

781-2013-02-21

Note Title

2013-02-19

In computer:

$light1_broken \leftarrow sw_up$
 $\wedge power \wedge unlit_light1.$
 $sw_up.$
 $power \leftarrow lit_light2.$
 $unlit_light1.$
 $lit_light2.$

In user's mind:

- $light1_broken$: light #1 is broken
- sw_up : switch is up
- $power$: there is power in the building
- $unlit_light1$: light #1 isn't lit
- lit_light2 : light #2 is lit

Conclusion: $light1_broken$

- The computer doesn't know the meaning of the symbols
- The user can interpret the symbol using their meaning

This will be used next class.

CNF and DNF

How to convert a prop. calculus formula to CNF & DNF.

Two main techniques.

One involves using the substitution theorem and a number of equivalences, including:

1. The "definition" of implication $P \rightarrow Q \equiv \neg P \vee Q$

2. The "definition" of equivalence $P \leftrightarrow Q \equiv (P \rightarrow Q) \wedge (Q \rightarrow P)$

3. Idempotency $(F \wedge F) \equiv F$, $(F \vee F) \equiv F$

4. Commutativity $(F \wedge G) \equiv (G \wedge F)$, $(F \vee G) \equiv (G \vee F)$

5. Associativity $((F \wedge G) \wedge H) \equiv (F \wedge (G \wedge H)) (\equiv (F \wedge G \wedge H))$

$((F \vee G) \vee H) \equiv (F \vee (G \vee H)) (\equiv (F \vee G \vee H))$

6. Absorption $(F \wedge (F \vee G)) \equiv F$, $(F \vee (F \wedge G)) \equiv F$

7. Distributivity $(F \wedge (G \vee H)) \equiv ((F \wedge G) \vee (F \wedge H))$

$(F \vee (G \wedge H)) \equiv ((F \vee G) \wedge (F \vee H))$

8. Double Negation $\neg \neg F \equiv F$

9. De Morgan's Laws $\neg (F \wedge G) \equiv (\neg F \vee \neg G)$

$\neg (F \vee G) \equiv (\neg F \wedge \neg G)$

10. Tautology Laws

$(F \vee G) \equiv F$ if F is a tautology

$(F \wedge G) \equiv G$ if F is a tautology

11. Unsatisfiability Laws

$(F \vee G) \equiv G$ if F is unsatisfiable

$(F \wedge G) \equiv F$ if F is unsatisfiable

To convert F to CNF,

1. Push negation inwards, using

$$\neg\neg G \Rightarrow G$$

$$\neg(G \wedge H) \Rightarrow (\neg G \vee \neg H)$$

$$\neg(G \vee H) \Rightarrow (\neg G \wedge \neg H)$$

until no subformulas of the form on the LHS of these rules occur.

2. Substitute in F each occurrence of a subformula of the form

$$(F \vee (G \wedge H)) \Rightarrow ((F \vee G) \wedge (F \vee H))$$

$$((F \wedge G) \vee H) \Rightarrow ((F \vee H) \wedge (G \vee H))$$

until no such subformulas occur.

The resulting formula is in CNF (except for possible occurrences of literals).

(End of method)

Another technique is based on truth tables.

For formula F , to convert to DNF, take each row of its truth-table and build a conjunction with a positive literal of the corresponding variable is assigned 1 (t), and a negative literal of the corresponding variable is assigned ϕ .

To convert F to CNF, interchange the roles of 0

and 1, and construct a disjunction for each row of the truth table.

Example. Convert to DNF and CNF the following formula

$$F = ((\neg A \rightarrow B) \wedge ((A \wedge \neg C) \leftrightarrow B))$$

A	B	C	$(\neg A \rightarrow B)$	$(A \wedge \neg C)$	$((A \wedge \neg C) \leftrightarrow B)$	F
0	0	0	0	0	1	0
0	0	1	0	0	1	0
0	1	0	1	0	0	0
0	1	1	1	0	0	0
1	0	0	1	1	0	0
1	0	1	1	0	1	0
1	1	0	1	0	1	1
1	1	1	1	0	0	0

The equivalent DNF formula is

$$(A \wedge \neg B \wedge C) \vee (A \wedge B \wedge \neg C)$$

The equivalent CNF formula is

$$(A \vee B \vee C) \wedge (A \vee B \vee \neg C) \wedge (A \vee \neg B \vee C) \wedge (A \vee \neg B \vee \neg C) \wedge$$

$$(\neg A \vee B \vee C) \wedge (\neg A \vee \neg B \vee \neg C), \text{ which is very opaque,}$$

d/c this is also an equivalent CNF formula.

