Mathematical Logic 2016

Lecture 4: Normal forms

Instructor: Ashutosh Gupta

TIFR, India

Compile date: 2016-08-12

Where are we and where are we going?

We have seen

- propositional logic syntax and semantics
- truth tables as methods for deciding SAT
- some common equivalences

We will learn

- Various normal forms
  - NNF (seen in the previous lecture)
  - CNF
  - DNF
  - k-SAT
- DNF formula minimization
- Encoding sat problems

Some terminology

- Propositional variables are also referred as atoms
- A literal is either an atom or its negation
- A clause is a disjunction of literals.

Since $\vee$ is associative, commutative and absorbs multiple occurrences, a clause may be referred as a set of literals

Example 4.1

- *p is an atom but $\neg p$ is not.*
- *$\neg p$ and p both are literals.*
- *$p \vee \neg p \vee p \vee q$ is a clause*
- *$\{p, \neg p, q\}$ is the same clause*

Topic 4.1

Conjunctive normal form

Conjunctive normal form(CNF)

Definition 4.1

A formula is in CNF if it is a conjunction of clauses.

Since $\wedge$ is associative, commutative and absorbs multiple occurrences, a CNF formula may be referred as a set of clauses

Example 4.2

- $\neg p$ *and* p *both are in CNF.*
- $(p \vee \neg q) \wedge (r \vee \neg q) \wedge \neg r$ *in CNF.*
- $\{(p \vee \neg q), (r \vee \neg q), \neg r\}$ *is the same CNF formula.*
- $\{\{p, \neg q\}, \{r, \neg q\}, \{\neg r\}\}$ *is the same CNF formula.*

Exercise 4.1

Write a formal grammar for CNF

CNF conversion

Theorem 4.1

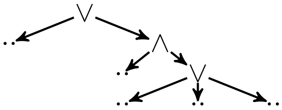
For every formula F there is another formula F' in CNF s.t. $F \equiv F'$.

Proof.

Let us suppose we have

- removed $\oplus$, $\Rightarrow$, $\Leftrightarrow$ using the equivalences seen earlier,
- converted the formula in NNF, and
- flattened $\land$ and $\lor$.

Now the formulas have the following form with literals at leaves.



After the push formula size grows! Why should the procedure terminate?

Since $\lor$ distributes over $\land$, we can always push $\lor$ inside $\land$.

Eventually, we will obtain a formula that is CNF. □

CNF conversion terminates

Theorem 4.2

The procedure of converting a formula in CNF terminates.

Proof.

For a formula F, let $\nu(F) \triangleq$ the maximum height of $\vee$,$\wedge$ alternations in F.
Consider a formula $F(G)$ such that

$$G = \bigvee_{i=0}^{m} \bigwedge_{j=0}^{n_i} G_{ij}.$$

After the push we obtain $F(G')$, where

$$G' = \bigwedge_{j_1=0}^{n_1} \dots \bigwedge_{j_m=0}^{n_m} \underbrace{\bigvee_{i=0}^{m} G_{ij_i}}_{\nu(\qquad)<\nu(G)}$$

Observations

- G' is either the top formula or the parent connective(s) are $\wedge$
- G_{ij} is either a literal or an $\vee$ formula

We need to apply flattening to keep $F(G')$ in the form(of the previous slide). ...

CNF conversion terminates (contd.)

(contd.)

Due to Köing lemma, the procedure terminates.(why?) □

Exercise 4.2

Consider a set of balls that are labelled with positive numbers. We can replace a k labelled ball with any number of balls with labels less than k. Using Köing lemma, show that the process always terminates.

Hint: in the above theorem, the bag is the subformulas of $F(G)$.

CNF examples

Example 4.3

Consider $(p \Rightarrow (\neg q \wedge r)) \wedge (p \Rightarrow \neg q)$

$\equiv (\neg p \vee (\neg q \wedge r)) \wedge (\neg p \vee \neg q)$

$\equiv ((\neg p \vee \neg q) \wedge (\neg p \vee r)) \wedge (\neg p \vee \neg q)$

$\equiv (\neg p \vee \neg q) \wedge (\neg p \vee r) \wedge (\neg p \vee \neg q)$

Exercise 4.3

Convert the following formulas into CNF

1. $\neg((p \Rightarrow q) \Rightarrow ((q \Rightarrow r) \Rightarrow (p \Rightarrow r)))$
2. $(p \Rightarrow (\neg q \Rightarrow r)) \wedge (p \Rightarrow \neg q) \Rightarrow (p \Rightarrow r)$

Conjunctive normal form(CNF) (2)

- A unit clause contains only one single literal.
- A binary clause contains two literals.
- A ternary clause contains three literals.
- We naturally extend definition of the clauses to empty set of literals. We refer to $\perp$ as empty clause.

