innovate achieve lead

BITS Pilani
Pilani Campus

CS/IS F214
Logic in Computer Science

MODULE: **PREDICATE LOGIC**

Semantics –

- Model Checking and Semantic Entailment

SEMANTICS: INTERPRETATION INVOLVING VARIABLES

Semantics: Interpretation: Variables

- Evaluating formulas with quantified variables:
  - e.g. $\forall \mathbf{X}\ \exists \mathbf{Y}\ \phi$,
- requires evaluation(s) of the sub-formula ϕ with
  - a value, say v1, used for all free occurrences of **X** and
  - a value, say v2, used for all free occurrences of **Y**
- How do we express this in our syntax?

Interpretation : Variables and Values

- Evaluating formula $\forall$**X** $\exists$**Y** ϕ requires evaluation(s) of the sub-formula ϕ with
  - a value, say v1, used for all free occurrences of **X** and
  - a value, say v2, used for all free occurrences of **Y**
- Can we express this as **ϕ[v1/X][v2/Y]**?
  - Is substitution defined on (semantic) values?
  - Recall that ϕ [t/X] was defined as

  replacing free occurrences of **X** *in* ϕ *with term* **t** *to get a new formula* **ϕ1**

    - What is the difference between this and the evaluation intended above?
- Using substitution in this context – for instance ϕ **[v1/X][v2/Y]** would only yield a new formula (i.e. not a value) !!
  - Why?

Semantics: Evaluation vs. Substitution

- Using substitution in the context of evaluation won't work!
  - e.g. consider the set of predicate logic formulas with
    - **F** = {***succ***, ***zero***} where ***zero*** is a constant and ***succ*** is a unary function, and **P** = { = }:
  - What are the formulas you can write in this language?
    - e.g. $\forall$**X** ϕ where ϕ is defined as $\neg$**(succ(X) = zero)**
- Now define a model for **F**,**P** : e.g.
  - The universe: *set of natural numbers*
  - ***zero*** has the meaning 0 (of the Platonic world)
  - ***succ*** has the meaning successor (of the Platonic world)
  - = has the usual meaning
- What is the difference between the substitution **ϕ[succ(succ(zero))/X]** and trying to evaluate ϕ when **X** has the value 2?
  - Why is **ϕ[2/X]** syntactically meaningless?

SEMANTICS: INTERPRETATION

Semantics: Interpretation : Need for Notation

- Evaluating a formula such as $\forall \mathbf{X}\ \exists \mathbf{Y}\ \phi$:

```
for each value v1 in U {
    for each value v2 in U {
        res = evaluate ϕ with X=v1 and Y=v2 ;
        if (res) then break; // get out of this loop
        else continue;
    }
    if  (res) then continue;
    else return false;
}
return true;
```

We need notation for this!

Interpretation: Lookup Tables - Definition

- We interpret formulas relative to an ***environment***:
  - i.e. *a context in which each variable has a specific value*
- We need to define an **environment**:
  - we will use a ***lookup table*** (i.e. *a function from variables to values*)
    - to denote that some variables have been assigned specific values.
- A ***look-up table*** (or **environment**) *is a function*:
  - ι : **Var --> A** where **Var** is the set of all variables (i.e. symbols) in a given formula and **A** is the universe in our (chosen) model.

Interpretation – Lookup Tables – Extension

- In evaluating our example formula ∀**X** ∃**Y** ϕ,
  - we can use the lookup-table ɩ where ɩ(X) = v1 and ɩ(Y) = v1

```
/* The evaluation will now look like this: */
for each value v1 in U {
    update ɩ such that ɩ(X) = v1 ;
    for each value v2 in U {
        update ɩ such that ɩ(Y) = v2
        res = evaluate ϕ with ɩ ;
        if (res) then break;
        else continue;
    }
    if (res) then continue;
    else return false;
}
return true;
```

We need notation for update(s) on a look-up table!