$$A \wedge (B \text{ XOR } C) \equiv A \wedge (B \vee C) \wedge (\neg B \vee \neg C)$$

Let's try the other method (for CNF):

$$F = ((\neg A \rightarrow B) \wedge (A \wedge \neg C \leftrightarrow B)) \equiv$$

$$\equiv ((\neg \neg A \vee B) \wedge ((A \wedge \neg C) \rightarrow B) \wedge (B \rightarrow (A \wedge \neg C))) \equiv$$

$$\equiv (A \vee B) \wedge (\neg(A \wedge \neg C) \vee B) \wedge (\neg B \vee (A \wedge \neg C)) \equiv$$

$$\equiv (A \vee B) \wedge ((\neg A \vee C) \vee B) \wedge (\neg B \vee (A \wedge \neg C)) \equiv$$

$$\equiv (A \vee B) \wedge ((\neg A \vee C \vee B) \wedge ((\neg B \vee A) \wedge (\neg B \vee \neg C))) \equiv$$

$$\equiv \underline{(A \vee B)} \wedge (\neg A \vee B \vee C) \wedge \underline{(A \vee \neg B)} \wedge (\neg B \vee \neg C) \equiv$$

$$\equiv A \wedge (\neg A \vee B \vee C) \wedge (\neg B \vee \neg C) \equiv$$

$$\equiv A \wedge (B \vee C) \wedge (\neg B \vee \neg C)$$

Resolution

The (propositional) resolution rule.

Assume: a formula in ^(conjunctive) clausal form;

Starts with F in CNF, so

$$F = (L_{1,1} \vee \dots \vee L_{1,n_1}) \wedge \dots \wedge (L_{k,1} \vee \dots \vee L_{k,n_k})$$

(F is a conjunction of k clauses, where the i th clause contains n_i literals)

Write each clause as a set, so F is expressed as a set of clauses;

$$F = \{ \{L_{1,1}, \dots, L_{1,n_1}\}, \dots, \{L_{k,1}, \dots, L_{k,n_k}\} \}$$

Definition Resolvent.

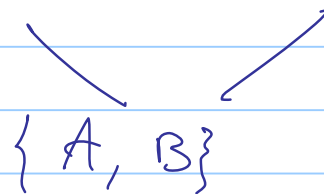
C_1 and C_2 are clauses. Then R is a resolvent of C_1 and C_2 if there is a literal $L \in C_1$ such

that $\bar{L} \in C_2$ and $R = (C_1 - \{L\}) \cup (C_2 - \{\bar{L}\})$.

$$\bar{L} = \begin{cases} \neg A_i & \text{if } L = A_i \\ A_i & \text{if } L = \neg A_i \end{cases}$$

Graphical notation $\{A, \neg C\}$ $\{A, B, C\}$

The resolvent
of $\{A, \neg C\}$ and $\{A, B, C\}$



Theorem. Let F be a CNF formula, represented

as a set of clauses. Let R be a resolvent of two clauses C_1 and C_2 in F . Then, F and $F \cup \{R\}$ are equivalent.

Definition Let F be a set of clauses. Then

$\text{Res}(F)$ is defined as;

$$\text{Res}(F) = F \cup \{R \mid R \text{ is a resolvent of two clauses in } F\}$$

Furthermore, define

$$\text{Res}^0(F) = F$$

$$\text{Res}^{n+1}(F) = \text{Res}_1(\text{Res}^n(F)) \text{ for } n \geq 0, \text{ and}$$

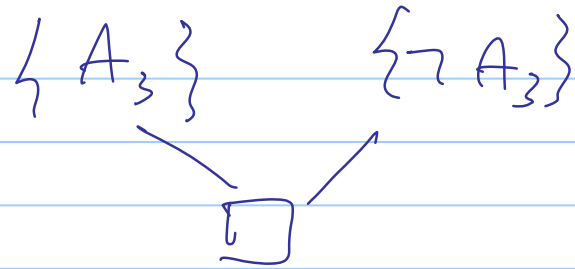
finally let

$$\text{Res}^*(F) = \bigcup_{n \geq 0} \text{Res}^n(F).$$

It can be proved that for every finite F , there is a k s.t. $\text{Res}^k(F) = \text{Res}^*(F)$.

