Knowledge Representation and Reasoning with First Order (FO) Logic

Ramón Béjar

Departament d'Informàtica
Universitat de Lleida

Esquema

A Gentle Reminder about FO Logic

FO syntax (what we can write with FO sentences):
The alphabet for writing FO logic sentences:

- Set of symbols for variables: $\{X_1, X_2, \ldots\}$
- Set of symbols for constants: $\{c_1, c_2, \ldots\}$
- Set of symbols for predicates: $\{P_1^{n_1}, P_2^{n_2}, \ldots\}$
- Set of symbols for functions: $\{f_1^{n_1}, f_2^{n_2}, \ldots\}$
- Logical connectives: $\{\wedge, \vee, \rightarrow, \leftarrow, \neg\}$
- Quantifiers: $\{\forall, \exists\}$

Terms:

- Constants and variables are terms
- If f_i^n is a function symbol of arity n, and $t_1, t_2, \ldots, t_n$ are n terms, then $f_i^n(t_1, t_2, \ldots, t_n)$ is also a term

A Gentle Reminder about FO Logic

FO syntax (what we can write with FO sentences):

An atomic formula is a predicate $P^n(t_1, \dots, t_n)$, where P^n_i is a symbol predicate of arity n and $t_1, t_2, \dots, t_n$ are n terms.

A Literal L is a positive $P^n(t_1, \dots, t_n)$ or negative $\neg P^n(t_1, \dots, t_n)$ atomic formula.

FO formulas:

1. Any atomic formula is a FO formula
2. If A and B are FO formulas and X_1 is a variable symbol, then $(A \wedge B)$, $(\neg A)$, $(A \leftarrow B)$, $(A \rightarrow B)$, $(A \vee B)$, $(\exists X_1\ A)$ y $(\forall X_1\ A)$ are also FO formulas

We consider here only closed formulas, i.e. formulas where all the variables are bounded by a quantifier

A Gentle Reminder about FO Logic

FO semantics (giving meaning to FO sentences).
An interpretation $\mathcal{I}$ for a FO sentence is composed of:

- A domain: a non-empty set of objects D. It can be even of infinite size
- An interpretation function I that assigns:
  - to every constant symbol c an element $\mathrm{I(c)} \in \mathrm{D}$
  - to every function symbol $\mathrm{f_i^n}$, a function $\mathrm{I(f_i^n)}$ with domain $\mathrm{D^n}$ and range D
  - to every predicate symbol $\mathrm{P_i^n}$, a relation $\mathrm{I(P_i^n)}$ over $\mathrm{D^n}$ ($\mathrm{I(P_i^n) \subseteq D^n}$) : it indicates the tuples from $\mathrm{D^n}$ that make the predicate $\mathrm{P_i^n}$ true.

A Gentle Reminder about FO Logic

An interpretation $\mathcal{I}$ satisfies a formula F, denoted as $\models_{\mathcal{I}}$ F, on these cases:

- $\models_{\mathcal{I}} \mathrm{P_i}(\mathrm{t_1}, \ldots, \mathrm{t_n})$ iff $(\mathrm{I}(\mathrm{t_1}), \ldots, \mathrm{I}(\mathrm{t_n})) \in \mathrm{I}(\mathrm{P_i})$.
- $\models_{\mathcal{I}} (\neg \mathrm{F})$ iff $\not\models_{\mathcal{I}} \mathrm{F}$
- $\models_{\mathcal{I}} (\mathrm{F} \wedge \mathrm{G})$ iff $\models_{\mathcal{I}} \mathrm{F}$ and $\models_{\mathcal{I}} \mathrm{G}$
- $\models_{\mathcal{I}} (\mathrm{F} \vee \mathrm{G})$ iff $\models_{\mathcal{I}} \mathrm{F}$ or $\models_{\mathcal{I}} \mathrm{G}$
- $\models_{\mathcal{I}} (\mathrm{F} \rightarrow \mathrm{G})$ iff $\not\models_{\mathcal{I}} \mathrm{F}$ or $\models_{\mathcal{I}} \mathrm{G}$
- $\models_{\mathcal{I}} (\forall \mathrm{X_i} \mathrm{F})$ iff $\models_{\mathcal{I}} \mathrm{F}$ for any value of $\mathrm{X_i}$ in D.
- $\models_{\mathcal{I}} (\exists \mathrm{x_i} \mathrm{F})$ iff $\models_{\mathcal{I}} \mathrm{F}$ for some value of $\mathrm{X_i}$ in D.

