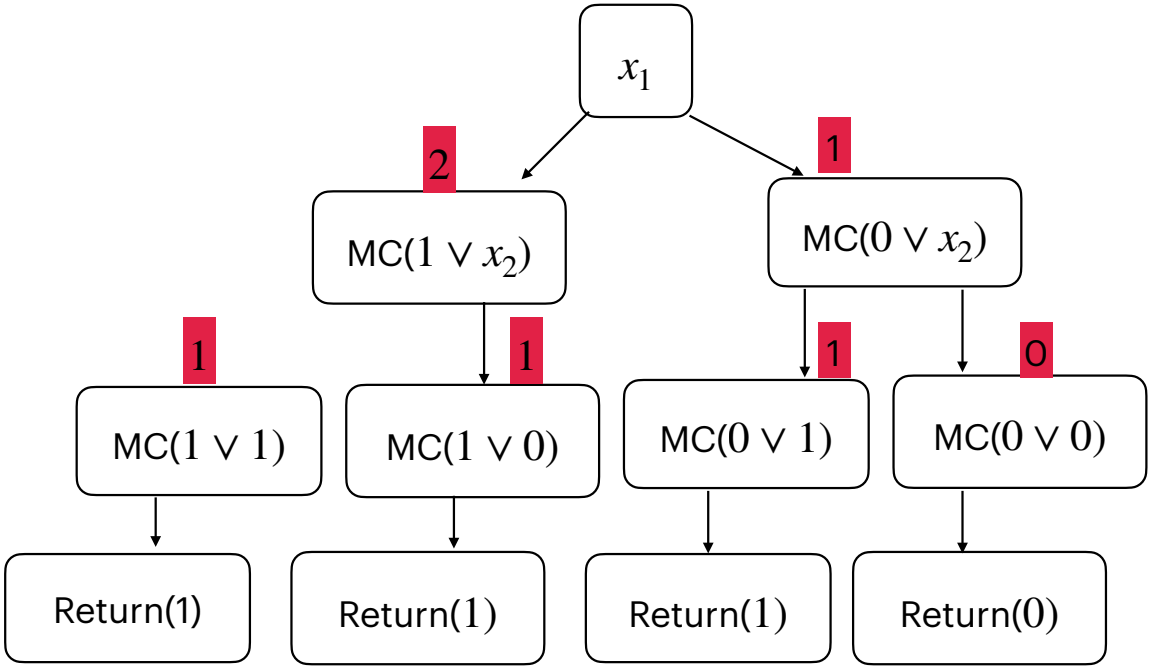
$return\ \ C_o + C_1\ \}$

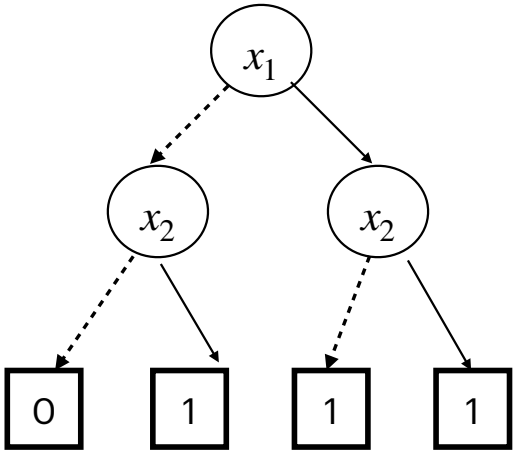
$F = x_1 \vee x_2$



EFX to SAT How many allocations exists?

