Propositional Logic – Resolution (part 1)

Logical deduction:

- To define a deduction relation or verify reasoning, we can use two techniques:
  - Model theory (truth table)
  - Proof theory (formal system).

Model theory	Demonstration theory (Proof theory)
$\boldsymbol{Q \vDash P}$	$\boldsymbol{Q \vdash P}$
P consequence of Q	P deducible from Q. from Q , we can deduce P
$\vDash \boldsymbol{P}$	$\vdash \boldsymbol{P}$
P is tautology	P is theorem
Semantics If we have n variables, then the truth table comprising 2^n lines	Syntax

- The proof theory of proof is based on the definition of the formal system.
- In this part, we will use the formal system that allows to simulate reasoning, and it uses calculations to prove a formula.
- It consists of:
  - Language (e.g. propositional, predicate),
  - Well-formed formulas
  - Axioms
  - Valid deduction rules (inference rules)
- **A formal system is correct:** it does not allow us to prove formulas that are not true.
- **A formal system is complete:** it allows us to prove everything that is true.

- **Robinson's resolution principle** is a system of inference, it is correct and complete.
- It is used particularly in logic programming systems.
- **Prolog** language is a logic programming language where the program is made up of a set of clauses.

Clause formatting

- In this part, we only use formulas in clausal form.
  - Literal: A literal is an atomic formula or its negation. p or$\neg p$
  - Clause: A clause is a disjunction of literals.$(l_1 \vee \ldots \vee\ l_n)$
    - Examples: $p \vee q, \neg r \vee\ \neg p$

- In the clausal form, only two connectors are used, negation and disjunction.$\neg/\vee$.
- To obtain the set of clauses from a formula:
  - Put the formula in CNF form,
  - We eliminate conjunction connectors,
  - Then, we obtain the set of clauses of the initial formula.

Example: Let the formula F: $(p\vee q)\Rightarrow(p\wedge q)$, find the set of equivalent clauses:

- We write the formula F under FNC:
- $(p\vee q)\Rightarrow(p\wedge q)$
- $\neg (p\vee q)\vee(p\wedge q) \equiv (\neg p\wedge\neg q)\vee(p\wedge q)$
- $(\neg p\vee q)\wedge(p\vee \neg q)$ FNC
- The set of clauses of formula F is: $\{\neg p\vee q;\ p\vee \neg q\}$

Propositional Resolution Rule

- Robinson's propositional resolution rule is a rule of logical inference.
- A rule of logical inference means a rule for deducing a new formula.

- The resolution rule is defined as:

$$\frac{\boldsymbol{p} \qquad (\boldsymbol{p}\Rightarrow\boldsymbol{q})}{\boldsymbol{q}} \quad \textbf{or} \quad \frac{\boldsymbol{p} \qquad (\neg\boldsymbol{p}\vee\boldsymbol{q})}{\boldsymbol{q}} \quad \textbf{or}\ \ \{\boldsymbol{p}; \neg\boldsymbol{p} \vee \boldsymbol{q}\} \vdash \boldsymbol{q}$$

- Reads:
  - from p and "$(\neg p \vee q)$", we can deduce q.
  - q is deducible from p and "$(\neg p \vee q)$ ".

- The resolution rule is generalized as:

$$\{p \vee l_1 \vee l_2 \vee \ldots \vee l_n; \neg p \vee l'_1 \vee l'_2 \vee \ldots \vee l'_m\} \vdash l_1 \vee l_2 \vee \ldots \vee l_n \vee l'_1 \vee l'_2 \vee \ldots \vee l'_m$$

$$\{C_1;\ C_2\} \vdash P$$

- We note:
    - $C_1 = p \vee l_1 \vee l_2 \vee \ldots \vee l_n$
    - $C_2; \neg p \vee l'_1 \vee l'_2 \vee \ldots \vee l'_m$
    - $P = l_1 \vee l_2 \vee \ldots \vee l_n \vee l'_1 \vee l'_2 \vee \ldots \vee l'_m$
- We say: From the clause C1 and the clause C2, we can derive the clause P.

Examples 1: Applying the resolution rule, find the deriving clause from C1 and C2:

- $C_1 = q \vee r; C_2 = \neg q \vee s$
    - The deriving clause is $p \vee s$
    - We note Res$(C_1, C_2) = p \vee s$
    - $q \vee r; \neg q \vee s \vdash p \vee s$

Examples 2: By applying the resolution rule, find the deriving clause.

- $C_1 = \neg p \vee q \vee \neg r; C_2 = \neg s \vee p \vee \neg r$
    - Res$(C_1, C_2) = q \vee \neg r \vee \neg s$ (removing redundancy of literals)
    - $\{\neg p \vee q \vee \neg r; \neg s \vee p \vee \neg t\} \vdash q \vee \neg r \vee \neg s$

Resolving two singleton clauses leads to the ***empty clause***; i.e. the clause consisting of no literals at all, as shown below. The derivation of the empty clause means that the database contains a contradiction.

- $C_1 = \neg p; C_2 = p$
    - Res$(C_1, C_2) = \square$ (empty clause)
    - $\{\neg p;\ p\} \vdash \square$

Note:

- The deriving clause P from two clauses C1 and C2 $\{C_1; C_2\} \vdash P$ is a logical consequence of C1 and C_2 $\{C_1; C_2\} \vDash P$

- The resolution method allows to highlight the semantic consequence.

Refutation

- The idea of proof by refutation is to demonstrate that the set is insatisfiable.$C \cup \{\neg P\}$

Refutation theorem:

- Let be C a set of clauses, $C \vdash P$ is equivalent to $C \cup \{\neg P\} \vdash \square$.

Demonstration:

- We have (1) $C \vdash P$
- We have (2) $\neg P \vdash \neg P$
- $C \cup \{\neg P\} \vdash \neg P \wedge P$ (1) and (2)
- So, $C \cup \{\neg P\} \vdash \square$

Exercise: Transform the following formulas into formulas in clausal form M

- ¬ (p ⇒ q) ∨ (p ∧ q)
- (p ∧ ¬q) ∨ (¬p ∧ q)

Find the resolving clauses in the following cases:

1. C1=¬q ∨ p C2=r ∨ ¬p ∨ s
2. C1=¬q ∨ p C2=q
3. C1=¬q ∨ ¬p C2=p ∨ q ∨ ¬r