Denotation in FOL (cont…)

It means to indicate which element of D is *denoted* by a term in an interpretation $\mathcal{J} = \langle \mathrm{D}, \mathrm{I} \rangle$.

1. If a term does not contain any variable

$$\|bestFriend(johnSmith)\|_{\mathcal{J}} = \mathrm{I}[bestFriend](\mathrm{I}[johnSmith])$$

2. If a term contains variables, we first define μ – a variable assignment over D. For a variable x, $\mu(x) \in \mathrm{D}$.

Denotation in FOL

It means to indicate which element of D is denoted by a term in an interpretation $\mathcal{J} = <\mathrm{D},\mathrm{I}>$.

1. If a term does not contain any variable

   $\|bestFriend(johnSmith)\|_{\mathcal{J}}=\mathrm{I}[bestFriend](\mathrm{I}[johnSmith])$

2. If a term contains variables, we first define μ – a variable assignment over D.
   For a variable x, $\mu(x) \in \mathrm{D}$.

The denotation of a term t, given $\mathcal{J}$ and μ, is written $\|t\|_{\mathcal{J},\mu}$ and is defined by the rules:

a) If x is a variable, then $\|x\|_{\mathcal{J},\mu}=\mu(x)$

b) If $t_1,\ldots,t_n$ are terms and f is a function symbol with n arguments, then
   $\|f(t_1,\ldots,t_n)\|_{\mathcal{J},\mu}=\mathrm{I}[f](\|t_1\|_{\mathcal{J},\mu},\ldots,\|t_n\|_{\mathcal{J},\mu})$

Example: $\|bestFriend(x)\|_{\mathcal{J},\mu}= \mathrm{I}[bestFriend](\mu(x))$
assuming that $\mu(x)=john$ then the denotation is $\mathrm{I}[bestFriend](john)$.

Obs. The denotation of a term is an element of D.

Satisfaction

We can say which formulas in FOL are true and which are false, according to an interpretation $\mathcal{I}$ and a variable assignment μ.

For example, Dog(bestFriend(johnSmith)) is true in $\mathcal{I}$ iff:

1. We use I to obtain the subset of D denoted by “Dog”
2. Find the object in D denoted by “bestFriend(johnSmith)”
3. The object found at step 2 belongs to the subset found at step 1.

Satisfaction

Given $\mathcal{I}$ and μ, we say that the formula α is satisfied in $\mathcal{I}$, μ, and we write $\mathcal{I},\mu \models \alpha$ according to the following rules:

1. $\mathcal{I},\mu \models P(t_1,\ldots, t_n)$ iff $<\|t_1\|_{\mathcal{I},\mu},\ldots,\|t_n\|_{\mathcal{I},\mu}> \in I[P]$;
2. $\mathcal{I},\mu \models t_1=t_2$ iff $\|t_1\|_{\mathcal{I},\mu}$ and $\|t_2\|_{\mathcal{I},\mu}$ are the same element of D;

3. $\mathcal{I},\mu \models \neg\alpha$ iff it is not the case that $\mathcal{I},\mu \models \alpha$ (noted as $\mathcal{I},\mu \not\models \alpha$);
4. $\mathcal{I},\mu \models \alpha\wedge\beta$ iff $\mathcal{I},\mu \models \alpha$ and $\mathcal{I},\mu \models \beta$;
5. $\mathcal{I},\mu \models \alpha\vee\beta$ iff $\mathcal{I},\mu \models \alpha$ or $\mathcal{I},\mu \models \beta$;

6. $\mathcal{I},\mu \models \exists x.\alpha$ iff $\mathcal{I},\mu' \models \alpha$ for some variable assignment μ′ that differs from μ on at most x;

   (example: $\mathcal{I},\mu \models \exists x.P(x,y)$ iff $\mathcal{I},\mu' \models P(x,y)$, which is equiv. to $<\mu'(x),\mu'(y)> \in I[P]$)

7 . $\mathcal{I},\mu \models \forall x.\alpha$ iff $\mathcal{I},\mu' \models \alpha$ for every variable assignment μ′ that differs from μ on at most x;

Satisfaction

- If α is a sentence (i.e. a formula without free variables), satisfaction does not depend on the given μ. We write $\mathscr{I} \models \alpha$ and read "α is true in the interpretation $\mathscr{I}$".

