

EXTENDS *Naturals, Sequences, BNFGrammars*

$CommaList(L) \triangleq L \& (tok(",") \& L)^*$

$AtLeast4(s) \triangleq Tok(\{s \circ s \circ s\} \& \{s\}^+)$

$ReservedWord \triangleq$

{ "ASSUME", "ELSE", "LOCAL", "UNION",
 "ASSUMPTION", "ENABLED", "MODULE", "VARIABLE",
 "AXIOM", "EXCEPT", "OTHER", "VARIABLES",
 "CASE", "EXTENDS", "SF_", "WF_",
 "CHOOSE", "IF", "SUBSET", "WITH",
 "CONSTANT", "IN", "THEN",
 "CONSTANTS", "INSTANCE", "THEOREM",
 "DOMAIN", "LET", "UNCHANGED",
 "BY", "HAVE", "QED", "TAKE",
 "DEF", "HIDE", "RECURSIVE", "USE",
 "DEFINE", "PROOF", "WITNESS", "PICK",
 "DEFS", "PROVE", "SUFFICES", "NEW",
 "LAMBDA", "STATE", "ACTION", "TEMPORAL",
 "OBVIOUS", "OMITTED", "LEMMA", "PROPOSITION" }

$Letter \triangleq OneOf("abcdefghijklmnopqrstuvwxyzABCDEFGHIJKLMNOPQRSTUVWXYZ")$

$Numeral \triangleq OneOf("0123456789")$

$NameChar \triangleq Letter \cup Numeral \cup \{"-\"}$

$Name \triangleq Tok((NameChar^* \& Letter \& NameChar^* \setminus (\{"WF_", "SF_"\} \& NameChar^+))$

$Identifier \triangleq Name \setminus Tok(ReservedWord)$

$IdentifierOrTuple \triangleq$

$Identifier \mid tok("<<") \& CommaList(Identifier) \& tok(">>")$

$NumberLexeme \triangleq$

$Numeral^+$

$\mid (Numeral^* \& \{".\" \} \& Numeral^+)$

$\mid \{"\\b", "\\B"\} \& OneOf("01")^+$

$\mid \{"\\o", "\\O"\} \& OneOf("01234567")^+$

$\mid \{"\\h", "\\H"\} \& OneOf("0123456789abcdefABCDEF")^+$

$Number \triangleq Tok(NumberLexeme)$

$ProofStepId \triangleq Tok(\{"<" \} \& (Numeral^+ \mid \{"*\" \}) \& \{">" \} \& (Letter \mid Numeral)^+)$

$BeginStepToken \triangleq Tok(\{"<" \} \& (Numeral^+ \mid \{"*", "+" \}) \& \{">" \} \& (Letter \mid Numeral)^* \& \{"." \}^*)$

$String \triangleq Tok(\{\backslash\} \& STRING \& \{\backslash\})$

$PrefixOp \triangleq Tok(\{-, \sim, \backslash\!not, \backslash\!neg, [], \langle \rangle, \text{"DOMAIN"}, \text{"ENABLED"}, \text{"SUBSET"}, \text{"UNCHANGED"}, \text{"UNION"}\})$

$InfixOp \triangleq Tok(\{ !, \#, \#\#, \$, \$\$, \%, \%\%, \&, \&\&, (+), (-), (.), (/), (\backslash X), *, **, +, ++, -, -+>, --, |, \dots, \dots, /, //, /=, /\backslash, ::, :=, >, <, <:, <=>, =, =<, =>, =|, >, >=, ??, @@, \backslash, \backslash/, ^, ^~, |, |- , |=, ||, ~>, \dots, <=, \backslash\!approx, \backslash\!geq, \backslash\!oslash, \backslash\!sqsupseteq, \backslash\!asympt, \backslash\!gg, \backslash\!otimes, \backslash\!star, \backslash\!bigcirc, \backslash\!in, \backslash\!prec, \backslash\!subset, \backslash\!bullet, \backslash\!intersect, \backslash\!preceq, \backslash\!subsubseteq, \backslash\!cap, \backslash\!land, \backslash\!propto, \backslash\!succ, \backslash\!cdot, \backslash\!leq, \backslash\!sim, \backslash\!succeq, \backslash\!circ, \backslash\!ll, \backslash\!simeq, \backslash\!supset, \backslash\!cong, \backslash\!lor, \backslash\!sqcap, \backslash\!supseteq, \backslash\!cup, \backslash\!o, \backslash\!sqcup, \backslash\!union, \backslash\!div, \backslash\!odot, \backslash\!sqsubset, \backslash\!uplus, \backslash\!doteq, \backslash\!ominus, \backslash\!sqsubsubseteq, \backslash\!wr, \backslash\!equiv, \backslash\!oplus, \backslash\!sqsupset, \backslash\!notin \})$

