Abstract Interpretation of Hibernate Query Language

Angshuman Jana, Raju Halder and Agostino Cortesi
{ajana.pcs, halder}@iitp.ac.in
Indian Institute of Technology Patna
cortesi@unive.it
Università Ca’ Foscari Venezia, Italy

Objectives
- We extend the Abstract Interpretation framework to Hibernate Query Language (HQL) as a way to support semantics-based sound approximation techniques.
- We define concrete and abstract semantics of Hibernate Query Language.
- It serves as a formal verification method for behavioral properties of persistent objects, rather than transient objects, which have permanent representations in the underlying databases.

Hibernate Query Language
- Hibernate Query language (HQL) [1] is an Object-Relational Mapping (ORM) language which remedies the paradigm mismatch between object-oriented languages and relational database models.
- It provides a unified platform for programmers to develop object-oriented application where high-level variables interact with underlying database attributes.
- Therefore, application programmers can develop object-oriented applications without knowing much detail about the underlying database.
- It is treated as an object-oriented variant of SQL.
- Various methods in “Session” are used to propagate object’s states from memory to the database (or vice versa) and to synchronize both states when a change is made to persistent objects.

Abstract Syntax of HQL
- Like OOP, the abstract syntax of a program p in HQL is defined as p = (c_main, L) where c_main is the main class, L are the other classes present in p.
- Similarly, a class c is defined as a triplet c = (init, F, M) where init is the constructor, F is the set of fields, and M is the set of member methods in c.

Session methods
m_ses ∈ M_ses
m_ses := (C, φ, OP)
where C ⊆ Class and φ represents ‘WHERE’ clause:
OP := SEL(f(⃗ x), r(⃗ h(⃗ x)), φ, g(⃗ x))
| UPD(⃗ x, ⃗ y)
| SAVE(φ)
| DEL()
where φ represents ‘HAVING’ clause and obj denotes an instance of a class.

HQL Programs
p ∈ P
p := (c_main, L)
where c_main ∈ Class is the main class and L ⊆ Class

State-of-the-art and Motivations
- It can be used for optimization of the code at class-level.
- The existing work on abstract interpretation of query languages [3] did not consider an access to the database operations through a high-level object oriented language.
- Our objective is fill up the gap in between these two theories, by extending the Abstract Interpretation theory to the case of HQL.
- As an application, the proposed framework can formally and automatically verify enterprise-policies in relational/non-relational database models.

Concrete Semantics of HQL
- We define the concrete semantics of HQL by specifying how the methods are executed on (e, s, φ) where e ∈ Env is an environment, s ∈ Store is a store, and ρ_d ∈ D is a database environment, resulting into new state (e′, s′, φ).
- The semantic definitions are expressed in terms of the semantics of database states SELECT, INSERT, UPDATE, DELETE [3].

Abstract Semantics
- The concrete semantics can be lifted to an abstract semantics by simply making correspondence of concrete objects (variables values, object instances, stores, states, traces, etc.) into abstract ones representing partial information on them.
- The abstract version of session methods are:
  OP := SEL(f(⃗ x), r(⃗ h(⃗ x)), φ, g(⃗ x))
  | UPD(⃗ x, ⃗ y)
  | SAVE(φ)
  | DEL()

  The abstract semantics of m_ses is defined in terms of the abstract semantic of INSERT^↓, UPDATE^↓, DELETE^↓, SELECT^↓.

Future Plan
- Extension to the language-based information-flow security analysis.
- Abstract slicing with respect to the properties of persistent objects.
- To build a static analyzer tool for HQL.

References