expressive

modular concurrent

The $\mathbb{K}$ Framework

easy practical

http://k-framework.org

executable scalable

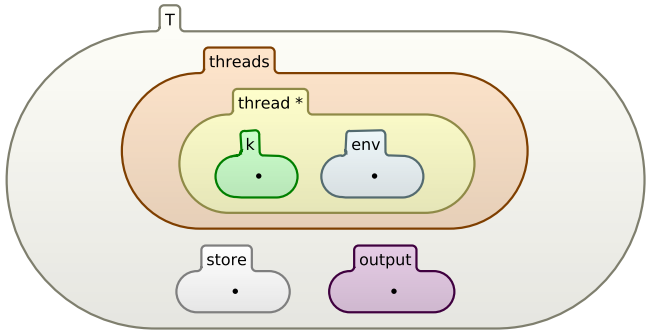
analyzable

David Lazar

lazar6@illinois.edu

The $\mathbb{K}$ Definition of POPLANG

MODULE POPLANG
IMPORTS SUBSTITUTION
SYNTAX *Val* ::= *#Int*
| *#Bool*
SYNTAX *Exp* ::= *Val*
| *#Id*
| *Exp* `+` *Exp* [strict]
| *Exp* `<=` *Exp* [strict]
| `randomBool`
| `callcc` *Exp* [strict]
| *λ#Id.Exp*
| *Exp* *Exp* [strict]
SYNTAX *Stmt* ::= `skip`
| *Stmt* `;` *Stmt*
| *#Id* `:=` *Exp* [strict(2)]
| `if` *Exp* `then` *Stmt* `else` *Stmt* [strict(1)]
| `while` *Exp* `do` *Stmt*
| `spawn` *Stmt*
| `print` *Exp* [strict]
SYNTAX *KResult* ::= *Val*
CONFIGURATION:



RULE I_1 `+` $I_2 \Rightarrow I_1 +_{Int} I_2$

RULE I_1 `<=` $I_2 \Rightarrow I_1 \leq_{Int} I_2$

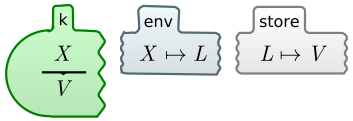
RULE `skip` $\Rightarrow$ •

RULE S_1 `;` $S_2 \Rightarrow S_1 \curvearrowright S_2$

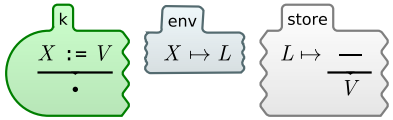
RULE `if true then` S_1 `else` — $\Rightarrow S_1$

RULE `if false then` — `else` $S_2 \Rightarrow S_2$

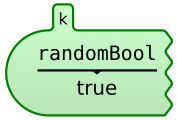
RULE



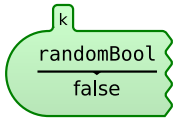
RULE



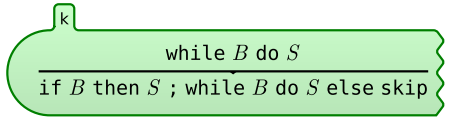
RULE



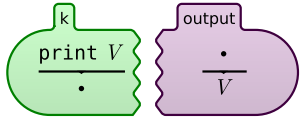
RULE



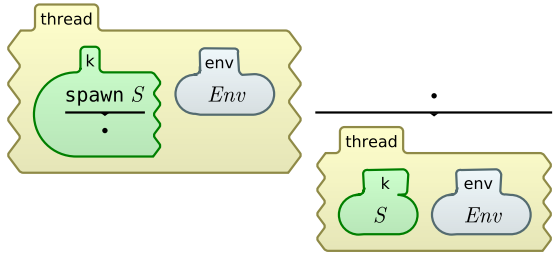
RULE



RULE

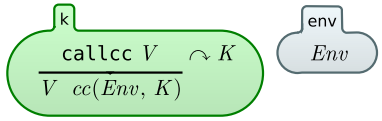


RULE

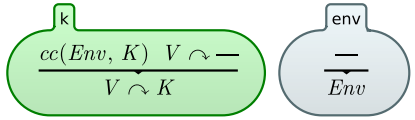


SYNTAX *KResult* ::= $cc(Map, K)$

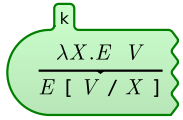
RULE



RULE



RULE



END MODULE