innovate achieve lead

BITS Pilani
Pilani Campus

CS/IS F214
Logic in Computer Science

MODULE: PROPOSITIONAL LOGIC

Semantics - Introduction

Propositional Logic - Semantics

- The semantics of propositional logic - i.e. meaning of sentences in propositional logic - has been discussed informally :
  - Truth Tables!
- Truth Tables (notation invented by Wittgenstien) define what a boolean (or propositional logic) operation means.
- Exercise:
  - Read and understand section 1.4.1 (The meaning of logical connectives).
    - Most of the section has been covered in tutorials and used in lectures and tutorials. So this should just be a quick review.

Soundness (of Proofs and Proof Rules)

- We saw proofs using ND rules.
  - Are all such proofs correct?
    - Is each proof rule correct?
  - Correctness relates to the meaning of the logical operations as given in truth tables
    - Does each proof rule (in ND) *preserve the meaning of the connectives* (i.e. operations) as given by the respective truth tables?
    - Recall the arguments given using truth tables when the rules were discussed.

Semantics of Formulas and Semantic Entailment

- To understand this, we define a notion of correctness of meaning (referred to as semantic entailment):
  - If for all valuations in which
    - all $\varphi_1, \varphi_2, \ldots, \varphi_n$ evaluate to TRUE, ψ also evaluates to TRUE
  - then we say that
    - $\varphi_1, \varphi_2, \ldots, \varphi_n \models \psi$
- **|=** denotes the ***semantic entailment*** relation.
- Example:
  - Verify: **p --> q |= !q --> !p**

Soundness (of Proof System.)

- **Soundness (of ND)**:
  - **If** $\varphi_1, \varphi_2, \ldots, \varphi_n \;|\!-_{ND}\; \psi$ then $\varphi_1, \varphi_2, \ldots, \varphi_n \;|\!= \psi$
    - or informally: *Anything provable - in ND - is TRUE*.
- [**Note:**
  - *This is a claim* – that **ND** is sound. At this point this is used to illustrate the concept of Soundness.
  - We will – in subsequent lectures – prove this claim.

End of Note.]

- More generally,
  - a proof system **D** is said to be sound, if anything provable in **D** is TRUE i.e. semantically obtainable.

Completeness – Natural Deduction and Propositional Logic

- Can *everything that is true* (of statements in propositional logic) *be proved using Natural Deduction*?
  - i.e. *is there a proof system* for propositional logic in which *every thing that is true can be proved*?

- This notion is referred to as the ***completeness of the proof system***
  - and therefore the ***completeness of propositional logic***.

Completeness of Propositional Logic

- **Completeness** (of Natural Deduction):
  - If $\varphi_1, \varphi_2, \ldots, \varphi_n \models \psi$ then $\varphi_1, \varphi_2, \ldots, \varphi_n \vdash_{ND} \psi$
    - Or informally:
      - *Everything TRUE is provable in ND*

- **Completeness** (of Propositional Logic):
  - There is a proof system **D** such that
    - if $\varphi_1, \varphi_2, \ldots, \varphi_n \models \psi$ then $\varphi_1, \varphi_2, \ldots, \varphi_n \vdash_{D} \psi$
  - for propositional formulas $\varphi_1, \varphi_2, \ldots, \varphi_n,$ and ψ

innovate achieve lead

BITS Pilani
Pilani Campus

CS/IS F214
Logic in Computer Science

MODULE: PROPOSITIONAL LOGIC

Semantic Equivalence and Normal Forms

Semantic Equivalence

- Two formulas ψ and φ are equivalent if:
  - $\psi \models \varphi$ and $\varphi \models \psi$
    - and we denote it as $\varphi \mathrel{=}\!\mid\!\mathrel{=} \psi$

- Prove the following equivalence:
  - p -->q =|= ¬q --> ¬p

- Sometimes the notation
  - $\varphi \equiv \psi$

  is used in place of
  - $\varphi \mathrel{=}\!\mid\!\mathrel{=} \psi$

Semantic Equivalence vs. Syntactic Equivalence

- Note that syntactic equivalence and semantic equivalence are two different notions:
- Two formulas φ and ψ are syntactically equivalent if

  ψ |-- φ and φ |-- ψ

  - and we denote it as

  ψ --|-- φ

- Prove the following (syntactic) equivalence:
  - **p -->q --|-- $\neg$q --> $\neg$p**

Double Implication

- Note that one can define a double implication operator:
    - ψ <--> φ
  - to denote
    - (ψ --> φ) $\wedge$ (φ --> ψ)

- Question:

  How does ψ <--> φ differ from ψ --|-- φ and from ψ =|= φ?

Normal Forms

- As seen in the example (for semantic equivalence)
  - there can be two or more *(syntactically) different formulae* that are *semantically equivalent*.
- Is there a standardized notation so that
  - *all equivalent formulae can be expressed using the (one) syntactic form*?
- Such forms do exist - they are referred to as ***canonical forms*** or ***normal forms***:
  - in particular, there are two commonly used forms:
    - Conjunctive Normal Form (CNF)
    - Disjunctive Normal Form (DNF)

Conjunctive Normal Form (CNF)

- A propositional logic formula φ is said to be in **CNF** if the formula is *a conjunction of sub-formulas* (or ***clauses***):

  i.e. it is of the form $\mathbf{C_1 \wedge C_2 \wedge \ldots \wedge C_n}$

  where each clause $\mathbf{C_i}$ is a *disjunction of literals*:

  i.e. it is of the form $\mathbf{L_{i1} \vee L_{i2} \vee \ldots \vee L_{im}}$

- where each literal $\mathbf{L_{ij}}$ is
  - either *an atomic proposition* (**p**) or the negation of *an atomic proposition* ($\neg$**p**) .

- In Boolean logic, the CNF is referred to as the ***Product-of-Sums*** (***POS***) form.

Conjunctive Normal Form - Example

- We saw that:
  - **p -->q =|= ¬q --> ¬p**

- We can write both these formulas in CNF:
  - **p --> q** can be written as:
    - **¬p ∨ q** (why?)
      - which is in CNF (of just one clause).
  - **¬q --> ¬p** can be written as:
    - **¬ ¬q ∨ ¬p**
  - which can be rewritten as
    - **q ∨ ¬p**
      - which is in CNF as well.
- Note that these two formulas – i.e. **p -->q** and **¬q --> ¬p** – have been rewritten as the same formula (*modulo commutativity*):
  - i.e. **q ∨ ¬p**

Disjunctive Normal Form (DNF)

- A propositional logic formula is said to be in **DNF** if the formula is a *disjunction of clauses*

  i.e. it is of the form $\mathbf{C_1 \lor C_2 \lor \ldots \lor C_n}$

  where each clause C_i is *a conjunction of literals*:

  i.e. . it is of the form $\mathbf{L_{i1} \land L_{i2} \land \ldots \land L_{im}}$

  where each literal $\mathbf{L_{ij}}$ is either *an atomic proposition* (**p**) or the *negation of an atomic proposition* ($\neg$**p**).

- In Boolean logic, the DNF is referred to as the ***Sum-of-Products*** (***SOP***) form.