Checking UML class diagrams instantiations

Giuseppe De Giacomo

Dipartimento di Ingegneria Informatica, Automatica e Gestionale
Sapienza Università di Roma

Outline

Outline

UML Class Diagram

An UML class diagram

- Captured by a finite set of logical axioms that describe universal properties (i.e., properties of all objects belonging to classes/associations).
- Represents intensional knowledge
- Corresponds to schema level information in database terms
- Corresponds to a set of constraints on class and association memberships
- Describes the semantics of the objects

UML class diagram instantiation

An UML class diagram instantiation, aka object diagram, describes properties of single objects or relationships between them.

- Assigns an extension to all classes, associations and attributes:
  - Classes are instantiated by sets of objects, i.e. unary relations;
  - Associations and attributes by binary or n-ary relations, involving classes;
  - Types are instantiated by sets of values, and are typically predefined: *String* is the set of all strings, *Integer* is the set of all integers, an so on;
  - Attributes are instantiated by binary relations, involving Types.
- Represents extensional knowledge.
- Corresponds to instance level information in database terms.
- Corresponds to a database in databases (though satisfying constraints!)
- Describes actual data

UML class diagram instantiation

We are going to formalize and study UML class diagram instantiations in two steps.

- first, we consider the case in which we have no ISA (or generalization) in the UML class diagram:
  - in this case, the UML class diagrams constructs act as constraints that must be satisfied in the instantiation for it to be correct.
- Then we consider the presence to ISAs (and generalizations)
  - In this case, we assume that the instantiation will be "partial" in that we do not explicit instantiate superclasses of subclasses. (Similarly association subsetting.); Notice this is what all Object Oriented Languages, i.e., Java, C++, etc., do.
  - In order to process such "partial" instantiations we need first to complete them using ISAs and Generalizations, this amount in a very simple form (polynomial) of reasoning.
  - Once completed through ISAs (and generalization), we can use all constructs as constraints that the completed instantiation must satisfy.

Outline

Instantiations of UML class diagram without ISAs

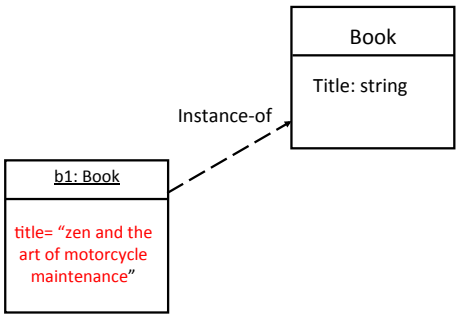
Instantiation of UML class diagrams without ISAs

In this case the instantiation of the diagram is a first order interpretation of the FOL theory corresponding to the diagram:

$$\mathcal{I} = (Obj^{\mathcal{I}}, C_1^{\mathcal{I}}, \ldots, C_n^{\mathcal{I}}, A_1^{\mathcal{I}}, \ldots, A_m^{\mathcal{I}}, T_1^{\mathcal{I}}, \ldots, T_\ell^{\mathcal{I}}, a_1^{\mathcal{I}}, \ldots, a_k^{\mathcal{I}}) \quad \text{where}$$

- $Obj^{\mathcal{I}}$ is the set of objects, all of which must appear in some class C_i, i.e., $Obj^{\mathcal{I}} = C_1^{\mathcal{I}} \cup \cdots \cup C_n^{\mathcal{I}}$. Notice:the actual domain of interpretation $\Delta^{\mathcal{I}}$ is $Obj^{\mathcal{I}} \cup T_1^{\mathcal{I}} \cup \cdots \cup T_\ell^{\mathcal{I}}$.
- For each class C and hence predicate $C(x)$ we have a set (unary relation) $C^{\mathcal{I}} \subseteq Obj^{\mathcal{I}}$.
- For each association A and hence predicate $A(x, y)$, we have a binary relation $A^{\mathcal{I}} \subseteq Obj^{\mathcal{I}} \times Obj^{\mathcal{I}}$, similarly or n-ary associations/predicates
- For each type T and hence predicate $T(x)$ we have a set (unary relation) of values $T^{\mathcal{I}}$. Notice: the interpretation of types is fixed (i.e. for strings $String^{\mathcal{I}} = String^{\mathcal{J}}$ is the set of all strings).
- For each attribute a binary predicate $a(x, y)$, we have a binary relation $a^{\mathcal{I}} \subseteq Obj^{\mathcal{I}} \times T^{\mathcal{I}}$ (where T is the type returned by attribute).

