Propositional Logic

Propositional logic is a subset of the predicate logic.

- Syntax
- Semantics
- Models
- Inference Rules
- Complexity

9

Syntax of Propositional Logic

- symbols
  - logical constants `True, False`
  - propositional symbols `P, Q,` …
  - logical connectives
    - conjunction $\wedge$, disjunction $\vee$,
    - negation $\neg$,
    - implication $\Rightarrow$, equivalence $\Leftrightarrow$
  - parentheses (,)
- sentences
  - constructed from simple sentences
  - conjunction, disjunction, implication, equivalence, negation

10

BNF Grammar Propositional Logic

Sentence → *AtomicSentence* | *ComplexSentence*

AtomicSentence → `True` | `False` | `P` | `Q` | `R` | `...`

ComplexSentence → (*Sentence*)

| *Sentence Connective Sentence*

| ¬ *Sentence*

Connective → ∧ | ∨ | ⇒ | ⇔

ambiguities are resolved through precedence ¬ ∧ ∨ ⇒ ⇔
or parentheses

e.g. ¬ *P* ∨ *Q* ∧ *R* ⇒ *S* is equivalent to (¬ *P)* ∨ *(Q* ∧ *R))* ⇒ *S*

11

Semantics of Propositional Logic

- interpretation of the propositional symbols and constants
  - symbols can be any arbitrary fact
    - sentences consisting of only a propositional symbols are satisfiable, but not valid
  - the constants `True` and `False` have a fixed interpretation
    - `True` indicates that the world is as stated
    - `False` indicates that the world is not as stated
- specification of the logical connectives
  - explicitly via truth tables

12

- propositions can be interpreted as any facts you want
  - e.g., P means "robins are birds", Q means "the Mike is dead", etc.
- meaning of complex sentences is derived from the meaning of its parts
  - one method is to use a truth table
  - all are easy except P => Q
    - this says that if P is true, then I claim that Q is true; otherwise I make no claim;
    - P is true and Q is true, then P => Q is true
    - P is true and Q is false, then P => Q is false
    - P is false and Q is true, then P => Q is true
    - P is false and Q is false, then P => Q is true

13

Validity and Satisfiability in Propositional Logic

- a sentence is *valid* or necessarily true if and only if it is true under all possible interpretations in all possible worlds
  - also called a *tautology*
  - since computers reason mostly at the syntactic level, valid sentences are very important
- a sentence is *satisfiable* iff there is some interpretation in some world for which it is true
- a sentence that is not satisfiable is *unsatisfiable*
  - also known as a *contradiction*

14

Truth Tables for Connectives in Propositional Logic

P	Q	$\neg P$	$P \wedge Q$	$P \vee Q$	$P \Rightarrow Q$	$P \Leftrightarrow Q$
False	*False*	*True*	*False*	*False*	*True*	*True*
False	*True*	*True*	*False*	*True*	*True*	*False*
True	*False*	*False*	*False*	*True*	*False*	*False*
True	*True*	*False*	*True*	*True*	*True*	*True*

15

Propositional Calculus

- properly formed statements that are either True or False
- syntax
  - logical constants, True and False
  - proposition symbols such as P and Q
  - logical connectives: and ^, or V, equivalence <=>, implies => and not ~
  - parentheses to indicate complex sentences
- sentences in this language are created through application of the following rules
  - True and False are each (atomic) sentences
  - Propositional symbols such as P or Q are each (atomic) sentences
  - Enclosing symbols and connective in parentheses yields (complex) sentences, e.g., (P ^ Q)

16

Complex Sentences in Propositional Calculus

- Combining simpler sentences with logical connectives yields complex sentences
  - conjunction
    - sentence whose main connective is and: P ^ (Q V R)
  - disjunction
    - sentence whose main connective is or: A V (P ^ Q)
  - implication (conditional)
    - sentence such as (P ^ Q) => R
    - the left hand side is called the premise or antecedent
    - the right hand side is called the conclusion or consequent
    - implications are also known as rules or if-then statements
  - equivalence (biconditional)
    - (P ^ Q) <=> (Q ^ P)
  - negation
    - the only unary connective (operates only on one sentence)
    - e.g., ~P

17

Inference Rules

They are more efficient than truth tables.

- Modus ponens
- Modus tollens
- Syllogism
- Resolution
- And-elimination
- And-introduction
- Or-introduction
- Double-negation elimination
- Unit resolution

18

Modus ponens

- eliminates =>

```
(X => Y),    X
_____________
      Y
```

  - If it rains, then the streets will be wet.
  - It is raining.
  - Infer the conclusion: The streets will be wet. (affirms the antecedent)

19

Modus tollens

(X => Y), ¬Y

_____

¬ X

- If it rains, then the streets will be wet.
- The streets are not wet.
- Infer the conclusion: It is not raining.

- NOTE: Avoid the fallacy of affirming the consequent:
  - If it rains, then the streets will be wet.
  - The streets are wet.
  - *cannot* conclude that it is raining.
- If Bacon wrote Hamlet, then Bacon was a great writer.
  - Bacon was a great writer.
  - cannot conclude that Bacon wrote Hamlet.

