LIN241

Introduction to Semantics

Lecture 2

Logic and Arguments

Logic and Arguments

Why study logic in a semantics course?

- It will help us understand meaning relations between sentences.
- It will help us understand the meaning of logical words like conjunctions.
- We will find it useful to ask how natural languages differ from the artificial languages of logic.

Logic and Arguments

Logic: the formal study of valid arguments.

What is a valid argument?

- An argument is a sequence of premises with a conclusion.
- Valid arguments: whenever the premises are true, the conclusion is true.

  I.e. the conclusion is entailed by the premises.

Examples of valid arguments

- Modus Ponens:

  If it snows, it is cold.

  It snows.

  ∴ It is cold.

The "∴" sign means "therefore".

Examples of valid arguments

- Modus Tollens:

  If John is in love, he is happy.

  John is not happy.

  ∴ John is not in love.

Examples of invalid arguments

- Affirming the consequent:

  If it snows, it is cold.

  It is cold.

  ∴ It snows.

Examples of invalid arguments

- Denying the antecedent:

  If it snows, it is cold.

  ∴ If it does not snow, it is not cold.

Logic and Arguments

- Showing that an argument is valid:
  - show that every way to make its premises true also make its conclusion true.
- Showing that an argument is not valid:
  - show that there is a way to make its conclusion false while all its premises are true.

Logic and Arguments

- Logically valid arguments are valid in virtue of their form.
- Validity is preserved by substitution of non-logical words:

  If John is in love, he is happy.

  John is not happy.

  ∴ John is not in love.

Logic and Arguments

- Logically valid arguments are valid in virtue of their form.
- Validity is preserved by substitution of non-logical words:

If the cat is a baseball player, I am Italian.

I am not Italian.

∴ The cat is not a baseball player.

Logic and Arguments

- In a sense, logic is the study of **logical words.**
- In a logic, **logical words** are called **logical constants.**
- Propositional Logic (PL) is a formal system that deals with only one type of logical constants, called truth-functional operators.
- They correspond to the negation of natural languages, and a number of subordinating and coordinating conjunctions like **and**, **or** or **if**.

Logic and Arguments

- By comparing the logical constants of PL to the logical words of natural languages we can:
  - learn more about the properties of these words,
  - try to understand whether "being a logical word" is a property that is relevant to the grammar of natural languages,
  - ask whether languages vary in the way they implement the kind of concepts that are represented as logical constants in PL.

Propositional logic: Syntax

Propositional logic

PL that has only two types of expressions:

- symbols that represent atomic propositions,
- logical operators on these propositions.

A proposition is something that is true or false.

In PL, a proposition is denoted by a formula.

Formulas are atomic when they have no structure.

Propositional logic

The sentence **Hellen read the book** has no structure that we can analyze in PL:

- This sentence is true or false, but it has no proper part that is true or false.

So if we want to translate it as a formula of PL, we translate it as an atomic formula.

In PL, atomic formula are written using letters of the alphabet: p, q, r, s, p', q'...

Propositional logic

Besides atomic formulas, we have logical **connectives** that allow us to build complex formulas from simpler formulas.

Here are some connectives that we can define in PL:

- not: ~
- and: &
- or: ∨
- if … then: →
- if and only if: ↔

Propositional logic

- Lexicon of PL:
  - An infinite set of propositional variables: p, q, r, s, p', p", ...
  - A finite set of connectives: ~, &, ∨, →, ↔
  - Parentheses: (,)

Propositional logic

In addition to the lexicon, we have a set of syntactic rules that define what kind of formulas we can form in PL.

A formula that conforms to the syntax of PL is called a well-formed formula (wff).

Propositional logic

- Syntax of PL:

  1. Any propositional variable is itself a well-formed formula (wff)

  2. If F is a wff, then ~F is a wff

  3. If F and F' are wff, then (F & F'), (F ∨ F'), (F → F'), (F ↔ F') are wffs

  4. Nothing else is a wff

- We can omit parentheses when this does not create ambiguity.

Propositional logic

- Examples of well formed formulas (according to the rules we adopted):

p	~~p
(p & p)	~(p ∨ q)
(~(p ∨ q) & p)	~((p ∨ q) & p)
~(p ↔ (r ∨ s))	(p ↔ ~(r ∨ s))
((p & q) ∨ (s & r))	((((p → q) → ~r) ↔ s) ∨ (t & u))

- Examples formulas that are not well formed (according to our rules):

pqr	(p
p~	∨q
(~p ↔ r ∨ s)	→ ∨ ∧
pq→	→∧pq ∨pq

Translating sentences of English into PL

We will use a set of informal guidelines:

1. identify simple clauses without connectives
2. write a small dictionary (a key) in which each simple clause is associated with an atomic formula of PL
3. replace the simple clauses by their translation as atomic formula of PL
4. translate any connectives into PL
5. represent the order in which the atomic formula combine with connectives using parentheses

Examples of translation

- **If it snows, it is cold.**

  s = it snows

  c = it is cold

  translation: $s \rightarrow c$

Examples of translation

- **It is not sunny, and if it snows, it is cold.**

  u = it is sunny

  s = it snows

  c = it is cold

  Translation: ~u & (s → c)

  ⚠ incorrect translation: (~u & s) → c

  ⚠ incorrect translation: ~(u & (s → c))

A couple of more complex examples

- Alice and Bob are both spies.
- If either Alice or Bob is a spy, then the code has been broken.
- If neither Alice nor Bob is a spy, then the code remains unbroken.
- The German embassy will be in an uproar, unless someone has broken the code.
- Either the code has been broken or it has not, but the German embassy will be in an uproar regardless.

