Standard Forms and Resolution

P. Danziger

1 Conjunctive Normal Form

We wish to form a standard language for manipulating compound logical expressions.

Definition 1

- An *atom* is a statement p.
- A *literal* is a statement p or its negation $\sim p$, i.e. an atom or its negation.

  Literals are usually denoted by lowercase letters from the end of the alphabet x, so $x_i = p_i$ or $y_i = \sim q_i$. Here x_i is a *positive literal* and y_i is a *negative literal.*
- A *clause* is an OR ($\vee$) of a number of literals $x_1 \vee x_2 \vee \ldots \vee x_k$.

  The *length* of a clause is the number of literals in it.
- A logical expression is in *Conjunctive Normal Form* (CNF) if it is composed of a number of clauses ANDed ($\wedge$) together. $(x_1 \vee x_2 \vee \ldots \vee x_k) \wedge (y_1 \vee y_2 \vee \ldots \vee y_k) \wedge \ldots \wedge (z_1 \vee z_2 \vee \ldots \vee z_k)$.
- If all of the clauses of a CNF expression have the same length, k, we say that the expression is in $k-$CNF.

Theorem 2 *Any logical expression without quantifiers can be put in CNF.*

Proof: Any expression can be put in Disjunctive Normal Form by considering its truth table and applying the algorithm for translating this to a logical expression. By repeated application of the distributive rule this may be transfered to CNF.

Example 3

1. The following are in CNF:

   (a) $p \vee q$

   This is a single clause of length 2. This expression is in $2-$CNF.

   (b) $p \wedge q$

   In this example p is a clause and q is a clause, each are of length 1. This expression is in $1-$CNF.

   (c) $(p_1 \vee p_2 \vee p_3) \wedge (\sim p_1 \vee p_3) \wedge (\sim p_2 \vee \sim p_3)$

   In this example there are three clauses: $(p_1 \vee p_2 \vee p_3)$, $(\sim p_1 \vee p_3)$ and $(\sim p_2 \vee \sim p_3)$, the first of length 3, the rest of length 2. This expression is not in $k-$CNF for any k.

(d) $(p \vee q) \wedge (\sim p \vee \sim q)$
In this example there are two clauses: $(p \vee q)$ and $(\sim p \vee \sim q)$. This expression is in 2−CNF.

2. The following are **not** in CNF:

   (a) $\sim(p \vee \sim q)$.
   (b) $(p \wedge \sim q \wedge \sim r) \vee (\sim p \wedge \sim q \wedge r)$.
   (c) $((p_1 \vee p_2) \wedge (\sim p_1 \vee p_3)) \vee (\sim p_2 \vee \sim p_3)$.
   (d) $p \Rightarrow \sim q$.

   Though all can be translated to an equivalent CNF form.

3. Put the following in CNF

   (a) $\sim(p \vee \sim q)$.
   $\equiv \sim p \wedge q$ (DeMorgan & Double Negation)
   (b) $(p \wedge q) \vee (\sim q \wedge r)$
   $\equiv (p \vee (\sim q \wedge r)) \wedge (q \vee (\sim q \wedge r))$ (Distribution)
   $\equiv (p \vee \sim q) \wedge (p \vee r) \wedge (q \vee \sim q) \wedge (q \vee r)$
   (c) The expression E whose truth table is given by the following:

p	q	r	E	
T	T	T	F	
T	T	F	F	
T	F	T	F	
T	F	F	T	$\leftarrow (p \wedge \sim q \wedge \sim r)$
F	T	T	F	
F	T	F	F	
F	F	T	T	$\leftarrow (\sim p \wedge \sim q \wedge r)$
F	F	F	F	

   Applying the method for putting an expression in DNF:

$$E \equiv (p \wedge \sim q \wedge \sim r) \vee (\sim p \wedge \sim q \wedge r)$$

   Applying distribution gives:

$$\begin{aligned} E \equiv\; & (p \vee \sim p) \wedge (p \vee \sim q) \wedge (p \vee r) \wedge (\sim q \vee \sim p) \wedge (\sim q \vee \sim q) \wedge \\ & (\sim q \vee r) \wedge (\sim r \vee \sim p) \wedge (\sim r \vee \sim q) \wedge (\sim r \vee r). \end{aligned}$$

2 Satisfiability

Given a formula in CNF it is possible to ask whether there is an assignment of truth values to the variables so that the formula evaluates to true. If there is such an assignment the formula is said to be *satisfiable* if not it is said to be *unsatisfiable*.
If the original formula is in k−CNF the problem of determining satisfiability is called k−SAT.