$PostfixOp \triangleq Tok(\{^+, ^*, ^\#, ""\})$

$TLAPlusGrammar \triangleq$

LET $P(G) \triangleq$
 $\wedge G.Module ::= AtLeast4("-)$
 $\quad \& tok("MODULE") \& Name \& AtLeast4("-)$
 $\quad \& (Nil | (tok("EXTENDS") \& CommaList(Name)))$
 $\quad \& (G.Unit)^*$
 $\quad \& AtLeast4(=)$

$\wedge G.Unit ::=$
 $G.VariableDeclaration$
 $| G.ConstantDeclaration$
 $| G.Recursive$
 $| G.UseOrHide$
 $| (Nil | tok("LOCAL")) \& G.OperatorDefinition$
 $| (Nil | tok("LOCAL")) \& G.FunctionDefinition$
 $| (Nil | tok("LOCAL")) \& G.Instance$

```

| (Nil | tok("LOCAL")) & G.ModuleDefinition
| G.Assumption
| G.Theorem & (Nil | G.Proof)
| G.Module
| AtLeast4("-")

^ G.VariableDeclaration ::=
  Tok({ "VARIABLE", "VARIABLES" }) & CommaList(Identifier)

^ G.ConstantDeclaration ::=
  Tok({ "CONSTANT", "CONSTANTS" }) & CommaList(G.OpDecl)

^ G.Recursive ::= tok("RECURSIVE") & CommaList(G.OpDecl)

^ G.OpDecl ::=
  Identifier
  | Identifier & tok("(") &
    CommaList(tok("-")) & tok(")")
  | PrefixOp & tok("-")
  | tok("-") & InfixOp & tok("-")
  | tok("-") & PostfixOp

^ G.OperatorDefinition ::=
  (Nil | tok("LOCAL"))
  & ( G.NonFixLHS
    | PrefixOp & Identifier
    | Identifier & InfixOp & Identifier
    | Identifier & PostfixOp)
  & tok("==")
  & G.Expression

^ G.NonFixLHS ::=
  Identifier
  & ( Nil
    | tok("(")
      & CommaList(Identifier | G.OpDecl)
      & tok(")"))

^ G.FunctionDefinition ::=
  (Nil | tok("LOCAL"))
  & Identifier
  & tok("[") & CommaList(G.QuantifierBound) & tok("]")
  & tok("==")
  & G.Expression

^ G.QuantifierBound ::=
  (IdentifierOrTuple | CommaList(Identifier))
  & tok("\\in")
  & G.Expression

```

$$\begin{aligned}
\wedge G.Instance & ::= \\
& \quad tok("INSTANCE") \\
& \quad \& Name \\
& \quad \& (Nil \mid tok("WITH") \& CommaList(G.Substitution)) \\
\wedge G.Substitution & ::= \\
& \quad (Identifier \mid PrefixOp \mid InfixOp \mid PostfixOp) \\
& \quad \& tok("<-") \\
& \quad \& G.Argument \\
\wedge G.Argument & ::= G.Expression \mid G.Opname \mid G.Lambda \\
\wedge G.Lambda & ::= tok("LAMBDA") \& CommaList(Identifier) \\
& \quad \& tok(":") \& G.Expression \\
\wedge G.OpName & ::= \\
& \quad (Identifier \mid PrefixOp \mid InfixOp \mid PostfixOp \mid ProofStepId) \\
& \quad \& (tok("!") \\
& \quad \quad \& (Identifier \mid PrefixOp \mid InfixOp \mid PostfixOp \\
& \quad \quad \quad \mid Tok(\{ "<<", ">>", "@" \} \cup Numeral^+)) \\
& \quad \quad)^* \\
\wedge G.OpArgs & ::= tok("(") \& CommaList(G.Argument) \& tok(")") \\
\wedge G.InstOrSubexprPrefix & ::= \\
& \quad ((Nil \mid ProofStepId \& tok("!")) \\
& \quad \quad \& ((Identifier \& (Nil \mid G.OpArgs) \\
& \quad \quad \quad \mid Tok(\{ "<<", ">>", ":" \} \cup Numeral^+) \\
& \quad \quad \quad \mid G.OpArgs \\
& \quad \quad \quad \mid (PrefixOp \mid PostfixOp) \& tok("(") \& G.Expression \& tok(")") \\
& \quad \quad \quad \mid InfixOp \& tok("(") \& G.Expression \& tok(",") \\
& \quad \quad \quad \quad \& G.Expression \& tok(")") \\
& \quad \quad) \\
& \quad \quad \& tok("!") \\
& \quad \quad)^* \\
& \quad) \setminus Nil \\
\wedge G.GeneralIdentifier & ::= \\
& \quad (G.InstOrSubexprPrefix \mid Nil) \& Identifier \\
& \quad \mid ProofStepId \\
\wedge G.ModuleDefinition & ::= G.NonFixLHS \\
& \quad \quad \& tok("==") \\
& \quad \quad \& G.Instance \\
\wedge G.Assumption & ::= \\
& \quad Tok(\{ "ASSUME", "ASSUMPTION", "AXIOM" \}) \\
& \quad \quad \& (Nil \mid Name \& tok("==")) \& G.Expression
\end{aligned}$$

