Artificial Intelligence

Knowledge Representation II

Lecture 7

(22 September, 1999)

Tralvex (Rex) Yeap
University of Leeds

Content: Knowledge Representation II

- Quick Review on Lecture 6
- Alt.Revision on L6
- Predicate Calculus
- Predicate Calculus Syntax
- First Order Predicate Calculus (FOPC)
- FOPC Equivalence
- Representation of English Language
- FOPC Inference Rules
- Some Useful Inference Rules
- Unification
- Substitution
- Most General Unifier
- Summary
- Students' Mini Research Presentation by Group C
- What's in Store for Lecture 8

Quick Review on Lecture 6

- Issues in Knowledge Representation
- Why logic in knowledge representation?
- Logical Symbols and Quantifiers
- How to use logical notation for representing knowledge
- A Set-Theoritical definition of L_{PC}
- Class Activity 1: Workouts on L_{PC}
- Ways of reasoning in L_{PC}
- Natural Deduction
- Semantic Tableau
- Truth Tables
- Relationships between (1) Natural Deduction, (2) Semantic Tableau, and (3) Truth Table (Model Theory) Methods

Introduction

- A proposition is a statement about the world that may be either true or false.
- Examples of propositions (“properly formed statements”):

  Bob’s car is blue.

  Seven plus six equals twelve.

  John is mary’s uncle.
- Each of the sentences is a proposition - not to be broken down into its constituent parts. i. e., we simply assign true, say, to “John is mary’s uncle” with no regard for what “uncle” means
- Examples of non- propositions:

  Mary’s uncle

  Seven plus four

  cannot assign truth value to them.

Introduction (cont.)

- Propositions are denoted by propositional symbols such as: P, Q, R, S,.
- Truth symbols are**: true (or T), false (or F).**
- Single propositions by themselves are not very interesting.
- We need to express complex propositions:

  The book is on the table or it is on the chair.

  If Socrates is a man then he is mortal.
- We can use connectives such as:

And	$\wedge$	conjunction
Or	$\vee$	disjunction
Not	$\sim$	negation
Implies	$\rightarrow$	implication
equivalent	$=$	equivalence

- Sentences in the propositional calculus are formed from these atomic symbols according to the syntax rules.

Alt.Revision on L6

Propositional Calculus Sentences (Syntax)

- Every propositional symbol and truth symbol is a sentence.

  e. g., true, P, R.

- The negation of a sentence is a sentence.

  e. g., ~P, ~false

- The conjunction of two sentences is a sentence.

  e. g., P Q, P $\wedge$ Q

- The disjunction of two sentences is a sentence.

  e. g., Q $\vee$ R

- The implication of one sentence for another is a sentence.

  e. g., P $\rightarrow$ Q

- The equivalence of two sentences is a sentence

Well-formed Formulae

- Legal sentences are also called *well- formed formulae* or WFFs

$P \wedge Q$, $P \vee Q$, $P \rightarrow Q$

$P \wedge Q$ → Conjuncts

$P \vee Q$ → Disjuncts

$P \rightarrow Q$ → premise, antecedent; conclusion, consequent

- (,), [,] can be used to group symbols into sub-expressions
- $(P \vee Q) = R$ is not the same as $P \vee (Q = R)$
- A WFF : $(P \rightarrow (Q \wedge R)) = \sim P \vee \sim Q \vee R$

Alt.Revision on L6

Propositional Calculus Semantics

- Given the truth values of propositions, what are the truth values of compound expressions formed from them?

P	**Q**	**P $\wedge$ Q**	**P $\vee$ Q**	**P $\rightarrow$ Q**	**~P**	**P = Q**
T	**T**	T	T	T	F	T
T	**F**	F	T	F	F	F
F	**T**	F	T	T	T	F
F	**F**	F	F	T	T	T

- The above table is also known as the truth table.

