Knowledgebase Representation

Predicate Logic

What is a Logic?

- A language with concrete rules
  - No ambiguity in representation (may be other errors!)
  - Allows unambiguous communication and processing
  - Very unlike natural languages e.g. English
- Not to be confused with logical reasoning: Logics are languages, reasoning is a process (may use logic)
- Syntax
  - Rules for constructing legal sentences in the logic
  - Which symbols we can use (English: letters, punctuation)
  - How we are allowed to combine symbols
- Semantics
  - How we interpret (read) sentences in the logic
  - Assigns a meaning to each sentence
- Example: “All lecturers are seven foot tall”
  - A valid sentence (syntax)
  - And we can understand the meaning (semantics)
  - This sentence happens to be false (there is a counterexample)

Propositional Logic

- A proposition is a collection of declarative statements that has either a truth value "true” or a truth value "false".
- A propositional consists of propositional variables and connectives.
- We denote the propositional variables by capital letters (A, B, etc).
- The connectives connect the propositional variables.
- Some examples:

  "Man is Mortal“ is a proposition.

  "12 + 9 = 3 – 2” is a proposition.

  "A is less than 2“ is not a Proposition.

Propositional Logic

Connectives:

In propositional logic generally we use five connectives which are :

- OR (∨)
- AND (∧)
- Negation/ NOT (¬)
- Implication / if-then (→):

  A→B is the proposition "if A, then B".

  It is false if A is true and B is false. The rest cases are true.

A	B	A ⇒ B
0	0	1
0	1	1
1	0	0
1	1	1

- Equivalent: If and only if (⇔) means (A ⇒ B) ∧ (B ⇒ A)

  It is true when A and B are same, i.e. both are false or both are true.

Propositional Logic

Examples:

- It is hot

  p

- It is not hot

  $\urcorner p$

If it is raining, then will not go to mountain

  $p \rightarrow \urcorner q$

- The food is good, and the service is good

  $x \wedge y$

- If The food is good and the service is good then the restaurant is good

  $x \wedge y \rightarrow z$

Propositional Logic

- ***Equivalence rules:***

$$
\begin{aligned}
(\alpha \wedge \beta) &\equiv (\beta \wedge \alpha) \quad \text{commutativity of } \wedge \\
(\alpha \vee \beta) &\equiv (\beta \vee \alpha) \quad \text{commutativity of } \vee \\
((\alpha \wedge \beta) \wedge \gamma) &\equiv (\alpha \wedge (\beta \wedge \gamma)) \quad \text{associativity of } \wedge \\
((\alpha \vee \beta) \vee \gamma) &\equiv (\alpha \vee (\beta \vee \gamma)) \quad \text{associativity of } \vee \\
\neg(\neg\alpha) &\equiv \alpha \quad \text{double-negation elimination} \\
(\alpha \Rightarrow \beta) &\equiv (\neg\beta \Rightarrow \neg\alpha) \quad \text{contraposition} \\
(\alpha \Rightarrow \beta) &\equiv (\neg\alpha \vee \beta) \quad \text{implication elimination} \\
(\alpha \Leftrightarrow \beta) &\equiv ((\alpha \Rightarrow \beta) \wedge (\beta \Rightarrow \alpha)) \quad \text{biconditional elimination} \\
\neg(\alpha \wedge \beta) &\equiv (\neg\alpha \vee \neg\beta) \quad \text{de Morgan} \\
\neg(\alpha \vee \beta) &\equiv (\neg\alpha \wedge \neg\beta) \quad \text{de Morgan} \\
(\alpha \wedge (\beta \vee \gamma)) &\equiv ((\alpha \wedge \beta) \vee (\alpha \wedge \gamma)) \quad \text{distributivity of } \wedge \text{ over } \vee \\
(\alpha \vee (\beta \wedge \gamma)) &\equiv ((\alpha \vee \beta) \wedge (\alpha \vee \gamma)) \quad \text{distributivity of } \vee \text{ over } \wedge
\end{aligned}
$$

Predicate logic

- Limitation of propositional logic: no properties of objects and relationships between objects in propositions.
- If there are n people and m locations, representing the fact that some person moved from one location to another requires mn^2 separate symbols.
- Predicate logic includes a richer ontology:

  Move(x, y, z) for person x moved from location y to z.
- Predicates:

Property(object): New York is raining: Rain(N)

- Functions: f(x1,x2,…,xn):

I like cheese: Like(me, cheese)   Like(x,y)

- Quantifiers: qualify values of variables
  - True for all objects (Universal): $\forall$X. likes(X, apples)
  - Exists at least one object (Existential): $\exists$X. likes(X, apples)

Predicate logic

Examples:-

- All basketball players are tall

  $\forall X$ play(X, basketball) $\rightarrow$tall(X)

- John like anyone who likes books

  like(X,book) $\rightarrow$ like(john,X)

- Nobody likes taxes

  $\neg$ $\exists X$ likes(X,taxes)

- There is a person who writes computer class

  $\exists X$ write(X,computer class)

- John did not study but he is lucky

  $\neg$study(john) $\wedge$ lucky(john)

Predicate logic

Examples:

- Every car is owned by someone.