$$F = x_1 \vee x_2$$



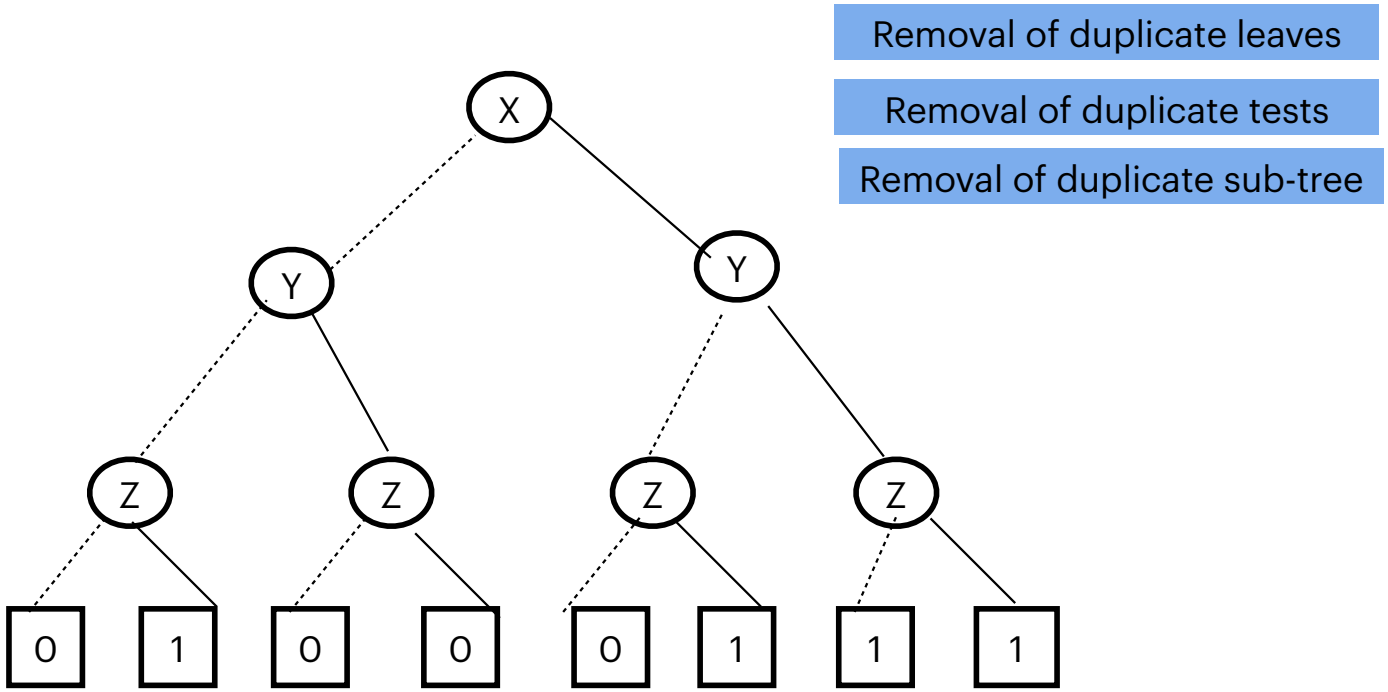


In OBDD, Model count is Sum of leaf nodes.

Model Counting

ROBDD — Reduced Ordered Binary Decision Diagrams

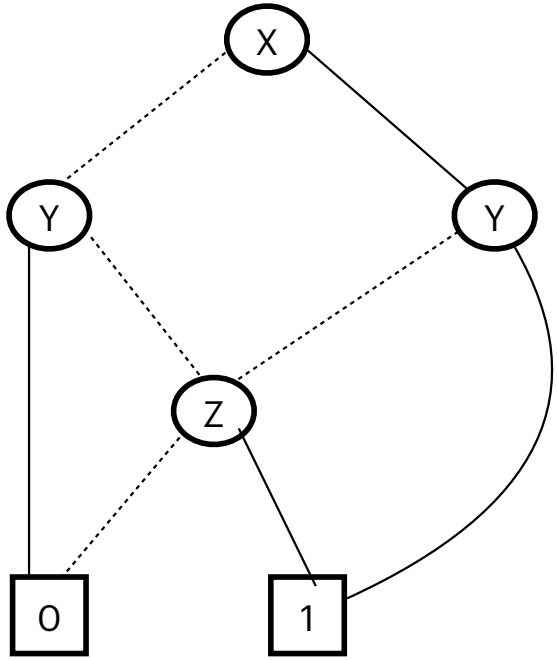
$F = (x \wedge y) \vee (\neg y \wedge z)$