- For the propositional subset of FOL, we write I[α]=1 or 0 according to whether $\mathscr{I} \models \alpha$ or not.

- If S is a set of sentences, we write $\mathscr{I} \models S$ if all the sentences in S are true in $\mathscr{I}$. We say that $\mathscr{I}$ is a logical model of S.

The pragmatics in FOL

It specifies how well-formed expressions are to be used. To be able to reason, concepts like “Dog”, “DemocraticCountry” should be given an intended interpretation.

Logical consequence

Although the interpretation of the nonlogical symbols defines the semantic interpretation in FOL, there are still connections between sentences that do not depend on the meaning of those symbols.

Example: assume that α and β are sentences in FOL.

Let $\gamma=\neg(\beta\wedge\neg\alpha)$.

If $\mathcal{I}$ is an interpretation where α is true then γ is true under $\mathcal{I}$ and its truth value does not depend on how we understand the nonlogical symbols in α and β.
We say that α logically entails γ, or γ is a logical consequence of α.

The pragmatics in FOL

For a set of sentences S and a sentence α, we say that α is a logical consequence of S (or S logically entails α) iff for every interpretation $\mathscr{I}$ with $\mathscr{I} \models S$ then $\mathscr{I} \models \alpha$.
Or equivalently, there is no interpretation $\mathscr{I}$ where $\mathscr{I} \models S \cup \{\neg\alpha\}$.
We write $S \models \alpha$.

A sentence α is logically valid, and we write $\models \alpha$, if it is a logical consequence of the empty set (i.e. α is valid iff $\forall\mathscr{I}$, $\mathscr{I} \models \alpha$).

If $S=\{\alpha_1,\ldots, \alpha_n\}$ finite and α is a sentence, then $S \models \alpha$ iff $[(\alpha_1\wedge\ldots\wedge\alpha_n) \supset \alpha]$ is valid.

The pragmatics in FOL

The logical entailment is the key of a knowledge-based system.

For example, if Fido is a dog, then a reasoning system should be able to conclude that Fido is a mammal.

If a set of sentences S entails a sentence α, then α is true in every interpretation where S is true. Other sentences that are not entailed by S may or may not be true, but a knowledge-based system must conclude that the entailed sentences are true.

The pragmatics in FOL

The logical entailment is the key of a knowledge-based system.

For example, if Fido is a dog, then a reasoning system should be able to conclude that Fido is a mammal.

If a set of sentences S entails a sentence α, then α is true in every interpretation where S is true. Other sentences that are not entailed by S may or may not be true, but a knowledge-based system must conclude that the entailed sentences are true.

If we have an interpretation $\mathcal{I}$ where Dog(fido) is true, then the system can conclude that ¬¬Dog(fido) and (Dog(fido)∨Happy(john)) are true. These conclusions are logically safe but this is not the kind of reasoning we would be interested in.

Something more useful would be if a system concludes from Dog(fido) that Mammal(fido).

The pragmatics in FOL

We can find an interpretation where Dog(fido) is true and Mammal(fido) is false.

For example,

$\mathcal{I}$=<D,I>

D={d}

I[Dog]={d}

I[P]={} for every other predicate P≠Dog.

I[f](d,…,d)=d

We have $\mathcal{I} \models$ Dog(fido) but $\mathcal{I} \models \neg$Mammal(fido).