Example 4.4

- $(p \wedge q \wedge \neg r)$ *has three unit clauses*
- $(p \vee \neg q \vee \neg s) \wedge (p \vee q) \wedge \neg r$ *has a ternary, a binary and a unit clause*

Exercise 4.4

a. Show F' obtained from the procedure may be exponentially larger than F
b. Give a linear time algorithm to prove validity of a CNF formula

Tseitin's encoding

We can translate every formula into CNF without exponential explosion using Tseitin's encoding by introducing fresh variables.

1. Assume input formula F is NNF without $\oplus$, $\Rightarrow$, and $\Leftrightarrow$.
2. Find a $G_1 \wedge \cdots \wedge G_n$ that is just below a $\vee$ in $F(G_1 \wedge \cdots \wedge G_n)$
3. Replace $F(G_1 \wedge .. \wedge G_n)$ by $F(p) \wedge (\neg p \vee G_1) \wedge .. \wedge (\neg p \vee G_n)$, where p is a fresh variable
4. goto 2

Exercise 4.5

Convert the following formulas into CNF using Tseitin's encoding

1. $(p \Rightarrow (\neg q \wedge r)) \wedge (p \Rightarrow \neg q)$
2. $(p \Rightarrow q) \vee (q \Rightarrow \neg r) \vee (r \Rightarrow q) \Rightarrow \neg(\neg(q \Rightarrow p) \Rightarrow (q \Leftrightarrow r))$

Exercise 4.6

Modify the encoding such that it works without the assumptions at step 1

Hint: Download sat solver \$wget http://fmv.jku.at/limboole/limboole1.1.tar.gz look for function tseitin in file limboole.c

Tseitin's encoding preserves satisfiability

Theorem 4.3
if $m \models F(p) \wedge (\neg p \vee G_1) \wedge \cdots \wedge (\neg p \vee G_n)$ *then* $m \models F(G_1 \wedge \cdots \wedge G_n)$

Proof.
Assume $m \models F(p) \wedge (\neg p \vee G_1) \wedge \cdots \wedge (\neg p \vee G_n)$.

Case $m \models p$:

- Therefore, $m \models G_i$ for all $i \in 1..n$.
- Therefore, $m \models G_1 \wedge \cdots \wedge G_n$.
- Therefore, $m \models F(G_1 \wedge \cdots \wedge G_n)$.

Case $m \not\models p$ and $m \not\models G_1 \wedge \cdots \wedge G_n$:

- Due to the substitution theorem, $m \models F(G_1 \wedge \cdots \wedge G_n)$

...

Tseitin's encoding preserves satisfiability(contd.)

Proof(contd.)

Case $m \not\models p$ and $m \models G_1 \wedge \cdots \wedge G_n$:

- Since $F(G_1 \wedge \cdots \wedge G_n)$ is in NNF, p occurs only positively in F.
- Therefore, $m[p \mapsto 1] \models F(p)_{\text{(why?)}}$.
- Since p does not occur in G_is, $m[p \mapsto 1] \models G_1 \wedge \cdots \wedge G_n$.
- Due to the substitution theorem, $m[p \mapsto 1] \models F(G_1 \wedge \cdots \wedge G_n)$
- Therefore, $m \models F(G_1 \wedge \cdots \wedge G_n)$. $\square$

Exercise 4.7

Show if $\not\models F(p) \wedge (\neg p \vee G_1) \wedge .. \wedge (\neg p \vee G_n)$ *then* $\not\models F(G_1 \wedge .. \wedge G_n)$

Topic 4.2

Disjunctive normal form

Disjunctive normal form(DNF)

Definition 4.2

A formula is in DNF if it is a disjunction of conjunctions of literals.

Theorem 4.4

For every formula F there is another formula F' in DNF s.t. $F \equiv F'$.

Proof.

Proof is similar to CNF. □

Exercise 4.8

a. Give the formal grammar of DNF

b. Give a linear time algorithm to prove satisfiability of a DNF formula

Topic 4.3

k-sat

k-sat

Definition 4.3

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

Example 4.5

- $(p \wedge q \wedge \neg r)$ *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 satisfiablity

Theorem 4.5

For each k-SAT formula F there is a 3-SAT formula F' with linear blow up such that F is sat iff F' is sat.

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 do not appear in F.
Let G' be the following set of clauses

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

We show G is sat iff G' is sat. ...

Exercise 4.9

Convert the following CNF in 3-SAT

- $(p \vee \neg q \vee s \vee \neg t) \wedge (\neg q \vee x \vee \neg y \vee z)$

3-SAT satisfiability(cont. I)

Proof(contd. from last slide).

Recall

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

Assume $m \models G'$:

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

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

Due to ith 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, exists $i \in 1..k$ $m(\ell_i) = 1$. Therefore $m \models G$.

...

