# Proving Reachability Modulo Theories

Ștefan Ciobâcă

Faculty of Computer Science
Alexandru Ioan Cuza University, Iași, Romania
stefan.ciobaca@info.uaic.ro

July 8th, 2017

(joint work with Dorel Lucanu)

# Summary

Given

$$S = \{ \quad l_1 \twoheadrightarrow r_1 \texttt{ if } \phi_1, \\ \ldots, \\ l_n \twoheadrightarrow r_n \texttt{ if } \phi_n\},$$

how to prove

$$S \vdash u \to^* v \text{ if } \phi?$$

# Summary

Given

$$S = \{ \quad l_1 \twoheadrightarrow r_1 \texttt{ if } \phi_1, \\ \dots, \\ l_n \twoheadrightarrow r_n \texttt{ if } \phi_n \},$$

how to prove

$$S \vdash u \to^* v \text{ if } \phi?$$

Example. Given:

$$S = \{ \quad init(n) \twoheadrightarrow loop(0, n) \texttt{ if } true, \\ loop(s, i) \twoheadrightarrow loop(s + i, i - 1) \texttt{ if } i > 0 \\ loop(s, 0) \twoheadrightarrow done(s) \texttt{ if } true$$

prove

$$S \vdash init(n) \to^* done(n(n + 1)/2) \text{ if } n \geq 0.$$

# Plan

# Transition Systems

Let $(M, \rightsquigarrow)$ be a transition system, where $\rightsquigarrow \subseteq M \times M$. We write $\gamma \rightsquigarrow \gamma'$ for $(\gamma, \gamma') \in \rightsquigarrow$. An element $\gamma \in M$ is *irreducible* if there is no $\gamma' \in M$ such that $\gamma \rightsquigarrow \gamma'$.

# Transition Systems

Let $(M, \rightsquigarrow)$ be a transition system, where $\rightsquigarrow \subseteq M \times M$. We write $\gamma \rightsquigarrow \gamma'$ for $(\gamma, \gamma') \in \rightsquigarrow$. An element $\gamma \in M$ is *irreducible* if there is no $\gamma' \in M$ such that $\gamma \rightsquigarrow \gamma'$.

## Definition (Execution Path)

The set of *(complete) execution paths* is coinductively defined by the folowing rules:

$$\frac{}{\gamma}\ \gamma \in M, \gamma \text{ irreducible} \qquad \frac{\tau}{\gamma_0 \circ \tau}\ \gamma_0 \rightsquigarrow hd(\tau)$$

where the function $hd$ is itself defined coinductively by $hd(\gamma) = \gamma$ and $hd(\gamma_0 \circ \tau) = \gamma_0$.

# State and Reachability Predicates

## Definition (State and Reachability Predicates)

A *state predicate* is a subset $P \subseteq M$. A *reachability predicate* is a pair $P \Rightarrow Q$, where $P$ and $Q$ are state predicates. A state predicate $P$ is *runnable* if for all $\gamma \in P$ there exists $\gamma' \in M$ such that $\gamma \rightsquigarrow \gamma'$.

# Derivative of a State Predicate

### Definition (Derivative of a State Predicate)

Given a state predicate $P$, the derivative of $P$ is the state predicate $\partial(P)$, defined by $\partial(P) = \{\gamma' \mid \gamma \rightsquigarrow \gamma' \text{ for some } \gamma \in P\}$.

# Satisfaction of Reachability Predicates

## Definition (Satisfaction of a Reachability Predicate)

An execution path $\tau$ *satisfies* a reachability predicate $P \Rightarrow Q$, written $\tau \vDash^{\forall} P \Rightarrow Q$, iff $\langle \tau, P \Rightarrow Q \rangle \in \nu\, \widehat{\mathrm{EPSRP}}$, where EPSRP consists of the following rules:

$$\frac{}{\langle \tau, P \Rightarrow Q \rangle}\; hd(\tau) \in P \cap Q \qquad \frac{\langle \tau, \partial(P) \Rightarrow Q \rangle}{\langle \gamma_0 \circ \tau, P \Rightarrow Q \rangle}\; \gamma_0 \in P, \gamma_0 \rightsquigarrow hd(\tau).$$