Removal of duplicate leaves

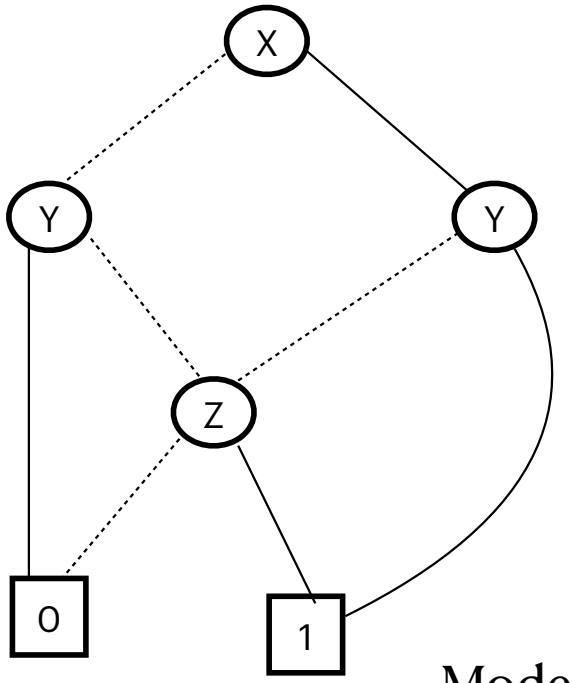
Removal of duplicate tests

Removal of duplicate sub-tree

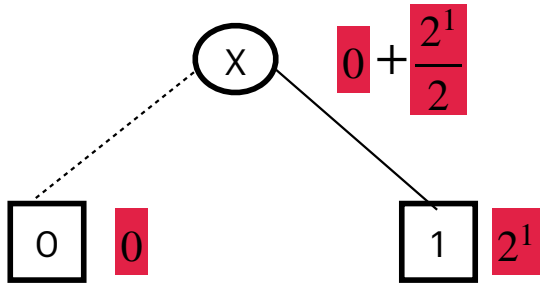


Model Counting

$F = (x \wedge y) \vee (\neg y \wedge z)$



Key Observation: We are fixing a variable as we move from the child to the parent node.

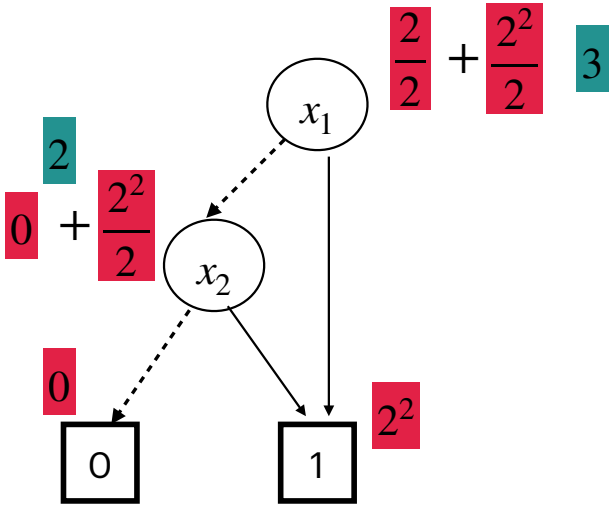


Bottom-up approach.

Model Counting in ROBDD?

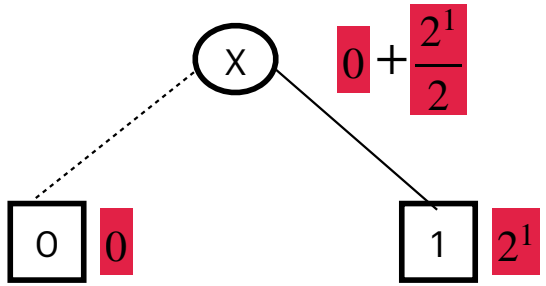
Model Counting

$F = x_1 \vee x_2$



Model Counting in ROBDD?