We use ɩ**[X |-> a]** to denote the look-table where variable **X** has been mapped to value **a** and any other variable **Y** to ɩ**(Y)**

Semantics: Interpretation: Notation

- Evaluating the formula $\forall$**X** $\exists$**Y** ϕ, requires evaluation(s) of the sub-formula ϕ with
  - a value, say v1, for all free occurrences of X and
  - a value, say v2, for all free occurrences of Y
- This evaluation can then be denoted as $\mathbf{M} \models_{\iota[X|\text{->}v1][Y|\text{->}v2]} \phi$
  - which is read as
    - ϕ *holds under the environment that maps* **X** *to* **v1** *and* **Y** *to* **v2** *in the model* **M**
      - i.e. ϕ *evaluates to true when* **X** *maps to* **v1** *and* **Y** *maps to* **v2** *in the model* **M**

Semantics: Model-Checks Relation

- Given a model **M** for a pair (**F**,**P**) and a given environment ι,

  we define the ***model-checks*** *relation* $\mathbf{M} \models_{\iota} \phi$ *for each formula* ϕ (by ***structural induction***):

  - **Induction Basis:**
    - ϕ is of the form $\mathbf{p(t_1, t_2, \ldots\, t_n\,)}$
      - for each **i** from **1** to **n**:
        - evaluate term $\mathbf{t_i}$ using **M** and ι to obtain value $\mathbf{a_i}$
      - $M \models_{\iota}$ $\mathbf{p(t_1, t_2, \ldots\, t_n\,)}$ holds iff $\mathbf{(a_1, a_2, \ldots\, a_n\,)}$ is in $\mathbf{p^M}$
        - where $\mathbf{p^M}$ is the meaning of **p** in model **M**

  …

Semantics – Evaluating Terms

- Define **evalTerm(t, M, ℓ)** by structural induction on the definition of terms:
  - case **t** is a constant **c** :
    - return $\mathbf{c_M}$
  - case **t** is a variable **X** :
    - return ℓ(**X**)
  - case **t** is a function term of the form $\mathbf{f(t_1, t_2, \ldots\ t_n)}$
    - return $\mathbf{f_M}$**(evalTerm(t_1,M, ℓ),**
      evalTerm(t_2,M, ℓ),
      …
      evalTerm(t_n,M, ℓ))

Semantics: Model-Checks Relation – Induction Step

- Given a model **M** for a pair (**F**,**P**) and a given environment ι,

  we define the model-checks relation $\mathbf{M} \models_{\iota} \phi$ *for each formula* ϕ (by ***structural induction***):

  - **Induction Basis:**
    - ϕ is of the form $\mathbf{p(t_1, t_2, \ldots t_n)}$

      ...
  - **Induction Step:**
    - ϕ is of the form $\forall \mathbf{X}\, \psi$:
      - $\mathbf{M} \models_{\iota} \phi$ holds <u>*iff*</u> $\mathbf{M} \models_{\iota[x|->a]} \psi$ holds for all **a** in **A**

        where **A** is the universe in **M**
    - ϕ is of the form $\exists \mathbf{X}\, \psi$
      - $\mathbf{M} \models_{\iota} \phi$ holds <u>*iff*</u> $\mathbf{M} \models_{\iota[x|->a]} \psi$ holds for some **a** in **A**

        where **A** is the universe in **M**

    ...

Semantics: Model-Checks Relation – Induction Step [2]

- Given a model **M** for a pair (**F**,**P**) and a given environment ι,

  we define the model-checks relation $\mathbf{M} \models_{\iota} \phi$ *for each formula* ϕ (by ***structural induction***):
  - **Induction Basis:**
    - ϕ is of the form $\mathbf{p(t_1, t_2, \ldots t_n)}$ **:** **…**
  - **Induction Step:**
    - ϕ is of the form $\forall \mathbf{X}\ \psi$: **…**
    - ϕ is of the form $\exists \mathbf{X}\ \psi$: **…**
    - ϕ is of the form $\neg\psi$:
      - $\mathbf{M} \models_{\iota} \phi$ holds iff $\mathbf{M} \models_{\iota} \psi$ does not hold

