A General Static Analysis Framework Based on a Transitional Semantics

20190065 김기환 (kimkihwan@kaist.ac.kr)

* transitional style semantics (concrete) =
* state = = : program label = next part of program

: machine state = memory state, program continuation, etc.

* set of states = where is set of labels and is set of machine state
* where

*note. can be defined by relation lifting of ‘’ with functor*

* reachable states (concrete semantic) =

= where

*note. Kleene’s theorem holds because is CPO with infimum and upper bound , and is continuous.*

* concrete semantic function =
* concrete semantic domain = where partial order is
* abstract domain =
* abstract semantic function =
* program-label-wise reachability (flow sensitive) =

collection of all state -> label-wise collection -> label-wise abstraction

*note. since program syntax is fixed, we assume set of labels is also finite and fixed*

* Galois connection =

with

* where

1.

2.

3. This function partitions the input based on label

4.

5.

* soundness of abstract transition =

*note. means we’re handling abstraction of set of states, not just single state.*

* soundness of abstract union =

*note. abstract union must cover all state of union of concretization*

* How to make sound static analysis =

1. Define set of memory and set of labels. Set of labels is fixed and finite.

2. Define concrete domain and semantics.

3. Define abstract domain and semantics

4. Check and are complete partial order set, and make Galois connection.

5. Check soundness of abstract transition and abstract union

6. Choose finite height or widening. It will form sound static analysis.

* Some skill for widening =

1. use worklist to prevent rescan not necessary labels

2. use widening only for loop and use upper bound union for other labels