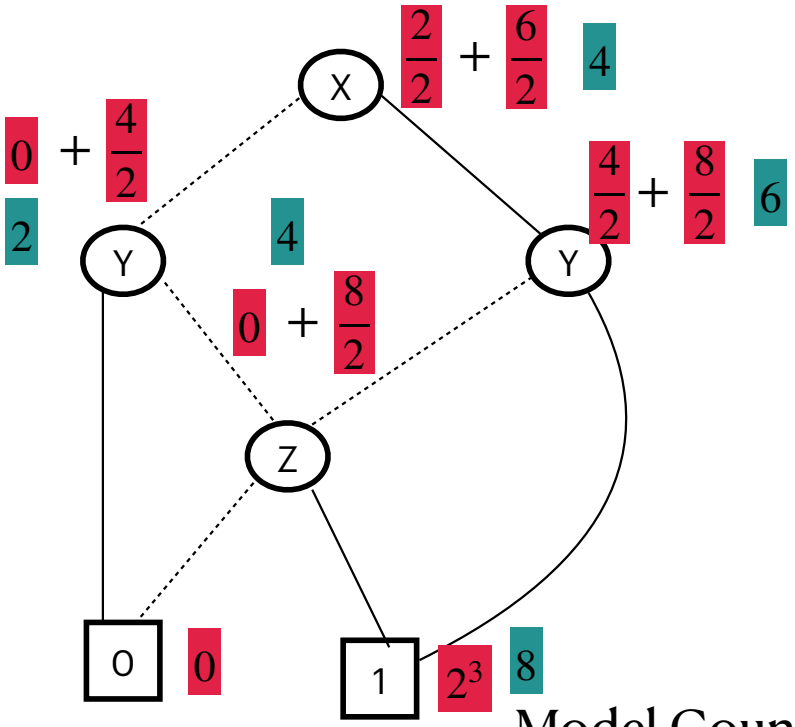
Key Observation: We are fixing a variable as we move from the child to the parent node.



Bottom-up approach.

Model Counting

$$F = (x \wedge y) \vee (\neg y \wedge z)$$



$$|Models(F)| = 4$$

Model Counting in ROBDD?

Model Counting

ROBDD vs CNF

	CNF	ROBDD
SAT	NP-Hard	$O(\lvert F_{ROBDD} \rvert)$
Model Count	#P	$O(\lvert F_{ROBDD} \rvert)$
UNSAT	Co-NP	$O(1)$

Model Counting in d-DNNF

(Deterministic Decomposable Normal Negation Form)

Tools like d4, c2d, DSharp for conversion

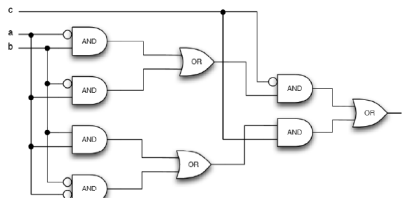
Just like ROBDD, may result in exponential size formula, but model counting is linear in the size of the formula

Efficient model counter, GANAK

By Shubham Sharma, a dual-degree student from IITK as his MTP project

Interested in Formal Methods/ Automated Reasoning ?

```
PC1 (char[] SP, char[] UI) {
        for (int i=0; i<UI.length(); i++) {
                if(SP[i] != UI[i]) return No;
        }
        return Yes;
}
```



