Lecture 9

Wednesday, February 19, 2020 5:04 PM

- Project proposals due by Feb 26
  - If you haven't already, come meet me either tomorrow (all day) or on Friday (3–5pm)

- Recap of last lecture
  Unit propagation, DPLL,
  Alg-1, Alg-2 : simple algorithms
  Horn-clauses : polynomial time solution methods

- 2-SAT in polynomial time
  Implication graph
- Schaeffer's dichotomy theorem
- DPLL + clause learning

– Decision heuristics / restarts

Unit propagation & Horn-SAT.

Ex: a and $(\neg a \vee b)$
and c
and $(\neg b \vee \neg c \vee d)$
and $(\neg c \vee \neg d \vee \neg a)$

a and $a \Rightarrow b$
and c
and $bc \Rightarrow d$
and $cda \Rightarrow$ false.

Horn-clause: At most 1 positive literal (≤ 1)

① p ② $\overline{n_1}$ or $\overline{n_2}$ or ... or $\overline{n_k}$ or p

③ $\overline{n_1} \vee \overline{n_2}$ or ... or $\overline{n_k}$ (or false)

②′ n_1 and n_2 and ... and $n_k \Rightarrow p$

③′ n_1 and n_2 and ... and $n_k \Rightarrow$ false.

Ex: and $(\neg a \vee b)$

and

and $(\neg b \vee \neg c \vee d)$

and $(\neg c \vee \neg d \vee \neg a)$

$a \Rightarrow b$

and

$bc \Rightarrow d$

and $acd \Rightarrow \text{false}$.

Claim: $\{a \mapsto \text{false}, b \mapsto \text{false}, c \mapsto \text{false}, d \mapsto \text{false}\}$

is a satisfying assignment.

Ex: $(ab \Rightarrow c)$ and $(c \Rightarrow a)$ and $(a \Rightarrow d)$ and $(d \Rightarrow b)$ and a. and $(ae \Rightarrow \text{false})$.

Solution: $\{a, b, c, d \mapsto \text{true}, e \mapsto \text{false}\}$

is a satisfying assignment.

Act-2: 2-SAT in polynomial time.

Act-2: 2-SAT in polynomial time.

A formula φ is in 2-CNF form if:

1. it is in CNF form, and
2. each clause has at most 2 literals.

Question: Given a 2-CNF formula φ,
is it satisfiable?

Goal: Introduce implication graphs.

Question: Aren't all 2-SAT instances φ
also Horn-SAT instances?

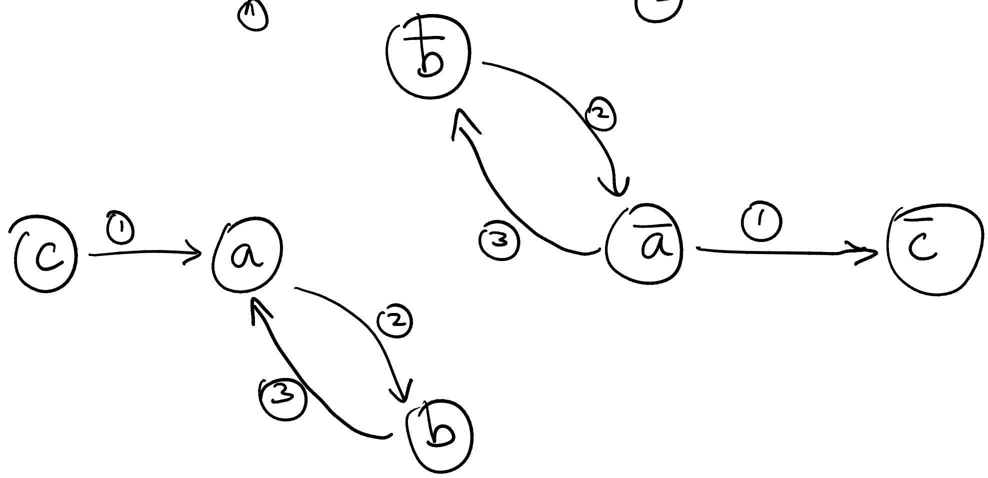
Cex: $\bar{a} \Rightarrow b$ is the same as $(\bar{\bar{a}} \text{ or } b)$
$\equiv (a \text{ or } b)$

So, no. Some 2-CNF clauses are not
Horn clauses.