Example 4
Find an assignment so that the following formulas are satisfiable:

1. $p \vee \sim p$

   This expression is in 2−SAT. Any assignment $p = T$ or $p = F$ will satisfy this expression.

2. $p \wedge \sim p$

   This expression is in $1 - SAT$. No assignment of values to p will satisfy this expression. Thus this is unsatisfiable.

3. $(p \vee q \vee \sim r) \wedge (\sim p \vee q \vee \sim r) \wedge (p \vee \sim q \vee \sim r)$

   This expression is in 3−SAT. The assignment $p = T$, $q = T$, $r = F$ will satisfy this expression. Note that this is not the only assignment that works, for example $p = F$, $q = F$, $r = F$ also satisfies the expression.

4. $(p \vee q \vee r) \wedge (p \vee q \vee \sim r) \wedge (p \vee \sim q \vee r) \wedge (p \vee \sim q \vee \sim r) \wedge (\sim p \vee q \vee r) \wedge (\sim p \vee q \vee \sim r) \wedge (\sim p \vee \sim q \vee r) \wedge (\sim p \vee \sim q \vee \sim r)$.

   This expression is in 3−SAT. This formula is not satisfiable.

The following important Theorems are known about k−SAT:

- There is a Polynomial time algorithm for solving 2-SAT
- (Cook's Theorem) 3-SAT is NP complete.

  This means that there is no known polynomial time algorithm to solve an arbitrary instance of 3-SAT. Further, it is known to be one of a set of computationally hard problems.

  One of the central questions of theoretical Computer Science is whether such a polynomial time algorithm exists. This can be succinctly stated as does $P = NP$?

Much work has gone into building SAT solvers. Such machines can be used as automated theorem provers, but the "3-SAT is hard" bound limits their usefulness.

3 Skolem Standard Form or Prenex Form

Note Skolem Standard Form is also often called *prenex* form. Prenex form is used to extend CNF to the case where there are quantifiers.

Definition 5 *A statement is in* Prenex *form or* Skolem Standard Form *if all the quantifiers appear at the beginning of the statement before any predicates and the remainder of the statement is in CNF, with the predicates as literals.*

Theorem 6 *Any logical expression can be put in prenex form.*

Proof: If quantifiers do not appear at the beginning they may be moved there using the Rule for combining quantified statements.

Example 7

1. The following are in Prenex form.
   (a) $\forall x, P(x) \vee Q(x)$.
   (b) $\forall x \in S,\ \exists y \in T, P(x) \wedge (Q(x,y) \vee R(y))$
   (c) $\forall x \in S.\ \forall y \in T, P(x) \wedge (Q(x,y) \vee \sim R(y))$
2. The following are **not** in Prenex form.
   (a) $\forall x, P(x) \vee \exists y, Q(x,y)$.
   (b) $\forall x, P(x) \vee (Q(x,y) \wedge \sim R(x,y))$.
3. Put the following in prenex form.
   (a) $\forall x, P(x) \vee \exists y, Q(x,y)$.
   $\equiv \forall x, \exists y, P(x) \vee Q(x,y)$.
   (b) $\forall x, (\exists y, P(x,y)) \Rightarrow (\exists y, Q(x,y))$
   Note that in the original expression the variable y is used for two different things. We replace one of them with a z.
   $\equiv \forall x, \sim(\exists y, P(x,y)) \vee (\exists z, Q(x,z))$
   $\equiv \forall x, \forall y,\ \sim P(x,y) \vee (\exists z, Q(x,z))$
   $\equiv \forall x, \forall y, \exists z, \sim P(x,y) \vee Q(x,z)$.

4 Prolog and Unification

The programming language was developed in the 1970s to make it easy to code complex logical deductions. The main reason for this was to facilitate programming for artificial intelligence. See p. 107 of the book for a brief description and example of Prolog in action.
One of the central ideas which is implemented in Prolog is the notion of *Unification*. At its simplest level unification is similar to variable assignment, an unassigned variable is given a value. The value may be a logical atom, a compound statement or another variable. However, in Prolog it is possible to make assignments by implication. It is thus possible for unification to fail, if the implication leads to a contradiction or is ambiguous.

Definition 8 *Given a set of assignments $X = \{x_1 = a_1, \ldots, x_k = a_k\}$, where x_i are variables, and two expressions E_1 and E_2. We say X* unifies *E_1 and E_2 if they are the same expression after making the variable assignments given in X.*

Example 9
In the following examples X, Y etc are unassigned variables, p, q etc. are atoms and A, B etc are assigned variables.