Example 1



Alphabet

$B(x), t(x, y), \text{String}(x)$

UML Class Diagram Axioms

$\forall x, y.t(x, y) \rightarrow B(x) \wedge \text{String}(y)$
$\forall x.B(x) \rightarrow 1 \leq \sharp\{y \mid t(x, y)\} \leq 1$

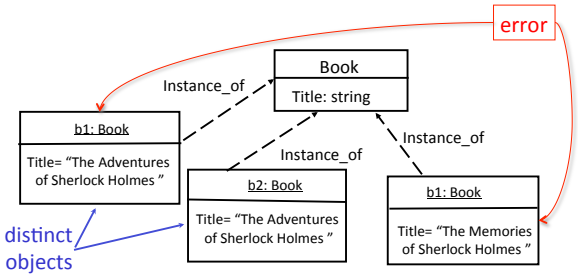
Instantiation

$\mathcal{I} = (Obj^{\mathcal{I}}, B^{\mathcal{I}}, String^{\mathcal{I}}, t^{\mathcal{I}})$

$B^{\mathcal{I}} = \{b1\}$
$t^{\mathcal{I}} = \{(b1, "zen")\}$
$String^{\mathcal{I}} = all\ strings$
$Obj^{\mathcal{I}} = \{b1\}$

After some examples, we will stop using object diagrams and directly use instantiations as FOL interpretations

Example 2



Alphabet

$B(x), t(x, y), \text{String}(x)$

UML Class Diagram Axioms

$\forall x, y.t(x, y) \rightarrow B(x) \wedge \text{String}(y)$
$\forall x.B(x) \rightarrow 1 \leq \sharp\{y \mid t(x, y)\} \leq 1$

Instantiation

$\mathcal{I} = (Obj^{\mathcal{I}}, B^{\mathcal{I}}, String^{\mathcal{I}}, t^{\mathcal{I}})$

$B^{\mathcal{I}} = \{b1, b2\}$
$t^{\mathcal{I}} = \{(b1, \text{“ash”}), (b2, \text{“ash”}), (b1, \text{“msh”})\}$
$String^{\mathcal{I}} = all\ strings$
$Obj^{\mathcal{I}} = \{b1, b2\}$

Is this instantiation correct? NO!

The second axiom is violated by $b1$: each x in B must have only one y in t.

Instantiations of UML class diagram without ISAs

In this setting, let Γ be (the set of FOL axioms corresponding to) the UML class diagram (notice there are no ISA assertions) and $\mathcal{I}$ an instantiation.

Correctness of the instantiation

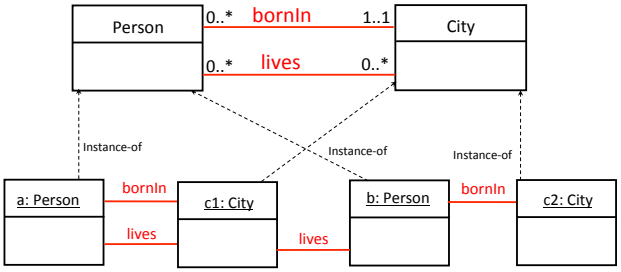
(All axioms in) Γ acts as constraints over $\mathcal{I}$.

$$\text{check } \mathcal{I} \models \Gamma$$

- if $\mathcal{I} \models \Gamma$ then $\mathcal{I}$ is a correct instantiation of the UML class diagram Γ
- Otherwise, $\mathcal{I}$ is not correct, and has to be discarded!