Alt.Revision on L6
WFF Equivalence

- $\sim(\sim P) = P$
- $(P \vee Q) = (\sim P \rightarrow Q)$ [or $(\sim P \vee Q) = (P \rightarrow Q)$]

P	Q	$\sim P$	$\sim P \vee Q$	$P \rightarrow Q$	$(\sim P \vee Q) = (P \rightarrow Q)$
T	T	T	T	T	T
T	F	F	T	F	F
F	T	F	T	T	F
F	F	F	F	T	T

- De Morgan’s laws:

  $\sim(P \vee Q) = (\sim P \wedge \sim Q)$

  $\sim(P \wedge Q) = (\sim P \vee \sim Q)$

- Distributive laws:

  $P \vee (Q \wedge R) = (P \vee Q) \wedge (P \vee R)$

  $P \wedge (Q \vee R) = (P \wedge Q) \vee (P \wedge R)$

Alt.Revision on L6

WFF Equivalence

- Commutative laws:

  $(P \wedge Q) = (Q \wedge P)$

  $(P \vee Q) = (Q \vee P)$

- Associative Laws:

  $((P \wedge Q) \wedge R) = (P \wedge (Q \wedge R))$

  $((P \vee Q) \vee R) = (P \vee (Q \vee R))$

- Contrapositive laws:

  $(P \rightarrow Q) = (\sim Q \rightarrow \sim P)$

- These identities can be used to change an expression into a syntactically different but logically equivalent form.
- Can be used to prove the equivalence of two expressions (instead of using truth table).

Alt.Revision on L6

WFF Example

- $(P \to (Q \wedge R)) = ((P \to Q) \wedge (P \to R))$

  is a *tautology* – always true no matter what propositions are substituted for P, Q, and R.

  It states:

  Saying “P implies Q and R” is the same as saying “P implies Q and P implies R”.

- Proof:

  $$\begin{aligned} LHS &= (P \to (Q \wedge R)) \\ &= {\sim}P \vee (Q \wedge R) \\ &= ({\sim}P \vee Q) \wedge ({\sim}P \vee R) \text{ (distributive law)} \\ &= ((P \to Q) \wedge (P \to R)) \\ &= RHS \end{aligned}$$

Alt.Revision on L6

Rules of Inference

- Allow the deduction of new sentences from previously given sentences.

  e. g., If we know that “John is an uncle” is true and
  that “If John is an uncle then John is male ” is true.
  Then we can conclude that “John is male” is true.

  Let P = John is an uncle
  Q = John is male
  Hence if P is true and $P \rightarrow Q$ is true
  then Q is true.

- This is known as the **modus ponens** rule, or the “$\rightarrow$ ” elimination rule.

Rules of Inference (cont.)

- Propositional logic is limited. For example if we represent the propositions

  P: Socrates is a man

  Q: Plato is a man,

  there is nothing common between the symbols P and Q that captures the fact that both of them are men. We need something that can represent the ***constituents*** of the sentence.

- If we are able to represent them as

  man(Socrates)

  man(Plato)

  that will show that Socrates and Plato share some common properties.

- Also, we would like to be able to represent something like

  All men are mortal

  so that the following inference can be made:

  All men are mortal

  Socrates is a man

  Socrates is mortal

Predicate Calculus

- Retain connectives such as $\sim \quad \wedge \quad \vee \quad \rightarrow \quad =$
- Instead of looking at sentences that are of interest merely for their truth values, predicate calculus is used to represent statements about specific *objects* or *individuals* .
- Examples of individuals:

  you, this page of lecture, the number 1, Socrates
- A predicate is that which says something about the subject.

  e. g., <u>The book</u> <u>is red</u>.

  subject   color of the book

  represented as:

  is-red(book) or simply red(book)

  is-red: predicate

  book: argument
- · A predicate statement takes the value **true** or **false** .

Predicate Calculus (cont.)

- red(book) is true if the book is red, false if it is not, then ~red(book) becomes false.
- Predicate with one argument is called a 1- place predicate.
- A predicate can have more than 1 argument:

  e. g., color(book, red)

  mother(john, mary)

  greater- than(7, 4)

  transfer($1000, chartered- bank, ocbc)
- The number of arguments of a predicate is called its arity.
- An atomic sentence (or, atomic expression, or atom) is a predicate of arity *n* followed by *n* terms enclosed in parenthesis separated by commas.
- book, red, john, mary, 7, 4 … are constants.
- We need variables and quantifiers to express sentences such as

  "Everyone likes ice cream"

  "Peter has some friends"

Predicate Calculus (cont.)