1. $X = p$, X is unified with the atom p.

2. $X = (p \vee q) \wedge r$, X is unified with the expression given.
3. $p = q$, unification fails the atoms are different.
4. $p = p$, unification succeeds.
5. $X = Y$, Unification succeeds, X and Y are now synonymous. If Y is already unified to an atom, X is now bound to that atom as well.
6. $X = A$, $A = p \vee q$, X is unified with A, since A is unified with $p \vee q$, X is as well.
7. $f(X) = f(p)$, succeeds X is unified with the atom p.
8. $f(X, p) = f(Y, Z)$, succeeds X is unified with the variable Y and Z with p. Note that this unification will fail if Z is already unified to a different atom or expression.
9. $f(X, A) = f(Y, B)$, This unification will fail if A and B are already unified to different atoms. It succeeds if A is already unified with B, or such a unification is consistent, X is then unified with the variable Y.
10. $f(X) = g(Y)$, unification fails, different head.
11. $f(X) = f(Y, Z)$, unification fails, different arity.

5 Resolution

The resolution rule is a valid argument which allows us to simplify a CNF expression.

$$\begin{array}{l} (p \vee q) \wedge (\sim p \vee r) \\ \hline \therefore (q \vee r) \end{array}$$

This can be restated as

$$\begin{array}{l} (p \vee q) \\ (\sim p \vee r) \\ \hline \therefore (q \vee r) \end{array}$$

We may prove such a statement as we would any valid form using a truth table.

					Premise	Conc.	
p	q	r	$p \vee q$	$\sim p \vee r$	$(p \vee q) \wedge (\sim p \vee r)$	$q \vee r$	
T	T	T	T	T	T	T	← c.r
T	T	F	T	F	F	T	
T	F	T	T	T	T	T	← c.r
T	F	F	T	F	F	F	
F	T	T	T	T	T	T	← c.r
F	T	F	T	T	T	T	← c.r
F	F	T	F	T	F	T	
F	F	F	F	T	F	F	

The conclusion is true in all critical rows, so the form is valid.
Since q can itself be a compound expression, this can be generalized to:

$$\frac{(p \vee q_1 \vee q_2 \vee \ldots q_k) \wedge (\sim p \vee r_1 \vee r_2 \vee \ldots \vee r_{k'})}{\therefore (q_1 \vee q_2 \vee \ldots q_k) \wedge (r_1 \vee r_2 \vee \ldots \vee r_{k'})}$$

If the expression involves quantifiers in prenex form we may still resolve it by applying the following rules:

1. Search for 2 clauses containing the same predicate P, but positive in one and negative in the other.
2. Unify the arguments appearing in the two clauses. Note that this may fail.
3. We may now apply the resolution rule above to eliminate P. The remaining predicates act on the resolved form.

We may use this scheme to prove forms with quantifiers.

Example 10

1. Consider Universal Modus Ponens

$$\begin{array}{ll} \forall x, (P(x) \Rightarrow Q(x)) & \equiv \ \forall x, \ \sim P(x) \vee Q(x) \\ P(a) & \\ \hline \therefore Q(a) & \end{array}$$

   (a) P appears negated in the first statement and positive in the second.

$$\begin{array}{l} \forall x, \ \sim P(x) \vee Q(x), \\ P(a) \end{array}$$

   (b) In the first clause we have $P(x)$, in the second $P(a)$, The unification $P(x) = P(a)$ unifies x with a.

   (c) Applying the unification $x = a$ and dropping the now redundant quantifier for x gives, via the resolution rule:

$$\begin{array}{l} \sim P(a) \vee Q(a), \\ P(a) \\ \hline \therefore Q(a) \end{array}$$

2. Consider Generalized transitivity of implication

$$\begin{array}{lcl} \forall x, (P(x) \Rightarrow Q(x)) & \equiv & \forall x, (\sim P(x) \vee Q(x)) \\ \forall y, (Q(y) \Rightarrow R(y)) & \equiv & \forall y, (\sim Q(y) \vee R(y)) \\ \hline \therefore \forall z, (P(z) \Rightarrow R(z)) & \equiv & \forall z, (\sim P(z) \vee R(z)) \end{array}$$

   (a) Q appears positive in the first statement and negated in the second.

$$\begin{array}{l} \forall x, (\sim P(x) \vee Q(x)) \\ \forall y, (\sim Q(y) \vee R(y)) \end{array}$$

(b) Unification of two free free variables x and y sets them equal.