Defn. The empty resolvent is the resolvent of $C_1 = \{L\}$ and $C_2 = \{\bar{L}\}$.

This is also called the empty clause, and is indicated by \square .



The Resolution Theorem (of propositional logic)

[J.A. Robinson proved this for FOL around 1960.]

A clause set F is unsatisfiable iff $\square \in \text{Res}^*(F)$.

Example (Ex 33 Schöningh) Using resolution,

show that $(A \wedge B \wedge C) \Rightarrow \perp$ is a consequence of

The clause set $F = \{ \neg A, B \}, \{ \neg B, C \}, \{ A, \neg C \}, \{ A, B, C \}$

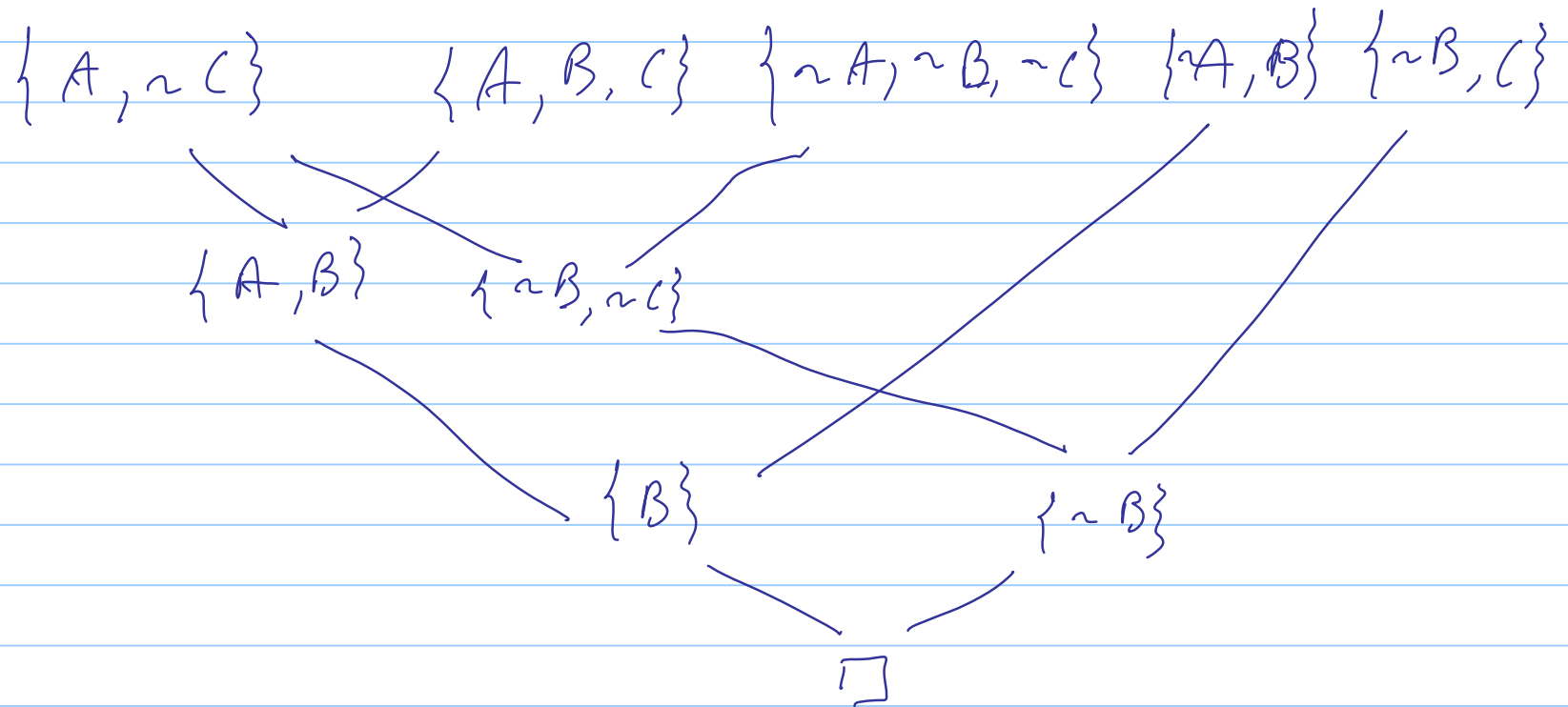
Recall: $F \models G$ iff $\vdash F \rightarrow G$ iff $F \wedge \neg G$ is unsatisfiable