# Satisfaction of Reachability Predicates

## Definition (Satisfaction of a Reachability Predicate)

An execution path $\tau$ *satisfies* a reachability predicate $P \Rightarrow Q$, written $\tau \models^{\forall} P \Rightarrow Q$, iff $\langle \tau, P \Rightarrow Q \rangle \in \nu\, \widehat{\mathrm{EPSRP}}$, where EPSRP consists of the following rules:

$$\frac{}{\langle \tau, P \Rightarrow Q \rangle}\ hd(\tau) \in P \cap Q \qquad \frac{\langle \tau, \partial(P) \Rightarrow Q \rangle}{\langle \gamma_0 \circ \tau, P \Rightarrow Q \rangle}\ \gamma_0 \in P, \gamma_0 \rightsquigarrow hd(\tau).$$

## Definition (Demonically Valid Predicates, Coinductively)

We say that $P \Rightarrow Q$ is *demonically valid*, and we write

$$(M, \rightsquigarrow) \models^{\forall} P \Rightarrow Q,$$

iff $P \Rightarrow Q \in \nu\, \widehat{\mathrm{DVP}}$, where DVP consists of the following rules:

$$[\![\mathrm{Subsumption}]\!]\ \frac{}{P \Rightarrow Q}\ P \subseteq Q \qquad [\![\mathrm{Step}]\!]\ \frac{\partial(P \setminus Q) \Rightarrow Q}{P \Rightarrow Q}\ P \setminus Q \text{ runnable.}$$

# Plan

# Builtins

## Definition (Builtin Signature)

A *builtin signature* $\Sigma^{\mathsf{b}} \triangleq (S^{\mathsf{b}}, F^{\mathsf{b}})$ is any many-sorted signature that includes the following distinguished objects:

- a sort *Bool* together with two constants $\top$ and $\bot$ of sort *Bool*,
- the propositional operations $\neg : Bool \to Bool$, $\wedge, \vee, \to \; : Bool \times Bool \to Bool$, and
- an equality predicate $=^{?} : s \times s \to Bool$ for each sort $s \in S^{\mathsf{b}}$.

# Builtins

## Definition (Builtin Signature)

A *builtin signature* $\Sigma^{\mathrm{b}} \triangleq (S^{\mathrm{b}}, F^{\mathrm{b}})$ is any many-sorted signature that includes the following distinguished objects:

- a sort *Bool* together with two constants $\top$ and $\bot$ of sort *Bool*,
- the propositional operations $\neg : Bool \to Bool$, $\wedge, \vee, \to\ : Bool \times Bool \to Bool$, and
- an equality predicate $=^{?} : s \times s \to Bool$ for each sort $s \in S^{\mathrm{b}}$.

## Definition (Builtin Model)

A *builtin model* $M^{\mathrm{b}}$ is a model of a builtin signature $\Sigma^{\mathrm{b}}$, where the interpretation of the distinguished objects of the builtin signature is fixed as follows: $M^{\mathrm{b}}_{Bool} = \{\top, \bot\}$, $M^{\mathrm{b}}_{\top} = \top$, $M^{\mathrm{b}}_{\bot} = \bot$, $M^{\mathrm{b}}_{=^?}(a, b) = \top$ iff $a = b$, $M^{\mathrm{b}}_{\neg}(\top) = \bot$, $M^{\mathrm{b}}_{\neg}(\bot) = \top$, $M^{\mathrm{b}}_{\wedge}(\top, b) = M^{\mathrm{b}}_{\wedge}(b, \top) = b$, $M^{\mathrm{b}}_{\wedge}(\bot, b) = M^{\mathrm{b}}_{\wedge}(b, \bot) = \bot$, and so on.

# Signature Modulo Builtins

## Definition (Signature Modulo a Builtin Model)

A *signature modulo a builtin model* is a tuple $\Sigma \triangleq (S, \leq, F, M^{\mathrm{b}})$ consisting of

- an order-sorted signature $(S, \leq, F)$, and
- a builtin $\Sigma^{\mathrm{b}}$-model $M^{\mathrm{b}}$, where $\Sigma^{\mathrm{b}} \triangleq (S^{\mathrm{b}}, F^{\mathrm{b}})$ is a builtin subsignature of $(S, \leq, F)$.