$$\begin{aligned}
\wedge G.Theorem ::= & Tok(\{\text{"THEOREM"}, \text{"PROPOSITION"}, \text{"LEMMA"}\}) \\
& \& (Nil \mid Name \& tok(=\)) \& (G.Expression \mid G.AssumeProve) \\
\wedge G.AssumeProve ::= & tok(\text{"ASSUME"}) \\
& \& CommaList(G.Expression \mid G.New \mid G.AssumeProve) \\
& \& tok(\text{"PROVE"}) \\
& \& G.Expression \\
\wedge G.New ::= & (((Nil \mid tok(\text{"NEW"})) \\
& \& (Nil \mid tok(\text{"CONSTANT"}) \mid tok(\text{"VARIABLE"}) \mid tok(\text{"STATE"}) \\
& \quad \mid tok(\text{"ACTION"}) \mid tok(\text{"TEMPORAL"})) \\
&) \setminus Nil) \\
& \& ((Identifier \& tok(\text{"\in"})) \& G.Expression) \mid G.OpDecl) \\
\wedge G.Proof ::= & G.TerminalProof \\
& \mid G.NonTerminalProof \\
\wedge G.TerminalProof ::= & (tok(\text{"PROOF"}) \mid Nil) \\
& \& (tok(\text{"BY"}) \& G.UseBody \\
& \quad \mid tok(\text{"OBVIOUS"}) \\
& \quad \mid tok(\text{"OMITTED"}) \\
&) \\
\wedge G.NonTerminalProof ::= & (Nil \mid tok(\text{"PROOF"})) \\
& \& G.Step^* \\
& \& G.QEDStep \\
\wedge G.Step ::= & BeginStepToken \\
& \& ((G.UseOrHide \\
& \quad \mid ((Nil \mid tok(\text{"DEFINE"})) \\
& \quad \& (G.OperatorDefinition \\
& \quad \quad \mid G.FunctionDefinition \\
& \quad \quad \mid G.ModuleDefinition)^+ \\
& \quad) \\
& \quad \mid G.Instance \\
& \quad \mid tok(\text{"HAVE"}) \& G.Expression \\
& \quad \mid tok(\text{"WITNESS"}) \& CommaList(G.Expression) \\
& \quad \mid tok(\text{"TAKE"}) \& (CommaList(G.QuantifierBound) \\
& \quad \quad \mid CommaList(Identifier)) \\
& \quad) \\
& \mid (((Nil \mid tok(\text{"SUFFICES"}) \\
& \quad \& (G.Expression \mid G.AssumeProve) \\
& \quad) \\
& \mid (tok(\text{"CASE"}) \& G.Expression)
\end{aligned}$$

```

        | ( tok("PICK")
          & ( CommaList(G.QuantifierBound) | CommaList(Identifier))
          & tok(":")
          & G.Expression
        )
      )
    & (Nil | G.Proof)
  )
)
^ G.QEDStep ::=
  BeginStepToken & tok("QED") & (Nil | G.Proof)
^ G.UseOrHide ::= Tok({"USE", "HIDE"}) & G.UseBody
^ G.UseBody ::= ( (Nil | CommaList(G.Expression |
                    tok("MODULE") & Name))
  & (Nil | Tok({"DEF", "DEFS"})
    & CommaList(G.OpName |
                tok("MODULE") & Name))
  ) \ Nil

^ G.Expression ::=
  Name & (Nil | tok("(") & CommaList(Identifier) & tok(")"))
  & tok(":") & G.Expression
| G.InstOrSubexprPrefix
  & (Tok({"<<", ">>", ":"} ∪ Numeral+) | G.OpArgs)
| G.GeneralIdentifier & (Nil | G.OpArgs)
| PrefixOp & G.Expression
| G.Expression & InfixOp & G.Expression
| G.Expression & PostfixOp
| tok("(") & G.Expression & tok(")")
| Tok({"\\A", "\\E"})
  & CommaList(G.QuantifierBound)
  & tok(":")
  & G.Expression
| Tok({"\\A", "\\E", "\\AA", "\\EE"})
  & CommaList(Identifier)
  & tok(":")