A model for a FO formula Γ is an interpretation that satisfies Γ

Esquema

Logical Consequences for FO Logic Formulas

Given a FO formula Γ, and property formula P, we would like to answer this question:

$$\Gamma \models P$$

Meaning: Is P true in all the models of Γ (a logical consequence of Γ) ?

Example: Given the formula Γ

1. no_barro_zapatos(pepe).
2. no_barro_zapatos(juan).

2. $\forall$X (no_es_asesino(X) $\leftarrow$ no_ha_saltado(X)).
3. $\forall$X (no_ha_saltado(X) $\leftarrow$ no_barro_zapatos(X)).

Question: Is $\Gamma \models \exists$Y no_es_asesino(Y) true ?

Logical Consequences for FO Logic Formulas

An important difference between propositional logic and first order logic, that will cause troubles when reasoning with FO formulas, is that a FO formula can have models of arbitrary size, even of infinite size !

A model has infinite size when its domain of objects is infinite

And even worse: a FO formula can have infinite models !

Logical Consequences for FO Logic Formulas

Example: A formula Γ to define the even numbers (but it can be interpreted with many other domains):

1. even_number(0).
2. $\forall$X even_number(two_more(X)) $\leftarrow$ even_number(X).

This formula has models of infinite size. For example:

- Domain $= \mathbb{N} \cup \{0\} \;=\; \{0, 1, 2, \ldots,\}$
- two_more(X) $=$ X $+ 2$
- even_number(X) $= \{$ X $|$ X is even $\}$

But we have actually infinite models for this formula ! Why ?

Logical Consequences for FO Logic Formulas

So FO formulas may have infinite models, and some of them of infinite size. Then, how can we check all the models of a formula ?

Or if we need to check whether $\Gamma \models P$, how can we determine whether $\Gamma \cup \{\neg P\}$ has no models ?

There are some good news: when looking for models of a FO formula, we can focus on a particular class of interpretations: Herbrand interpretations

Herbrand Interpretations for FO Logic Formulas

When looking for models, or trying to prove that there are no models, for a formula, if the formula is in clausal form, it is enough to focus on Herbrand interpretations

In a Herbrand interpretation for a formula F:

1. The domain D is defined inductively as follows:
    1. All the constant symbols in F are in D. If there are no constant symbols in F, we introduce a new one: a
    2. If $t_1, t_2, \ldots, t_n$ are elements in D, not necessarily distinct, and f_i^n is a function symbol in F with arity n, then $f_i^n(t_1, t_2, \ldots, t_n)$ is also an element of D
2. The value assigned to each constant c is the constant c itself (that belongs to D)
3. The value assigned to $f_i(t_1, t_2, \ldots, t_n)$ will be $f_i(t_1, t_2, \ldots, t_n)$ itself (that also belongs to D)
4. The only choice is the set of relations associated with the predicate symbols, but always over the Herbrand domain

Herbrand Interpretations for FO Logic Formulas

So, the Herbrand domain is build composing the constant and function symbols in all possible ways

Working only with Herbrand interpretations makes, sometimes, the search for a proof of the unsatisfiability of a formula much simpler

A particular nice case is when there are no function symbols on the formula. In that case, the Herbrand domain is of finite size, so in this case the space of possible distinct interpretations is also finite !

There are proof procedures based directly on Herbrand models, and for the resolution based procedure we explain we will use always Herbrand interpretations

FO Logic Clausal Form

As with propositional logic, with FO logic we will work with a clausal normal form in order to be able to obtain refutation proofs when trying to prove logical consequences.

A clause is a disjunction of literals, where all the variables that appear in any literal are universally quantified

$$\forall X_1 \forall X_2 \ldots \forall X_n (L_1 \lor L_2 \lor \ldots \lor L_k)$$

A FO formula is in clausal form if it is a conjunction of clauses

FO Logic Clausal Form

Examples:

- $$\forall X_1(p(X_1)) \wedge \forall X_2(r(X_2, g_1(X_2)) \vee \neg r(X_2, g_2(X_2))) \wedge (r(a))$$

  Equivalent version as a set of clauses and omitting the quantifiers:
  $\{\{p(X_1)\}, \{r(X_2, g_1(X_2)), \neg r(X_2, g_2(X_2))\}, \{r(a)\}\}$

- $$\forall X_1 \forall X_2(p(X_1) \vee q(X_2, f(X_1)))\wedge$$
  $$\forall X_3 \forall X_4 \forall X_5(r(X_3) \vee \neg q(X_4, f(X_4)) \vee \neg h(X_5, X_3))$$

  Equivalent version:
  $\{\{p(X_1), q(X_2, f(X_1))\}, \{r(X_3), \neg q(X_4, f(X_4)), \neg h(X_5, X_3)\}\}$

Resolution for FO Logic

There is an algorithm for obtaining a satisfiability preserving clausal form formula from a FO formula not in clausal form

Is similar to the propositional logic version, but the fact of using only universal quantifiers in clauses makes necessary to transform existential quantifiers with the so called Skolem constants and functions

Check the Computational Logic lecture notes !

