Artificial Intelligence

CC-421

Basics of First-Order Logic

Like propositional logic, first-order logic consists of a formal language with a syntax and semantics that give meaning to the well-formed strings.

Syntax

The alphabet of first-order logic contains the following symbols:

1. Constants: A constant is a symbol like 'Socrates', 'John', 'B', and '1'.
2. Predicates: The symbols 'True' and 'False' and other symbols like 'married', 'love', and 'brother'.
3. Functions: Symbols like 'mother', 'weight', and 'height'.
4. Variables: A variable is a lower-case alphabetic character like x, y, or z.
5. Operators: $\neg, \vee, \wedge, \Rightarrow, \Leftrightarrow$.
6. Quantifiers: The symbols $\forall$ (for all) and $\exists$ (there exists).
7. Grouping symbols: The open and closed parentheses and the comma.

The rules for creating well-formed strings are as follows:

1. A term is one of the following:
   (a) A constant symbol
   (b) A variable symbol
   (c) A function symbol followed by one or more terms separated by commas and enclosed in parentheses
2. An atomic formula is one of the following:
   (a) A predicate symbol
   (b) A predicate symbol followed by one or more terms separated by commas and enclosed in parentheses
   (c) Two terms separated by the $=$ symbol
3. A formula is one of the following:
   (a) An atomic formula
   (b) The operator $\neg$ followed by a formula
   (c) Two formulas separated by $\vee, \wedge, \Rightarrow$, or $\Leftrightarrow$
   (d) A quantifier following by a variable followed by a formula
4. A sentence is a formula with no free variables.

A variable in a formula is free if it is not quantified by the $\forall$ or the $\exists$ symbol; otherwise it is bound. The formula on the left in the next expression contains the free variable y and therefore is not a sentence, whereas the formula on the right does not contain any free variables and therefore is a sentence.

$\forall x$ jump(x,y) $\forall x$ tall(x).

In first-order logic we have a **domain of discourse.** This domain is a set and each element in the set is called an entity. Each constant symbol identifies one entity in the domain.

Example 1 If we are considering all individuals living in a certain home, our constant symbols could be their names. If there are five such individuals, the constant symbols might be ‘Mary’, ‘Fred’, ‘Sam’, ‘Laura’, and ‘Dave’.

A predicate denotes a relationship among a set of entities or a property of a single entity. For example,

married(Mary,Fred)
young(Sam)

Los operadores de la lógica proposicional funcionan de la misma manera:

$$\neg\mathrm{married}(\mathrm{Mary},\mathrm{Fred})$$

denotes that Mary and Fred are not married. The formula

$$\neg\mathrm{married}(\mathrm{Mary},\mathrm{Fred}) \wedge \mathrm{young}(\mathrm{Sam})$$

denotes that Mary and Fred are not married and that Sam is young.

The equality operator is used to denote that two terms refer to the same entity. For example,

Mary = mother(Laura)

denotes that Mary and mother(Laura) refer to the same entity.

The quantifier $\forall$ denotes that some formula is true for all entities in the domain of discourse.

The quantifier $\exists$ denotes that some formula is true for at least one entity in the domain.

Semantics

In first-order logic we first specify a signature, which determines the language. Given a language, a model has the following components:

1. A nonempty set D of **entities** called a **domain of discourse**.
2. An **interpretation**, which consists of the following:
   (a) An entity in D is assigned to each of the constant symbols. Ordinarily, every entity is assigned to a constant symbol.
   (b) For each function, an entity is assigned to each possible input of entities to the function.
   (c) The predicate 'True' is always assigned the value T, and the predicate 'False' is always assigned the value F.
   (d) For every other predicate, the value T or F is assigned to each possible input of entities to the predicate.

Here are some related examples.

Example2 Suppose our application is considering the five individuals mentioned in Example 1, and we are only concerned with discussing whether any of them are married and whether any of them are young. Our signature can be as follows:

1. Constant Symbols = {Mary, Fred, Sam, Laura, Dave }.
2. Predicate Symbols = {married, young}- . The predicate ‘married’ has arity two and the predicate ‘young’ has arity one.