Remember: checking $\mathcal{I} \models \Gamma$ means checking if all axioms in Γ evaluate to true in the interpretation $\mathcal{I}$.

Example 3



Alphabet

$P(x), C(x), b(x,y), \ell(x,y)$

UML Class Diagram Axioms

$\forall x, y.b(x,y) \rightarrow P(x) \wedge C(y)$
$\forall x.P(x) \rightarrow 1 \leq \sharp\{y \mid b(x,y)\} \leq 1$
$\forall x, y.\ell(x,y) \rightarrow P(x) \wedge C(y)$

Instantiation

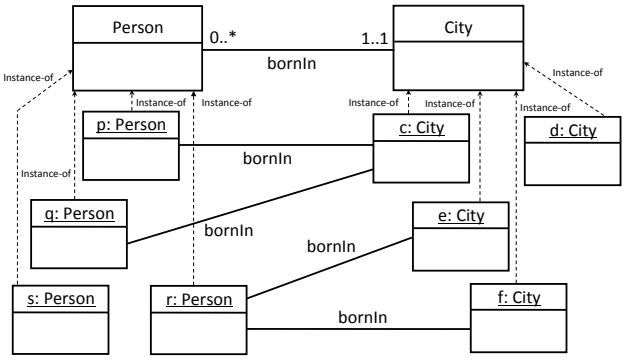
$\mathcal{I} = (Obj^{\mathcal{I}}, P^{\mathcal{I}}, C^{\mathcal{I}}, b^{\mathcal{I}}, \ell^{\mathcal{I}})$

$P^{\mathcal{I}} = \{a, b\}$
$C^{\mathcal{I}} = \{c1, c2\}$
$b^{\mathcal{I}} = \{(a, c1), (b, c2)\}$
$\ell^{\mathcal{I}} = \{(a, c1), (b, c1)\}$
$Obj^{\mathcal{I}} = \{a, b, c1, c2\}$

Is this instantiation correct? Yes!

All axioms are true in $\mathcal{I}$.

Example 4



Alphabet

$P(x), C(x), b(x, y)$

UML Class Diagram Axioms

$\forall x, y.b(x, y) \rightarrow P(x) \wedge C(y)$
$\forall x.P(x) \rightarrow 1 \leq \sharp\{y \mid b(x, y)\} \leq 1$

Instantiation

$\mathcal{I} = (Obj^{\mathcal{I}}, P^{\mathcal{I}}, C^{\mathcal{I}}, b^{\mathcal{I}})$

$P^{\mathcal{I}} = \{p, q, r, s\}$
$C^{\mathcal{I}} = \{c, d, e, f\}$
$b^{\mathcal{I}} = \{(p, c), (q, c), (r, e), (r, f)\}$
$\ell^{\mathcal{I}} = \{(a, c1), (b, c1)\}$
$Obj^{\mathcal{I}} = \{p, q, r, s, c, d, e, f\} (= P^{\mathcal{I}} \cup C^{\mathcal{I}})$

Is this instantiation correct? NO!

For both s and r the second axiom is false:

- s is not born anywhere
- r is born in two cities e, f

Querying instantiation of UML class diagram without ISAs

If the instantiation is correct with the UML class diagram we can query it using FOL queries being in fact a FOL interpretation.

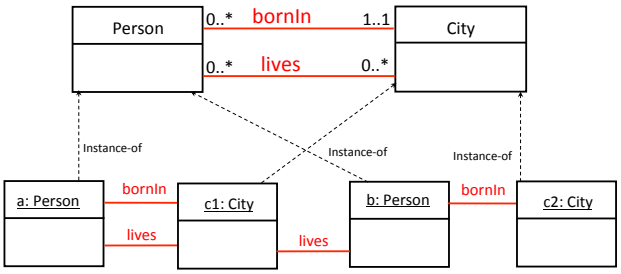
Querying

- if $\mathcal{I} \models \Gamma$, then compute q over $\mathcal{I}$ as in FOL ignoring Γ – like in for relational databases (NB no ISAs for now).
- Otherwise, querying $\mathcal{I}$ is meaningless, since $\mathcal{I}$ is NOT and instantiation of the UML class diagram Γ.