A couple of more complex examples

- **Alice and Bob are both spies.**
  - Key:

    p: Alice is a spy.

    q: Bob is a spy.
  - Translation:

    p & q

A couple of more complex examples

- **If either Alice or Bob is a spy, then the code has been broken.**
  - Key:

    p: Alice is a spy.

    q: Bob is a spy.

    r: The code has been broken.
  - Translation:

    $(p \lor q) \rightarrow r$

A couple of more complex examples

- **If neither Alice nor Bob is a spy, then the code remains unbroken.**
  - Key:

    p: Alice is a spy.

    q: Bob is a spy.

    r: The code has been broken.
  - Equivalent translations:

    $\sim(p \vee q) \rightarrow \sim r$

    $(\sim p \,\&\, \sim q) \rightarrow \sim r$

A couple of more complex examples

- **The German embassy will be in an uproar, unless someone has broken the code.**
    - Key:

      p: The German embassy will be in an uproar.

      q: Someone has broken the code.
    - Equivalent translations:

      $\sim p \rightarrow q$

      $\sim q \rightarrow p$

      $p \vee q$

A couple of more complex examples

- **Either the code has been broken or it has not, but the German embassy will be in an uproar regardless.**
  - Key:

    p: The German embassy will be in an uproar.

    q: The code has been broken.
  - Translation:

    $(q \lor \sim q) \& p$

    This is equivalent to simply: p

Propositional logic: Semantics

Propositional logic: Semantics

We will think about the meaning of formulas of PL truth-conditionally.

Formulas have truth-values: T(rue) or F(alse).

They also have truth-conditions.

In PL, we don't have much to say about the truth-conditions of atomic formulas. We just say whether they are true or false.

However, we can express the truth-value of a complex formula as a function of the truth-values of its atomic parts and the logical structure of the formula.

Propositional logic: Semantics

We express the truth-conditions of complex formulas using truth tables (TT).

A TT is a table that represents all possible ways to assign a truth-value to a well formed formula of PL, which we call **valuations**.

Using TT, we can define the meaning of the connectives of PL.

Negation

Negation reverses truth-values:

p	~ p
T	F
F	T

Conjunction

Conjunction requires the two conjuncts to be true:

p	q	p & q
T	T	T
T	F	F
F	T	F
F	F	F

Conjunction

- In PL, we translate conjunctions of verb phrases and conjunctions of noun phrases as if they were conjunctions of clauses, whenever the two are equivalent:
  - Read **John smokes and drinks** as **John smokes and John drinks**.
  - Read **Susan and Jim smoke** as **Susan smokes and Jim smokes**.

Conjunction

- Here are three connectives of English that we will translate into PL as "&":
  - **and**
  - **but**
  - **although**
- Can you identify aspects of their meaning in English that are lost in translation?

Conjunction

- Lost in translation:
  - Order:
    - **John walked in and sat down.**
  - Concession:
    - **John read the book, but he doesn't remember it.**
  - Concession:
    - **Although John read the book, he doesn't remember it.**

Disjunction

Disjunction (more precisely: inclusive disjunction) requires at least one of the two disjuncts to be true:

p	q	p ∨ q
T	T	T
T	F	T
F	T	T
F	F	F

Disjunction

- In PL, we translate disjunctions of verb phrases and disjunctions of noun phrases as if they were disjunctions of clauses, whenever the two are equivalent:
  - Read **John smokes or drinks** as **John smokes or John drinks.**
  - Read **Susan or Jim smoke** as **Susan smokes or Jim smokes.**

Disjunction

- Which aspects of the meaning of **or** in the following sentence is lost in the translation using inclusive disjunction?
  - **You can have cheese or dessert.**
  - **The book is in the office or in the library.**

Disjunction

- Lost in translation:
    - Exclusion:
        - **You can have cheese or dessert (but not both).**
    - Ignorance:
        - **The book is in the office or in the library (I don't know which).**

Material implication (if … then)

p	q	$p \rightarrow q$
T	T	T
T	F	F
F	T	T
F	F	T

Material implications have an antecedent (to the left of the arrow) and a consequent (to the right).

Material implication is false only when the antecedent is true and the consequent is false.