System

Satisfies

Properties

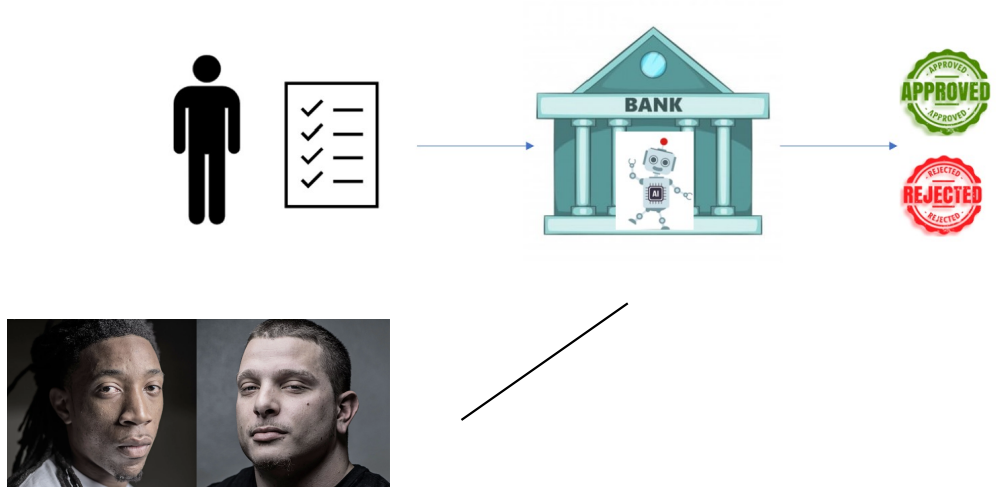
$$S(I,O) \models P(I,O)$$

Is the always the case that S satisfies Property P?

How often S satisfies P?

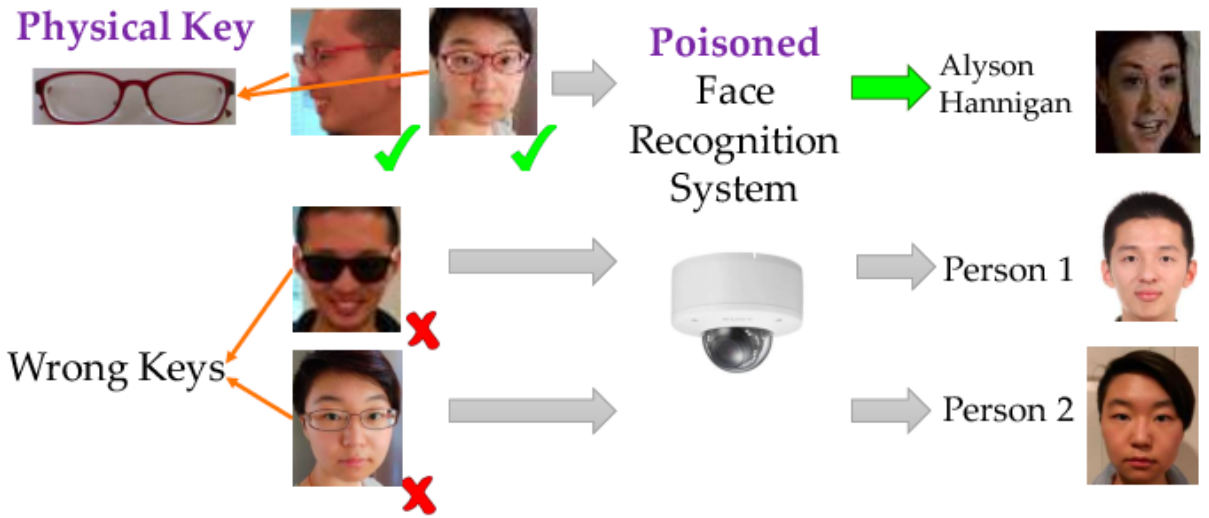
Why S doesn't satisfy P?

Interested in Formal Methods/ Automated Reasoning ?



Do two individuals with different skin colors but the same income, education, etc., receive the same prediction? Can you reason about it?

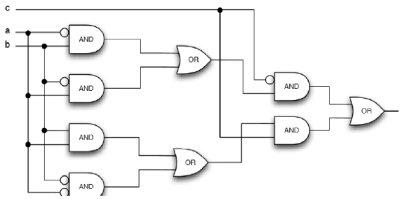
Interested in Formal Methods/ Automated Reasoning ?



When is a model not secure? Can you reason about it?

Interested in Formal Methods/ Automated Reasoning ?

```
PC1 (char[] SP, char[] UI) {
        for (int i=0; i<UI.length(); i++) {
            if(SP[i] != UI[i]) return No;
        }
        return Yes;
}
```



System

Satisfies

Properties

$S(I,O) \models P(I,O)$

Is the always the case that S satisfies Property P?

How often S satisfies P?

Why S doesn't satisfy P?

https://priyanka-golia.github.io/