Food for thought: Can we convert

Food for thought: Can we convert 2-CNF formulas φ into ~~equivalent~~ equisatisfiable Horn-SAT instances?

Ex: $\underbrace{(c \Rightarrow a)}_{1}$ and $\underbrace{(a \Rightarrow b)}_{2}$ and $\underbrace{(b \Rightarrow a)}_{3}$



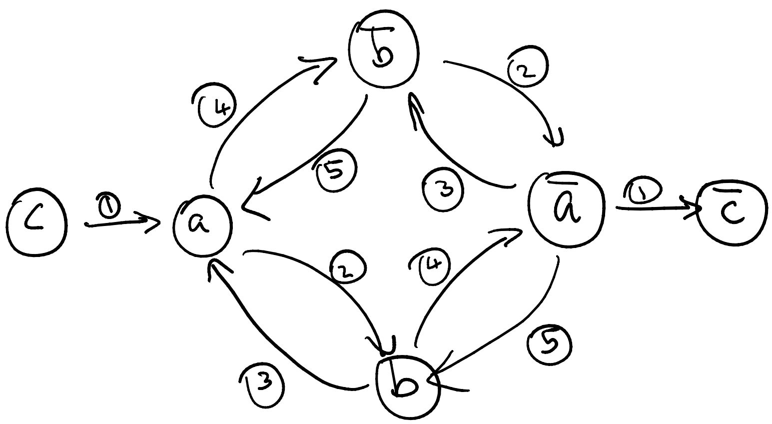
- Always pick a node or its negation (exclusive)
- If you pick a node, pick everything downstream

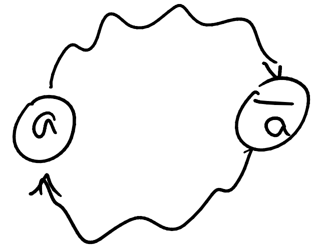
Contrapositive: $(c \Rightarrow a) \equiv (\bar{a} \Rightarrow \bar{c})$

Note: It is possible to retrieve φ from its implication graph.

its implication graph.

Ex: $\underbrace{(c \Rightarrow a)}_{1}$ and $\underbrace{(a \Rightarrow b)}_{2}$ and $\underbrace{(b \Rightarrow a)}_{3}$ and $\underbrace{(a \Rightarrow \bar{b})}_{4}$ and $\underbrace{(\bar{a} \Rightarrow b)}_{5}$





- Complaint 1: Loop containing variable & its negation

- Complaint 2: Path containing variable & its negation.

- Conjecture: Problem unsat because of complaints.

- Conjecture 1: Problem is SAT iff $\forall$ literal l

- **Conjecture ①:** Problem is SAT iff $\forall$ literal l all paths starting from l, path does not contain its negation $\bar{l}$.

Cex: $a \Rightarrow \bar{a}$ is SAT but violates conjecture ①.

a	$\bar{a}$	$a \Rightarrow \bar{a}$
T	F	F
F	T	T

$a \Rightarrow \bar{a}$ is its own contrapositive.

$$a \xrightarrow[\text{①}]{\text{①}} \bar{a}$$

- **Conjecture ②:** Problem is SAT iff $\exists$ path π st. $\forall$ literal l, π either contains l or $\bar{l}$.

Observation: $(a \Rightarrow \bar{a})$ also violates conjecture ②.

Theorem: φ (2-CNF) is satisfiable iff $\forall$ variables v, $\nexists$ loop $v \rightsquigarrow \bar{v} \rightsquigarrow v$.