Material implication (if … then)

- What aspect of the meaning of the conditional is lost in the translation using material implication?
  - **If you mow the lawn, I'll give you five dollars?**
  - **If you heat iron, it turns red.**

Material implication (if … then)

- These conditional sentences tend to be interpreted as **if and only if**:
  - **If you mow the lawn, I'll give you five dollars?**
  - **If you heat iron, it turns red.**

Material implication (if … then)

- Note that the following sentence is odd in English, but fine in PL:
  - **If we are in Toronto, my name is Guillaume.**
- Material implication does not require that the antecedent be relevant to the consequent.

Material implication (if … then)

- It is generally assumed that **if S then S'** and **S only if S'** have the same translation into PL, using material implication.
- Sometimes, this translation works fine:
  - **If all men are mortal, then Aristotle is mortal.**
  - **All men are mortal only if Aristotle is mortal.**
- But the equivalence breaks down in some cases:
  - **If you're boiled in oil, you'll die.**
  - **You'll be boiled in oil only if you die.**

Biconditional (if and only if)

The biconditional requires that the two formulas it relates have the same truth-value:

p	q	$p \leftrightarrow q$
T	T	T
T	F	F
F	T	F
F	F	T

Truth tables of complex formulas

Now that we have defined TT for the connectives of PL, we can make TT for more complex formulas.

Here is an example: **r & (~p → q)**

p q r	~ p	~ p → q	r & (~ p → q)
T T T	F	T	T
T T F	F	T	F
T F T	F	T	T
T F F	F	T	F
F T T	T	T	T
F T F	T	T	F
F F T	T	F	F
F F F	T	F	F

Truth tables of complex formulas

- Rules for building truth tables for a complex formula F:
  - First columns:
    - possible truth-values of the atomic parts of F.
  - Intermediate columns:
    - possible truth-values of non-atomic parts of F.
  - Final column:
    - possible truth-values of F.
- The truth-values of a non-atomic formula are represented below its main connective.

Truth tables of complex formulas

- Each row of a TT corresponds to a **valuation**: a unique way to assign a truth-value to each atomic formula.
- The truth-values of non-atomic formulas is determined by:
    - the assignment of truth-values to their atomic parts.
    - the TT for connectives of PL.

Propositional logic: contradiction

- A contradiction is a formula that is false in every possible valuation.
- Example: **q & ~q**

q	~ q	q & ~ q
T	F	F
F	T	F

Propositional logic: tautology

- A tautology is a formula that is true in every possible valuation.
- Example: **q** ∨ **~q**

q	~ q	q ∨ ~ q
T	F	T
F	T	T

Propositional logic: equivalences

- Two WFF of PL are equivalent just in case they are true in the same valuations.
- Relation between Logical Equivalence between formulas and Biconditional formulas:
  - If a biconditional formula ($p \leftrightarrow r$) is a tautology, then the two formulas p and r are logically equivalent.

    This is written as $p \Leftrightarrow r$.

Propositional logic: equivalences

- We can show that two formulas are equivalent by drawing their truth-tables.
- The following TT for p & q and for q & p show that these formulas are equivalent:

p	q	p & q	q & p
T	T	T	T
T	F	F	F
F	T	F	F
F	F	F	F

Propositional logic: equivalences

- Here are some easy equivalences of PL:

  1. $p \,\&\, p \Leftrightarrow p$
  2. $p \lor p \Leftrightarrow p$
  3. $p \,\&\, q \Leftrightarrow q \,\&\, p$
  4. $p \lor q \Leftrightarrow q \lor p$
  5. $(p \,\&\, q) \,\&\, r \Leftrightarrow p \,\&\, (q \,\&\, r)$
  6. $(p \lor q) \lor r \Leftrightarrow p \lor (q \lor r)$

Propositional logic: equivalences

- Here are some less straightforward equivalences of PL:

  1. p & (q ∨ r) ⇔ (p & q) ∨ (p & r)

  2. p ∨ (q & r) ⇔ (p ∨ q) & (p ∨ r)

  3. ~(p & q) ⇔ ~p ∨ ~q

  4. ~(p ∨ q) ⇔ ~p & ~q

  5. p → q ⇔ ~(p & ~q)

  6. p → q ⇔ ~p ∨ q

  7. p → q ⇔ ~q → ~p

Propositional logic: equivalences

- You can check these equivalences on paper, then check your answers using the following program:
  - https://mrieppel.net/prog/truthtable.html
  - Check how the program works by entering the following symbols: p & (q v r) , (p & q) v (p & r)

Propositional logic: validity

- Let us go back to arguments:
    - showing that an argument is valid: show that every way to make its premises true also make its conclusion true.
    - showing that an argument is not valid: show that there is a way to make its conclusion false while all its premises are true

We can now do this using truth-tables.

Propositional logic: validity

- Show that the following argument is not valid:

  $P \rightarrow Q$

  $\sim P$

  $\therefore \sim Q$

- To do so, we build a truth table where every premise and the conclusion appear as column.

Propositional logic: validity

- Show that the following argument is not valid:

$P \rightarrow Q$

$\sim P$

$\therefore \sim Q$

p	q	p → q	~ p	~ q
T	T	T	F	F
T	F	F	F	T
F	T	T	T	F
F	F	T	T	T

- The second valuation starting from the bottom shows that this argument is not valid.