The pragmatics in FOL

So, there is no logical connection between the two sentences. To create it, we need to include in S a statement that connects the nonlogical symbols involved:

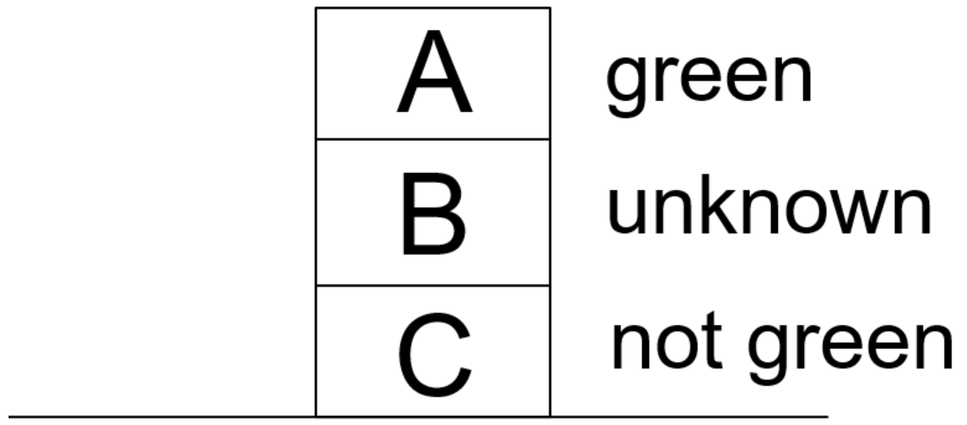
$$\forall x.Dog(x) \supset Mammal(x)$$

Thus, Mammal(x) becomes the logical consequence of Dog(x) and we rule out all the interpretations where the set of dogs is not included into the set of mammals.

“Truth in the intended interpretation”

Reasoning based on logical consequence allows only safe, logically guaranteed conclusions in a knowledge-based system.

Exercise 1



Question: Is there a green block directly on top of a non-green one?

Formalization in FOL

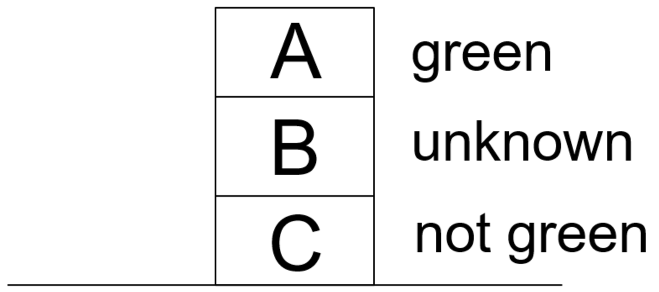
- a, b, c the names of the blocks
- G unary predicate symbol for “green”
- O binary predicate symbol for “on”

S={O(a,b), O(b,c), G(a), ¬G(c)}}

The sentence α is $\exists x \exists y. G(x) \wedge \neg G(y) \wedge O(x,y)$.

We want to prove that $S \models \alpha$.

Exercise 1



Let $\mathcal{I}$ be a logical model for S, $\mathcal{I} \models S$

$$\left.\begin{array}{r}\text{1. Suppose that } \mathcal{I} \models G(b) \\ \neg G(c),\ O(b,c) \in S\end{array}\right\} \Rightarrow \mathcal{I} \models G(b) \wedge \neg G(c) \wedge O(b,c)$$

$$\Rightarrow \mathcal{I} \models \exists x \exists y.\ G(x) \wedge \neg G(y) \wedge O(x,y)$$

$$\left.\begin{array}{r}\text{2. Suppose that } \mathcal{I} \models \neg G(b) \\ G(a),\ O(a,b) \in S\end{array}\right\} \Rightarrow \mathcal{I} \models G(a) \wedge \neg G(b) \wedge O(a,b)$$

$$\Rightarrow \mathcal{I} \models \exists x \exists y.\ G(x) \wedge \neg G(y) \wedge O(x,y)$$

Thus, α is a logical consequence of S.

Exercise 2 - The barber’s paradox
(formulated by Bertrand Russell)

Anyone who does not shave himself must be shaved by the barber.

Whomever the barber shaves, must not shave himself.

Show that no barber can fulfill these requirements (i.e. there is no interpretation where both sentences are true).

Exercise 2 - The barber’s paradox
(formulated by Bertrand Russell)

Anyone who does not shave himself must be shaved by the barber.

Whomever the barber shaves, must not shave himself.

