Patryk Konieczny – patrykk2

Assignment 5

Partial Evaluation –

To partially evaluate an expression, use .eval on a Set Expression from SetOps defined in the DSL. Partial evaluation returns an evaluated set only if the variables are defined and bound to a set. If there is a variable that does not have a binding, then an expression is returned with the Var() expression and any bound variables evaluated to sets. Partial evaluations are valid for all set operations. It also removes the error for trying to evaluate a variable that is not bound. The partial evaluation was implemented by returning changing the type of the eval function to the union type of SetOps | SetType. I also changed the parameters of certain expressions to be of type SetOps | SetType. This allowed me to either return a set when an expression had a binding or return an expression that was a combination of a sets and expressions.

.map() –

The .map function is used to optimize the Set DSL expressions. To use the map use .map on the SetOps expressions that are available to optimize. The map function partially evaluates the expression first and the optimizes the function with the passed function.

Available optimizations –

.map(e => new SetDSL.unionEval(e))

This optimization transforms an expression that is trying to union an empty set with a non-empty set. This set returns the expression of the non-empty set. This optimization is only allowed to be used in this case.

.map(e => new SetDSL.interEval(e))

This optimization transforms an expression that is trying to intersect an empty set with a non-empty set. This set returns an empty set. This optimization is only allowed to be used in this case.

.map(e => new SetDSL.complimnetEval(e))

This optimization transforms an expression that is trying to compliment itself. This set returns an empty set. This optimization is only allowed to be used in this case.