One particular model has these components:

1. The domain of discourse D is the set of these five particular individuals.
2. The interpretation is as follows:
   a. A different individual is assigned to each of the constant symbols.
   b. The truth value assignments are given by these tables:

x	Mary	Fred	Sam	Laura	Dave
young(x)	F	F	T	T	T

x \ y	Mary	Fred	Sam	Laura	Dave
Mary	F	T	F	F	F
Fred	T	F	F	F	F
Sam	F	F	F	F	F
Laura	F	F	F	F	F
Dave	F	F	F	F	F

married(x, y)

Example3 Suppose our application is considering the three individuals named Dave, Gloria, and Ann, and we are concerned with discussing the mother of each individual and whether each individual loves the other. Our signature can be as follows:

1. Constant Symbols = {Dave, Gloria,Ann }.
2. Predicate Symbols = {love} . The predicate 'love' has arity two.
3. Function symbols = {mother}. The function 'mother' has arity one.

One particular model has these components:

1. The domain of discourse D is the set of these three particular individuals.
2. The interpretation is as follows:
   a. A different individual is assigned to each of the constant symbols.
   b. The truth value assignments are given by these tables:
   c. The function assignments are given by this table:

x	Dave	Gloria	Ann
mother(x)	Gloria	Ann	-

y / x	Dave	Gloria	Ann
Dave	F	F	F
Gloria	T	T	T
Ann	T	T	F

love(x, y)

Notice that no assignment is made for the mother of Ann. Technically, every entity must be assigned a value by a function. If Ann's mother is not one of the entities, we can simply use a dummy symbol like the dash (-) as our assignment.

Once a model is specified, the truth values of all sentences are assigned according to the following rules:

1. The truth values for sentences developed with the symbols $\neg, \wedge, \vee, \Rightarrow$, and $\Leftrightarrow$ are assigned in the same way as done in propositional logic.

2. The truth value for two terms connected by the $=$ symbol is T if both terms refer to the same entity; otherwise it is F.

3. The truth value for $\forall x \ \mathrm{p}(x)$ has value T if $\mathrm{p}(x)$ has value T for every assignment to x of an entity in the domain D; otherwise it has value F.

4. The truth value for $\exists x \ \mathrm{p}(x)$ has value T if $\mathrm{p}(x)$ has value T for at least one assignment to x of an entity in the domain D; otherwise it has value F.

5. The operator precedence is as follows: $\neg, =, \wedge, \vee, \Rightarrow, \Leftrightarrow$.

6. The quantifiers have precedence over the operators.

7. Parentheses change the order of the precedence.

Example 4: Suppose we have the model and interpretation in Example 2. Recall that the truth value assignments in that interpretation are given by this table:

x \ y	Dave	Gloria	Ann
Dave	F	F	F
Gloria	T	T	T
Ann	T	T	F

love(x, y)

Then we have the following:

1. The sentence : **$\exists x\ \forall y$ loves(x, y)**

has value T because love(Gloria,y) has value T for every value of y. This sentence means that there is someone who loves everybody. It is true because Gloria loves everyone.

2. The sentence: **$\forall x\ \exists y$ loves(x,y)**

has value F because love(Dave,y) does not have value T for any value of y. This sentence means that everyone loves someone. It is false because Dave does not love anyone.

Validity and Logical Implication

Definition If sentence s has value T under interpretation I, we say that I satisfies s, and we write I ⊧ s. A sentence is satisfiable if there is some interpretation under which it has value T. By example the sentence $\forall x$ human(x) is satisfied in any interpretation that assigns T to human(x) for every individual x in the domain of discourse.

Definition A formula that contains free variables is satisfied by an interpretation if the formula has value T regardless of which individuals from the domain of discourse are assigned to its free variables.

The formula loves(Socrates, y) is satisfied by any interpretation that assigns T to loves(Socrates, y) for every individual y in the domain of discourse.

Definition A formula is valid if it is satisfied by every interpretation.

The formula loves(Socrates, y) $\vee$ $\neg$ loves(Socrates, y) is valid. Regardless of which individual in the domain of discourse is assigned to y, the formula is true in every interpretation. So it is valid.