- ∀ for all, for every (universal quantifier)
- ∃ there exists (some) (existential quantifier)

  Examples:

  $\forall X$ likes(X, ice_ cream)

  $\exists Y$ friends(Y, Peter)

- The quantifier specifies the extension of the variable (the total number of objects it applies, or the range of values it can take).
- In addition, we also allow *functions*

  -- have a fixed number of arguments (arity)

  -- return (or evaluate to) objects instead of truth values.

  e. g., uncle- of(mary) = john

  plus(4, 3) = 7

- Arguments can be constants, variables, or functions.

  e. g., father- of(father- of(john))

- Sometime we use something called a *term* , which is either a constant, variable, or function expression.

Predicate Calculus Syntax

- Every atomic sentence is a sentence.
- If s is a sentence, so is ~s.
- If s_1 and s_2 are sentences, so is $s_1 \wedge s_2$;
- …. so is $s_1 \vee s_2$;
- …. so is $s_1 \rightarrow s_2$;
- …. so is $s_1 = s_2$;
- If X is a variable and *s* a sentence, then $\forall$X s is a sentence.
- …. then $\exists$X s is a sentence.
- For example:

  $\forall X\ \forall Y$ father(X, Y) $\vee$ mother(X, Y) $\rightarrow$ parent(X, Y)

  is a well- formed predicate calculus sentence.

First Order Predicate Calculus (FOPC)

- First-order predicate calculus permits quantification over individuals but not over predicates and functions.

  e. g., the statement:

  "All predicates have only one argument"

  cannot be expressed in first- order predicate calculus:

  $\forall$p arity(p(X), 1) – not a WFF

  This needs higher- order predicate calculi.

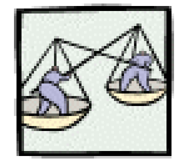
First Order Predicate Calculus (cont.)

Quantifier Scope

The scope of a quantifier is that part of the string of formulae to which the quantifier applies.

e. g., $\underbrace{\exists X\, [p(X) \vee q(X)]}_{\text{scope of } \exists X} \wedge \underbrace{\forall Y\, [p(Y) \vee r(Y)]}_{\text{scope of } \forall Y}$

FOPC Equivalence



- $\sim\exists X\ p(X) = \forall X\ \sim p(X)$
- $\sim\forall X\ p(X) = \exists X\ \sim p(X)$
- $\forall X\ p(X) = \forall Y\ p(Y)$
- $\exists X\ p(X) = \exists Y\ p(Y)$
- $\forall X\ (p(X) \wedge q(X)) = \forall X\ p(X) \wedge \forall Y\ q(Y)$
- $\exists X\ (p(X) \vee q(X)) = \exists X\ p(X) \vee \exists Y\ q(Y)$

Examples:

$\forall X[red(X) \vee green(X)]$

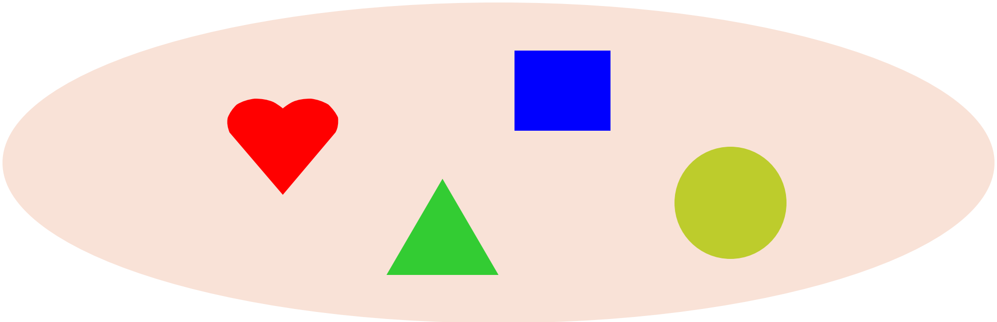
$\forall X\ red(X) \vee \forall Y\ green(Y)$

$\forall X[red(X) \vee square(X)] = \forall X\ red(X) \vee \forall Y\ square(Y)$

$\exists X[red(X) \vee square(X)] = \exists X\ red(X) \vee \exists Y\ square(Y)$

$\forall X[red(X) \vee square(X)] \neq \forall\ X\ red(X) \vee \exists Y\ square(Y)$

FOPC Equivalence (cont.)



$\forall X[\text{red}(X) \vee \text{square}(X)] \neq \forall X\ \text{red}(X) \vee \exists Y\ \text{square}(Y)$

LHS = T

RHS = $\forall X\ \text{red}(X) \vee \exists Y\ \text{square}(Y)$

$= F \vee F$

$= F$

Representation of English Language

- Almost any English sentence may be represented in first- order predicate calculus.
- There is no unique mapping of sentence into predicate calculus expressions.

