Semantically Lifted Programs (SLP)

An SLP maps its program state to a knowledge graph and enable semantic reflection. Performing queries on its own configuration using semantic web technologies.

![Diagram of SLP](image)

The direct mapping μ results in an OWL knowledge graph, where the TBox contains axioms from two sources: Axioms Tconf describing the configuration and domain knowledge Tuser provided by the user. Both TBoxes are static and do not change during execution.

Typing Semantically Lifted State Access using Query Containment

The following models a cloud system that uses the domain knowledge about overloaded platforms to dynamically reschedule tasks.

```java
unit reschedule()

 Scheduler(List<Platform> platformList)
 ... Server(List<Task> taskList) ...

 l:= access("SELECT ?x WHERE {?x a Overloaded}");
 this.adaptPlatforms(l);
```

Will the above query always return a list of Platform instances?
- Depends on axioms for Overload and the entailment regime
- Type system and TBox interact

How to ensure that a query access is type-safe, if part of the knowledge requires reasoning about the user-provided TBox?

Typing Semantically Lifted State Access using Concept Subsumption

There are no practical algorithms for query containment under expressive entailment regimes. We, thus, resort to an approximation that reduces the check to concept subsumption with respect to a TBox. If the query can be over-approximated by some DL concept C, then it suffices to check whether C subsumed by the concept ClassT, to ensure that the query return something of type T.

In the, again slightly simplified, rule below, the first premise is changed to express this situation. For tree-shaped conjunctive queries, such a concept C can be easily and efficiently computed directly from the query.

```
\Gamma \vdash \text{access}(\exists y.\varphi(x).e_1, \ldots, e_n) : \text{Unit}
```

Conclusion

We are able to give a type system to semantically lifted programs and implemented it for the Semantic Micro Object Language (SMOL) [1]. This is a first approach to directly couple programming languages and DL reasoning with static guarantees about the interactions while still maintaining a separation of concerns between programming and domain modelling.


Further Research

- **Query Containment** Type checking requires only asymmetrical query containment: the right-hand side is simple concept query. Can this asymmetry be used for more practical containment checks under entailment regimes?
- **Concept Approximation** Can we approximate more complex classes of queries to approximate type checking?

Download and Contact

SLPs are implemented in the SMOL interpreter, try it out under:

github.com/Edkamb/SemanticObjects

- Eduard Kamburjan
- Egor V. Kostylev

UiO: University of Oslo