Validity and Logical Implication

Definition A sentence is a contradiction if there is no interpretation that satisfies it. The sentence $\exists x\ (\text{human}(x) \land \neg\ \text{human}(x))$ is not satisfiable under any interpretation and is therefore a contradiction.

Definition Given two formulas A and B, if $A \Rightarrow B$ is valid, we say that A logically implies B and we write $A \Rrightarrow B$.

$\text{human(Socrates)} \land (\text{human(Socrates)} \Rightarrow \text{mortal(Socrates)}) \Rrightarrow \text{mortal(Socrates)}.$

Definition Given two formulas A and B, if $A \Leftrightarrow B$ is valid, we say that A is logically equivalent to B and we write $A \equiv B$.

$\text{human(Socrates)} \Rightarrow \text{mortal(Socrates)} \equiv \neg\text{human(Socrates)} \lor \text{mortal(Socrates)}.$

Theorem The following logical equivalences hold. (A and B are variables denoting arbitrary predicates. Furthermore, they could have other arguments besides x.):

1. $\neg\exists x\ A(x) \equiv \forall x\ \neg A(x)$
2. $\neg\forall x\ A(x) \equiv \exists x\ \neg A(x)$
3. $\exists x\ (A(x) \vee B(x)) \equiv \exists x\ A(x) \vee \exists x\ B(x)$
4. $\forall x\ (A(x) \wedge B(x)) \equiv \forall x\ A(x) \wedge \forall x\ B(x)$
5. $\forall x\ A(x) \equiv \forall y\ A(y)$
6. $\exists x\ A(x) \equiv \exists y\ A(y)$

Derivation Systems

We write arguments showing the list of premises followed by the conclusion as follows:

1. A_1
2. A_2

$\vdots$

$n.$ $\underline{A_n}$

B

If the argument is sound, we write

$$A_1, A_2, \ldots, A_n \models B,$$

and if it is a fallacy, we write

$$A_1, A_2, \ldots, A_n \not\models B.$$

Example Consider the following argument:

1. man(Socrates)
2. man(Socrates) $\Rightarrow$ human(Socrates)

$$\frac{\text{man(Socrates)} \Rightarrow \text{human(Socrates)}}{\text{human(Socrates)}}$$

We have the following derivation for this argument:

	Derivation	Rule
1	man(Socrates)	Premise
2	man(Socrates) $\Rightarrow$ human(Socrates)	Premise
3	human(Socrates)	1, 2, MP

Universal Instantiation

The universal instantiation (UI) rule is as follows:

$$\forall x\, A(x) \models A(t),$$

where t is any term. This rules says that if A(x) has value T for all entities in the domain of discourse, then it must have value T for term t. Consider the following argument:

1. man(Socrates)
2. $\forall x$ man(x) $\Rightarrow$ human(x)

---

human(Socrates)

We have the following derivation for this argument:

	Derivation	Rule
1	man(Socrates)	Premise
2	$\forall x$ (man(x) $\Rightarrow$ human(x))	Premise
3	man(Socrates) $\Rightarrow$ human(Socrates)	2, UI
4	human(Socrates)	1, 3, MP

Consider the following argument:

1. $\forall x$ (father(Sam, x) $\Rightarrow$ son(x, Sam) $\vee$ daughter(x, Sam))
2. father(Sam, Dave)
3. $\neg$daughter(Dave, Sam)

son(Dave, Sam)

We have the following derivation for this argument:

	Derivation	Rule
1	$\forall x$ (father(Sam, x) $\Rightarrow$ son(x, Sam) $\vee$ daughter(x, Sam))	Premise
2	father(Sam, Dave)	Premise
3	$\neg$daughter(Dave, Sam)	Premise
4	father(Sam, Dave) $\Rightarrow$ son(Dave, Sam) $\vee$ daughter(Dave, Sam)	1, UI
5	son(Dave, Sam) $\vee$ daughter(Dave, Sam))	2, 4, MP
6	son(Dave, Sam)	3, 5, DS

t can be a bound variable?