∴ Negate G and show that $F \wedge \neg G$ is unsat

$\neg G$: $\{ \neg A, \neg B, \neg C \}$. Show that

$F \cup \neg G$: $\{ \neg A, B \}, \{ \neg B, C \}, \{ A, \neg C \}, \{ A, B, C \}, \{ \neg A, \neg B, \neg C \}$
is unsat, i.e., by the res thm, show that

$$\text{Res}^*(FC \cap G) = \square.$$



Show by resolution that the following formula is unsatisfiable:

$$F = (P \vee Q) \wedge (P \vee \neg Q) \wedge (\neg P \vee Q) \wedge (\neg P \vee \neg Q)$$

$$\{P, Q\} \quad \{P, \neg Q\} \quad \{\neg P, Q\} \quad \{\neg P, \neg Q\}$$

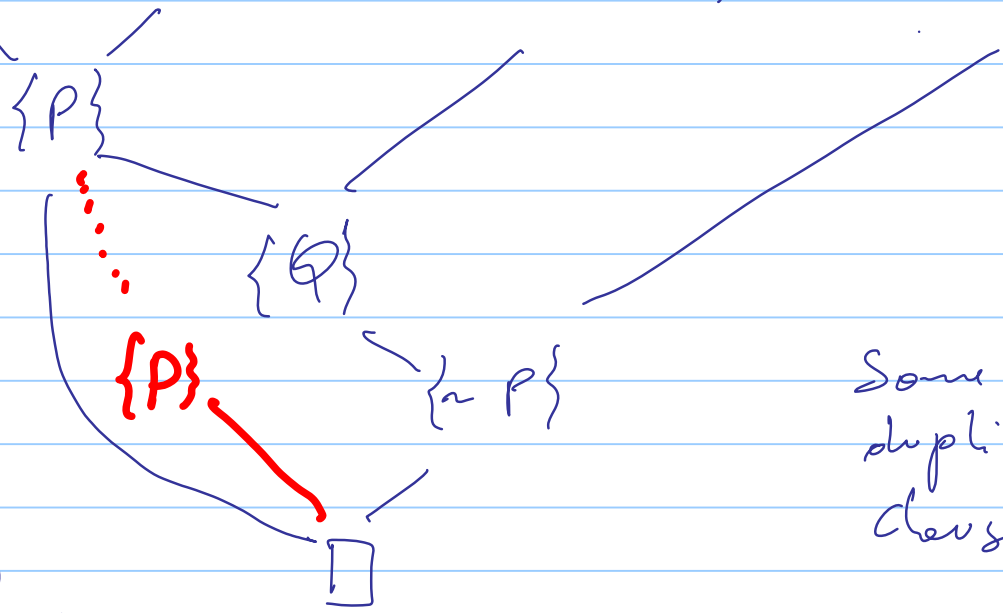
$$\{P\}$$

$$\{\neg P\}$$

□

Another proof of the same:

$\{P, Q\}$ $\{P, \sim Q\}$ $\{\sim P, Q\}$ $\{\sim P, \sim Q\}$



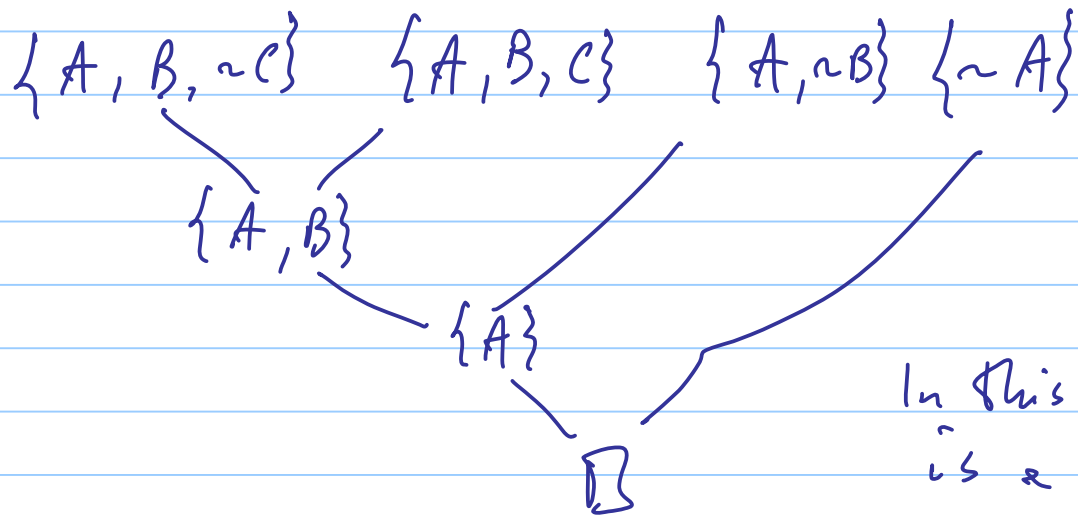
This edge → shows that this is a refute Non graph (Schöningh)