(c) We may now combine the quantifiers for x and y and apply the resolution rule to get

$$\begin{array}{l} \forall x, (\\ \qquad \sim P(x) \vee Q(x) \\ \qquad \sim Q(x) \vee R(x) \\ \quad) \\ \hline \qquad \therefore \forall x, \sim P(x) \vee R(x) \end{array}$$

But $\forall x, \sim P(x) \vee R(y) \equiv \forall z, (P(z) \Rightarrow R(z))$

3. The following pair of clauses fail at the unification step:

$$\begin{array}{l} \forall x, \forall y P(x,y) \vee Q(y) \\ \forall x, \sim P(x,a) \vee Q(b) \end{array}$$

Though this is a candidate for a resolution rule, since P appears in the first clause and $\sim P$ in the second it fails at the unification step. At this point we would apply the unifications $P(x,y) = P(x,a)$, which would unify y to a and $Q(y) = Q(b)$, which would unify y to b.

Resolutions are used to simplify a set of expressions in the hope of producing a simple expression. We can use this in an attempt to solve the satisfiability problem above.

Theorem 11 *If a set of expressions can be reduced to the empty set by resolution then the AND of those expressions is unsatisfiable.*

SWP.
We thus start with an expression in CNF or Prenex form, we break this up into its constituent clauses and attempt to resolve the result into the empty clause.
Note that this may fail to terminate if the original expression is in fact satisfiable.

5.1 Horn Clauses

Another alternative is to use Horn Clauses.

Definition 12 *A* Horn Clause *is a clause of a CNF expression which has exactly one literal positive.*

Note that given a Horn clause

$$(\sim p_1 \vee \sim p_2 \vee \dots \sim p_{k-1} \vee p_k) \equiv (p_1 \wedge p_2 \wedge \dots p_{k-1}) \rightarrow p_k$$

and so the variable p_k can now be replaced by the expression $p_1 \wedge p_2 \wedge \dots p_{k-1}$.

6 Exercises

1. Convert the following to CNF. state how many clauses the result has and the length of each clause. If the result is in $k-$CNF for some k give k.

   (a) $\sim(p \wedge q)$

   (b) $(p \wedge \sim q \wedge \sim r) \vee (\sim p \wedge \sim q \wedge r)$.

   (c) $(p_1 \vee p_2) \wedge (\sim p_1 \vee p_3) \vee (\sim p_2 \vee \sim p_3)$.

   (d) The expression E whose truth table is given by the following:

p	q	r	E
T	T	T	F
T	T	F	T
T	F	T	F
T	F	F	T
F	T	T	F
F	T	F	F
F	F	T	F
F	F	F	T

2. Determine whether the following expressions are satisfiable. If they are give an assignment to the variables so that the expression evaluates to true.

   (a) $(p \vee q) \wedge (p \vee \sim q) \wedge (\sim p \vee \sim q)$

   (b) $(p \vee q) \wedge (p \vee \sim q) \wedge (\sim p \vee q)$

   (c) $(p \vee q) \wedge (p \vee \sim q) \wedge (\sim p \vee q) \wedge (\sim p \vee \sim q)$

   (d) $(p \vee q \vee r) \wedge (p \vee \sim q \vee \sim r) \wedge (\sim p \vee q \vee r)$

3. Put the following in prenex form

   (a) $(\forall x, P(x)) \vee (\exists x, P(x))$.

   (b) $((\forall x, P(x)) \vee (\exists x, P(x))) \wedge (\forall y, Q(x))$.

   (c) $\sim(\forall x, P(x) \vee \exists y,\ \sim Q(y)) \Rightarrow (\forall x, R(x))$

   (d) $\sim(\forall x, P(x)) \Rightarrow (\forall x, Q(x))$

4. Explain the following unifications by giving the assignments. If the unifications fails explain why.

   (a) $X = Y$, $Y = p$.

   (b) $f(g(x), y) = f(g(a), b)$.

   (c) $f(g(x), y) = f(g(a), x)$.

   (d) $f(g(x), x) = f(g(a), b)$.

5. Use Resolution to prove Generalised Modus Tollens:

$$\begin{array}{l} \forall x, (P(x) \Rightarrow Q(x)) \\ \sim Q(a) \\ \hline \therefore \sim P(a) \end{array}$$

6. Use Resolution to prove Generalised Division into Cases:

$$\begin{array}{l} \forall x, P(x) \vee Q(x)) \\ \forall y, P(y) \Rightarrow R(y) \\ \forall z, Q(z) \Rightarrow R(z) \\ \hline \therefore \forall x, R(x)) \end{array}$$

(Hint: you need to use resolution twice.)