```
MODULE KWALA-SYNTAX
    Syntax
  SYNTAX #Id ::= object
                    null
  SYNTAX Variable ::= #Int
  SYNTAX VariableName ::= v #Int
  SYNTAX Selector ::= <init>()V
  SYNTAX TypeName ::= \#Id
                           | #Id / TypeName
  SYNTAX TypeReference ::= < #Id , TypeName >
  SYNTAX FieldReference ::= < #Id , TypeName , #Id , TypeReference >
  SYNTAX MethodReference ::= < #Id , TypeName , Selector >
  SYNTAX Params ::= List{Variable,"," }
  SYNTAX NewInstructionBase ::= Variable =new TypeReference @ #Int
  {\tt SYNTAX} \quad \textit{NewInstruction} ::= \textit{NewInstructionBase}
                                NewInstructionBase ( Params )
  SYNTAX GetInstruction ::= Variable =getfield FieldReference Variable
                               Variable =getstatic FieldReference
  SYNTAX PutInstruction ::= putfield Variable = Variable FieldReference
                               | putstatic Variable FieldReference
  SYNTAX PhiInstruction ::= Variable =phi( Params )
  SYNTAX PhiPhiInstruction ::= Variable =phiphi( Params )
  {\tt SYNTAX} \quad \textit{InvokeSpecialInstruction} ::= {\tt invokespecial} \, \textit{MethodReference} \, \, \textit{Params} \, @ \, \#\textit{Int} \, \, {\tt exception} \colon \textit{Variable} \, \\
  {\tt SYNTAX} \quad \textit{Instruction} ::= \textit{NewInstruction}
                             GetInstruction
                             PutInstruction
                             PhiInstruction
                             PhiPhiInstruction \\
                             return
                             {\it Invoke Special Instruction}
                             noinstruction
                             main
  SYNTAX BBEdge := \#Id \rightarrow \#Id;
  SYNTAX BlockBody ::= List{Instruction,";"}
  \mathtt{SYNTAX} \quad Block ::= \#Id : \{ \; BlockBody \; \}
  {\tt SYNTAX} \quad \textit{TaskUnit} ::= \textit{BBEdge}
                           Block
                           start
                           analysis
  SYNTAX \quad \textit{Task} ::= \textit{TaskUnit}
                    Task Task
END MODULE
MODULE KWALA
  IMPORTS KWALA-SYNTAX
  SYNTAX RelList ::= List{RelElement, ", "}
    Semantics
    Configuration
  CONFIGURATION:
                                                                   class *
                                                                                                  method *
                                                                                                                                                      basicBlocks
                                                                                                                                                         basicBlock *
                                                                  className
                                                                                                  methodName
                                                                                                                                                          blockName
                                                                                                                                                                                  block
                                                                                                                             edges
                                                                                                      null
                                                                                                                                                              null
                                                                                                                                                                                 noinstruction
                                                                    object
                                                                               invocations
                                                                                          invocation *
                                                                                                                                basicBlocks
                                                                                                                                    basicBlock *
                                                                                                                                                            block
                                                                                         variables
                                                                                                                                    blockName
                                                                                                                                                           noinstruction
                                                                                                                                        null
                                                                                                                                               nextObject
                                                                                                                     nextPointer
                                                                                                                                                     0
                                                                   constraints
                                                                                                                                                                 object *
                                                                                                           fieldRead
                                                                                                                                  fieldWrite
                                                                                                                                                                  name
                                                                                                                                                                                     fields
                                                                    pointsTo
                                                                                          copy
                                                                                                                                                                     0
    Processing Basic Blocks
   RULE T_1 T_2 \Rightarrow T_1 \curvearrowright T_2
   RULE I_1 ; BBl_2 \Rightarrow I_1 \curvearrowright BBl_2
  SYNTAX ListItem ::= [ \#Id , \#Id ]
   RULE
               BB_1 \rightarrow BB_2;
                                       [ BB_1 , BB_2 ]
                                           method
               BB:{ Bl}
   RULE
                                              basicBlock
                                               blockName
                                                                       block
                                                   BB
                                                                         Bl
                                       method
               BB :{}
   RULE
                                          basicBlock
                                           blockName
                                                                  block
                                                                 noinstruction
                                               BB
   RULE
              start
                                   class
                                                            method
                                   className
                                                             methodName
                                                                                     edges
                                                                                                      basicBlocks
                                     "blah"
                                                               "blahah"
                                  basicBlocks
                                     BBls
   RULE
               analysis
                                                                 invocation
                                                                                         basicBlocks
                                                                 variables
                                                                                             BBls
    Gathering Constraints
  {\tt SYNTAX} \quad \textit{RelElement} ::= (\textit{FieldReference} \mid \textit{>} \# \textit{Int} )
                            | (| #Int |)
  SYNTAX K ::= RelList
    Phi functions
               invocation
                    block
                                               variables
                                                                      nextPointer
                                                                                              сору
   RULE
                    V_{\it 1} =phi( P )
                                                                         NP
                  V_1 = \text{phiphi}(P)
                                             VV_1 \mapsto NP
                                                                      NP +_{Int} \mathbf{1}
               invocation
                   block
                                                 variables
   RULE
                  V_{1} =phiphi( V_{2} , P )
                                                 \mathsf{v}\ V_1 \mapsto P_1 \ -\!\!\!\!-\ \mathsf{v}\ V_2 \mapsto P_2
                                                                                                     RL_1
                                                                                              (|P_2|), RL_1
                     V_1 = \text{phiphi}(P)
               invocation
                   block
                  V_1 =phi()
   RULE
    Get field
              invocation
                                                   variables
                                                                                            nextPointer
                                                                                                                   fieldWrite
                    block
                  V_1 =getfield F V_2
                                                                                               NP
   RULE
                                                                                           \overline{NP +_{Int} \mathbf{1}}
                                                 VV_1 \mapsto NP
                                                                                                                  P_2\mapsto ( F\mid > NP ) ,
    Put field
                    block
                                                variables
                                                                                         fieldRead
   RULE
                  putfield V_1 = V_2 F
                                               \mathbf{v} \ V_1 \mapsto P_1 \ - \ \mathbf{v} \ V_2 \mapsto P_2
                                                                                        P_1 \mapsto (F \mid P_2)
    New instruction
              invocation
                                                                                                                    pointsTo
                    block
                                              variables
                                                                      nextPointer
                                                                                             nextObject
                  V_1 =new TR @ —
                                                                         NP
                                                                                                NO
   RULE
                                                                                            \overline{NO} +_{Int} \mathbf{1}
                                                                                                                  \overline{NP}\mapsto (|NO|) , •
                                                                     NP +_{Int} \mathbf{1}
                                            V V_1 \mapsto NP
                                                                                                                                                   ∫object
                                                                                                                                                    name
                                                                                                                                                                    fields
                                                                                                                                                     NO
```

stask

PGM

END MODULE