Notice that the constraints Γ that form the UML class diagram play no role in the evaluation, once we are assured that the instance is correct with respect Γ.

Example 5

Let's consider again Example 3.



Alphabet

$P(x), C(x), b(x, y), \ell(x, y)$

UML Class Diagram Axioms

$\forall x, y.b(x, y) \rightarrow P(x) \wedge C(y)$

$\forall x.P(x) \rightarrow 1 \leq \sharp\{y \mid b(x, y)\} \leq 1$

$\forall x, y.\ell(x, y) \rightarrow P(x) \wedge C(y)$

Instantiation

$\mathcal{I} = (Obj^{\mathcal{I}}, P^{\mathcal{I}}, C^{\mathcal{I}}, b^{\mathcal{I}}, \ell^{\mathcal{I}})$

$P^{\mathcal{I}} = \{a, b\}$

$C^{\mathcal{I}} = \{c1, c2\}$

$b^{\mathcal{I}} = \{(a, c1), (b, c2)\}$

$\ell^{\mathcal{I}} = \{(a, c1), (b, c1)\}$

$Obj^{\mathcal{I}} = \{a, b, c1, c2\}$

The instantiation is correct so we can query it. For example we may ask:

$q(x) \rightarrow \exists z, y.\ell(x, z) \wedge \ell(y, z) \wedge x \neq y$

The answer in this case would be $x = a, x = b$.

Notice in querying we ignore the UML class diagram axioms and consider only the instantiation, exactly as we do for databases. Querying is done by evaluating q over $\mathcal{I}$

Instead, asking queries to Example 4, which is not a correct instantiation, is meaningless!

Outline

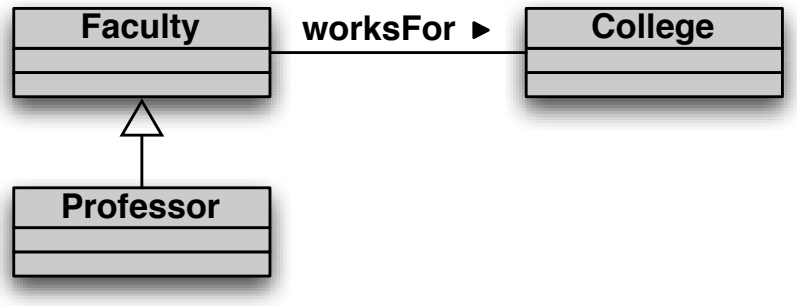
Complete Instantiation of UML class diagram ISAs

ISAs and complete instantiation

Given a complete instantiation ISAs (and hence the ISA part of generalizations) can be considered as additional constraints of the form $\forall x.A(x) \rightarrow B(x)$ requiring that each object in the extension of the subclass A is also in the extension of the class B.

Hence in case of a complete instantiation there is no difference in the treating UML class diagram with or without ISAs.

Example of ISA and complete instantiation



$\mathcal{I}$: $\text{Faculty}^{\mathcal{I}} = \{ \texttt{john}, \texttt{mary}, \texttt{paul} \}$
$\text{Professor}^{\mathcal{I}} = \{ \texttt{john}, \texttt{paul} \}$
$\text{College}^{\mathcal{I}} = \{ \texttt{collA}, \texttt{collB} \}$
$\text{worksFor}^{\mathcal{I}} = \{ (\texttt{john},\texttt{collA}),(\texttt{mary},\texttt{collB}) \}$

For each class and association we have a complete extension in the instantiation $\mathcal{I}$.

It is easy to check that $\mathcal{I}$ is a correct instantiation with respect to the UML class diagram!