3-SAT satisfiability(cont. II)

Proof(contd. from last slide).

Recall

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

Assume $m \models G$:
There is a $m(\ell_i) = 1$.
Let $m' = m[x_2 \mapsto 1, .., x_{i-1} \mapsto 1, x_i \mapsto 0, \ldots, x_{k-2} \mapsto 0]$
Therefore, $m' \models G'_{\text{(why?)}}$.

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

Exercise 4.10

a. Complete the above argument.

b. Show a 3-SAT formula cannot be converted into 2-SAT using Tseitin's encoding

Topic 4.4

Encoding in SAT

SAT encoding

Since SAT is a NP-complete problem, therefore any NP-hard problem can be encoded into SAT in polynomial size.

We will look into a few interesting examples.

Encoding into CNF

CNF is the form of choice

- Most problems specify collection of restrictions on solutions
- Each restriction is usually of the form

$$\text{if-this} \Rightarrow \text{then-this}$$

The above constraints are naturally in CNF.

"Even if the system has hundreds and thousands of formulas, it can be put into CNF piece by piece without any multiplying out"

– Martin Davis and Hilary Putnam

Exercise 4.11

Which of the following two encodings of ite(p, q, r) is in CNF?

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

Coloring graph

Problem:

color a graph$(\{v_1, \ldots, v_n\}, E)$ with at most d colors s.t. if $(v_i, v_j) \in E$ then color of v_1 is different from v_2.

SAT encoding

Variables: p_{ij} for $i \in 1..n$ and $j \in 1..d$. p_{ij} is true iff v_i is assigned jth color.
Clauses:

- Each vertex has at least one color

$$\text{for each } i \in 1..n \qquad (p_{i1} \vee \cdots \vee p_{id})$$

- if $(v_i, v_j) \in E$ then color of v_1 is different from v_2.

$$(\neg p_{ik} \vee \neg p_{jk}) \qquad \text{for each } k \in 1..d, \quad (v_i, v_j) \in 1..n$$

Exercise 4.12

a. Encode: "every vertex has at most one color."

b. Do we need this constraint to solve the problem?

Pigeon hole principle

Prove:

if we place $n+1$ pigeons in n holes then there is a hole with at least 2 pigeons

The theorem holds true for any n, but we can prove it for a fixed n.

SAT encoding

Variables: p_{ij} for $i \in 0..n$ and $j \in 1..n$. p_{ij} is true iff pigeon i sits in hole j.
Clauses:

- Each pigeon sits in at least one hole

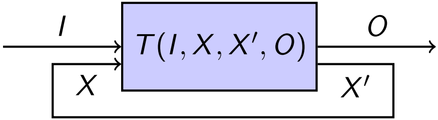
$$\text{for each } i \in 0..n \qquad (p_{i1} \vee \cdots \vee p_{in})$$

- There is at most one pigeon in each hole.

$$(\neg p_{ik} \vee \neg p_{jk}) \qquad \text{for each } k \in 1..n, \quad i < j \in 1..n$$

Bounded model checking

Consider a Mealy machine



- I is a vector of variables representing input
- O is a vector of variables representing output
- X is a vector of variables representing current state
- X' is a vector of variables representing next state

Prove: After n steps, the machines always produces output O that satisfies some formula $F(O)$.

Bounded model checking encoding

SAT encoding:

Variables:

- $I_0, \dots, I_{n-1}$ representing input at every step
- $O_1, \dots, O_n$ representing output at every step
- $X_0, \dots, X_n$ representing internal state at every step

Clauses:

- Encoding system runs

$$T(I_0, X_0, X_1, O_1) \wedge \dots \wedge T(I_{n-1}, X_{n-1}, X_n, O_n)$$

- Encoding property

$$\neg F(O_n)$$

If the encoding is unsat the property holds.

Topic 4.5

Problems

CNF

Exercise 4.13

Convert the following formulas into CNF

1. $(p \Rightarrow q) \lor (q \Rightarrow \neg r) \lor (r \Rightarrow q) \Rightarrow \neg(\neg(q \Rightarrow p) \Rightarrow (q \Leftrightarrow r))$

P=NP argument

Exercise 4.14

What is wrong with the following proof of P=NP? Give counterexample.

Tseitin's encoding does not explode and proving validity of CNF formulas has a linear time algorithm. Therefore, we can convert every formula into CNF in polynomial time and check validity in linear time. As a consequence, we can solve sat of F in linear time by checking validity of $\neg F$ in linear time.

Validity

Exercise 4.15

Give a procedure like Tseitin's encoding that converts a formula into another formula while preserving validity. Prove correctness of your transformation.

SAT encoding

Exercise 4.16

Encode N-queens problem in a SAT problem.
N-queens problem: Place n queens in $n \times n$ chess such that none of the queens threatens each other.

End of Lecture 4