Logic for AI

Master 1 IFI

Université
Nice
Sophia Antipolis

Andrea G. B. Tettamanzi
Nice Sophia Antipolis University
Computer Science Department
andrea.tettamanzi@unice.fr

Session 1

Propositional Logic

Agenda

- Introduction
- Propositional Logic
  - Syntax
  - Semantics
  - Logical Entailment
  - Canonical Representation
  - Davis-Putnam Algorithm
  - Resolution
  - Formal Systems, Deduction, and Proof

Introduction

- One of the hallmarks of **intelligence** is the ability to reason
- If we want to build **intelligent machines**, we must be able to automate **reasoning**
- Logic is the **study of how we** (should) **reason**
- One of the oldest intellectual disciplines in human history
  - Aristotle (Ἀριστοτέλης, 384–322 BC), a pupil of Plato
  - Gottfried Wilhelm von Leibniz (1646–1716)
  - George Boole (1815–1864)
  - Bertrand Russel (1872–1970)
  - Alan Turing (1912–1954)
  - … and many others!

Introduction

- Logic plays an important role in several areas of CS
  - software engineering (specification and verification)
  - programming languages (semantics, logic programming)
  - **artificial intelligence** (knowledge representation and reasoning).
- Goals of this course
  - Provide general background in Logic
  - Enable access to more advanced topics in CS
  - In particular, (symbolic) artificial intelligence
  - Deal with uncertainty, imprecision, and incompleteness

Contents of the Course

- Part I – Basics
  - Propositional Logic: syntax and semantics
  - First Order Predicate Logic: syntax and semantics
  - Natural Deduction
  - Unification and Resolution

- Part II – Non-Monotonic Logic and Approximate Reasoning
  - Argumentation Theory
  - Belief Revision and Update
  - Fuzzy Logic
  - Possibility Theory

Credits

I’m indebted to many colleagues. In particular:

- Michael Genesereth & Eric Kao (Stanford)
- P. Clemente (ENSI Bourges)

What is Logic?

- Logic is the study of information encoded in the form of logical sentences (or formulas).
- Each sentence *S* divides the set of possible worlds into
  - The set of worlds in which *S* is true (models of *S*)
  - The set of worlds in which *S* is false (counter-models of *S*)
- A set of premises logically entails a conclusion ⇔ the conclusion is true in every world in which all of the premises are true
- A logic consists of
  - A language with a formal syntax and a precise semantics
  - A notion of logical entailment
  - Rules for manipulating expressions in the language.

Why Do We Need “Formal” Logic?

- Why not study Logic using just natural language?
  - Natural language can be ambiguous
    - The boy saw the girl with the telescope
    - British Left Waffles On Falkland Islands
  - Long sentences may be too complex
  - Failing to understand the meaning of a sentence can lead to errors in reasoning
    - Bad sex is better than nothing.
      Nothing is better than good sex.
      Therefore, bad sex is better than good sex”
- These difficulties can be eliminated by using a formal language

Propositional Languages

- A propositional signature is a set of primitive symbols, called propositional constants.
- A propositional constant symbolizes a simple sentence, like
  - "it is raining" → *r*
  - "the tank is empty" → *e*
- Given a propositional signature, a propositional sentence is either
  - a member of the signature or
  - a compound expression formed from members of the signature. (Details to follow.)
- A propositional language is the set of all propositional sentences that can be formed from a propositional signature.

Compound Sentences

- Negations: $\neg$*raining*

  The argument of a negation is called the *target*.
- Conjunctions: (*raining* $\wedge$ *snowing*)

  The arguments of a conjunction are called *conjuncts*.
- Disjunctions: (*raining* $\vee$ *snowing*)

  The arguments of a disjunction are called *disjuncts*.
- Implications: (*raining* $\Rightarrow$ *cloudy*)

  The left argument of an implication is the *antecedent*.

  The right argument is the *consequent*.
- Equivalences: (*cloudy* $\Leftrightarrow$ *raining*)

Propositional Interpretation

- A propositional interpretation is a function mapping every propositional constant in a propositional language to the truth values T or F.

$$\mathcal{I} : \mathrm{Constants} \to \{F, T\}$$

$$\begin{array}{ccc} p & \stackrel{\mathcal{I}}{\mapsto} & T \\ q & \stackrel{\mathcal{I}}{\mapsto} & F \\ r & \stackrel{\mathcal{I}}{\mapsto} & T \end{array} \qquad \begin{array}{ccc} p^{\mathcal{I}} & = & T \\ q^{\mathcal{I}} & = & F \\ r^{\mathcal{I}} & = & T \end{array}$$

- We sometimes view an interpretation as a Boolean vector of values for the items in the signature of the language (when the signature is ordered): *TFT*

Sentential Interpretation

- A sentential interpretation is a function mapping every propositional sentence to the truth values T or F.

$$
\begin{aligned}
p^{\mathcal{I}} &= T & (p \vee q)^{\mathcal{I}} &= T \\
q^{\mathcal{I}} &= F & (\neg q \vee r)^{\mathcal{I}} &= T \\
r^{\mathcal{I}} &= T & ((p \vee q) \wedge (\neg p \vee r))^{\mathcal{I}} &= T
\end{aligned}
$$

- A propositional interpretation defines a sentential interpretation by application of operator semantics.