We further assume that the only builtin constants in $\Sigma$ are the elements of the builtin model, i.e., $F^{\mathrm{b}}_{\varepsilon,s} = M^{\mathrm{b}}_s$. $\Sigma^{\mathrm{b}}$ is called the *builtin subsignature* of $\Sigma$ and $\Sigma^{\mathrm{c}} = (S, \leq, (F \setminus F^{\mathrm{b}}) \cup \bigcup_{s \in S^{\mathrm{b}}} F^{\mathrm{b}}_{\varepsilon,s})$ the *constructor subsignature* of $\Sigma$.

# Model Modulo Builtins

**Definition (Model $M^{\Sigma}$ Generated by a Signature Modulo a Builtin Model)**

Let $\Sigma \triangleq (S, \leq, F, M^{\mathrm{b}})$ be a signature modulo a builtin model. The model $M^{\mathrm{b}}$ is extended in a protected way to a $(S, \leq, F)$-model $M^{\Sigma}$ defined as follows:

- $M^{\Sigma}_{s} = T_{\Sigma^{\mathrm{c}},s}(M^{\mathrm{b}})$ for each $s \in S \setminus S^{\mathrm{b}}$, i.e. $M^{\Sigma}_{s}$ includes the constructor terms;
- $M^{\Sigma}_{f} = M^{\mathrm{b}}_{f}$ for each $f \in F^{\mathrm{b}}$;
- $M^{\Sigma}_{f}$ is the term constructor $M^{\Sigma}_{f}(t_1, \ldots, t_n) = f(t_1, \ldots, t_n)$ for each $f \in F \setminus F^{\mathrm{b}}$.

# Constraint Formulae

## Definition (Constraint Formulae)

The set $\mathrm{CF}(\Sigma, X)$ of *constraint formulae* over variables $X$ is inductively defined as follows:

$$\phi ::= b \mid t_1 =^? t_2 \mid (\exists x)\phi' \mid \neg\phi' \mid \phi_1 \wedge \phi_2$$

where $b$ ranges over $T_{\Sigma,Bool}(X)$, $t_i$ over $T_{\Sigma,s_i}(X)$ such that $s_1$ and $s_2$ are in the same connected component, and $x$ ranges over all variables.

# Constraint Formulae

## Definition (Constraint Formulae)

The set $\mathrm{CF}(\Sigma, X)$ of *constraint formulae* over variables $X$ is inductively defined as follows:

$$\phi ::= b \mid t_1 =^? t_2 \mid (\exists x)\phi' \mid \neg\phi' \mid \phi_1 \wedge \phi_2$$

where $b$ ranges over $T_{\Sigma,Bool}(X)$, $t_i$ over $T_{\Sigma,s_i}(X)$ such that $s_1$ and $s_2$ are in the same connected component, and $x$ ranges over all variables.

## Definition (Semantics of Constraint Formulae)

The satisfaction relation $\models$ is inductively defined over the model $M^\Sigma$, valuations $\alpha : Var \to M^\Sigma$, and formulae $\phi \in \mathrm{CF}(\Sigma, X)$, as follows:

- $M^\Sigma, \alpha \models b$ iff $\alpha(b) = true$, where $b \in T_{\Sigma,Bool}(X)$;
- $M^\Sigma, \alpha \models t_1 =^? t_2$ iff $\alpha(t_1) = \alpha(t_2)$;
- $M^\Sigma, \alpha \models (\exists x)\phi$ iff there exists $a \in M_s$ (where $x \in X_s$) such that $M^\sigma, \alpha[x \mapsto a] \models \phi$;

# Constrained Terms

## Definition (Constrained Terms)

A *constrained term* $\varphi$ of sort $s \in S$ is a pair $(t \mid \phi)$ with $t \in T_{\Sigma,s}(X)$ and $\phi \in \mathrm{CF}(\Sigma, X)$.

# Constrained Terms

## Definition (Constrained Terms)

A *constrained term* $\varphi$ of sort $s \in S$ is a pair $(t \mid \phi)$ with $t \in T_{\Sigma,s}(X)$ and $\phi \in \mathrm{CF}(\Sigma, X)$.

