

SAT with Low Complexity :

k-SAT, XOR SAT and Horn Clauses

HE Mingxin, Max

CS104: program07 @ yeah.net

CS108: mxhe1 @ yeah.net

I2ML(H) Spring 2023 (CS104|CS108)

Exercises 08 : Reading and More

Record your time spent (in 0.1 hours) with brief tasks and durations in your learning log by hand writing!

- 1) Read [textB-ch01-1.6-SAT-solvers.pdf](#) (in 2 weeks)
- 2) Work on Assignment 3&4...

Topic 8.1

k -SAT

k -SAT

Definition 8.1

A k -SAT formula is a CNF formula and has at most k literals in each of its clauses.

Example 8.1

- ▶ $(p \wedge q \wedge \neg r)$ is 1-SAT
- ▶ $(p \vee \neg p) \wedge (p \vee q)$ is 2-SAT
- ▶ $(p \vee \neg q \vee \neg s) \wedge (p \vee q) \wedge \neg r$ is 3-SAT

3-SAT Satisfiability

Theorem 8.1

For each k -SAT formula F there is a 3-SAT formula F' with linear blow up such that F and F' are equisatisfiable.

Proof.

Consider F a k -SAT formula with $k \geq 4$.

Consider a clause $G = (\ell_1 \vee \cdots \vee \ell_k)$ in F , where ℓ_i are literals.

Let x_2, \dots, x_{k-2} be variables that are not in $\text{Vars}(F)$.

Let G' be the following set of clauses

$$(\ell_1 \vee \ell_2 \vee x_2) \wedge \bigwedge_{i \in 2..k-3} (\neg x_i \vee x_{i+1} \vee \ell_{i+1}) \wedge (\neg x_{k-2} \vee \ell_{k-1} \vee \ell_k).$$

We show F is sat iff $(F - \{G\}) \cup G'$ is sat.

...

Thinking Exercise 8.1

Convert $(p \vee \neg q \vee s \vee \neg t) \wedge (\neg q \vee x \vee \neg y \vee z)$ into a 3-SAT formula

3-SAT satisfiability(cont. I)

Proof(contd. from last slide).

Recall

$$G' = (\ell_1 \vee \ell_2 \vee x_2) \wedge \bigwedge_{i \in 2..k-3} (\neg x_i \vee x_{i+1} \vee \ell_{i+1}) \wedge (\neg x_{k-2} \vee \ell_{k-1} \vee \ell_k).$$

Assume $m \models (F - \{G\}) \cup G'$:

Assume for each $i \in 1..k$, $m(\ell_i) = 0$.

Due to the first clause $m(x_2) = 1$.

Due to i th clause, if $m(x_i) = 1$ then $m(x_{i+1}) = 1$.

Due to induction, $m(x_{k-2}) = 1$.

Due to the last clause of G' , $m(x_{k-2}) = 0$. **Contradiction.**

Therefore, there is $i \in 1..k$ such that $m(\ell_i) = 1$. Therefore $m \models G$. Therefore, $m \models F$...

3-SAT Satisfiability (cont. II)

Proof(contd. from last slide).

Recall

$$G' = (\ell_1 \vee \ell_2 \vee x_2) \wedge \bigwedge_{i \in 2..k-3} (\neg x_i \vee x_{i+1} \vee \ell_{i+1}) \wedge (\neg x_{k-2} \vee \ell_{k-1} \vee \ell_k).$$

Assume $m \models F$:

Therefore, $m \models G$.

There is a $m(\ell_i) = 1$.

Let $m' \triangleq m[x_2 \mapsto 1, \dots, x_{i-1} \mapsto 1, x_i \mapsto 0, \dots, x_{k-2} \mapsto 0]$.

Therefore, $m' \models F - \{G\}$ and $m' \models G'_{(\text{why?})}$. Therefore, $m' \models (F - \{G\}) \cup G'$.

G' contains $3(k-2)$ literals. In the worst case, the formula size will increase 3 times. □

Thinking Exercise 8.2

- Complete the above argument.
- Show a 3-SAT cannot be converted into a 2-SAT via Tseitin's encoding.
- When is the worst case?

