CS 245

Notes 5

Standardized formats

for formulae in Form(LP)

can aid in reading,

understanding, analyzing

and the automated processing

of formulae.

In particulars, gaining an intuitive understanding of deeply nested formulae may be challenging.

E.g.

$$A_1 \to (A_2 \to \dots (A_n \to A))$$

One such normal form, used in several applications, is conjunctive normal form (CNF).

Formulae in CNF have the feature that there are at most two levels of nesting.

In addition, the $\neg$ operator is applied only to atoms.

To describe how to write formulae in CNF we will also describe a dual format Disjunctive Normal Form (DNF).

Def. If $p \in Atom(LP)$ then p is a literal and $\neg p$ is a literal.

Def. If $A_1, \ldots, A_k$ are literals then $(A_1 \wedge \ldots \wedge A_k)$ is a conjunctive clause, and $(A_1 \vee \ldots \vee A_k)$ is a disjunctive clause.

Def. A disjunctions of conjunctive clauses is a formula in DNF.

A conjunction of disjunctive clause is a formula in CNF.

So for instance

$$(A_{1,1} \wedge \dots \wedge A_{1,n_1}) \vee \dots \vee (A_{k,1} \wedge \dots \wedge A_{k,n_k})$$

is a formula in DNA

while

$$(A_{1,1} \vee \dots \vee A_{1,n_1}) \wedge \dots \wedge (A_{k,1} \vee \dots \vee A_{k,n_k})$$

is in CNF.

Here, the $A_{i,j}$ are all literals, that is, atoms or the negation of atoms.

So if a formula is in CNF or DNF its only binary connectives are $\wedge$ and $\vee$, and negation is applied only to atoms.

Fact: For any formula $A \in Form(L^p)$ there is a formula B in DNF such that $A \dashv\vdash B$.

Fact: for any formula $A \in Form(L^p)$ there is a formula B in CNF such that $A \dashv\vdash B$.

Show that A is equiv to B in DNF.

Idea: Consider $A \in Form(L^P)$, and build the truth table for A.

If A is a contradiction then $(p \wedge \neg p)$ is equivalent to A and $(p \wedge \neg p)$ is in DNF.

Suppose A is not a contradiction. Consider the case where A has 3 atoms, a_1, a_2 and a_3.

From the truth table, find all rows for which A evaluates to 1.

For each row construct a clause as follows. Each atom will contribute one literal to the clause.

If atom a_i has value 1 in the row then a_i appears in the clause.

If atom a_i has value 0 in the row then $\neg a_i$ appears in the clause.

Then take the disjunction of all the clauses, the resulting formula is in DNF and is equivalent to the original formula.

Example: Suppose A has three atoms a_1, a_2, and a_3.

Furthers, suppose A has value 1 under the three valuations

$a_1^t = 1$, $a_2^t = 1$, $a_3^t = 0$

and

$a_1^{t_1} = 1$, $a_2^{t_1} = 0$, $a_3^{t_1} = 1$

and

$a_1^{t_2} = 0$, $a_2^{t_2} = 0$, $a_3^{t_2} = 1$.

Otherwise $A^{t'} = 0$.

Then

$$B = (a_1 \wedge a_2 \wedge \neg a_3) \vee (a_1 \wedge \neg a_2 \wedge a_3) \vee (\neg a_1 \wedge \neg a_2 \vee a_3)$$

is in DNF and $B \not\equiv A$.

We can also use the rules for rewriting equivalent formulae to transform formulae in $\text{form}(L^{\omega})$ into DNF. Similarly formulae in DNF can be transformed into equivalent formulae in CNF and vice versa.

Our analysis of, for instance whether $A, B \vdash C$, has reasoned about both the syntax and semantics of the formulae.

A purely syntactic analysis style may allow for machine (computer) generated proofs or machine checkable proofs.

We consider two such proof systems, resolution and natural deduction.

For both resolution and natural deduction, the expectation is that any proof demonstration should be mechanically checkable purely based on rules about the syntactic structure of the formulae.

A proof will then consist of a list of formulae.

Typically, any premises are listed first. Then a formula may be added to the list if some rule of inference is then applied.

Here is an example rule of inference.

If we have deduced that $A \vee \neg B$ holds and that B holds we can then deduce that A holds.

This rule is given as:

$$\frac{A \vee \neg B \quad B}{A}$$

The premises of the rule are given above the line.

The conclusion of the rule below the line.

Given a set of premises, Σ, and a formula A that we deduce using a proof system, we write

$$\Sigma \vdash A.$$

Notice the similarity with $\Sigma \models A$.

In general inference rules are given as:

$$\frac{A_1 \quad \dots \quad A_k}{B}$$

If all the A_i have been proven (appear in the current proof) then we may infer B.

The proof systems we will look at, resolution and natural deduction, are sound.

That is, if we show that $\Sigma \vdash A$ then we can conclude that $\Sigma \models A$.

Resolution is of some interest because it is often used in building automated proof systems.

Resolution proofs have two special features.

The formulae in the proof are all in CNF.

Successful deductions produce contradictions denoted by $\perp$.

Resolution proofs use the inference rule

$$\frac{A \vee B \qquad \neg B \vee C}{A \vee C}$$

There are two special cases:

Unit resolution

$$\frac{A \vee B \quad \neg B}{A}$$

Contradiction:

$$\frac{A \qquad \neg A}{\bot}$$

A proof using the resolution system is finished when $\perp$ is derived.

In this way resolution is known as a refutation based proof system.

Example:

Show that

$\{p, q, \neg p \vee \neg q\} \vdash \bot$

1.	p	premise
2.	q	premise
3.	$\neg p \vee \neg q$	premise
4.	$\neg q$	1, 3
5.	$\bot$	2, 4

Since resolution proofs derive contradictions, if we want to show $\Sigma \vdash A$ we show

$$\Sigma \cup \{\neg A\} \vdash \bot .$$

We can then conclude $\Sigma \vdash A$.

To apply the resolution method: convert the formulae to CNF and then separate the conjunctions.

In this way our previous deduction showed $\{p, q\} \vdash p \wedge q$.

Example: Show

$\{p \to q, q \to r\} \vdash p \to r$

Convert to CNF

$\{\neg p \vee q, \neg q \vee r\} \vdash \neg p \vee r$

Show that

$$\{\neg p \vee q,\ \neg q \vee r,\ p,\ \neg r\} \vdash \bot$$

1.	$\neg p \vee q$	premise
2.	$\neg q \vee r$	premise
3.	p	premise
4.	$\neg r$	premise
5.	q	1, 3
6.	r	2, 5
7.	$\bot$	4, 6