  $\forall X\ Car(X) \rightarrow \exists Y\ Own(Y,X)$

- Everybody owns a car

  $\forall X\ \exists Y\ Own(X,Y) \land\ Car(Y)$

- Ali owns a car

  $\exists X\ Own(Ali, X) \land\ Car(X)$

- Nobody pass the exam

  $\urcorner\ (\exists X\ pass(X,exam))$

- Not everybody pass the exam

  $\urcorner\ (\forall X\ pass(X,exam))$

Resolution

- It is a technique for proving theorems in propositional and predicate calculus.
- It describes a way of finding contradictions in the databases with a minimum use of substitutions.
- It proves a theorem by negating the statement to be proved and adding this negated goal to the set of axioms that are known to be true.
- It uses the inference rules to show that this leads to a contradiction.
- If the negated goal is inconsistent with the given set of axioms, it follows that the original goal is consistent.

Resolution

- The Resolution by Refutation Algorithm includes the following steps:-

a) Convert the statements to Predicate Logic.

b) Convert the statements from Predicate Logic to CNF “Conjunctive Normal Form”.

c) Add the negation of what is to be proved to the CNF.

d) Resolve the clauses to producing new clauses and producing a contradiction by generating the empty clause.

Resolution

The statements that produced from predicate logic method are nested and very complex to understand, so this will lead to more complexity in resolution stage , therefore the following algorithm is used to convert the predicate logic to clause form:-

1. Eliminate all $(\rightarrow)$

   by replacing each instance of the form $(P \rightarrow Q)$ by expression $(\neg P \vee Q)$

2. Reduce the scope of negation

   $\neg(\neg a) \equiv a$

   $\neg(\forall X)\ b(X) \equiv \exists X \neg\ b(X)$

   $\neg(\exists X)\ b(X) \equiv \forall X \neg b(X)$

   $\neg(a \wedge b) \equiv \neg a \vee \neg b$

   $\neg(a \vee b) \equiv \neg a \wedge \neg b$

Resolution

3. Standardize variables: rename all variables so that each quantifier has its own unique variable name.

   For example: $\forall X\ a(X) \vee \forall X\ b(X) \equiv \forall X\ a(X) \vee \forall Y\ b(Y)$

4. Move all quantifiers to the left without changing their order.

   For example: $\forall X\ a(X) \vee \forall Y\ b(Y) \equiv \forall X\ \forall Y\ a(X) \vee b(Y)$

5. Eliminate existential quantification by using the equivalent function.

   For example: $\exists X\ p(X) \equiv p(C)$ where C is a constant.

   $\forall X\ \exists Y\ (mother(X, Y)) \equiv \forall X\ (mother(X, m(X)))$

   $\forall X\ \forall Y\ \exists Z\ \forall W\ (food(X, Y, Z, W) \equiv \forall X\ \forall Y\ \forall W\ (food(X,Y,f(X,Y), W)$

6. Remove universal quantification symbols.

   For example: $\forall X\ \forall Y\ (p(X, Y, f(X, Y))) \equiv p(X, Y, f(X, Y))$

Resolution

7. Distribute "and" over "or" to get a conjunction of disjunctions called conjunctive normal form.

For example:

$a \vee (b \vee c) \equiv (a \vee b) \vee c$

$a \wedge (b \wedge c) \equiv (a \wedge b) \wedge c$

$a \vee (b \wedge c) \equiv (a \vee b) \wedge (a \vee c)$

$a \wedge (b \vee c) \equiv (a \wedge b) \vee (a \wedge c)$

8. Split each conjunct into a separate clause.

For example:

$(\neg a(X) \vee \neg b(X) \vee e(W)) \wedge (\neg b(X) \vee \neg d(X,f(X)) \vee e(W))$

changed to

$\neg a(X) \vee \neg b(X) \vee e(W)$

$\neg b(X) \vee \neg d(X,f(X)) \vee e(W)$

Resolution

9. Standardize variables apart again so that each clause contains variable names that do not occur in any other clause.

For example:

$(\neg a(X) \vee \neg b(X) \vee e(W)) \wedge (\neg b(X) \vee \neg\, d(X,f(X)) \vee e(W))$

$\neg a(X) \vee \neg b(X) \vee e(W)$

$\neg b(Y) \vee \neg d(X,f(X)) \vee e(V)$

Resolution

c) Add the negation of what is to be proved to the CNF forms.

d) Resolve the clauses to producing new clauses and producing a contradiction by generating the empty clause.