Some authors (eg. Loveland) duplicate reused clauses

Another example - Show that

$F = \{\{A, B, \neg C\}, \{\neg A\}, \{A, B, C\}, \{A, \neg B\}\}$ is

unsatisfiable



This is
an example
of a
resolution graph.
In this case, the r. g.
is a tree.

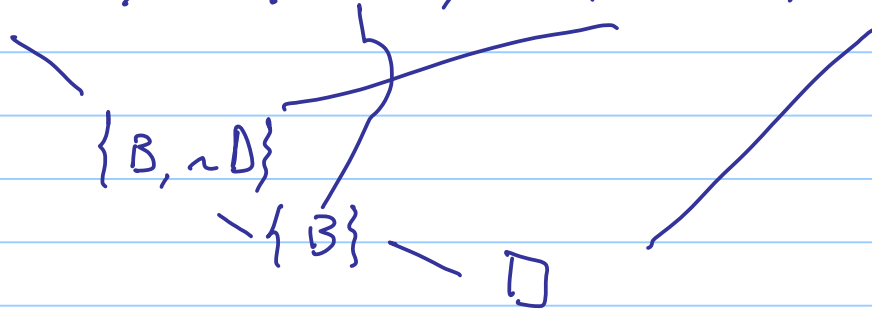
Exercise 34 [Solving]

Using resolution, show that

$F = (\neg B \wedge \neg C \wedge D) \vee (\neg B \wedge \neg D) \vee (C \wedge D) \vee B$ is a tautology.
We show that $\neg F$ is unsatisfiable.

$\neg F = (B \vee C \vee \neg D) \wedge (B \vee D) \wedge (\neg(C \vee \neg D)) \wedge \neg B$, in clausal form

$\{B, C, \neg D\} \quad \{B, D\} \quad \{\neg C, \neg D\} \quad \{\neg B\}$



A formula F in CNF is a Horn formula if every disjunction^(clause) in F contains at most one positive literal.

We distinguish three kinds of Horn clauses

- ① A a fact ^{exactly and only} (one positive literal)
- ② $\neg v \wedge B_1 \vee \dots \vee \neg B_k$ a rule (one pos. literal & at least one negative literal)
- ③ $\neg B_1 \vee \dots \vee \neg B_k$ an integrity constraint (or indefinite clause) (only negative literals)

Horn clauses are often written in implicative form.

(Reminder/defn. : 1 stands for any tautology
0 " " " " contradiction.)

① $1 \rightarrow A$ (fact, usually written as A)

② $B_1 \wedge \dots \wedge B_k \rightarrow H$ (rule)

③ $B_1 \wedge \dots \wedge B_k \rightarrow 0$ (integrity constraint)

There is an efficient (linear-time) algorithm to test satisfiability of Horn formulas. Here it is:

Input: a Horn formula F (in implicational form)

1. Mark every occurrence of an atomic formula A in F if there is a clause $(\perp \rightarrow A)$ in F .

2. while there is a clause G in F of form (2) or (3),
where B_1, \dots, B_k are already marked (and H is
not yet marked),

do

— if G is of form (2),

then mark every occurrence of H

else output "unsatisfiable" and halt.

3. Output "satisfiable" and halt

(The satisfying assignment is given by the marking,
in that variable $A_i = 1$ iff A_i has a mark.)

[The above algorithm is sound & complete. No proof given. It provides the minimal model for F .]

Exercise 22 (Schöningh) Give an example of

a formula that does not have an equivalent Horn formula.

$A \vee B$, because this formula does not have a single minimal model.

Homework (Ex. 21 (Schöningh));

HW2

due Tuesday.

Apply the marking algorithm to

$$F = (\neg A \vee \neg B \vee \neg D) \wedge \neg E \wedge (\neg C \vee A) \wedge C \wedge B \wedge (\neg G \vee D) \wedge G$$

If the formula is unsatisfiable, please also show that by resolution.