Special Classes of Formulas

We will discuss the following polynomial time subclasses of SAT problem

- ▶ 2-SAT
- ▶ XOR-SAT
- ▶ Horn Clauses

Commentary: For the curious, Schaefer's dichotomy theorem identified subclasses of SAT problem that are polynomial and all others are NP-complete. The above subclasses play an important role in the theorem. The theorem suggests that no other subclass is polynomial.

Topic 8.2

2-SAT

2-SAT

Definition 8.2

A *2-sat formula* is a CNF formula that has *only* binary clauses

We assume that unit clauses are replaced by clauses with repeated literals.

Example 8.2

- ▶ $(\neg p \vee q) \wedge (\neg q \vee r) \wedge (\neg r \vee p) \wedge (r \vee q)$ is a 2-SAT formula
- ▶ $(p \vee p) \wedge (\neg p \vee \neg p)$ is a 2-SAT formula

Implication Graph

Definition 8.3

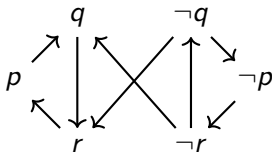
Let F be a 2-SAT formula such that $\text{Vars}(F) = \{p_1, \dots, p_n\}$. The *implication graph* (V, E) for F is defined as follows.

$$V \triangleq \{p_1, \dots, p_n, \neg p_1, \dots, \neg p_n\} \quad E \triangleq \{(\bar{\ell}_1, \ell_2), (\bar{\ell}_2, \ell_1) \mid (\ell_1 \vee \ell_2) \in F\},$$

where $\bar{p} = \neg p$ and $\overline{\neg p} = p$.

Example 8.3

Consider $(\neg p \vee q) \wedge (\neg q \vee r) \wedge (\neg r \vee p) \wedge (r \vee q)$.



Exercise: Implication Graph

Thinking Exercise 8.3

Draw implication graphs of the following

1. $(p \vee q) \wedge (\neg p \vee \neg q)$
2. $(p \vee \neg q) \wedge (q \vee p) \wedge (\neg p \vee \neg r) \wedge (r \vee \neg p)$
3. $(p \vee p) \wedge (\neg p \vee \neg p)$
4. $(p \vee \neg p) \wedge (p \vee \neg p)$

Properties of Implication Graph

Consider a formula F and its implication graph (V, E) .

Theorem 8.2

If there is a path from ℓ_1 to ℓ_2 in (V, E) then there is a path from $\bar{\ell}_2$ to $\bar{\ell}_1$.

Thinking Exercise 8.4

a. *Prove the above theorem.*

b. *Does the above theorem imply*

if there is a path from p to $\neg p$ in (V, E) then there is a path from $\neg p$ to p ?

Theorem 8.3

*For every strongly connected component(scc) $S \subseteq V$ in (V, E) , there is another scc S^c , called **complementary component**, that has exactly the literals that are negation of the literals in S .*

Proof.

Due to theorem 8.2.



Properties of Implication Graph (contd.)

Theorem 8.4

For each $m \models F$, if there is a path from ℓ_1 to ℓ_2 in (V, E) then if $m(\ell_1) = 1$ then $m(\ell_2) = 1$.

Theorem 8.5

For each $m \models F$ and each scc S in (V, E) , either

- ▶ *$m(\ell) = 1$ for each $\ell \in S$, or*
- ▶ *$m(\ell) = 0$ for each $\ell \in S$.*

Thinking Exercise 8.5

Prove the above theorems.

(use theorem in the previous slide)

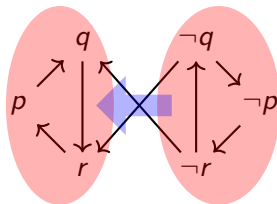
Reduced Implication Graph

Definition 8.4

For an implication graph (V, E) , the **reduced implication DAG** (V^R, E^R) is defined as follows.

$$V^R \triangleq \{S \mid S \text{ is a scc in } (V, E)\} \quad E^R \triangleq \{(S, S') \mid \text{there are } \ell \in S \text{ and } \ell' \in S' \text{ s.t. } (\ell, \ell') \in E\}$$

Example 8.4



Theorem 8.6

If $(S, S') \in E^R$ then $(S'^c, S^c) \in E^R$.