Since the instantiation is correct we can query it, for example:
Query: $q(x) \leftarrow \exists y.\, \text{Professor}(x) \wedge \text{College}(y) \wedge \text{worksFor}(x, y)$

Answer: $\{ \texttt{john} \}$

Instantiations of UML class diagram with ISAs

However, often in presence of ISAs the instantiation is NOT complete:

- For every object it is assumed that the instantiation state explicitly which are the most specific classes it is instance of.
- Then, the fact that the objects belongs also to all superclasses is only implicitly assumed.
- Typical examples are instantiations in virtually all object-oriented programming languages, e.g., Java, C^{++}, C$^{\sharp}$, Objective C, etc.

Instantiations of UML class diagram with ISAs

This incomplete instantiations require some automated reasoning to be supported by the system in order correctly answer to `instanct-of` operators.

Incomplete instantiation and reasoning

The form of automate reasoning is pretty simple and is based on "modus ponens":

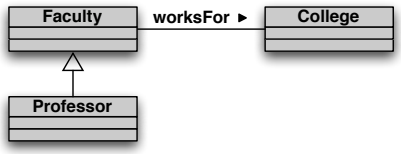
If a is an instance of A (ie, $A(a)$ is true in $\mathcal{I}$) and A ISA B (i.e., $\forall x.A(x) \rightarrow B(x)$), then a is also an instance of B (ie, $B(a)$ is true in $\mathcal{I}$).

Similarly for subset constraints between associations (ie, $\forall x, y.P(x, y) \rightarrow R(x, y)$).

In other words, the instantiation is (virtually) completed first by using all ISAs and then it is used as usual.

Example of ISA and incomplete instantiation

$\mathcal{I}$: Faculty$^{\mathcal{I}}$: {mary} – *incomplete!*
Professor$^{\mathcal{I}}$: { john, paul }
College$^{\mathcal{I}}$: { collA, collB }
worksFor$^{\mathcal{I}}$: { (john,collA),(mary,collB) }



This instantiation is incomplete!

- We know that mary belongs to Faculty and that john and paul belong to Professors.
- However, since we have that Professors ISA Faculty (ie, $\forall x.\text{Professors}(x) \rightarrow \text{Faculty}(x)$): we have (by reasoning - modus ponens) that john and paul belong also to Faculty.

So, the completed instantiation $\mathcal{I}$ is:
Faculty$^{\mathcal{I}}$ = {mary,john, paul}
Professor$^{\mathcal{I}}$ = { john, paul }
College$^{\mathcal{I}}$ = { collA, collB }
worksFor$^{\mathcal{I}}$:= { (john,collA),(mary,collB) }

Notice that the answer to the query $q(x) \leftarrow \text{Faculty}(x)$ of the complete instantiation is correctly: {mary, john, paul}, while over the incomplete one would be wrongly: {mary}.

Instantiations of UML class diagram with ISAs

To complete an incomplete instantiation we apply a chase procedure:

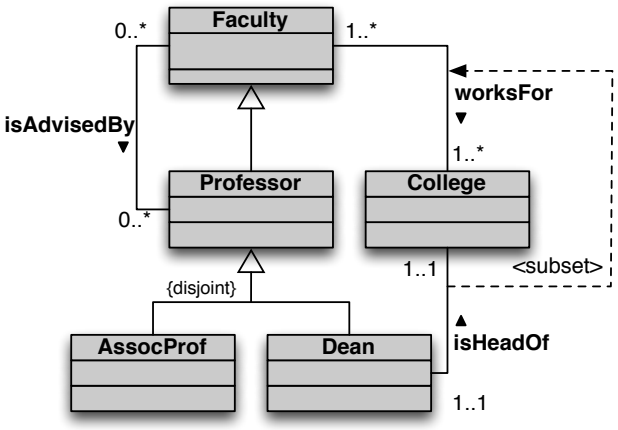
Chase of ISAs and subset constraints

Input: partial instantiation $\mathcal{I}$ and UML class diagram Γ
Output: completed $\mathcal{I}$