Example:

(1) John's mother is married to John's father

$$married(father(john), mother(john))$$

(2) Chicken is good and chicken is a kind of food,

$$good(chicken) \wedge is(chicken, food)$$

$$good(chicken) \wedge food(chicken)$$

(3) All food are edible

$$\forall X\ food(X) \rightarrow edible(X)$$

Representation of English Language (cont.)

- John lives in a yellow house

  lives(john, house- 1) $\wedge$ color(house- 1, yellow)

  $\exists$X house(X) $\wedge$ color(X, yellow) $\wedge$ lives(john, X)

- If the car belongs to John, then it is green.

  owns(john, car-1) $\rightarrow$ color(car-1, green)

  $\forall$X car(X) $\wedge$ owns(john, X) $\rightarrow$ color(X, green)

- John plays the piano or the violin

  plays(john, piano) $\vee$ plays(john, violin)

- Some people like snakes.

  $\exists$X (person(X) $\wedge$ likes(X, snakes))

  $\exists$X (person(X) $\wedge$ $\forall$Y (snake(Y) $\rightarrow$ likes(X, Y)))

- All students take exams

  $\forall$X student(X) $\rightarrow$ take-exam(X)

  $\forall$X student(X) $\rightarrow$ ($\exists$Y exam(Y) $\wedge$ take (X, Y))

Representation of English Language (cont.)

- All exams are difficult

  $\forall X$ exam(X) $\rightarrow$ difficult(X)

- Winston did not write Hamlet.

  ~write(winston, hamlet)

- Nobody wrote Hamlet.

  $\sim\exists X$ write(X, hamlet)

- Every city has a dogcatcher who has been bitten by every dog in town

  $\forall X$ {city(X) $\rightarrow\exists Y${dogcatcher(Y, Z) $\wedge$ $\forall Z${[dog(Z) $\wedge$ live- in(Z, X)] $\rightarrow$ bit(Z, Y)}}}

Try X=NY, Y=Dan, Z=Fido

FOPC Inference Rules

- To help infer new correct expressions from a set of true assertions.

  e. g., $\forall X$ human(X) $\rightarrow$ mortal(X) [all humans are mortal]

  human(socrates) [Socrates is a human]

  It should logically follow that:

  mortal(Socrates) [Socrates is mortal]

- **Satisfy:**

  An interpretation that makes a sentence true is said to *satisfy* that sentence.

- **Logically follows:**

  An expression X *logically follows* from a set of predicate expressions S if every interpretation that satisfies S also satisfies X.

FOPC Inference Rules (cont.)

- **Inference rules produce new correct sentences** based on syntactic form of given logical assertions:

  E. g., modus ponens

  if $P \rightarrow Q$

  and Q

  then P is true

- **Soundness**

  When every sentence X produced by an inference rule operating on a set of S of logical expressions logically follows from S, the inference rule is said to be sound. (e. g., modus ponens is sound).

FOPC Inference Rules (cont.)

- Sometimes, in heuristic and common sense reasoning, we use unsound rules of inference.

    E. g., abduction :

    If P $\rightarrow$ Q

    and we observe Q then conclude P

    If a student is sick , he will not attend lecture

    P Q

    sick(student) $\rightarrow$ not_ attend_ lecture(student)

    Observe not_ attend_ lecture(student)

    Conclude sick(student)

- **Completeness**

    If an inference rule is able to produce every erxpression that logically follows from S (a set of logical expressions), then it is said to be complete .

Some Useful Inference Rules

- Modus ponens

  If P is true and $P \longrightarrow Q$ is true

  then Q is true

- Modus tolens

  if $P \rightarrow Q$ is true and Q is false or ~Q is true

  then ~P is true

  e. g., sick(student) $\rightarrow$ not_ attend_ lecture(student)

  ~not_ attend_ lecture(student)

  produces: ~sick(student)

- Elimination

  if $P \wedge Q$ is true

  then P is true and Q is true

Some Useful Inference Rules (cont.)