Show that no barber can fulfill these requirements (i.e. there is no interpretation where both sentences are true).

$$\left[\begin{array}{l}\forall x.\neg \mathrm{Shave}(x,x) \supset \mathrm{Shave}(\mathrm{barber},x)\\ \forall x.\mathrm{Shave}(\mathrm{barber},x) \supset \neg \mathrm{Shave}(x,x)\end{array}\right.$$

Let $\mathcal{I}$ be an interpretation such that

$\mathcal{I} \models \forall x.\neg \mathrm{Shave}(x,x) \supset \mathrm{Shave}(\mathrm{barber},x) \Rightarrow$

$\mathcal{I} \models \neg \mathrm{Shave}(\mathrm{barber},\mathrm{barber}) \supset \mathrm{Shave}(\mathrm{barber},\mathrm{barber})$

Similarly, $\mathcal{I} \models \mathrm{Shave}(\mathrm{barber},\mathrm{barber}) \supset \neg \mathrm{Shave}(\mathrm{barber},\mathrm{barber})$

[recall that $\alpha \supset \beta$ is $\neg\alpha \vee \beta$]

It follows that $\mathcal{I} \models \mathrm{Shave}(\mathrm{barber},\mathrm{barber})$ and

$\mathcal{I} \models \neg \mathrm{Shave}(\mathrm{barber},\mathrm{barber})$

contradiction

Reasoning in FOL

There is no automated procedure in FOL to decide in all cases whether a sentence is entailed or not from others.

A reasoning process is

- Logically sound if whenever it produces α, then α is guaranteed to be a logical consequence.
- Logically complete if it is guaranteed to produce α whenever α is entailed.

Expressing knowledge – creating a knowledge base

Knowledge engineering – is the first step when creating a knowledge base – and it means deciding on the representation language followed by determining the kind of objects important to the agent, the properties those objects have and the relationships among them.

Expressing knowledge – creating a knowledge base

Knowledge engineering – is the first step when creating a knowledge base – and it means deciding on the representation language followed by determining the kind of objects important to the agent, the properties those objects have and the relationships among them.

Vocabulary

We start by identifying the essential entities in the agent’s world:

- Constant symbols
    - Persons: johnSmith
    - Institutions: government
    - Places: centralStation
- Description of the basic types of objects: Person(x), Country(x), Restaurant(x)
- The set of attributes of objects: Rich, Nice, Smart
- Express relationships: DaughterOf(anna,mary), MarriedTo(anna,john)
- Functions: bestFriendOf(john), firstChildOf(anna,john)

Expressing knowledge – creating a knowledge base

Basic Facts

They are represented by atomic sentences and negations of atomic sentences ($P(t_1,\ldots,t_n)$ and $t_1=t_2$)

- Man(john), Rich(mary), WorksFor(john,george)
- ¬HappilyMarried(john)
- bestFriendOf(john)=george
- y≠anna

Expressing knowledge – creating a knowledge base

Complex facts

Connectors are used to express various beliefs

$\forall y[Rich(y) \wedge Man(y) \supset Loves(y,mary)]$

$\forall y[Woman(y) \wedge y \neq jane \supset Loves(y,john)]$

We can express general facts

$\forall x \forall y[Loves(x,y) \supset \neg Blackmails(x,y)]$

(or $\neg \exists x \exists y[Loves(x,y) \wedge Blackmails(x,y)]$)

or incomplete knowledge

$Loves(jane,john) \vee Loves(jane,jim)$

$\exists x[Adult(x) \wedge Blackmails(x,john)]$

Expressing knowledge – creating a knowledge base

Relationships among predicates

If john is Man then Woman(john) should be false.

If MarriedTo(anna,john) is true then MarriedTo(john,anna) should be true.

But a KB does not generate by itself such inferences. We need to provide a set of facts about the terminology we are using.

