Verifying Static Aspects of UML in PROLOG.

Anudeep Medishetti -112675646
Stony Brook University.

Paper: "Verifying Static Aspects of UML models using Prolog”

Authors: Feng Sheng, Huibiao Zhu, Zongyuan Yang, Jiaqi Yin, Gang Lu

Paper Overview:

- Consistency check in UML models ensure stability in coding phase.
- Static and Dynamic aspects of UML models.
- EMOF equivalent in Prolog.
- Horizontal, Semantic and Syntactic consistency

Prolog Implementation

Implementation in XSB

- Acyclic Generalization.
- Association Existence
- Generalization Satisfaction
- Class and Object Existence

Method of Implementation

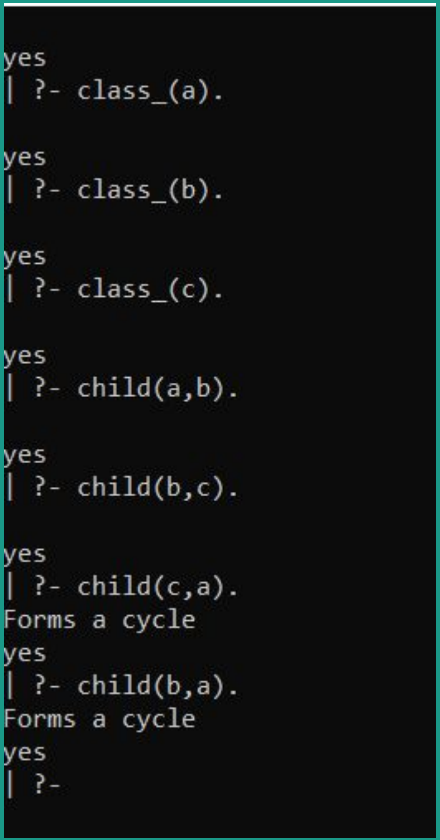
1. EMOF equivalent PROLOG clauses in XSB.

2. Rules defined in each clause to verify consistency checks.

3. User driven consistency checking process.

4. Storing and processing EMOF equivalents of classes, objects, associations, links etc in XSB.

Overview: Syntactic Consistency



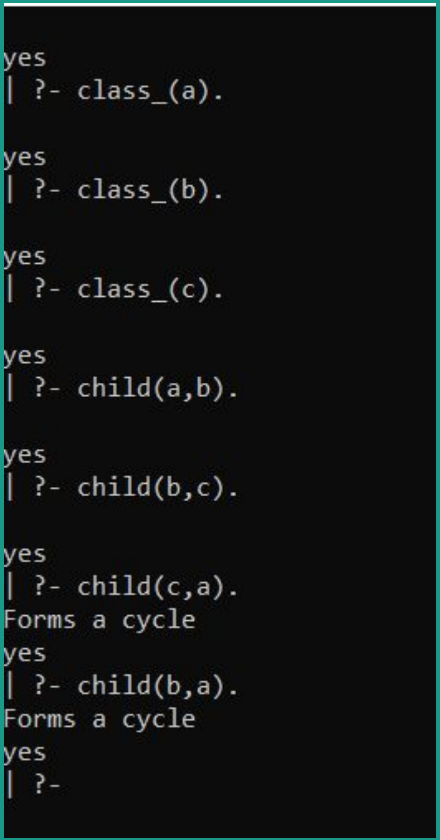
```
yes
| ?- class_(a).

yes
| ?- class_(b).

yes
| ?- class_(c).

yes
| ?- child(a,b).

yes
| ?- child(b,c).

yes
| ?- child(c,a).
Forms a cycle
yes
| ?- child(b,a).
Forms a cycle
yes
| ?-
```

Class Properties:

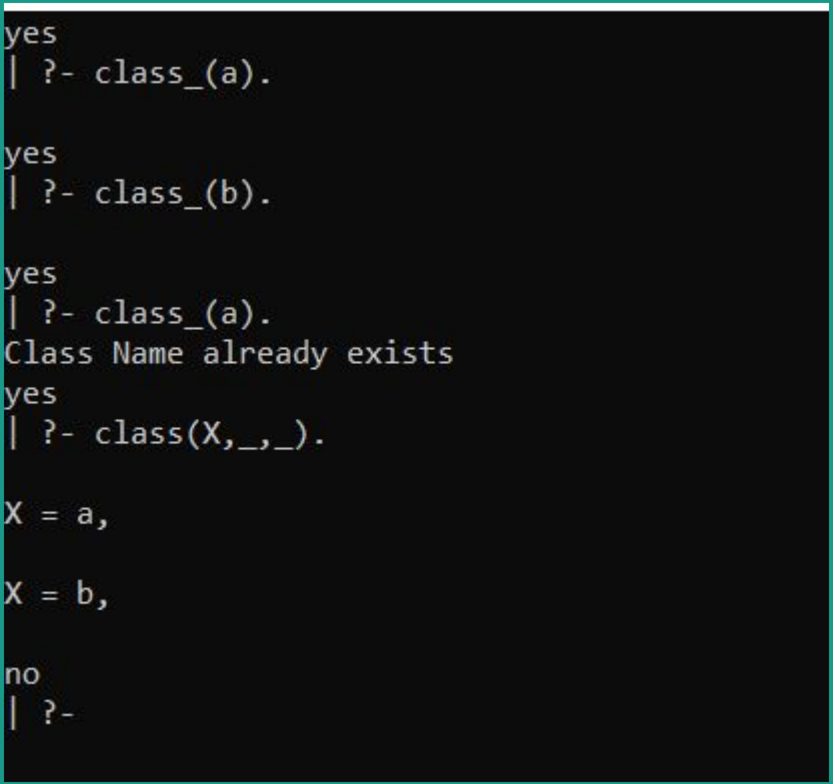
1. Class cannot be a child of itself.. directly or indirectly.
2. The adjacent consistency check shows, if it forms a cycle.