Thinking Exercise 8.6 Prove the above theorem.

Commentary: (V^R, E^R) is a graph over scc's of (V, E) . Please notice that (V^R, E^R) will always be a directed acyclic graph (DAG).

2-SAT Satisfiability

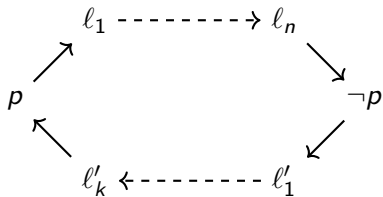
Theorem 8.7

A 2-SAT formula F is unsat iff there is a scc S in its implication graph (V, E) such that $\{p, \neg p\} \subseteq S$ for some p .

Proof.

Reverse direction

We assume $\{p, \neg p\} \subseteq S$.



There is a path that goes from p to $\neg p$.

Therefore, if p is true then $\neg p$ is true.

Therefore, p must be false.

Due to the path from $\neg p$ to p , if p is false then $\neg p$ is false.

Therefore, p must be true.

Therefore, F is unsat.

...

2-SAT Satisfiability (contd.)

Proof(contd.)

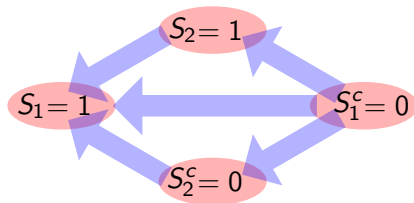
Fwd direction: Let us assume there is no such S .

We will construct a model of F as follows.

1. Initially all literals are unassigned.
2. While(some scc in V^R is unassigned)
 - 2.1 Let $S \in V^R$ be an unassigned scc whose all children are assigned 1.
 - 2.2 Assign literals of S to 1. Consequently, S^c is assigned 0.

...

Example 8.5



2-SAT Satisfiability (contd.)

Proof(contd.)

We need to show that step 2.1 always finds S with all children assigned 1.

claim: at step 2.1, there is an unassigned node whose all children are assigned

Choose an unassigned node.

Descend down if there is an unassigned child.

Since the DAG is finite, the process will terminate.

claim: an unassigned node can not have a child that is assigned 0.

If S is assigned 1, all its children are already 1.

Therefore, all the parents of S^c are already assigned 0(due to theorem 8.6).

Therefore, no node with 0 model has an unassigned parent.



Thinking Exercise 8.7

- Show that the procedure produces a satisfying model.
- Where did we use the fact that $\{p, \neg p\} \not\subseteq S$?

2-SAT is Polynomial

Theorem 8.8

A 2-SAT satisfiability problem can be solved in linear time.

Proof.

Due to the previous theorem.



Exercise: 2-SAT Solving

Thinking Exercise 8.8

Find a satisfying model of the following formula

1. $(\neg x \vee \neg y) \wedge (\neg y \vee \neg z) \wedge (\neg z \vee \neg x) \wedge (x \vee \neg w) \wedge (y \vee \neg w) \wedge (z \vee \neg w)$
2. $(p_0 \vee p_2) \wedge (p_0 \vee \neg p_3) \wedge (p_1 \vee \neg p_3) \wedge (p_1 \vee \neg p_4) \wedge (p_2 \vee \neg p_4) \wedge$
 $(p_0 \vee \neg p_5) \wedge (p_1 \vee \neg p_5) \wedge (p_2 \vee \neg p_5) \wedge (p_3 \vee p_6) \wedge (p_4 \vee p_6) \wedge (p_5 \vee p_6)$

Topic 8.3

XOR SAT

XOR-SAT

Definition 8.5

A formula is **XOR-SAT** if it is a conjunction of xors of literals.

Example 8.6

$(p \oplus r \oplus s) \wedge (q \oplus \neg r \oplus s) \wedge (p \oplus q \oplus \neg s) \wedge (p \oplus \neg q \oplus \neg r)$ is a XOR-SAT formula.