Proof: **Case 1:** If sat, then $\forall v$, $\nexists$ loop

Proof: Case 1: If sat, then $\forall v$, $\nexists$ loop

Case 1': If $\exists$ loop, then $\underbrace{\text{unsat}}_{\text{not sat}}$.

If $\exists$ loop, if sat then false.

$v \rightleftarrows \bar{v}$ $\quad \vec{p}$

$\vec{p}$ either contains v or $\bar{v}$.

If it contains v, $v \rightsquigarrow \bar{v}$ means p contains $\bar{v}$

If it contains $\bar{v}$, $\bar{v} \rightsquigarrow v$ means p also contains v.

Contradiction!

Case 2: If $\nexists$ loop, then sat.

In a graph, u & v are in the same SCC if $u \rightleftarrows v$

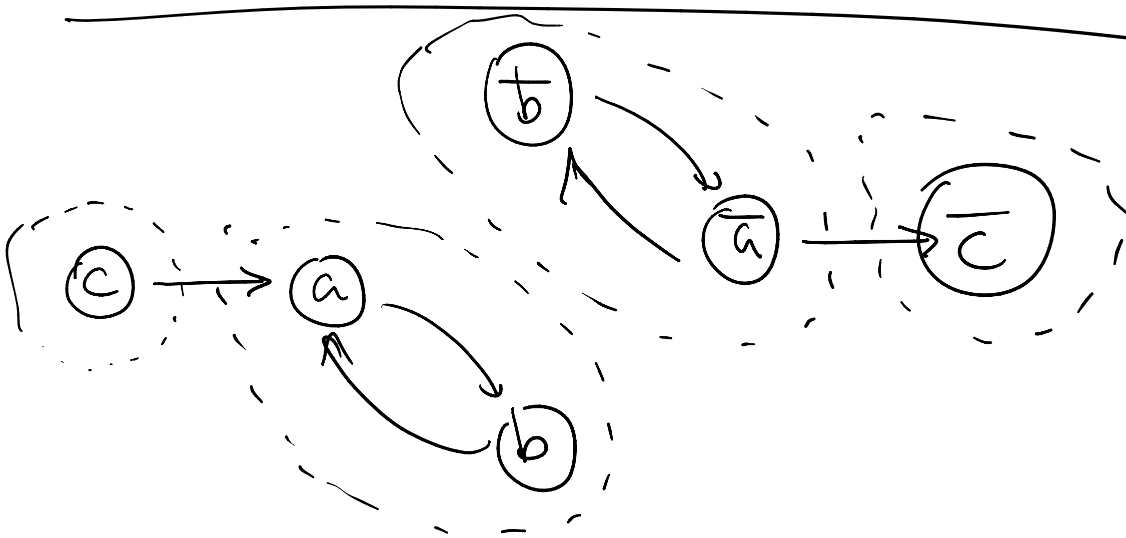
Theorem': φ is satisfiable iff $\forall$ variables v, v & $\bar{v}$ do not occur in the same SCC.

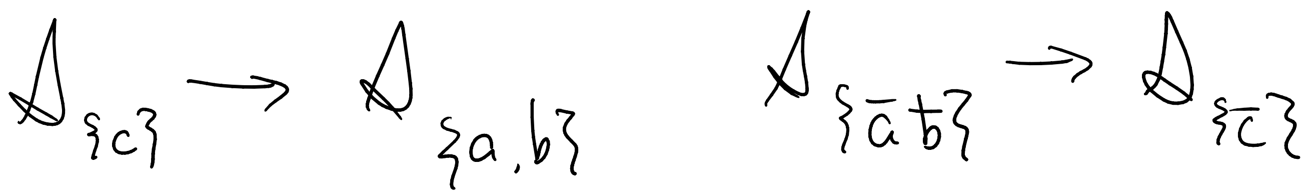
SCC $\longrightarrow$ DAG.

- Associate new vertex δ with each SCC.
- Draw edge $\delta \rightarrow \delta'$ if

$$\begin{array}{ccc} \delta & \rightarrow & \delta' \\ \cup\!\!\!| & & \cup\!\!\!| \\ u & \longrightarrow & v \end{array} \quad \& \; (\delta \neq \delta')$$

- This graph forms a DAG.
  (Food for thought!)