Resolution for FO Logic

In propositional logic, the resolution inference rule is very simple:

$$C_1 = \{L^1_1, L^1_2 \ldots, P, \ldots, L^1_n\}, C_2 = \{L^2_1, L^2_2 \ldots, \neg P, \ldots, L^2_m\} \vdash C_1 \cup C_2 \setminus \{P, \neg P\}$$

because in any model of $C_1 \wedge C_2$ only P or ¬P can be true, but not both !

In FO Logic, it is not always so easy to find contradictory literals in clauses !

Resolution for First Order Logic

When are two literals contradictory in CP1 ? That is, they cannot be true in the same interpretation ?

Some contradictory literals:

1. $\forall X\ p(X)$ and $\forall Y\ \neg p(Y)$: in any interpretation I, if $\forall X\ p(X)$ is true then $\forall Y\ \neg p(Y)$ is false, and viceversa.
2. $\forall X\ p(X)$ and $\neg p(b)$: in any interpretation I, if $\forall X\ p(X)$ is true then $\neg p(b)$ is false (because $p(b)$ must be true), and viceversa.
3. $\forall X\ \neg p(f(X))$ and $p(f(a))$: in any interpretation I, if $\forall X\ \neg p(f(X))$ is true then $p(f(a))$ is false (because $\neg p(f(a))$ must be true), and viceversa.

Resolution for FO Logic

To find whether two literals, with the same predicate symbol but one positive and the other negative, are really contradictory we can use an algorithm for finding a most general unifier (mgu)

In the previous example:

1. $\forall$X p(X) and $\forall$Y $\neg$p(Y): a mgu is {Y/X}
2. $\forall$X p(X) and $\neg$p(b): a mgu is {b/X}
3. $\forall$X $\neg$p(f(X)) and p(f(a)): a mgu is {a/X}

The mgu identifies the non-empty intersection of the domain of tuples of L_1 and L_2, that is the set of tuples where both literals coincide

Resolution for First Order Logic

Resolution for FO Logic: Given clauses C_1 and C_2 where there is a positive literal L_1 in C_1 and a negative literal L_2 in C_2 such that their predicate symbols are equal and can be unified with an mgu σ, then the following inference rule is sound:

$$C_1, C_2 \vdash ((C_1 \cup C_2) \setminus \{L_1, L_2\})\sigma$$

where σ is a mgu of L_1 and L_2

That is, if C_1 and C_2 are true in a same interpretation, L_1 and L_2 cannot be true on all their possible tuples, because the mgu σ proves that they have at least one common tuple in their domain

Resolution for First Order Logic

Example: Formula Γ:

$$\begin{array}{ll} \{ & q(a) \\ & \forall X\ p(X) \leftarrow q(X)\} \end{array}$$

Query: $\Gamma \models p(a)$?

Clausal form for $\Gamma \cup \{\neg p(a)\}$:

$$\{\{p(X), \neg q(X)\}, \{q(a)\}, \{\neg p(a)\}\}$$

Resolution for First Order Logic

Resolution proof for $\Gamma \cup \{\neg p(a)\}$ (two resolution steps):

1. $\{\neg p(a)\}, \{p(X), \neg q(X)\} \vdash$
   $\{\neg q(a)\}$ ($\sigma_1 = \{a/X\}$)

2. $\{\neg q(a)\}, \{q(a)\} \vdash \Box$

We obtain a resolution proof, that uses the substitution

$$\sigma_1 = \{a/X\}$$

that is actually serving to identify a particular case of the rule $\forall X\ (p(X) \leftarrow q(X))$ that generates the answer we are looking for: $p(a)$

Resolution for FO Logic

Example: Given the formula Γ

1. $\mathrm{ancestor}(\mathrm{jaime}, \mathrm{juan})$.
2. $\forall X_1 \forall Y_1\ (\mathrm{ancestor}(\mathrm{father}(X_1), Y_1) \leftarrow \mathrm{ancestor}(X_1, Y_1))$.

