Algorithm GenerateOCLBenchmarks

Input:

UML Metamodel M, OCL Metamodel OM, Benchmark Parameters P

Output:

Set of OCL Constraints C

1. Initialize:

C ← ∅ // Set of OCL constraints

2. Parse UML Metamodel M:

Extract classes (CL), attributes (AT), relationships (RL), and types

3. Parse OCL Metamodel OM:

Define valid grammar, operators, and typing rules

4. Define Benchmark Properties from P:

num\_constraints ← P.num\_constraints

expr\_types ← P.expr\_types

ast\_size ← P.ast\_size

quant\_depth ← P.quant\_depth

5. For i = 1 to num\_constraints:

a. Generate AST Skeleton:

If "basic" ∈ expr\_types:

AST ← BinaryNode(op, attr1, attr2)

If "quantifiers" ∈ expr\_types:

AST ← QuantifierNode(op, collection, condition)

If "navigation" ∈ expr\_types:

AST ← NavigationNode(relationship, attr)

b. Assign Typing Constraints:

For each node N in AST:

If N is an Operator:

Assign typing rules:

Example: For +, At ∈ {Int, Real} ∧ Bt ∈ {Int, Real}

Resolve constraints using SMT solver

c. Populate AST Nodes:

Replace placeholders:

- Attributes with valid attributes from M

- Operators with valid symbols (e.g., +, >)

- Literals with concrete values (e.g., 18, true)

d. Validate Constraint:

Ensure all nodes conform to:

- Typing constraints

- UML Metamodel rules

e. Add Constraint to C:

Format AST into `.ocl` syntax and add to C

6. Save Constraints:

Write C to "generated\_constraints.ocl"

7. Return C