- Disjointness – the assertion of one implies the negation of the other

  $\forall x[Man(x) \supset \neg Woman(x)]$

- Subtypes

  $\forall x[Surgeon(x) \supset Doctor(x)]$

Expressing knowledge – creating a knowledge base

- Exhaustiveness – two or more subtypes completely account for a supertype

  $\forall x[\mathrm{Adult}(x) \supset (\mathrm{Man}(x) \vee \mathrm{Woman}(x))]$

- Symmetry

  $\forall x \forall y[\mathrm{MarriedTo}(x,y) \supset \mathrm{MarriedTo}(y,x)]$

- Type restrictions – the arguments of a predicate are of certain types

  $\forall x \forall y[\mathrm{MarriedTo}(x,y) \supset \mathrm{Person}(x) \wedge \mathrm{Person}(y)]$

- Full definitions – predicates that are completely defined by a logical combination of other predicates

  $\forall x[\mathrm{RichMan}(x) \equiv \mathrm{Rich}(x) \wedge \mathrm{Man}(x)]$

Expressing knowledge – creating a knowledge base

Entailments

It means to derive implicit conclusions from explicit knowledge in KB.

Example 1

1. Rich(john)
2. Man(john)
3. $\forall y[Rich(y) \wedge Man(y) \supset Loves(y,jane)]$
4. john=ceoOf(insuranceCompany)
5. Company(insuranceCompany)

Question: Is there a company whose CEO loves Jane?

In FOL, $\exists x[Company(x) \wedge Loves(ceoOf(x),jane)]$

Expressing knowledge – creating a knowledge base

Let $\mathcal{J}$ an interpretation that is a logical model for KB.

$$\left.\begin{array}{l} \mathcal{J} \text{ satisfies 1,2,3 from KB} \Rightarrow \mathcal{J} \models \text{Loves(john,jane)} \\ \qquad\qquad\qquad\qquad\qquad\quad \mathcal{J} \models \text{john=ceoOf(insuranceCompany)} \end{array}\right\} \Rightarrow$$

$$\left.\begin{array}{l} \mathcal{J} \models \text{Loves(ceoOf(insuranceCompany),jane)} \\ \mathcal{J} \models \text{Company(insuranceCompany)} \end{array}\right\} \Rightarrow$$

$\mathcal{J} \models \text{Company(insuranceCompany)} \wedge \text{Loves(ceoOf(insuranceCompany),jane)}$

$\Rightarrow \mathcal{J} \models \exists x\ [\text{Company}(x) \wedge \text{Loves(ceoOf}(x)\text{,jane)}]$

$\square$

Expressing knowledge – creating a knowledge base

Obs. KB $\models \alpha \supset \beta$ iff KB U $\{\alpha\} \models \beta$

Example 2

1. $\exists x[\text{Adult}(x) \wedge \text{Blackmails}(x,\text{john})]$
2. $\forall x[\text{Adult}(x) \supset (\text{Man}(x) \vee \text{Woman}(x))]$
3. $\text{Loves}(\text{john},\text{jane})$
4. $\forall y[\text{Woman}(y) \wedge y \neq \text{jane} \supset \text{Loves}(y,\text{john})]$
5. $\forall x \forall y[\text{Loves}(x,y) \supset \neg \text{Blackmails}(x,y)]$

Question: If no man blackmails John, then is he blackmailed by someone he loves?

Expressing knowledge – creating a knowledge base

Obs. KB $\models \alpha \supset \beta$ iff KB U $\{\alpha\} \models \beta$

Example 2

1. $\exists x[\text{Adult}(x) \wedge \text{Blackmails}(x,\text{john})]$
2. $\forall x[\text{Adult}(x) \supset (\text{Man}(x) \vee \text{Woman}(x))]$
3. $\text{Loves}(\text{john},\text{jane})$
4. $\forall y[\text{Woman}(y) \wedge y \neq \text{jane} \supset \text{Loves}(y,\text{john})]$
5. $\forall x \forall y[\text{Loves}(x,y) \supset \neg \text{Blackmails}(x,y)]$

Question: If no man blackmails John, then is he blackmailed by someone he loves?