## Definition (Valuation Semantics of Constrained Terms)

The *valuation semantics* of a constrained term $(t \mid \phi)$ is the set of valuations $[\![(t \mid \phi)]\!] \triangleq \{\alpha : X \to M^{\Sigma} \mid M^{\Sigma}, \alpha \vDash \phi\}$.

# Constrained Terms

## Definition (Constrained Terms)

A *constrained term* $\varphi$ of sort $s \in S$ is a pair $(t \mid \phi)$ with $t \in T_{\Sigma,s}(X)$ and $\phi \in \mathrm{CF}(\Sigma, X)$.

## Definition (Valuation Semantics of Constrained Terms)

The *valuation semantics* of a constrained term $(t \mid \phi)$ is the set of valuations $\llbracket (t \mid \phi) \rrbracket \triangleq \{\alpha : X \to M^{\Sigma} \mid M^{\Sigma}, \alpha \vDash \phi\}$.

## Definition (State Predicate Semantics of Constrained Terms)

The *state predicate semantics* of a constrained term is defined by

$$\llbracket (t \mid \phi) \rrbracket \triangleq \{\alpha(t) \mid \alpha \in \llbracket (t \mid \phi) \rrbracket\}.$$

# Plan

# Constrained Rule Systems

## Definition (Constrained Rule Systems)

A *constrained rule* is a tuple $(l, r, \phi)$, often written as $l \twoheadrightarrow r \texttt{ if } \phi$, where $l, r$ are terms in $T_\Sigma(X)$ having sorts in the same connected component, and $\phi \in \mathrm{CF}(\Sigma, X)$.
A *constrained rule system* $\mathcal{R}$ is a set of rules. $\mathcal{R}$ defines a *transition relation* $\rightsquigarrow_{\mathcal{R}}$ on $M^\Sigma$ as follows: $t \rightsquigarrow_{\mathcal{R}} t'$ iff there exist a rule $l \twoheadrightarrow r \texttt{ if } \phi$ in $\mathcal{R}$, a context $c[\cdot]$, and a valuation $\alpha : X \to M^\Sigma$ such that $t = \alpha(c[l])$, $t' = \alpha(c[r])$ and $M^\Sigma, \alpha \models \phi$.

# Reachability Properties

## Definition (Reachability Properties of Constrained Rule Systems)

A *reachability formula* is a pair of constrained terms written as $\varphi \Rightarrow \varphi'$, which may share some variables. We say that a constrained rule system $\mathcal{R}$ *demonically satisfies* $\varphi \Rightarrow \varphi'$, and write

$$\mathcal{R} \vDash^{\forall} \varphi \Rightarrow \varphi'$$

iff $(M^{\Sigma}, \rightsquigarrow_{\mathcal{R}}) \vDash^{\forall} [\![\sigma(\varphi)]\!] \Rightarrow [\![\sigma(\varphi')]\!]$ for each substitution $\sigma : \mathit{var}(\varphi) \cap \mathit{var}(\varphi') \to M^{\Sigma}$.

# Derivatives of Constrained Terms

## Definition (Derivatives of Constrained Terms)

The *set of derivatives* of a constrained term $\varphi \triangleq (t \mid \phi)$ w.r.t. a rule $l \twoheadrightarrow r$ `if` $\phi_{lr}$ is

$$\begin{aligned}\Delta_{l,r,\phi_{lr}}(\varphi) \triangleq \{(c[r] \mid \phi') \mid {} & M^{\Sigma} \vDash \phi' \leftrightarrow (\phi \wedge t =^{?} c[l] \wedge \phi_{lr}),\\ & c[\cdot] \text{ an appropriate context},\\ & \phi' \text{ is satisfiable}\} \qquad (1)\end{aligned}$$

where the variables in $l \twoheadrightarrow r$ `if` $\phi$ are renamed such that *var*$(l \twoheadrightarrow r$ `if` $\phi)$ and *var*$(\varphi)$ are disjoint.
If $\mathcal{R}$ is a set of rules, then $\Delta_{\mathcal{R}}(\varphi) = \bigcup_{(l,r,\phi_{lr})\in\mathcal{R}} \Delta_{l,r,\phi_{lr}}(\varphi)$. A constrained term $\varphi$ is *$\mathcal{R}$-derivable* if $\Delta_{\mathcal{R}}(\varphi) \neq \emptyset$.