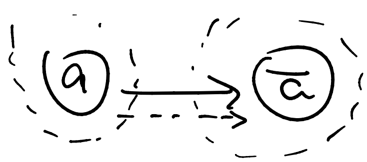


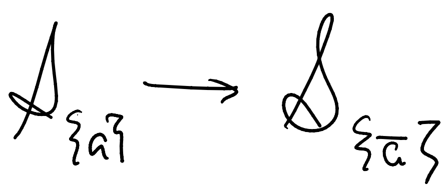
2-CNF ⇝ Implication graph

⇝ SCC

⇝ DAG ⇝ Topological order

<u>Ex</u>: $a \Rightarrow \bar{a}$



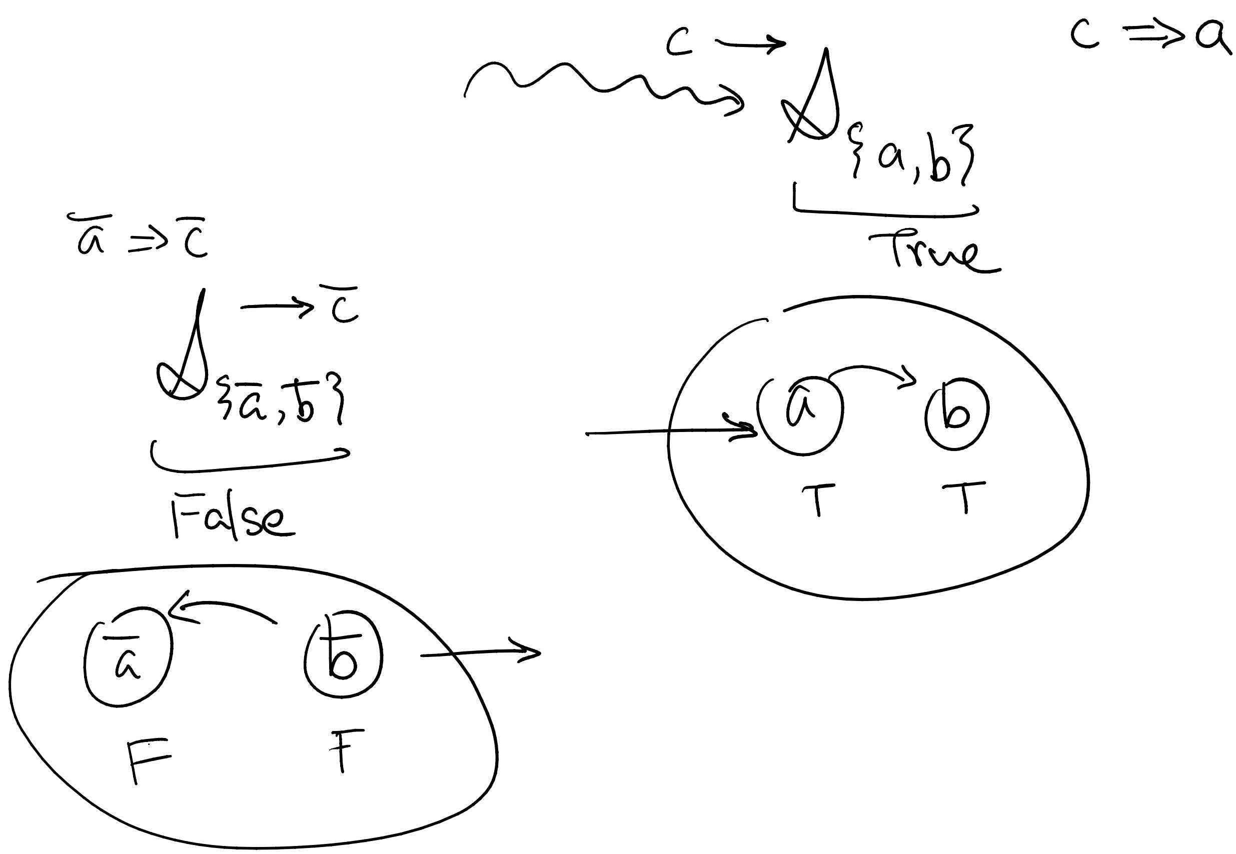


① Consider the SCCs, in reverse topological order.