$\mathcal{I}_{old} = \emptyset$;
$\mathcal{I}_{new} = \mathcal{I}$;
while ($\mathcal{I}_{new}$ and $\mathcal{I}_{old}$ are different) **do** {
  $\mathcal{I}_{old} := \mathcal{I}_{new}$;
  for each ($\forall x.A(x) \rightarrow B(x)$ in Γ) **do**
    for each ($a \in A^{\mathcal{I}_{new}}$) **do**
      $B^{\mathcal{I}_{new}} := B^{\mathcal{I}_{new}} \cup \{a\}$;
    similarly for each subset constraint $\forall x, y.P(x, y) \rightarrow R(x, y)$ in Γ
}
$\mathcal{I} := \mathcal{I}_{new}$;
return $\mathcal{I}$

In other words the chase applies in all possible ways all ISA and subset constraints (the order of application is not relevant since each of the application grows the instantiation monotonically) and returns the resulting completed instantiation.

Exercise



UML class diagram

$\forall x.\text{Professor}(x) \rightarrow \text{Faculty}(x)$
$\forall x.\text{AssocProf}(x) \rightarrow \text{Professor}(x)$
$\forall x.\text{Dean}(x) \rightarrow \text{Professor}(x)$
$\forall x.\text{AssocProf}(x) \rightarrow \neg\text{Dean}(x)$
$\forall x, y.\text{isHeadOf}(x, y) \rightarrow \text{worksFor}(x, y)$
...

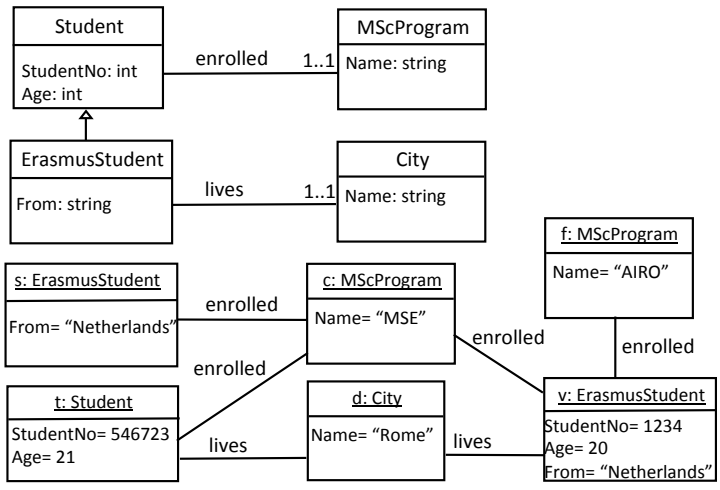
Instantiation

$\text{AssocProf}^{\mathcal{I}} : \{john, bill\}$
$\text{Dean}^{\mathcal{I}} : \{mary\}$
$\text{Professor}^{\mathcal{I}} : \{helen\}$
$\text{College}^{\mathcal{I}} : \{coll1\}$
$\text{isHeadOf}^{\mathcal{I}} : \{(mary, coll1)\}$
$\text{worksFor}^{\mathcal{I}} : \{(john, coll1), (bill, coll1), (helen, coll1)\}$

1. Complete the instantiation by chase of ISAs and subset constraints.
2. Is the (completed) instantiation correct with respect to the UML class diagram?
3. If so, answer the following queries:

$$q1(x) : \text{Faculty}(x)$$
$$q2(x, y) : \text{worksFor}(x, y)$$
$$q3(x) : \text{Dean}(x) \wedge \text{AssociateProf}(x)$$
$$q4(x) : \text{Professor}(x) \wedge \forall y.\text{College}(y) \rightarrow \text{worksFor}(x, y)$$

Another exercise



1. Generate the UML class diagram formalization in FOL and the incomplete instantiation reported in the object diagram.
2. Complete the instantiation by chase of ISAs and subset constraints.
3. Is the (completed) instantiation correct with respect to the UML class diagram?
4. If not change the (initial) instantiation to make it correct with respect to the UML class diagram.