Universal Generalization

The universal generalization (UG) rule is as follows:

A(e) for every entity e in the domain of discourse $\models$ $\forall$x A(x),

This rule says that if A(e) has value T for every entity e, then $\forall$x A(x) has value T. By example:

$$\begin{array}{l} 1.\ \forall x\ (\text{study}(x) \Rightarrow \text{pass}(x)) \\ \hline \forall x\ (\neg\text{pass}(x) \Rightarrow \neg\text{study}(x)) \end{array}$$

This argument says that if it is true that everyone who studies passes, then we can conclude that everyone who did not pass did not study. We will use UG and the deduction theorem (DT) to derive this argument:

	Derivation	Rule	Comment
1	$\forall x\ (\text{study}(x) \Rightarrow \text{pass}(x))$	Premise	
2	$\text{study}(e) \Rightarrow \text{pass}(e)$	UI	Substitute arbitrary entity e.
3	$\neg\text{pass}(e)$	Assumption	We assume $\neg\text{pass}(e)$.
4	$\neg\text{study}(e)$	2, 3, MT	
5	$\neg\text{pass}(e) \Rightarrow \neg\text{study}(e)$	DT	We discharge $\neg\text{pass}(e)$.
6	$\forall x\ \neg\text{pass}(x) \Rightarrow \neg\text{study}(x)$	UG	

Notice in the previous example that we showed the conclusion had value T for an arbitrary entity e and then applied UG. If we only know that a formula has value T for a specific entity, we cannot use UG.

For example, if we only know young(Dave) or young(e) where e represents a specific entity, we cannot use UG. An entity e introduced with existential instantiation (which will be discussed shortly) is an example of a case where e represents a specific entity.

Existential Generalization

The existential generalization (EG) rule is as follows:

$$A(e) \models \exists x\, A(x),$$

where e is an entity in the domain of discourse. This rule says that if A(e) has value T for some entity e, then $\exists x\, A(x)$ has value T. Consider the following argument and derivation:

1. $\text{man}(\text{Socrates})$
2. $\forall x\ \text{man}(x) \Rightarrow \text{human}(x)$

$$\frac{\qquad}{\exists x\ \text{human}(x)}$$

	Derivation	Rule
1	$\text{man}(\text{Socrates})$	Premise
2	$\forall x\ \text{man}(x) \Rightarrow \text{human}(x)$	Premise
3	$\text{man}(\text{Socrates}) \Rightarrow \text{human}(\text{Socrates})$	2, UI
4	$\text{human}(\text{Socrates})$	1, 3, MP
5	$\exists x\ \text{human}(x)$	4, EG

The variable x may not appear as a free variable in formula A(e) when we apply EG. For example, suppose we know that father(x,e) has value T for some entity e. We cannot conclude that **$\exists x$ father(x, x)** as this would mean there is some entity that is its own father.

We would need to use another variable such as y and conclude that **$\exists y$ father(x, y).**

Existential Instantiation

The existential instantiation (EI) rule is as follows:

$\exists x\ A(e) \models A(e)$,

for some entity e in the domain of discourse. This rule says that if $\exists x\ A(x)$ has value T,
then A(e) has value T for some entity e . Consider the following argument:

1. $\exists x \text{ man}(x)$
2. $\forall x \text{ man}(x) \Rightarrow \text{human}(x)$

$\exists x \text{ human}(x)$

We have the following derivation for this argument:

	Derivation	Rule
1	$\exists x \text{ man}(x)$	Premise
2	$\forall x \text{ man}(x) \Rightarrow \text{human}(x)$	Premise
3	$\text{man}(e)$	1, EI
4	$\text{man}(e) \Rightarrow \text{human}(e)$	2, UI
5	$\text{human}(e)$	3, 4, MP
6	$\exists x \text{ human}(x)$	5, EG

The variable used in EI cannot appear elsewhere as a free variable. For example, suppose we conclude **man(e)** using EI. We cannot later conclude **monkey(e)** using EI, for this would mean e is both a man and a monkey. Rather, we must use a different variable such as f and conclude **monkey(f)**.