Operator Semantics

ϕ	$\neg\phi$
F	T
T	F

ϕ	ψ	$\phi \wedge \psi$
F	F	F
F	T	F
T	F	F
T	T	T

ϕ	ψ	$\phi \vee \psi$
F	F	F
F	T	T
T	F	T
T	T	T

ϕ	ψ	$\phi \Rightarrow \psi$
F	F	T
F	T	T
T	F	F
T	T	T

ϕ	ψ	$\phi \Leftrightarrow \psi$
F	F	T
F	T	F
T	F	F
T	T	T

Multiple Interpretations

- Logic does not prescribe which interpretation is "correct". In the absence of additional information, one interpretation is as good as another.
- Examples:
  - Different days of the week
  - Different locations
  - Beliefs of different people
- We may think of each interpretation as a *possible world*
- The set of all interpretations (possible worlds) is

$$\Omega = \{F, T\}^{\mathrm{Constants}} \qquad \|\Omega\| = 2^{\|\mathrm{Constants}\|}$$

Truth Tables

- A truth table is a table of all possible interpretations for the propositional constants in a language (i.e., a representation of Ω).

p	q	r
F	F	F
F	F	T
F	T	F
F	T	T
T	F	F
T	F	T
T	T	F
T	T	T

One row per interpretation

One column per constant

For a language with *n* constants, there are 2^n interpretations

Properties of Sentences

Valid (tautologies)

A sentence is *valid* if and only if *every* interpretation satisfies it.

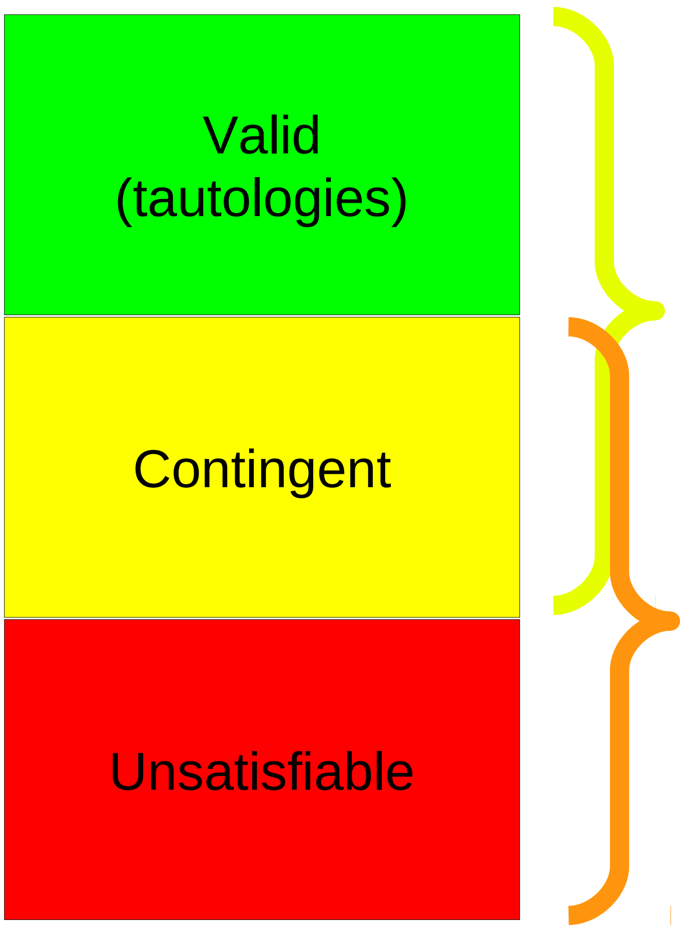
Contingent

A sentence is *contingent* if and only if *some* interpretation satisfies it and *some* interpretation falsifies it.

Unsatisfiable

A sentence is *unsatisfiable* if and only if *no* interpretation satisfies it.

Properties of Sentences



A sentence is *satisfiable* if and only if it is either valid or contingent.

A sentence is *falsifiable* if and only if it is either contingent or unsatisfiable.