- Introduction

  if P is true and Q is true

  then $P \wedge Q$ is true

- Universal instantiation / $\forall$-elimination

  For any WFF Φ that mentions a variable X,

  if we have

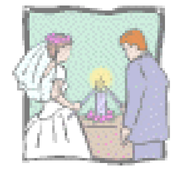
  $\forall X\ \Phi(X)$

  we can conclude $\Phi(a)$ for any a from the domain of X.

  e. g., $\forall X$ human(X) $\rightarrow$ mortal(X)

  then human(socrates) $\rightarrow$ mortal(socrates)

Unification



- In propositional calculus, two expressions match or are the same only if they are syntactically identical:

  e. g., $P \wedge (Q \vee R) \rightarrow S = P \wedge (Q \vee R) \rightarrow S$
  $\neq P \wedge (S \wedge Q) \rightarrow R$

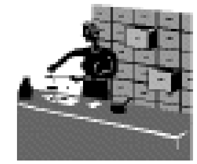
- variables in predicate calculus complicates matters.

  human(X) = human(socrates) *if X=socrates*

- Unification is an algorithm for determining the substitutions needed to make two predicate calculus expressions match.

- Unification and inference rules allows us to make inferences on a set of logical assertions. To do this, the logical data base must be expressed in an appropriate form.

Substitution



- To make, say p(X, X) and p(Y, Z) match, we may use

  X/ Y - X substitute for Y

  X/ Z - X substitute for Z

sometimes, we may need to substitute a function for a variable:

e. g., human(X) → mortal(X)

human(**father_ of(plato)**)

↓ **father_ of(plato)**/ X

human(**father_ of(plato)**) → mortal(**father_ of(plato)**)

Substitution (cont.)

- Another example:

foo(fred, a, goo(Z))

{fred/ X, Z/ Y}

foo(X, a, goo(Y))

{W/ X, jack/ Y}

foo(W, a, goo(jack))

{Z/ X, moo(Z)/ Y}

foo(Z, a, goo(moo(Z)))

- A variable can be substituted by a constant, but any constant is a *ground instance* and may not be replaced

  red(X)

  red(square)

Substitution (cont.)

- Question: Unify q(X, tom) & q(mary, Y)

q(X, tom) & q(mary, Y)

{tom/ Y, mary/ X}

q(mary, tom) q(mary, tom)

- A variable must be substituted consistently across all occurrences of variable in both expressions being matched.

$p(X) \wedge q(Y, X)$ $p(W) \wedge q(Z, X)$

{W/ X, Y/ Z}

$p(W) \wedge q(Y, W)$ $p(W) \wedge q(Y, W)$

Substitution (cont.)

- A variable cannot be unified with a term containing that variable:

$$
\begin{tikzcd}
X \arrow[d] & \{P(X)/X\} & p(X) \arrow[d] \\
p(X) & & p(p(X)) \\
p(p(X)) & & p(p(p(X)))\ \ldots\ \bullet
\end{tikzcd}
$$

- If we apply a series of substitutions, the effect is the same as a single substitution that is a composition of the series of substitutions.

$$
\begin{tikzcd}
& \mathbf{p(X) \wedge q(Y, X)} \arrow[dl, "\{V/X, Z/Y\}"'] \arrow[dd, "\{a/X. f(b)/Y\}"] \\
\mathbf{p(V) \wedge q(Z, V)} \arrow[dr, "\{a/V, f(b)/Z\}"'] & \\
& \mathbf{p(a) \wedge q(f(b), a)}
\end{tikzcd}
$$

Most General Unifier

- Consider kick(X, Y) & kick(tom, Z)

  we can unify by either

  {tom/ X, Z/ Y} kick(tom, Z)

  kick(tom, Z)

  or

  {tom/ X, bob/ Y, bob/ Z} kick(tom, bob)

  kick(tom, bob)

- The first unification is more general than the second. The second generates ground instances where all variables are replaced by constants.

- The unification algorithm will find the most general substitution (unifier), generating ground instances only when necessary.

Summary

- A proposition is a statement with two possible values: true or false.
- A predicate is a statement about an *object* or *individual.* Again, it takes two possible values: true or false.
- Almost any English sentence may be represented in first- order predicate calculus.
- There is no unique mapping of sentence into predicate calculus expressions.
- Unification is an algorithm for determining the substitutions needed to make two predicate calculus expressions match.
- Unification and inference rules allows us to make inferences on a set of logical assertions.

Students’ Mini Research Presentation by Group C

What's in Store for Lecture 8

- Important: Lecture on 20th October 1999, ie. 3 weeks break.

- Knowledge Representation III

End of Lecture 7

Good Night.