# Derivatives of Constrained Terms

## Definition (Derivatives of Constrained Terms)

The *set of derivatives* of a constrained term $\varphi \triangleq (t \mid \phi)$ w.r.t. a rule $l \twoheadrightarrow r$ `if` $\phi_{lr}$ is

$$\begin{aligned}\Delta_{l,r,\phi_{lr}}(\varphi) \triangleq \{(c[r] \mid \phi') \mid\ & M^{\Sigma} \vDash \phi' \leftrightarrow (\phi \wedge t =^{?} c[l] \wedge \phi_{lr}),\\ & c[\cdot] \text{ an appropriate context},\\ & \phi' \text{ is satisfiable}\}\end{aligned} \tag{1}$$

where the variables in $l \twoheadrightarrow r$ `if` $\phi$ are renamed such that $var(l \twoheadrightarrow r$ `if` $\phi)$ and $var(\varphi)$ are disjoint.
If $\mathcal{R}$ is a set of rules, then $\Delta_{\mathcal{R}}(\varphi) = \bigcup_{(l,r,\phi_{lr}) \in \mathcal{R}} \Delta_{l,r,\phi_{lr}}(\varphi)$. A constrained term $\varphi$ is $\mathcal{R}$*-derivable* if $\Delta_{\mathcal{R}}(\varphi) \neq \emptyset$.

Let $\varphi \triangleq (t \mid \phi)$, $\mathcal{R}$ a constrained rule system, and $(M^{\Sigma}, \rightsquigarrow_{\mathcal{R}})$ the transition system defined by $\mathcal{R}$. Then $[\![\Delta_{\mathcal{R}}(\varphi)]\!] = \partial([\![\varphi]\!])$.

# Example

$$
\begin{aligned}
S = \{ \quad & init(n) \twoheadrightarrow loop(0, n) \texttt{ if } true, \\
& loop(s, i) \twoheadrightarrow loop(s + i, i - 1) \texttt{ if } i > 0 \\
& loop(s, 0) \twoheadrightarrow done(s) \texttt{ if } true \}
\end{aligned}
$$

# Example

$$
\begin{aligned}
S = \{ \quad & \mathit{init}(n) \twoheadrightarrow \mathit{loop}(0, n) \texttt{ if } \mathit{true}, \\
& \mathit{loop}(s, i) \twoheadrightarrow \mathit{loop}(s + i, i - 1) \texttt{ if } i > 0 \\
& \mathit{loop}(s, 0) \twoheadrightarrow \mathit{done}(s) \texttt{ if } \mathit{true} \}
\end{aligned}
$$

$$
\Delta_{\mathcal{R}}((\mathit{loop}(s, n) \mid \mathit{true})) = \begin{array}{l} \{ \\ \quad (\mathit{loop}(s + n, n - 1) \mid n > 0), \\ \quad (\mathit{done}(s) \mid n = 0) \\ \} \end{array}
$$

# Plan

# Proof System for Symbolic Execution

Figure: The DSTEP($\mathcal{R}$) Proof System

$$[\text{axiom}]\ \frac{}{(t \mid \phi) \Rightarrow \varphi'}\ M^{\Sigma} \models \phi \leftrightarrow \bot$$

$$[\text{subs}]\ \frac{(t'' \mid \phi'' \wedge \neg\phi''') \Rightarrow (t' \mid \phi')}{(t \mid \phi) \Rightarrow (t' \mid \phi')}\ \begin{array}{l} (t'' \mid \phi'') \Rightarrow \varphi' \equiv (t \mid \phi) \Rightarrow \varphi', \text{ and} \\ M^{\Sigma} \models \phi''' \leftrightarrow (\exists X)(t'' =^{?} t' \wedge \phi'), \text{ ar} \\ X \triangleq var(t', \phi') \setminus var(t'', \phi'') \end{array}$$

$$[\text{der}^{\forall}]\ \frac{\{(t^j \mid \phi^j) \Rightarrow \varphi' \mid (t^j \mid \phi^j) \in \Delta_{\mathcal{R}}((t'' \mid \phi''))\}}{(t \mid \phi) \Rightarrow \varphi'}$$