Query: $\Gamma \models \exists X_2 \exists Y_2\ \mathrm{ancestor}(X_2, Y_2)$?

In this example, we have many resolution proofs (actually infinite) associated with different answers for the query !

Resolution for FO Logic

Many resolution proofs, many answers:

- jaime is ancestor (one resolution step):

  1. $\{\neg\text{ancestor}(X_2, Y_2)\}, \{\text{ancestor}(\text{jaime}, \text{juan})\}$
     $\vdash \ \square$ with $\sigma_1 = \{\text{jaime}/X_2, \text{juan}/Y_2\}$

  Answer: $\sigma_1 = \{\text{jaime}/X_2, \text{juan}/Y_2\}$

- father(jaime) is ancestor (two resolution steps):

  1. $\{\neg\text{ancestor}(X_2, Y_2)\}, \{\text{ancestor}(\text{father}(X_1), Y_1), \neg\text{ancestor}(X_1, Y_1)\}$
     $\vdash \ \{\neg\text{ancestor}(X_1, Y_1)\}$ with $\sigma_1 = \{\text{father}(X_1)/X_2, Y_1/Y_2\}$

  2. $\{\neg\text{ancestor}(X_1, Y_1)\}, \{\text{ancestor}(\text{jaime}, \text{juan})\}$
     $\vdash \ \square$ with $\sigma_2 = \{\text{jaime}/X_1, \text{juan}/Y_1\}$

  Answer: $\sigma_1 \cdot \sigma_2 = \{\text{father}(\text{jaime})/X_2, \text{juan}/Y_2\}$

Resolution for FO Logic

- father(father(jaime)) is ancestor (three resolution steps):

1. $\{\neg \text{ancestor}(X_2, Y_2)\}, \{\text{ancestor}(\text{father}(X_1), Y_1), \neg \text{ancestor}(X_1, Y_1)\}$
   $\vdash \{\neg \text{ancestor}(X_1, Y_1)\}$ with $\sigma_1 = \{\text{father}(X_1)/X_2, Y_1/Y_2\}$

2. $\{\neg \text{ancestor}(X_1, Y_1)\}, \{\text{ancestor}(\text{father}(X_3), Y_3), \neg \text{ancestor}(X_3, Y_3)\}$
   $\vdash \{\neg \text{ancestor}(X_3, Y_3)\}$ with $\sigma_2 = \{\text{father}(X_3)/X_1, Y_3/Y_1\}$

3. $\{\neg \text{ancestor}(X_3, Y_3)\}, \{\text{ancestor}(\text{jaime}, \text{juan})\}$
   $\vdash \square$ with $\sigma_3 = \{\text{jaime}/X_3, \text{juan}/Y_3\}$

Answer: $\sigma_1 \cdot \sigma_2 \cdot \sigma_3 = \{\text{father}(\text{father}(\text{jaime}))/X_2, \text{juan}/Y_2\}$
Notice that in this resolution proof we have used a same clause two times (clause 2), renaming its set of variables in the second application.

Resolution for FO Logic

- father(father(father(jaime))) is ancestor: the resolution proof is similar to the previous one, but now using clause 2 three times
- father(father(father(father(jaime)))) is ancestor: the resolution proof is similar to the previous one, but now using clause 2 four times
- ...

This example has actually infinite answers, but we can find all of them using First Order Resolution in an ordered manner (in particular, they can be found using SLD-Resolution)

Resolution for FO Logic

Example: A formula Γ about the power of love:

1. $\mathrm{i\_love\_you}(d_0)$.
2. $\forall X_1\ \mathrm{geq}(X_1, d_0)$.
3. $\forall Y_2 \forall X_2\ \mathrm{i\_love\_you}(Y_2) \leftarrow \mathrm{geq}(Y_2, X_2), \mathrm{i\_love\_you}(X_2)$.

Clausal form (all the variables universally quantified):

$$\begin{array}{l} \{\{\mathrm{i\_love\_you}(d0)\}, \\ \{\mathrm{geq}(X_1, d0)\}, \\ \{\mathrm{i\_love\_you}(Y_2), \neg\mathrm{geq}(Y_2, X_2), \neg\mathrm{i\_love\_you}(X_2)\}\} \end{array}$$

What I would like to check:

$$\Gamma \models \forall Y_3\ \mathrm{i\_love\_you}(Y_3)$$

Will I always love you ?