…

Semantics: Model-Checks Relation – Induction Step [3]

- Given a model **M** for a pair (**F**,**P**) and a given environment ι,
  we define the model-checks relation $\mathbf{M} \models_\iota \phi$ *for each formula* ϕ:
  - **Induction Basis:**
    - ϕ is of the form $\mathbf{p(t_1, t_2, \ldots t_n)}$: ...
  - **Induction Step:**
    - ϕ is of the form $\forall \mathbf{X}\ \psi$: ...
    - ϕ is of the form $\exists \mathbf{X}\ \psi$: ...
    - ϕ is of the form $\neg\psi$: ...
    - ϕ is of the form $\psi\mathbf{1} \wedge \psi\mathbf{2}$
      - $\mathbf{M} \models_\iota \phi$ holds ***iff*** $\mathbf{M} \models_\iota \psi\mathbf{1}$ holds and $\mathbf{M} \models_\iota \psi\mathbf{2}$ holds
    - ϕ is of the form $\psi\mathbf{1} \vee \psi\mathbf{2}$
      - $\mathbf{M} \models_\iota \phi$ holds ***iff*** $\mathbf{M} \models_\iota \psi\mathbf{1}$ holds or $\mathbf{M} \models_\iota \psi\mathbf{2}$ holds
    - ϕ is of the form $\psi\mathbf{1}$ **-->** $\psi\mathbf{2}$
      - $\mathbf{M} \models_\iota \phi$ holds ***iff*** $\mathbf{M} \models_\iota \psi\mathbf{2}$ holds whenever $\mathbf{M} \models_\iota \psi\mathbf{1}$ holds

    ...

Model-Checks Relation

- Given a model **M** for a pair (**F**,**P**) and a given environment ι, we define the model-checks relation $\mathbf{M} \models_{\iota} \phi$ for each formula ϕ :
  - ϕ is of the form $\mathbf{p(t_1, t_2, \ldots t_n)}$
    - evaluate each term $\mathbf{t_i}$ using **M** and ι to obtain $\mathbf{a_i}$;

      $\mathbf{M} \models_{\iota} \mathbf{p(t_1, t_2, \ldots t_n)}$ holds iff $\mathbf{(a_1, a_2, \ldots a_n)}$ is in $\mathbf{p^M}$
  - ϕ is of the form $\forall \mathbf{X}\, \psi$:
    - $\mathbf{M} \models_{\iota} \phi$ holds iff $\mathbf{M} \models_{\iota[x|\text{->}a]} \psi$ holds for all **a** in **A**
  - ϕ is of the form $\exists\, \mathbf{X}\, \psi$:
    - $\mathbf{M} \models_{\iota} \phi$ holds iff $\mathbf{M} \models_{\iota[x|\text{->}a]} \psi$ holds for some **a** in **A**
  - ϕ is of the form $\neg\psi$:
    - $\mathbf{M} \models_{\iota} \phi$ holds iff $\mathbf{M} \models_{\iota} \psi$ does not hold
  - ϕ is of the form $\psi\mathbf{1} \wedge \psi\mathbf{2}$:
    - $\mathbf{M} \models_{\iota} \phi$ holds iff $\mathbf{M} \models_{\iota} \psi\mathbf{1}$ holds and $\mathbf{M} \models_{\iota} \psi\mathbf{2}$ holds
  - ϕ is of the form $\psi\mathbf{1} \vee \psi\mathbf{2}$:
    - $\mathbf{M} \models_{\iota} \phi$ holds iff $\mathbf{M} \models_{\iota} \psi\mathbf{1}$ holds or $\mathbf{M} \models_{\iota} \psi\mathbf{2}$ holds
  - ϕ is of the form $\psi\mathbf{1}$ **-->** $\psi\mathbf{2}$:
    - $\mathbf{M} \models_{\iota} \phi$ holds iff $\mathbf{M} \models_{\iota} \psi\mathbf{2}$ holds whenever $\mathbf{M} \models_{\iota} \psi\mathbf{1}$ holds