In FOL, $\forall x[\text{Man}(x) \supset \neg \text{Blackmails}(x,\text{john})] \supset \exists y[\text{Loves}(\text{john},y) \wedge \text{Blackmails}(y,\text{john})]$

Let $\mathcal{I} \models$ KB and $\mathcal{I} \models \forall x[\text{Man}(x) \supset \neg \text{Blackmails}(x,\text{john})]$

We want to show that $\mathcal{I} \models \exists y[\text{Loves}(\text{john},y) \wedge \text{Blackmails}(y,\text{john})]$

Expressing knowledge – creating a knowledge base

$$\left.\begin{array}{l} \text{1. } \exists x[\mathrm{Adult}(x) \wedge \mathrm{Blackmails}(x,john)] \\ \text{2. } \forall x[\mathrm{Adult}(x) \supset (\mathrm{Man}(x) \vee \mathrm{Woman}(x))] \\ \quad \forall x[\mathrm{Man}(x) \supset \neg \mathrm{Blackmails}(x,john)] \end{array}\right\} \Rightarrow \text{6. } \mathcal{I} \vDash \exists x\, [\mathrm{Woman}(x) \wedge \mathrm{Blackmails}(x,john)]$$

$$\left.\begin{array}{l} \text{4. } \forall y[\mathrm{Woman}(y) \wedge y \neq jane \supset \mathrm{Loves}(y,john)] \\ \text{5. } \forall x \forall y[\mathrm{Loves}(x,y) \supset \neg \mathrm{Blackmails}(x,y)] \end{array}\right\} \Rightarrow \text{7. } \mathcal{I} \vDash \forall y[\mathrm{Woman}(y) \wedge y \neq jane \supset \neg \mathrm{Blackmails}(y,john)]$$

$$\left.\begin{array}{l} \text{from 6,7} \Rightarrow \mathcal{I} \vDash \mathrm{Blackmails}(jane,john) \\ \text{3. } \mathcal{I} \vDash \mathrm{Loves}(john,jane) \end{array}\right\} \Rightarrow$$

$\mathcal{I} \vDash \mathrm{Loves}(john,jane) \wedge \mathrm{Blackmails}(jane,john) \Rightarrow$

$\mathcal{I} \vDash \exists y[\mathrm{Loves}(john,y) \wedge \mathrm{Blackmails}(y,john)]$

□

Expressing knowledge – creating a knowledge base

Abstract individuals

In FOL we represent facts in a domain, but we can get greater flexibility in representation if we map objects onto predicates and functions.

Reification means to transform a sentence into an object, by creating new abstract individuals.

Expressing knowledge – creating a knowledge base

Abstract individuals

In FOL we represent facts in a domain, but we can get greater flexibility in representation if we map objects onto predicates and functions.

Reification means to transform a sentence into an object, by creating new abstract individuals.

For instance, we can say that John purchases a bike:

- Purchases(john,bike)
- Purchases(john,bike,oct12)
- Purchases(john,bike,oct12,1000RON) …

The arity of “Purchases” depends on the level of the details that we want to express.

Expressing knowledge – creating a knowledge base

Abstract individuals

We will consider “purchase” to be an abstract individual called, let’s say, p17. We can now describe this purchase using predicates and functions:

Purchase(p17)^agent(p17)=john^object(p17)=bike^
cost(p17)=1000RON^time(p17)=16

Instead of time(p17)=16 we can say
time(p17)=t19^hour(t19)=16^minute(t19)=23

The advantage now is that the arities of the predicate and function symbols are determined in advance.

Expressing knowledge – creating a knowledge base

Other types of facts

- Statistical and probabilistic facts
    - Half of the companies are profitable
    - Most of the students work
    - There are 10% chances that tomorrow will be sunny.

- Default and prototypical facts – usually true unless stated otherwise
    - Cars have four wheels.
    - Birds fly.

Expressing knowledge – creating a knowledge base

Exercise

KB

Tony, Mike and John belong to the Alpine Club. Every member of the Alpine Club who is not a skier is a mountain climber. Mountain climbers do not like rain, and anyone who does not like snow is not a skier. Mike dislikes whatever Tony likes, and likes whatever Tony dislikes. Tony likes rain and snow.

a) Represent the above sentences in FOL, using a consistent vocabulary (which you must define);
b) Prove that the sentences logically entail that there is a member of the Alpine Club who is a mountain climber but not a skier.