Example of Tautology

p	q	r	$p \Rightarrow q$	$q \Rightarrow r$	$(p \Rightarrow q) \vee (q \Rightarrow r)$
F	F	F			
F	F	T			
F	T	F			
F	T	T			
T	F	F			
T	F	T			
T	T	F			
T	T	T			

Example of Tautology

p	q	r	$p \Rightarrow q$	$q \Rightarrow r$	$(p \Rightarrow q) \vee (q \Rightarrow r)$
F	F	F	T	T	
F	F	T	T	T	
F	T	F	T	F	
F	T	T	T	T	
T	F	F	F	T	
T	F	T	F	T	
T	T	F	T	F	
T	T	T	T	T	

Example of Tautology

p	q	r	$p \Rightarrow q$	$q \Rightarrow r$	$(p \Rightarrow q) \vee (q \Rightarrow r)$
F	F	F	T	T	T
F	F	T	T	T	T
F	T	F	T	F	T
F	T	T	T	T	T
T	F	F	F	T	T
T	F	T	F	T	T
T	T	F	T	F	T
T	T	T	T	T	T

More Valid Sentences (Tautologies)

Double Negation: $p \Leftrightarrow \neg\neg p$

deMorgan's Laws:

$$\neg(p \wedge q) \quad \Leftrightarrow \quad \neg p \vee \neg q$$
$$\neg(p \vee q) \quad \Leftrightarrow \quad \neg p \wedge \neg q$$

Implication Introduction: $p \Rightarrow (q \Rightarrow p)$

Implication Distribution:

$$(p \Rightarrow (q \Rightarrow r)) \Rightarrow ((p \Rightarrow q) \Rightarrow (p \Rightarrow r))$$

Axiomatizability

- A set of boolean vectors of length *n* is *axiomatizable* in propositional logic if and only if there is a signature of size *n* and a set of sentences from the corresponding language such that the vectors in the set correspond to the set of interpretations satisfying the sentences.
- A set of sentences defining a set of vectors is called the *axiomatization* of the set of vectors.
- Example:
  - Set of Boolean Vectors: { TFF, FTF, FTT }
  - Signature: $\{p, q, r\}$
  - Axiomatization: $(p \wedge \neg q \wedge \neg r) \vee (\neg p \wedge q)$

Logical Entailment

- A set of premises Δ logically entails a conclusion φ, written

$$\Delta \models \phi$$

  if and only if every interpretation that satisfies the premises also satisfies the conclusion.
- Examples:

$$\{p\} \models p \vee q$$

$$\{p\} \not\models p \wedge q$$

$$\{p, q\} \models p \wedge q$$

Logical Entailment ≠ Logical Equivalence!

Truth Table Method

- Method for computing whether a set of premises logically entails a conclusion
  1) Form a truth table for the propositional constants occurring in the premises and conclusion; add a column for the premises and a column for the conclusion
  2) Evaluate the premises for each row in the table
  3) Evaluate the conclusion for each row in the table
  4) If every row that satisfies the premises also satisfies the conclusion, then the premises logically entail the conclusion

Logical Entailment and Satisfiability

- **Unsatisfiability Theorem**: $\Delta \models \phi$ if and only if $\Delta \cup \{\neg\phi\}$ is unsatisfiable.
- Proof:
  - $[\Rightarrow]$: Suppose that $\Delta \models \phi$. If an interpretation satisfies Δ, then it must also satisfy ϕ. But then it cannot satisfy . Therefore, $\Delta \cup \{\neg\phi\}$ is unsatisfiable.
  - $[\Leftarrow]$: Suppose that $\Delta \cup \{\neg\phi\}$ is unsatisfiable. Then every interpretation that satisfies Δ must fail to satisfy $\neg\phi$, i.e., it must satisfy ϕ. Therefore, $\Delta \models \phi$.
- Corollary: we can determine logical entailment by determining satisfiability (proof by refutation).

Satisfaction

- Method to find all propositional interpretations that satisfy a given set of sentences:
    1) Form a truth table for the propositional constants.
    2) For each sentence in the set and each row in the truth table, check whether the row satisfies the sentence. If not, cross out the row.
    3) Any row remaining satisfies all sentences in the set. (Note that there might be more than one.)