Resolution for First Order Logic

So we have to check the unsatisfiability of
$\Gamma \cup \{\neg\forall Y_3 \text{ i\_love\_you}(Y_3)\}$:

$$\begin{array}{l}\{\{\text{i\_love\_you}(d0)\},\\ \{\text{geq}(X_1, d0)\},\\ \{\text{i\_love\_you}(Y_2), \neg\text{geq}(Y_2, X_2), \neg\text{i\_love\_you}(X_2)\},\\ \{\neg\forall Y_3 \text{ i\_love\_you}(Y_3)\}\ \}\end{array}$$

A little problem about the last clause:

$$\neg\forall Y_3 \text{ i\_love\_you}(Y_3) \equiv \exists Y_3 \neg\text{i\_love\_you}(Y_3)$$

but this is not an universally quantified clause (it is existentially quantified)

So, we cannot perform resolution with this clause.

Resolution for First Order Logic

But we can transform the last clause, call it C, to a clause C′ that will preserve the satisfiability (or unsatisfiability) of $\Gamma \cup C$

Generate a new constant symbol (it does not appear in any of the clauses of Γ) call it c, then:

$$C' = \{\neg i\_love\_you(c)\}$$

This generated constant c will act as a "witness" for the possible proof for the existence of such an object (an object c that satisfies $\neg$i_love_you(c)

This constant is called an Skolem constant

Resolution for First Order Logic

Resolution proof for $\Gamma \cup \{\neg \text{i\_love\_you}(c)\}$:

1. $\{\neg \text{i\_love\_you}(c)\}, \{\text{i\_love\_you}(Y_2), \neg \text{geq}(Y_2, X_2), \neg \text{i\_love\_you}(X_2)\}$
   $\{\neg \text{geq}(c, X_2), \neg \text{i\_love\_you}(X_2)\}$ $(\sigma_1 = \{c/Y_2\})$

2. $\{\neg \text{geq}(c, X_2), \neg \text{i\_love\_you}(X_2)\}, \{\text{geq}(X_1, d0)\} \vdash$
   $\{\neg \text{i\_love\_you}(d_0)\}$ $(\sigma_2 = \{c/X_1, d_0/X_2\})$

3. $\{\neg \text{i\_love\_you}(d_0)\}, \{\text{i\_love\_you}(d_0)\} \vdash \Box$

Observe that in every resolvent, one of the clauses is the resolvent of the previous resolution step, and we start with our goal clause $\{\neg \text{i\_love\_you}(c)\}$

The Power of First Order Logic versus Propositional Logic

The previous example shows a very powerful feature of FO Logic: we can express, and prove, properties satisfied by sets of infinite size, because the previous proof shows that the predicate i_love_you is satified by any object of the domain of a model of Γ, even for an infinite domain $\{d_0, d_1, d_2, \ldots\}$:

$$\Gamma \models \{\mathrm{i\_love\_you}(d_0), \mathrm{i\_love\_you}(d_1), \mathrm{i\_love\_you}(d_3), \ldots\}$$

This has been proved with a single formula, and a single resolution proof !

Can you do the same with Propositional logic ?

When $\Gamma \cup \neg G$ is satisfiable

When G is not a logical consequence of Γ, so there is no resolution proof for $\neg G$, that means that there is at least a model of Γ where $\neg G$ is true.

For example, here:

1. even_number(0).
2. $\forall X$ even_number(two_more(X)) $\leftarrow$ even_number(X).

Query G: $\forall Y$ even_number(Y)

We do not have a resolution proof of $\Gamma \cup \neg G$. Check it !

When $\Gamma \cup \neg G$ is satisfiable

What is a Herbrand model for $\Gamma \cup \neg G$?

$\neg G \equiv \neg\forall Y\, \text{even\_number}(Y) \equiv \exists Y \neg\text{even\_number}(Y) \equiv \neg\text{even\_number}(w)$

where w is a new constant (an Skolem constant)
A Herbrand model for $\Gamma \cup \{\neg\text{even\_number}(w)\}$:

- $$\text{Domain} = \{0, w, \text{two\_more}(0), \text{two\_more}(w), \text{two\_more}(\text{two\_more}(0)), \text{two\_more}(\text{two\_more}(w)), \ldots\}$$
- even_number(X) true for $X = \{0, \text{two\_more}(0), \text{two\_more}(\text{two\_more}(0))\}$
- even_number(X) false for $X = \{w, \text{two\_more}(w), \text{two\_more}(\text{two\_more}(w))\}$

Is there any other Herbrand model for $\Gamma \cup \{\neg\text{even\_number}(w)\}$?