Solving XOR-SAT

Since xors are negation of equality, we may eliminate variables via substitution.

Theorem 8.9

For a variable p , formula G , and formula F , if $p \notin \text{Vars}(G)$ then $(p \oplus G) \wedge F$ and $F[\neg G/p]$ are equisatisfiable.

Thinking Exercise 8.9

Prove the above theorem.

Example : solving XOR-SAT

Example 8.7

Consider $(p \oplus r \oplus s) \wedge (q \oplus \neg r \oplus s) \wedge (p \oplus q \oplus \neg s) \wedge (p \oplus \neg q \oplus \neg r)$

Eliminate p :

Due to the first xor: $p \Leftrightarrow \neg r \oplus s$

Substitution: $(q \oplus \neg r \oplus s) \wedge (\neg r \oplus s \oplus q \oplus \neg s) \wedge (\neg r \oplus s \oplus \neg q \oplus \neg r)$

Simplification: $(q \oplus \neg r \oplus s) \wedge (\neg r \oplus \neg q) \wedge (s \oplus \neg q)$

Eliminate r :

Due to the second xor: $r \Leftrightarrow \neg q$

Substitution: $(q \oplus \neg \neg q \oplus s) \wedge (s \oplus \neg q)$

Simplification: $s \wedge (s \oplus \neg q)$

Example: solving XOR-SAT (contd.)

Eliminate q :

Due to the second xor: $q \Leftrightarrow s$

After substitution: s

Solution: $m(s) = 1$ $m(q) = m(s) = 1$ $m(r) = m(\neg q) = 0$ $m(p) = m(\neg r \oplus s) = 0$

Thinking Exercise 8.10

Find a satisfying model of the following formula

► $(p \oplus r \oplus s) \wedge (q \oplus r \oplus s) \wedge (\neg p \oplus q \oplus \neg s) \wedge (p \oplus \neg q \oplus \neg r)$

Topic 8.4

Horn Clauses

Horn Clauses

Definition 8.6

A *Horn clause* is a clause that has the following form

$$\neg p_1 \vee \cdots \vee \neg p_n \vee q,$$

where $p_1, \dots, p_n \in \text{Vars}$, and $q \in \text{Vars} \cup \{\perp\}$.

A *Horn formula* is a set of Horn clauses, which is a conjunction of the Horn clauses.

The clauses with \perp literals are called *goal clauses* and others are called *implication clauses*.

Example 8.8

The following set is a Horn formula

$$\{p, \neg q \vee \neg r \vee \neg t \vee p, \neg p \vee q, \neg p \vee \neg r \vee t, \neg p \vee \neg q \vee t, \neg r \vee \perp, \neg p \vee \neg q \vee \neg t \vee \perp\}$$

Implication View of the Horn Clauses

We may view a Horn clause

$$\neg p_1 \vee \cdots \vee \neg p_n \vee q$$

as

$$p_1 \wedge \cdots \wedge p_n \Rightarrow q.$$

Example 8.9

The following is an implication view of the Horn formula from previous slide.

$$\{\top \Rightarrow p, \quad q \wedge r \wedge t \Rightarrow p, \quad p \Rightarrow q, \quad p \wedge r \Rightarrow t, \quad p \wedge q \Rightarrow t, \quad r \Rightarrow \perp, \quad p \wedge q \wedge t \Rightarrow \perp\}$$

Note $\top \Rightarrow p$ means p , which is a Horn clause without negative literals

Horn Satisfiability

Algorithm 8.1: HORNSAT(Hs, Gs)

Input: Hs : implication clauses, Gs : goal clauses

Output: model/unsat

```
1  $m := \lambda x.0$ ;  
2 while  $m \not\models (p_1 \wedge \dots \wedge p_n \Rightarrow p) \in Hs$  do  
3    $m := m[p \mapsto 1]$ ;  
4 if  $m \not\models (q_1 \wedge \dots \wedge q_k \Rightarrow \perp) \in Gs$  then return unsat ;  
5 return  $m$ 
```

Thinking Exercise 8.11

Solve $\{\top \Rightarrow p, \quad q \wedge r \wedge t \Rightarrow p, \quad p \Rightarrow q, \quad p \wedge r \Rightarrow t, \quad p \wedge q \Rightarrow t, \quad r \Rightarrow \perp, \quad p \wedge q \wedge t \Rightarrow \perp\}$

Thinking Exercise 8.12

What is the maximum number of times the truth value of a clause in Hs changes during the algorithm? Give a supporting argument for the answer and an example that exhibits the situation.

