Propositional logic part 03 proof system

Conjunctive and disjunctive normal form

1. <u>CNF</u>

A formula in conjunctive normal form (CNF) is a conjunction of clauses, where a clause is a disjunction of literals. CNF formulas are used in the context of automated theorem proving and in solving the SAT problem (particularly in the DPLL algorithm).

Example
All of the following expressions are in CNF:

- A ∧ B
- A
- (A ∨B) ∧C

(A ∨ ¬B ∨¬C ∨ ¬D) ∧ (¬D∨ E∨F)

I.1. **Conversion Algorithm to CNF**

Step 1: Eliminate all occurrences of the connectors → and ↔ by replacing them:

F → G with ¬F∨G
And F ↔G with (¬F∨G) ∧ (¬G∨F)

Step 2: Apply De Morgan's laws to move the ¬ inward by replacing:

¬(F∨G) par ¬F∧¬G
¬(F∧G) par ¬F∨¬G

Step 3: Eliminate double negations by replacing

¬¬ F with F.

Step 4: Apply the distributive rules by replacing

F∨(G ∧H) par (F∨G) ∧ (F∨H)
(F∧G)∨H par (F∨H) ∧ (G∨H)

<u>2. DNF</u>

A disjunctive normal form (DNF) is a normalization of a logical expression that is a disjunction of conjunctive clauses. It is used in automated theorem proving. A logical expression is in DNF if and only if it is a disjunction of one or more conjunctions of one or more literals.

Example
All of the following expressions are in DNF:

- A ∨ B

- A
- (A ∧ B) ∨C
- (A ∧ ¬B ∧¬C ∧ ¬D) ∨(¬D∧ E∧F)

2.1. Conversion   Algorithm to DNF

The same algorithm of CNF only replacing **∧ by ∨ and ∨ by ∧** in step **2** and **4**

Exercise

transform the given formulae into conjunctive normal form (apply the conversion algorithm

1. ((b ∨ c) =⇒ a ∨ d
2. (P ∧ (Q ⇒ R) ⇒ S)

Solution

1.

Step 1 : ((b ∨ c) =⇒ a ∨ d ≡ (¬(b ∨ c) ∨ a) ∨ d

Step 2 : (¬(b ∨ c) ∨ a) ∨ d ≡ ((¬b ∧ ¬c) ∨ a) ∨ d

Step 3 : /

Step 4 : ((¬b ∧ ¬c) ∨ a) ∨ d≡((¬b ∨ a) ∧ (¬c ∨ a)) ∨ d ≡ ((¬b ∨ a) ∨ d) ∧ ((¬c ∨ a)) ∨ d) ≡ (¬b ∨ a ∨ d) ∧ (¬c ∨ a ∨ d).

2.

(P ∧ (Q ⇒ R) ⇒ S)

(P ∧ (Q ⇒ R) ⇒ S) → ¬(P ∧ (Q ⇒ R)) ∨ S
→ (¬P ∨ ¬(Q ⇒ R)) ∨ S
→ (¬P ∨ ¬(¬Q ∨ R)) ∨ S)
→ (¬P ∨ (Q ∧ ¬R)) ∨ S
→ ((¬P ∨ Q) ∧ (¬P ∨ ¬R)) ∨ S
→ (¬P ∨ Q ∨ S) ∧ (¬P ∨ ¬R ∨ S)

Proof system

The truth table is a simple and reliable method for checking the validity of a logical deduction. However, this statement is not always true, especially if the number of propositional variables exceeds a certain limit, such as 8, 16, or more. In these cases, using a truth table to verify the validity of a logical deduction becomes impractical.

To find an alternative method for verifying the validity of logical deductions, we must turn to the mathematical theorems.

1. deduction theorem (DT):

We have the following deduction theorem (DT):

F* |= *C if and only if |= ***(F → C)***

This theorem can be expressed as: "C is deduced from F if and only if (F→C) is a tautology .

2. Refutation theorem (RT) :

F* |= *C if and only if ***F ∪{¬C} is inconsistent***

3. Resolution Schemes

3.1.*Modus Ponens*: it is the simplest scheme ,{Contrôle, Contrôle → Note}|= Note

3.2.General inference rule : Among the various existing reasoning schemes, we will focus on the resolution rule, of which Modus Ponens and Modus Tollens are actually special cases. This inference rule is expressed as follows:

{ X ∨A,¬X ∨B} |= A∨B.

If A and B are two ***complementary clauses*** (which contain the literals ***Φ*** and ¬***Φ,*** respectively), then we can deduce the ***new clause C***, called the ***resolvent***, obtained by combining all the literals ***from A and B except for Φ and ¬Φ."***

3.3.Transitivity Rule of Propositional Calculus

p→q, q→r |= p→r

4. Proof by Resolution in Propositional Logic

• Method

– Negation of the conclusion

– Conversion to clause form

– Application of the resolution principle until obtaining the empty clause

– Conclusion.