

# IMP

This is the symbolic semantics of IMP enriched with reachability logic.

This semantics receives as input programs generated from reachability formulas. Additionally, the semantics contains the circularities also generated from reachability formulas.

MODULE IMP-SYNTAX

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

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

SYNTAX  Block ::= {}
          | {Smt}
          | Id : Block

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

SYNTAX  Pgm ::= int Ids ; Smt

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

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

K tool issues

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

SYNTAX  Int ::= (Int) [bracket]

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

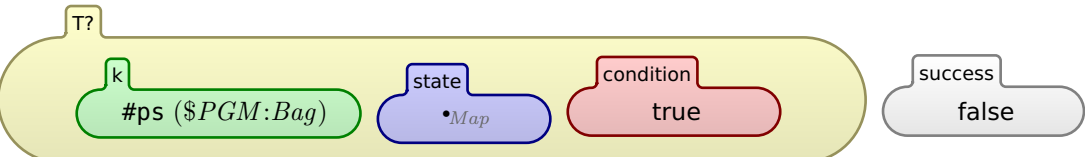
END MODULE

MODULE IMP

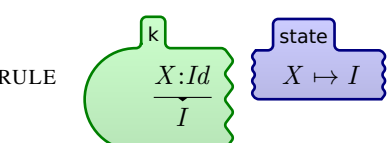
```
SYNTAX  KResult ::= Int
          | Bool
```

The configuration of IMP is enriched with cells <frozen> and <goal>. Cell <ruleConstraints> will store some labels which are meant to block the first application of the circularity rule. The cell <goal> contains the current goal. Whenever multiple rules can be applied to the same configuration, the current goal will be splitted into multiple goals.

CONFIGURATION:



The initial semantics of IMP.



RULE 
$$\frac{I1:Int + I2:Int}{I1 +_{Int} I2}$$

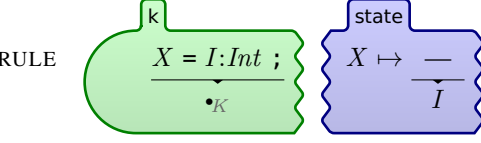
RULE 
$$\frac{I1:Int - I2:Int}{I1 -_{Int} I2}$$

RULE 
$$\frac{I1:Int \leq I2:Int}{I1 \leq_{Int} I2}$$

RULE 
$$\frac{! T:Bool}{\neg_{Bool} T}$$

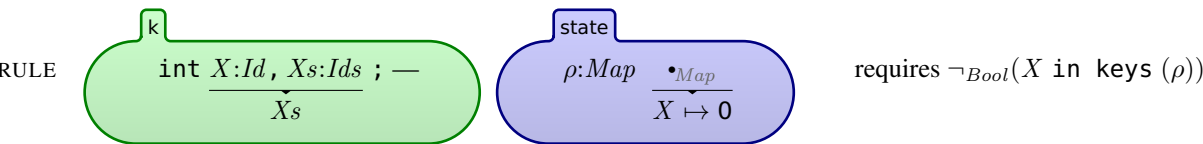
RULE 
$$\frac{\{ \}}{\bullet_K}$$
 [structural]

RULE 
$$\frac{\{ S \}}{S}$$
 [structural]



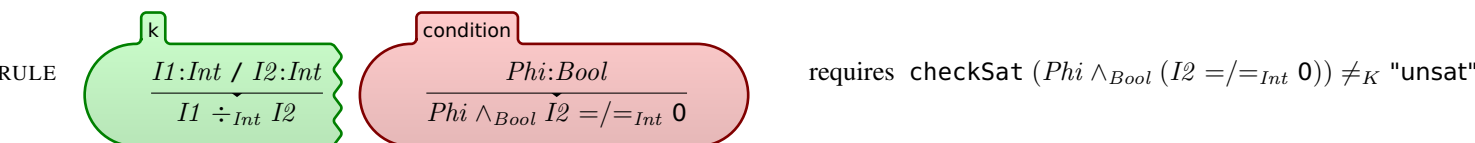
RULE 
$$\frac{S1 \ S2}{S1 \frown S2}$$
 [structural]

RULE 
$$\frac{\text{while } (B)S}{\text{if } (B)\{S \ \text{while } (B)S \ \text{else } \{\}\}}$$
 [transition]



RULE 
$$\frac{\text{int } \bullet_{ids} ; S}{S}$$
 [structural]

Symbolic semantics - the transformed rules



RULE 
$$\frac{B1:Bool \ \&\& \ B2:BExp}{B2}$$
 [transition]

RULE 
$$\frac{B1:Bool \ \&\& \ B2:Bool}{\text{false}}$$
 [transition]

RULE 
$$\frac{B1:Bool \ \&\& \ B2:Bool}{\text{false}}$$
 [transition]

RULE 
$$\frac{\text{if } (B:Bool)S \ \text{else } —}{S}$$
 [transition, computational]

RULE 
$$\frac{\text{if } (B:Bool)— \ \text{else } S}{S}$$
 [transition, computational]

These rules must be generated from reachability formulas given as input and added to the semantics at runtime. Since we don't have this possibility now, we added them manually

For each reachability formula given as input we have two corresponding generated rules: - one corresponding to circularity deduction rule - one checking if the final configuration implies the righ-hand side of the formula (corresponding to consequence deduction rule).

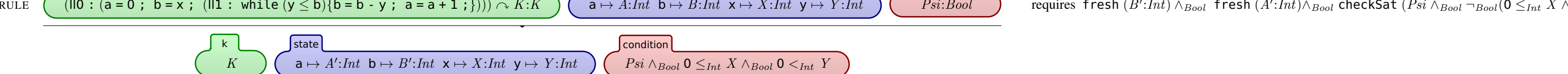
```
SYNTAX  Id ::= Token{"a"}
          | Token{"b"}
          | Token{"x"}
          | Token{"y"}
          | Token{"ll0"}
          | Token{"ll1"}
          | Token{"ll2"}

SYNTAX  Pgm ::= check0
          | check1
          | check2

SYNTAX  Bag ::= success
```



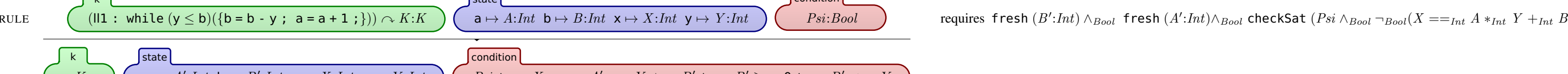
[transition]



[transition]



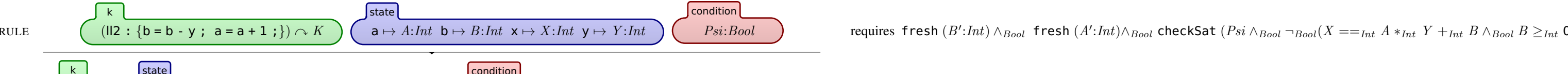
[transition]



[transition]



[transition]



[transition]



[transition]

Utils



[structural]

END MODULE