There are two ways to do this, the first is **backward resolution** and the second is **forward resolution.**

Resolution

Example:

Use the Resolution Algorithm to prove that “John is happy” with regard the following story:

Everyone passing their AI exam and winning the lottery is happy. But everyone who studies or lucky can pass all their exams, John did not study but he is lucky. Everyone who is lucky wins the lottery. Prove that John is happy.

Solution:

a) Convert all statements to predicate logic.

$\forall X$ pass(X, AI exam) $\wedge$ win(X,lottery) $\rightarrow$ happy(X)

$\forall X$ study(X) $\vee$ lucky(X) $\rightarrow$ $\forall E$ pass(X, E)

$\neg$study(john) $\wedge$ lucky(john)

$\forall X$ lucky (X) $\rightarrow$win (X,lottery)

happy (john)?

Resolution

b) Convert the statements from predicate logic to CNF forms.

Rule 1: Eliminate all (→)

∀X ˥(pass(X, AI exam) ∧ win(X,lottery)) ∨ happy(X)

∀X ˥(study(X) ∨ lucky(X)) ∨ ∀E pass(X, E)

˥study(john) ∧ lucky(john)

∀X ˥(lucky (X)) ∨ win (X,lottery)

happy (john)?

Resolution

Rule 2: Reduce the scope of negation

$\forall X\ (\neg pass(X, AI\ exam) \vee \neg win(X, lottery)) \vee happy(X)$

$\forall X\ (\neg study(X) \wedge \neg lucky(X)) \vee \forall E\ pass(X, E)$

$\neg study(john) \wedge lucky(john)$

$\forall X\ (\neg lucky\ (X)) \vee win\ (X, lottery)$

happy (john)?

Resolution

Rule 3: Standardize variables

∀X (˥pass(X, AI exam) ∨ ˥win(X,lottery)) ∨ happy(X)

∀Y (˥study(Y) ∧ ˥lucky(Y)) ∨ ∀E pass(Y, E)

˥study(john) ∧ lucky(john)

∀Z (˥lucky (Z)) ∨ win (Z,lottery)

happy (john)?

Resolution

Rule 4: Move all quantifiers to the left

∀X (˥pass(X, AI exam) ∨ ˥win(X,lottery)) ∨ happy(X)

∀Y ∀E (˥study(Y) ∧ ˥lucky(Y)) ∨ pass(Y, E)

˥study(john) ∧ lucky(john)

∀Z (˥lucky (Z)) ∨ win (Z,lottery)

happy (john)?

Resolution

Rule 5: Eliminate existential quantification by using the equivalent function

Nothing to do.

Rule 6: Remove universal quantification symbols

(˥pass(X,AI exam) ∨ ˥win(X,lottery)) ∨ happy(X)

(˥study(Y) ∧ ˥lucky(Y)) ∨ pass(Y,E)

˥study(john) ∧ lucky(john)

(˥lucky(Z)) ∨ win(Z,lottery)

happy (john)?

Resolution

Rule 7: Distribute "and" over "or“ $(a \wedge b) \vee c \equiv c \vee (a \wedge b)$

┐pass(X,AI exam) ∨ ┐win(X,lottery) ∨ happy(X)

(┐study(Y)∧┐lucky(Y))∨ pass(Y,E)

The second statement become:

pass(Y, E) ∨ ┐study(Y) ∧ pass(Y, E) ∨ ┐lucky(Y)

┐study(john) ∧ lucky(john)

┐lucky(Z) ∨ win(Z,lottery)

happy(john)?

Resolution

Rule 8: Split each conjunct into a separate clause

˥pass(X,AI exam) ∨ ˥win(X,lottery) ∨ happy(X)

pass(Y,E) ∨ ˥study(Y)

pass(Y,E) ∨ ˥lucky(Y)

˥study(john)

lucky(john)

˥lucky(Z) ∨ win(Z,lottery)

happy(john)?

Resolution

Rule 9: Standardize variables apart again

˥pass(X,AI exam) ˅ ˥win(X,lottery) ˅ happy(X)

pass(Y,E) ˅ ˥study(Y)

pass(M,G) ˅ ˥lucky(M)

˥study(john)

lucky(john)

˥lucky(Z) ˅ win(Z,lottery)

happy(john)?