```

```

& G.Expression

| tok("CHOOSE")
  & IdentifierOrTuple
  & (Nil | tok("\\in") & G.Expression)
  & tok(":")
  & G.Expression

| tok("{")
  & (Nil | CommaList(G.Expression))
  & tok("}")

| tok("{")
  & IdentifierOrTuple & tok("\\in") & G.Expression
  & tok(":")
  & G.Expression
  & tok("}")

| tok("{")
  & G.Expression
  & tok(":")
  & CommaList(G.QuantifierBound)
  & tok("}")

| G.Expression & tok("[") & CommaList(G.Expression)
  & tok("]")

| tok("[")
  & CommaList(G.QuantifierBound)
  & tok("|->")
  & G.Expression
  & tok("]")

| tok("[") & G.Expression & tok("->")
  & G.Expression & tok("]")

| tok("[")
  & CommaList(Name & tok("|->") & G.Expression)
  & tok("]")

| tok("[")
  & CommaList(Name & tok(":") & G.Expression)
  & tok("]")

| tok("[")
  & G.Expression
  & tok("EXCEPT")
  & CommaList( tok("!")

```

$$\begin{aligned} & \& tok(\text{"."}) \& Name \\ & \quad | \quad tok(\text{"["}) \& CommaList(G.Expression) \& tok(\text{"]"})^+ \\ & \quad \& tok(\text{"="}) \& G.Expression \\ & \& tok(\text{"]"}) \\ | \quad & G.Expression \& tok(\text{"."}) \& Name \\ | \quad & tok(\text{"<<"}) \& (CommaList(G.Expression) | Nil) \& tok(\text{">>"}) \\ | \quad & G.Expression \& (Tok(\{\text{"\\X"}, \text{"\\times"}\}) \\ & \quad \& G.Expression)^+ \\ | \quad & tok(\text{"["}) \& G.Expression \& tok(\text{"]-"}) \\ & \quad \& G.Expression \\ | \quad & tok(\text{"<<"}) \& G.Expression \& tok(\text{">>-"}) \& G.Expression \\ | \quad & Tok(\{\text{"WF-"}, \text{"SF-"}\}) \\ & \quad \& G.Expression \\ & \quad \& tok(\text{"("}) \& G.Expression \& tok(\text{"})") \\ | \quad & tok(\text{"IF"}) \quad \& G.Expression \\ & \quad \& tok(\text{"THEN"}) \quad \& G.Expression \\ & \quad \& tok(\text{"ELSE"}) \quad \& G.Expression \\ | \quad & tok(\text{"CASE"}) \\ & \quad \& (\text{LET } CaseArm \triangleq \\ & \quad \quad G.Expression \& tok(\text{"->"}) \& G.Expression \\ & \quad \text{IN } CaseArm \& (tok(\text{"["}) \& CaseArm)^*) \\ & \quad \& (Nil \\ & \quad | \quad (tok(\text{"["}) \& tok(\text{"OTHER"}) \& tok(\text{"->"}) \& G.Expression)) \\ | \quad & tok(\text{"LET"}) \\ & \quad \& (G.OperatorDefinition \\ & \quad \quad | G.FunctionDefinition \\ & \quad \quad | G.ModuleDefinition)^+ \\ & \quad \& tok(\text{"IN"}) \\ & \quad \& G.Expression \\ | \quad & (tok(\text{"/\\"}) \& G.Expression)^+ \\ | \quad & (tok(\text{"\\/"}) \& G.Expression)^+ \\ | \quad & Number \\ | \quad & String \\ | \quad & tok(\text{"@"}) \end{aligned}$$

IN *LeastGrammar(P)*
