

IMP

This is the symbolic semantics of IMP. It contains the normal semantics if IMP and the rules for symbolic execution, as described in the technical report available here: <https://fmse.info.uaic.ro/publications/156/>.

MODULE IMP-SYNTAX

```
SYNTAX  AExp ::= Int
        | Id
        | AExp / AExp [strict]
        | AExp + AExp [strict]
        | (AExp) [bracket]

SYNTAX  BExp ::= Bool
        | AExp ≤ AExp [seqstrict]
        | ! BExp [strict]
        | BExp && BExp [strict(1)]
        | (BExp) [bracket]

SYNTAX  Block ::= {}
        | { Stmt }

SYNTAX  Stmt ::= Block
        | Id = AExp ; [strict(2)]
        | if (BExp) Block else Block [strict(1)]
        | while (BExp) Block
        | Stmt Stmt

SYNTAX  Pgm ::= int Ids ; Stmt

SYNTAX  Ids ::= List{Id, ", "}
```

Programs are \mathbb{K} configurations.

```
SYNTAX  Stmt ::= #ps (Bag)
```

Assertions syntax

```
SYNTAX  Stmt ::= assert (BExp) ; [strict]
```

\mathbb{K} compiler issues

```
SYNTAX  Dummy ::= symInt [dummySymInt]
```

```
SYNTAX  Int ::= #symInt (Id) [onlyLabel, klabel(#symInt)]
```

```
SYNTAX  Id ::= Token{"a"}
        | Token{"b"}
        | Token{"c"}
        | Token{"sum"}
        | Token{"i"}
        | Token{"n"}
        | Token{"x"}
        | Token{"min"}
```

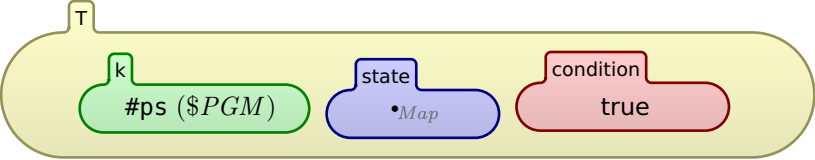
END MODULE

MODULE IMP

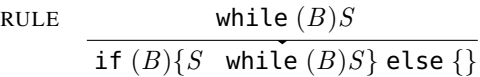
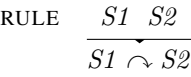
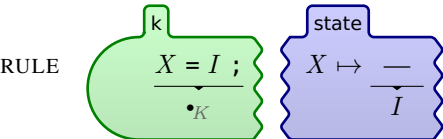
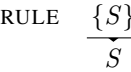
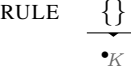
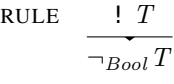
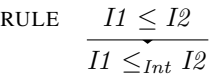
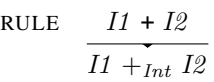
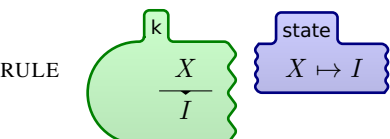
```
SYNTAX  KResult ::= Int
        | Bool
```

IMP configuration is enriched with cell condition.

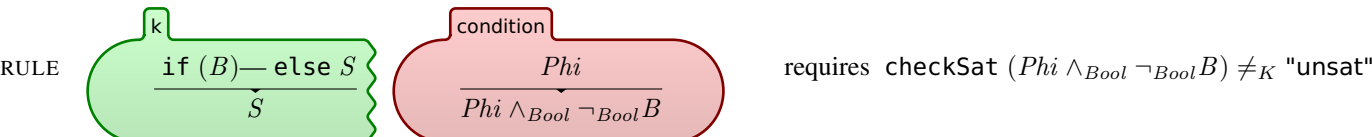
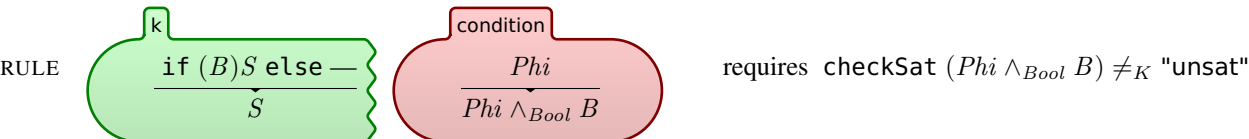
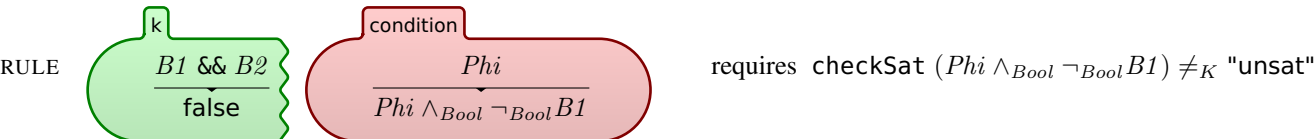
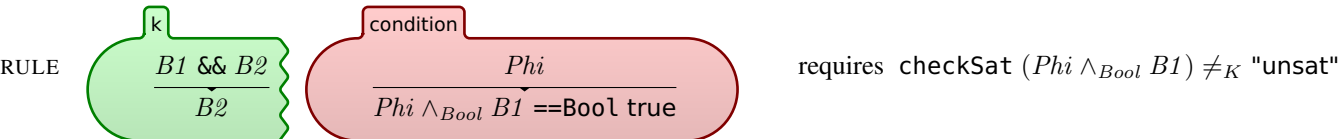
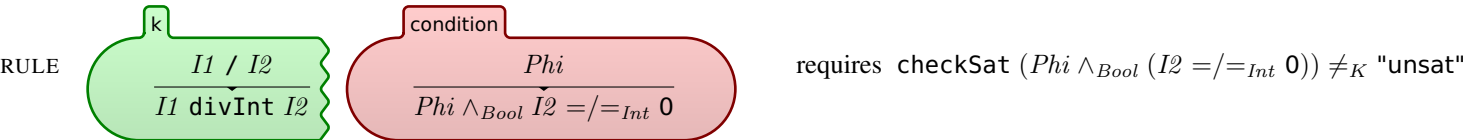
CONFIGURATION:



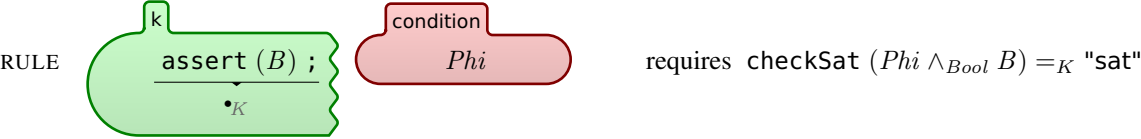
The concrete semantics of IMP which remains unchanged for symbolic execution



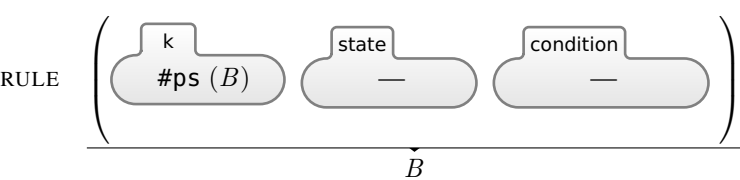
IMP symbolic semantics



Assert semantics: remain stuck when the assertion doesn' hold.



Load PGM.



END MODULE