Modus Ponens for First-Order Logic

Find the derivation of this argument:

1. mother(Mary,Scott)
2. sister(Mary,Alice)
3. $\forall x\ \forall y \forall z\ \text{mother}(x, y) \wedge \text{sister}(x, z) \Rightarrow \text{aunt}(z, y)$

---

aunt(Alice,Scott)

Unification

Definition Suppose we have two sentences A and B. A unification of A and B is a substitution θ of values for some of the variables in A and B that make the sentences identical. The set of substitutions θ is called the unifier.

Example Suppose that we have the two sentences **parents(Dave, y, z)** and **parents(y, Mary, Sam)**. Because the y variables in the two sentences are different variables, we rename the second y variable as x to obtain the sentence parents(x, Mary, Sam). Then

θ = {x/Dave, y/Mary, z/Sam}

unifies the two sentences into the sentence parents(Dave,Mary,Sam).

Ejemplo We cannot unify the sentences **parents(Dave, Nancy, z)** and **parents(y, Mary, Sam)** because Nancy and Mary are both constants and therefore cannot be substituted.

Unification

Definition A unifier θ is a most general unifier (MGU)) if every other unifier θ' is an instance of θ in the sense that θ' can be derived by making substitutions in θ.

Example Suppose that we have the two sentences **father(x, Sam)** and **father(y, z).** Then:

$\boldsymbol{\theta_1}$ **= {x/Dave, y=Dave, z/Sam}**

unifies the two sentences into the sentence father(Dave, Sam). Furthermore,

$\boldsymbol{\theta_2}$ **= {x/y , z/Sam}**

unifies the two sentences into the sentence **father(y, Sam).**

Unifier θ_2 in example is a most general unifier. The following algorithm returns a most general unifier of two sentences if they can be unified.

We want to unify the literals p(f(g(x)), y, z) and p(u, u, f(u)). Several unifiers are:

$$
\begin{array}{lllll}
\sigma_1: & & y/f(g(x)), & z/f(f(g(x))), & u/f(g(x)), \\
\sigma_2: & x/h(v), & y/f(g(h(v))), & z/f(f(g(h(v)))), & u/f(g(h(v))) \\
\sigma_3: & x/h(h(v)), & y/f(g(h(h(v)))), & z/f(f(g(h(h(v))))), & u/f(g(h(h(v)))) \\
\sigma_4: & x/h(a), & y/f(g(h(a))), & z/f(f(g(h(a)))), & u/f(g(h(a))) \\
\sigma_5: & x/a, & y/f(g(a)), & z/f(f(g(a))), & u/f(g(a))
\end{array}
$$

where σ_1 is the most general unifier. The other unifiers result from σ_1 through the substitutions $x/h(v)$, $x/h(h(v))$, $x/h(a)$, x/a.

Algorithm 3.1 Unification

Input: Two sentences A and B; an empty set of substitutions θ.
Output: A most general unifier of the sentences if they can be unified; otherwise failure.

```
Procedure unify(A, B, var θ);
scan A and B from left to right
    until A and B disagree on a symbol or A and B are exhausted;
if A and B are not exhausted
    let x and y be the symbols where A and B disagree;
    if x is a variable
        θ = θ ∪ {x/y};
        unify(subst(θ, A), subst(θ, B), θ);
    else if y is a variable
        θ = θ ∪ {y/x};
        unify(subst(θ, A), subst(θ, B), θ);
    else
        θ = Failure;
    endif
endif
```

The preceding algorithm calls a procedure **subst**, which takes as input a set of substitutions θ and a sentence A and applies the substitutions in θ to A.

Generalized Modus Ponens

Suppose we have sentences A, B, and C, and the sentence A $\Rightarrow$ B, which is implicitly universally quantified for all variables in the sentence.

The generalized modus ponens (GMP) rule is as follows:

A $\Rightarrow$ B, C, unify(A; C; θ) $\models$ subst(B, θ)

Find the derivation of this argument:

1. mother(Mary,Scott)
2. sister(Mary,Alice)
3. $\forall x\ \forall y \forall z\ \text{mother}(x, y) \land \text{sister}(x, z) \Rightarrow \text{aunt}(z, y)$

$$\frac{\qquad}{\text{aunt(Alice,Scott)}}$$