Class A child of Class B.
Class B child of Class C.
Class C can’t be child of A.

Overview: Syntactic Consistency

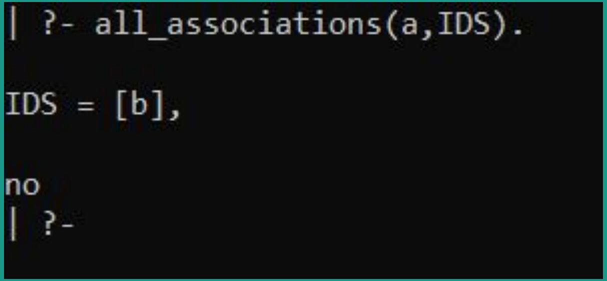


```
yes
| ?- class_(a).

yes
| ?- class_(b).

yes
| ?- class_(c).

yes
| ?- child(a,b).

yes
| ?- child(b,c).

yes
| ?- child(c,a).
Forms a cycle
yes
| ?- child(b,a).
Forms a cycle
yes
| ?-
```



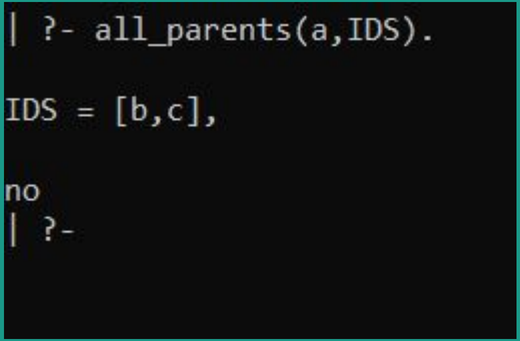
```
yes
| ?- class_(a).

yes
| ?- class_(b).

yes
| ?- class_(a).
Class Name already exists
yes
| ?- class(X,_,_).

X = a,

X = b,

no
| ?-
```

Overview: Horizontal Consistency



```
| ?- all_associations(a,IDS).

IDS = [b],

no
| ?-
```

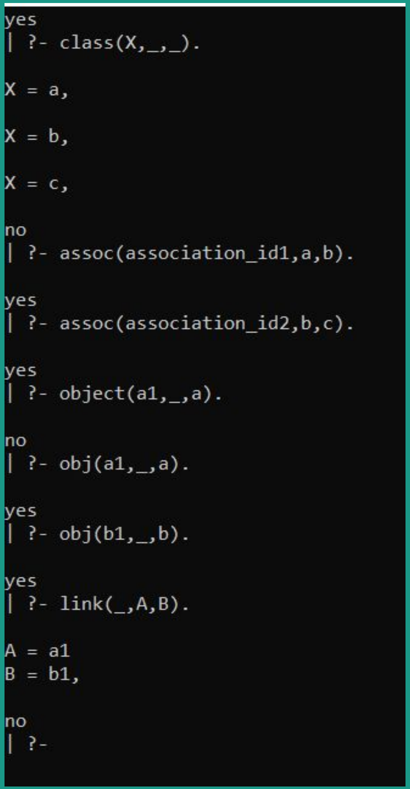
The adjacent screenshot shows associations of a class.



```
| ?- all_parents(a,IDS).

IDS = [b,c],

no
| ?-
```

The adjacent screenshot shows parents of a class.

Overview:



```
yes
| ?- class(X,_,_).

X = a,

X = b,

X = c,

no
| ?- assoc(association_id1,a,b).

yes
| ?- assoc(association_id2,b,c).

yes
| ?- object(a1,_,a).

no
| ?- obj(a1,_,a).

yes
| ?- obj(b1,_,b).

yes
| ?- link(_,A,B).

A = a1
B = b1,

no
| ?-
```

Class: Equivalent of a UML class.

Association: Associating two classes (creating a relationship)

Object: Instance of a class

Link: Instance of an association

Advantages :

- Consistencies can be checked in Linear time in Prolog.
- Prolog makes saving rules easier, when compared to other ways of checking consistencies