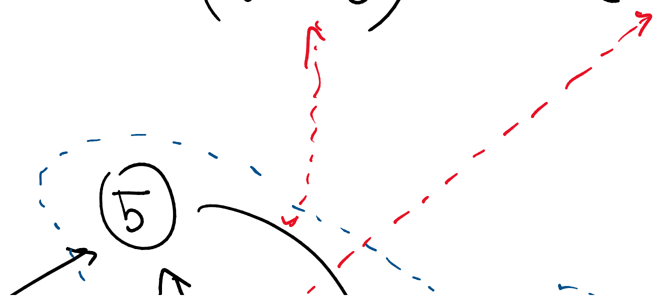
Set every literal of the SCC to true.

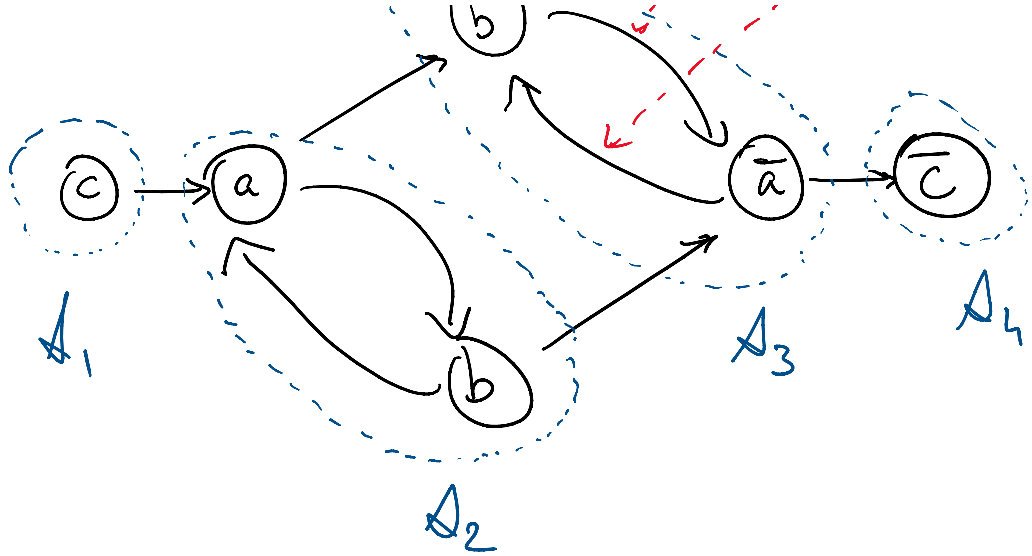
Set every literal of the complement to false

<u>Claim</u>: Assignment satisfies original formula.



<u>Ex</u>: $(c \Rightarrow a)$ and $(a \Rightarrow b)$ and $(b \Rightarrow a)$ and $(a \Rightarrow \bar{b})$





$\Delta_1 \to \Delta_2 \longrightarrow \Delta_3 \longrightarrow \Delta_4$

Step 1: Assign every literal of Δ_4 to T.

Δ_4 was the result of some clauses, {}

All clauses are satisfied.

Step 2: Same as assigning every literal of Δ_1 to F.

Δ_1 was the result of the same clauses as Δ_4.

All satisfied.

Step 3: Assign every literal

Step 3: Assign every literal
of $\mathcal{S}_3$ to true.

Clauses $(a \Rightarrow b)$ and $(b \Rightarrow a)$ both sat.

Step 4: Same as assigning every literal of $\mathcal{S}_2$ to false.

Only question: Clauses going between SCCs.

I claim: These are also satisfied.