Model-Checks Relation : Example

- Let $\boldsymbol{F}$ = { + } and $\boldsymbol{P}$ = { ≡ } and **M** a model for (***F***,***P***) be:

({ 0, 1, 2, 3, 4, 5, 6 }, addition modulo 7, congruent modulo 7)

- Check whether the following formula ϕ "model-checks":
  - $\forall X \ \forall Y \ \exists Z \ X + Y \equiv Z$
- $\mathbf{M} \ |=_{\{\}} \phi$ holds iff:
  - for each **j** from 0 to 6
  - for each **k** from 0 to 6
    - $\mathbf{M} \ |=_{\{ X \ |\text{->} \ j, \ Y \ |\text{->} \ k \}} \exists Z \ X + Y \equiv Z$ holds
  - i.e. $\mathbf{M} \ |=_{\{ X \ |\text{->} \ j, \ Y \ |\text{->} \ k, \ Z \ |\text{->} n \}} X + Y \equiv Z$ holds for some **n** = 0 to 6, for all **j** = 0 to 6, for all **k** = 0 to 6
  - i.e. $\mathbf{j} +_7 \mathbf{k} = \mathbf{n}$ **(mod 7)** holds for some **n** and for any 0<= **j** , **k** <= 6

Model-Checks Relation : Example (continued)

- Let F = { + } and P = { ≡ } and **M** a model for (F,P) be:

({ 0, 1, 2, 3, 4, 5, 6 }, addition modulo 7, congruent modulo 7)

- Check whether the following formula ϕ "model-checks":
  - **∀X ∀Y ∃Z X + Y ≡ Z**

```
modelChecks_M(ϕ) { // ϕ is ∀X ∀Y ∃Z X + Y ≡ Z
    for each j from 0 to 6 {
      for each k from 0 to 6 {
        for each n from 0 to 6 {
              res =  ((j+k)%7 == n);
              if (res) break;
        }
        if (!res) return FALSE;
        }
      if (!res) else return FALSE;
      }
    return TRUE;
```

Model Checking

- Exercise:
  - Write an algorithm to define the model-checks relation i.e. an algorithm MC
    - that takes a model M, a look-up table ι , and a predicate formula ϕ and
    - evaluates ϕ under M and ι.
  - [Hints:
    - Identify suitable representation i.e. data structures for:
      - ϕ (e.g. a parse tree)
      - M (a set of values for the universe, a map for symbols in ***F*** and ***P***)
      - ι
    - Follow the inductive definition of |= to write MC recursively.
    - What assumptions do you need to make this algorithm terminate?

  End of Hints]

SEMANTICS - SEMANTIC ENTAILMENT

Semantic Entailment

- Let Γ be (*a possibly infinite*) set of formulas and let ψ be a formula
  - in predicate logic using functions and predicates in F and P respectively.
- We define a ***semantic entailment*** relation as follows:
  - If ***M*** is a model of (**F,P**), then we say that

$$\Gamma \models^{M} \psi$$

i.e. Γ entails ψ under model ***M***

iff

for all environments (i.e. look-up tables) ι ,

M $\models_{\iota}$ ψ holds

whenever ***M*** $\models_{\iota}$ ϕ holds for all ϕ in Γ

Semantic Entailment

- Let Γ be (a possibly infinite) set of formulas and let ψ be a formula in predicate logic
  - using functions and predicates in **F** and **P** respectively.
- $\Gamma \models \psi$ holds ***iff***
  - $\Gamma \models^{M} \psi$ for all models ***M*** for **F** and **P**

Semantic Entailment

- Recall:
  - we had a potential infinity in evaluating a formula using quantifiers
    - if the universe is infinite,
      - then we cannot write an algorithm (that terminates) to evaluate a formula in predicate logic because
        - we may have to evaluate the formula for each of the values a variable can assume
- Now we have another infinity involved:
  - To understand the meaning of a formula - (i.e. *to evaluate a formula*):
    - *one has to evaluate the formula under all possible models*