Canonical Representation

- Syntactically distinct sentences can be equivalent (i.e., semantically identical)
- Sometimes, that can be impractical
- Idea: why don’t we reduce all sentences to a canonical form, so that checking them for equivalence becomes trivial?
- Conjunctive and Disjunctive Normal Form (resp. CNF and DNF)

Conjunctive Normal Form (CNF)

- A literal is a positive or negated constant, like p or $\neg$p
- A clause is the disjunction of a finite number of literals, i.e., a sentence of the form

$$(l_1 \vee l_2 \vee \ldots \vee l_n)$$

- A clause is valid if and only if it contains a pair of opposed literals, like p and $\neg$p.
- The empty clause F is the only unsatisfiable clause.
- A CNF is the conjunction of a finite number of clauses, i.e., a sentence of the form

$$(c_1 \wedge c_2 \wedge \ldots \wedge c_n)$$

Conjunctive Normal Form

- **Theorem**: for every propositional sentence, there exists an equivalent CNF
- Proof: we give an algorithm to transform any sentence into CNF

1) Eliminate the ⇔ and ⇒ operators:

$$(\phi \Leftrightarrow \psi) \rightarrow (\phi \Rightarrow \psi) \wedge (\psi \Rightarrow \phi) \qquad (\phi \Rightarrow \psi) \rightarrow (\neg\phi \vee \psi)$$

2) Apply as many times as possible the following rewrite rules:

$$\neg(\phi \vee \psi) \rightarrow (\neg\phi \wedge \neg\psi)$$
$$\neg(\phi \wedge \psi) \rightarrow (\neg\phi \vee \neg\psi)$$
$$\neg\neg\phi \rightarrow \phi$$

3) Apply as many times as possible the following rewrite rules:

$$\phi \vee (\psi \wedge \xi) \rightarrow (\phi \vee \psi) \wedge (\phi \vee \xi)$$
$$(\phi \wedge \psi) \vee \xi \rightarrow (\phi \vee \xi) \wedge (\psi \vee \xi)$$

The resulting CNF is equivalent to the initial sentence.

Conjunctive Normal Form

- A few details complete the algorithm of the previous slide:
  - Valid clauses can be deleted as soon as they appear
  - Repeated literals in the same clause can be simplified
  - If a clause *c* is included in another clause *c'* (*c* subsumes *c'*), then clause *c'* can be deleted
  - A CNF including an empty clause can be reduced to just the empty clause F.

- The CNF thus obtained is said to be “pure”.
- The algorithm always terminates after a finite number of steps and returns a CNF

Davis-Putnam Algorithm

- DP(S : pure CNF) : Boolean // Test whether S is satisfiable
  1) If S = Ø, then return V; If S = {F}, then return F; Otherwise
  2) Select a propositional constant *p* in S, giving priority to those such that (a) *p* or $\neg p$ occurs alone in a clause or (b) only *p* or $\neg p$ occurs in S
  3) Let S_p be the set of clause containing *p*, $S_{\neg p}$ those not containing *p*, and S'' the remaining clauses
  4) $S'_p \leftarrow S_p$ where *p* is set to F (thus, deleted from each clause)
  5) $S'_{\neg p} \leftarrow S_{\neg p}$ where *p* is set to T (thus $\neg p$ is deleted)
  6) Return $DP(S'_p \cup S'') \vee DP(S'_{\neg p} \cup S'')$.

Deduction (Proofs)

- Deduction:
  - Symbolic manipulation of sentences, rather than enumeration of interpretations (= truth assignments)
- Benefits:
  - Usually smaller than truth tables
  - Can be often found with less work

Resolution Principle

$$(l_1 \vee l_2 \vee \ldots \vee l_n \vee p)$$

$$\wedge$$

$$(l_1 \vee l_2 \vee \ldots \vee l_n \vee \neg p)$$

$$(l_1 \vee l_2 \vee \ldots \vee l_n)$$

Resolvent clause

Clausal Resolution

- To check whether a CNF S is satisfiable:
  1) Find two clauses in S, one containing literal *l* and the other containing ¬*l*, such that they have not yet been used together (if they cannot be found, terminate with result: “satisfiable”)
  2) Compute their resolvent (if it is the empty clause F, terminate with result: “unsatisfiable”)
  3) Add the resolvent to S
  4) Go back to Step 1.

- We can use resulution to construct proofs by refutation: to prove that S |= ϕ, prove that S$\cup$ $\{\neg\phi\}$ is unsatisfiable.

Example

$$S = \{p \overset{1}{\vee} q, p \overset{2}{\vee} r, \neg q \overset{3}{\vee} \neg r, \overset{4}{\neg} p\}$$

#	clause	from
5	$p \vee \neg r$	(1, 3)
6	q	(1, 4)
7	$p \vee \neg q$	(2, 3)
8	r	(2, 4)
9	p	(2, 5)
10	$\neg r$	(3, 6)
11	$\neg q$	(3, 8)
12	$\neg r$	(4, 5)
13	$\neg q$	(4, 7)
14	F	(4, 9)

Thank you for your attention