Topic 8.5

Problems

Unsat XOR-sat

Thinking Exercise 8.13

Give an unsat XOR-sat formula that has only xors with more than three arguments.

Unsat 2-SAT**

Thinking Exercise 8.14

Let us suppose we have n variables in a 2-CNF problem. What is the maximum number of clauses in the formula such that the formula is satisfiable?

Unsatisfiable Core of 2-SAT

Thinking Exercise 8.15

An unsatisfiable core of an unsatisfiable CNF formula is a (preferably minimal) subset of the formula that is also unsatisfiable. Give an algorithm to compute a minimal unsatisfiable core of 2-SAT formula.

Horn SAT true to false

Thinking Exercise 8.16

In the Horn solving algorithm, we started with all false model and incrementally turned the variables true.

- a. Is it possible to modify the algorithm such that it starts with all true model and finds satisfying model for the Horn clauses.*
- b. Can we also start with any initial model?*

Topic 8.6

Extra Lecture Slides : Recognizable Horn Clauses

Recognizing Horn Clauses

Sometimes a set of clauses are not immediately recognizable as Horn clause.

We may convert a CNF into a Horn formula by flipping the negation signs for some variables. Such CNF are called **Horn clause renamable**.

Definition 8.7

Let F be a CNF formula and m be a model. Let $\text{flip}(F, m)$ denote the formula obtained by flipping the variables that are assigned 1 in m .

Example 8.10

$$\text{flip}((p \vee \neg q \vee \neg s), \{p \mapsto 1, q \mapsto 0, s \mapsto 1, ..\}) = (\neg p \vee \neg q \vee s)$$

Thinking Exercise 8.17

$$\text{Calculate } \text{flip}((\neg p \vee q \vee \neg s), \{p \mapsto 1, q \mapsto 1, s \mapsto 0, ..\})$$

Renaming Horn Clauses

Theorem 8.10

A CNF formula $F = \{C_1, \dots, C_n\}$, where $C_i = \{\ell_{i1}, \dots, \ell_{i|C_i|}\}$ is Horn clause renamable iff the following 2-SAT formula is satisfiable.

$$G = \{\ell_{ij} \vee \ell_{ik} \mid i \in 1..n \text{ and } 1 \leq j < k \leq |C_i|\}$$

Proof.

Forward direction: there is a model m such that $\text{flip}(F, m)$ is a Horn formula

claim: $m \models G$

consider a clause $\ell_{ij} \vee \ell_{ik} \in G$

case $\ell_{ij} = p, \ell_{ik} = q$: one of them must flip, i.e., $m(p) = 1$ or $m(q) = 1$

case $\ell_{ij} = \neg p, \ell_{ik} = \neg q$: at least one must not flip, i.e., not $m(p) = m(q) = 1$

case $\ell_{ij} = \neg p, \ell_{ik} = q$: if p flips then q must, i.e., if $m(p) = 1$ then $m(q) = 1$

In all the three cases $m \models \ell_{ij} \vee \ell_{ik}$.

Renaming Horn Clauses (contd.)

Proof(contd.)

Reverse direction: Let $m \models G$. Let $F' = \text{flip}(F, m)$.

claim: F' is a Horn formula

Suppose F' is not a Horn formula.

Then, there are positive literals ℓ'_{ij} and ℓ'_{ik} in clause C_i in F' .

Therefore, $m \not\models \ell_{ij} \vee \ell_{ik}$ (why?). **Contradiction.**



Thinking Exercise 8.18

What is the complexity of checking if a formula is Horn clause renameable?

Thinking Exercise 8.19

Can you improve the above complexity?

End of Lecture 8