20

Syllogism

- chain implications to deduce a conclusion

  (X => Y), (Y => Z)

  _____

  (X => Z)

21

Resolution

(X v Y), (~Y v Z)

_____

(X v Z)

- basis for the inference mechanism in the Prolog language and some theorem provers

22

Complexity issues

- truth table enumerates 2^n rows of the table for any proof involving n symbol
  - it is complete
  - computation time is exponential in n
- checking a set of sentences for satisfiability is NP-complete
  - but there are some circumstances where the proof only involves a small subset of the KB, so can do some of the work in polynomial time
  - if a KB is monotonic (i.e., even if we add new sentences to a KB, all the sentences entailed by the original KB are still entailed by the new larger KB), then you can apply an inference rule locally (i.e., don't have to go checking the entire KB)

23

Inference Methods 1

- deduction sound
  - conclusions must follow from their premises; prototype of logical reasoning
- induction unsound
  - inference from specific cases (examples) to the general
- abduction unsound
  - reasoning from a true conclusion to premises that may have caused the conclusion
- resolution sound
  - find two clauses with complementary literals, and combine them
- generate and test unsound
  - a tentative solution is generated and tested for validity
  - often used for efficiency (trial and error)

24

Inference Methods 2

- default reasoning unsound
  - general or common knowledge is assumed in the absence of specific knowledge
- analogy unsound
  - a conclusion is drawn based on similarities to another situation
- heuristics unsound
  - rules of thumb based on experience
- intuition unsound
  - typically human reasoning method (no proven theory)
- nonmonotonic reasoning unsound
  - new evidence may invalidate previous knowledge
- autoepistemic unsound
  - reasoning about your own knowledge (self knowledge)

25

Predicate Logic

- Additional concepts (in addition to propositional logic)
  - complex objects
    - terms
  - relations
    - predicates
    - quantifiers
  - syntax
  - semantics
  - inference rules
  - usage

26

Objects in Predicate Logic

- distinguishable things in the real world
  - people, cars, computers, programs, ...
- frequently includes concepts
  - colors, stories, light, money, love, ...
- properties
  - describe specific aspects of objects
    - green, round, heavy, visible,
  - can be used to distinguish between objects

27

Relations in Predicate Logic

- establish connections between objects
- relations can be defined by the designer or user
  - neighbor, successor, next to, taller than, younger than, …
- functions are special types of relations
  - non-ambiguous: only one output for a given input

28

Syntax of Predicate Logic

- also based on sentences, but more complex
  - sentences can contain terms, which represent objects
- constant symbols: A, B, C, Square
  - stand for unique objects (in a specific context)
- predicate symbols: Adjacent-to, Younger-than, ...
  - describes relations between objects
- function symbols: SQRT, Cosine, …
  - the given object is related to exactly one other object

29

Semantics of Predicate Logic

- provided by interpretations for the basic constructs
  - usually suggested by meaningful names
- constants
  - the interpretation identifies the object in the real world
- predicate symbols
  - the interpretation specifies the particular relation in a model
  - may be explicitly defined through the set of tuples of objects that satisfy the relation
- function symbols
  - identifies the object referred to by a tuple of objects
  - may be defined implicitly through other functions, or explicitly through tables

30

BNF Grammar Predicate Logic

Sentence	→ *AtomicSentence*			
		| *Sentence Connective Sentence*		
		| *Quantifier Variable, ... Sentence*		
		| ¬ *Sentence*	| (*Sentence*)	
AtomicSentence	→ *Predicate(Term, …)*	| *Term = Term*		
Term	→ *Function(Term, …)*	| *Constant*	| *Variable*	
Connective	→ ∧ | ∨ | ⇒ | ⇔			
Quantifier	→ ∀ | ∃			
Constant	→ *A, B, C, X_1 , X_2, Jim, Jack*			
Variable	→ *a, b, c, x_1 , x_2, counter, position*			
Predicate	→ *Adjacent-To, Younger-Than,*			
Function	→ *Sqrt, Cosine*			

Ambiguities are resolved through precedence or parentheses.

31

Terms in Predicate Logic

- logical expressions that specify objects
- constants and variables are terms
- more complex terms are constructed from function symbols and simpler terms, enclosed in parentheses
  - basically a complicated name of an object
- semantics is constructed from the basic components, and the definition of the functions involved
  - either through explicit descriptions (e.g. table), or via other functions

32

Unification in Predicate Logic

- The process of finding substitution for variables to make arguments match is called unification.
  - a substitution is the simultaneous replacement of variable instances by terms, providing a "binding" for the variable
  - without unification, the matching between rules would be restricted to constants
  - ~F(y) V H(a,y), F(b) (b can be substitute for y)
  - unification itself is a very powerful and possibly complex operation
    - in many practical implementations, restrictions are imposed

33

Universal Quantification in Predicate Logic

- states that a predicate P is holds for all objects x in the universe under discourse
  $\forall x\ P(x)$
- the sentence is true if and only if all the individual sentences where the variable x is replaced by the individual objects it can stand for are true

34

Existential Quantification in Predicate Logic

- states that a predicate P holds for some objects in the universe
  $\exists x \; P(x)$
- the sentence is true if and only if there is at least one true individual sentence where the variable *x* is replaced by the individual objects it can stand for

35

Horn clauses or sentences in Predicate Logic

- class of sentences for which a polynomial-time inference procedure exists
  - $P_1 \wedge P_2 \wedge ... \wedge P_n \Rightarrow Q$
    where P_i and Q are non-negated atomic sentences
- not every knowledge base can be written as a collection of Horn sentences
- Horn clauses are essentially rules of the form
  - If $P_1 \wedge P_2 \wedge ... \wedge P_n$ then Q

36