Resolution

c) Add the negation of what is to be proved to the clause forms.

pass(X,AI exam) ∨ ˥win(X,lottery) ∨ happy(X)

pass(Y,E) ∨ ˥study(Y)

pass(M,G) ∨ ˥lucky(M)

˥study(john)

lucky(john)

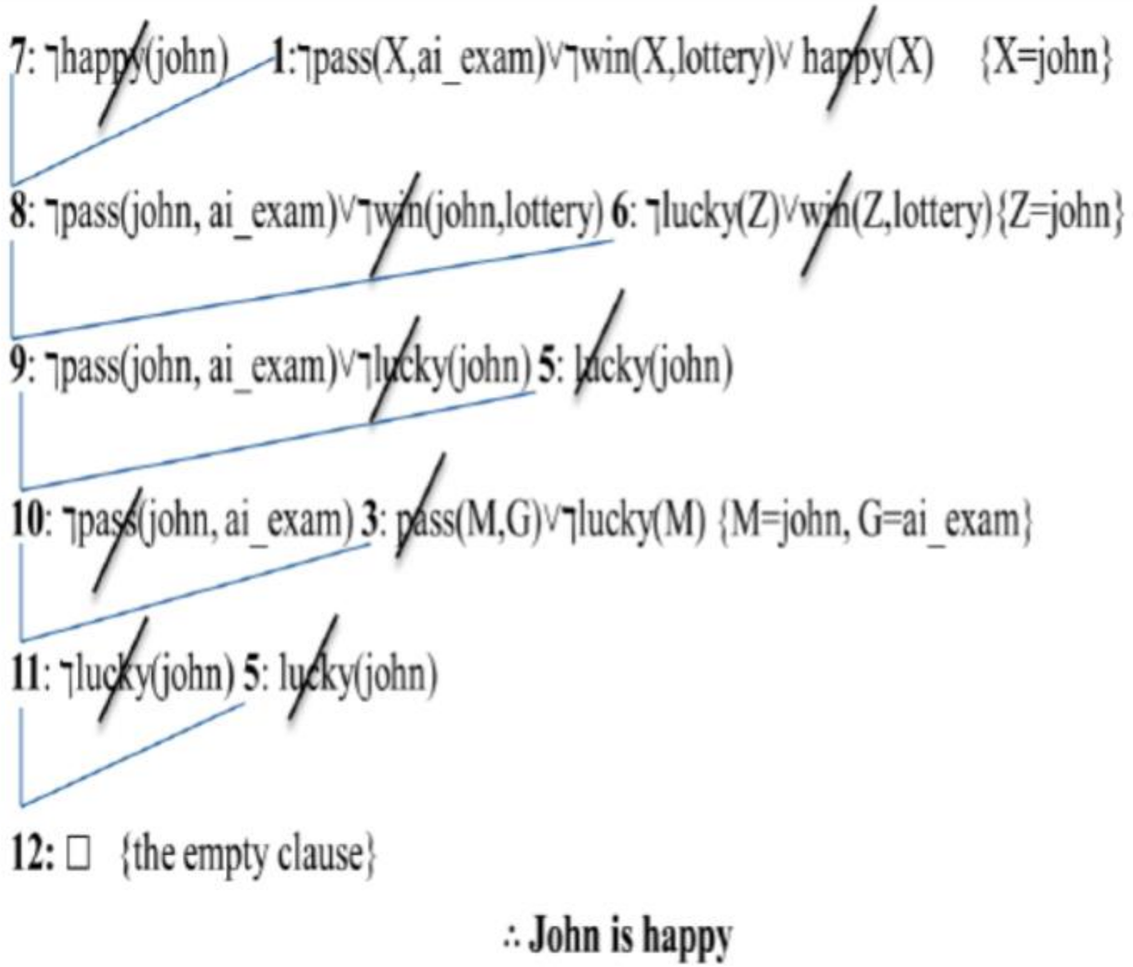
˥lucky(Z) ∨ win(Z,lottery)

˥happy(john).

Resolution

d) Resolve the clauses to producing new clauses and producing a contradiction by generating the empty clause.

- Backward Resolution:



Resolution

- Forward Resolution:

1:¬pass(X,ai_exam)∨~~¬win(X,lottery)~~∨happy(X) 6: ¬lucky(Z)∨~~win(Z, lottery)~~ {Z=X}

8: ¬ pass(X,ai_exam)∨happy(X) ∨ ~~¬lucky(X)~~ 5: ~~lucky(john)~~ {X=john}

9:~~¬ pass(john, ai_exam)~~∨happy(john) 3: ~~pass(M,G)~~∨¬lucky(M) {M=john, G=ai_exam}

10: happy(john) ∨ ~~¬lucky(john)~~ 5: ~~lucky(john)~~

11 : ~~happy(john)~~ 7: ~~¬happy(john)~~

12: □ {the empty clause}

∴ **John is happy**

Resolution

- Example 2:

a. Translate the following statements into predicate logic:

1. Anyone whom Mary loves is a football star.
2. Any student who does not pass does not play.
3. John is a student.
4. Any student who does not study does not pass.
5. Anyone who does not play is not a football star.

b. Using the above premises prove that

“if John does not study, then Mary does not love John”