$$\begin{array}{l} (t \mid \phi)\ \mathcal{R}\text{−derivable, and} \\ (t'' \mid \phi'') \Rightarrow \varphi' \equiv (t \mid \phi) \Rightarrow \varphi', \text{ and} \\ \phi'' \wedge \bigwedge \left\{ \neg(\exists Y)\phi^j \;\middle|\; \begin{array}{l} (t^j \mid \phi^j) \in \Delta_{\mathcal{R}}((t'' \mid \phi'')), \\ Y \triangleq var(t^j, \phi^j) \setminus var(t'', \phi'') \end{array} \right\} \text{ not satisfial} \end{array}$$

# Soundness

Let $\mathcal{R}$ be a constrained rule system. Then $\mathcal{R} \models^{\forall} \nu\, \widehat{\mathsf{DSTEP}}(\mathcal{R})$.

# Soundness

Let $\mathcal{R}$ be a constrained rule system. Then $\mathcal{R} \models^{\forall} \nu\,\widehat{\mathsf{DSTEP}}(\mathcal{R})$.
Not terribly useful!

# Circularity

## Definition (Demonic circular coinduction)

Let $G$ be a finite set reachability formulae. Then the set of rules DCC($\mathcal{R}, G$) consists of DSTEP($\mathcal{R}$) together with

$$[\text{circ}]\ \frac{\begin{array}{l}(t'_c \mid \phi'_c \wedge \phi \wedge \phi'')) \Rightarrow \varphi', \\ (t \mid \phi \wedge \neg\phi'') \Rightarrow \varphi'\end{array}}{(t \mid \phi) \Rightarrow \varphi'} \quad \begin{array}{l} M^{\Sigma} \models \phi'' \leftrightarrow (\exists var(t_c, \phi_c))(t =^{?} t_c \wedge \\ (t_c \mid \phi_c) \Rightarrow (t'_c \mid \phi'_c) \in G \end{array}$$

where the variables in $(t_c \mid \phi_c) \Rightarrow (t'_c \mid \phi'_c)$ are renamed such that $var(t_c, \phi_c) \cap var(t, \phi) = \emptyset$.

# Circularity

## Definition (Demonic circular coinduction)

Let $G$ be a finite set reachability formulae. Then the set of rules $\mathrm{DCC}(\mathcal{R}, G)$ consists of $\mathrm{DSTEP}(\mathcal{R})$ together with

$$[\mathrm{circ}]\ \frac{\begin{array}{c}(t'_c \mid \phi'_c \wedge \phi \wedge \phi'')) \Rightarrow \varphi', \\ (t \mid \phi \wedge \neg\phi'') \Rightarrow \varphi'\end{array}}{(t \mid \phi) \Rightarrow \varphi'} \quad \begin{array}{l} M^{\Sigma} \models \phi'' \leftrightarrow (\exists var(t_c, \phi_c))(t =^{?} t_c \wedge \varphi \\ (t_c \mid \phi_c) \Rightarrow (t'_c \mid \phi'_c) \in G \end{array}$$

where the variables in $(t_c \mid \phi_c) \Rightarrow (t'_c \mid \phi'_c)$ are renamed such that $var(t_c, \phi_c) \cap var(t, \phi) = \emptyset$.

## Definition

Let $PT$ be a proof tree of $\varphi \Rightarrow \varphi'$ under $\mathrm{DCC}(\mathcal{R}, G)$. A [circ] node in $PT$ is *guarded* iff it has as ancestor a $[\mathrm{der}^{\forall}]$ node. $PT$ is *guarded* iff all its [circ] nodes are guarded.

# Soundness of Circularity

### Definition

We write $(\mathcal{R}, G) \vdash^{\forall} \varphi \Rightarrow \varphi'$ iff there is a proof tree of $\varphi \Rightarrow \varphi'$ under $\mathrm{DCC}(\mathcal{R}, G)$ that is guarded. If $F$ is a set of reachability formulae, we write $(\mathcal{R}, G) \vdash^{\forall} F$ iff $(\mathcal{R}, G) \vdash^{\forall} \varphi \Rightarrow \varphi'$ for all $\varphi \Rightarrow \varphi' \in F$.

# Soundness of Circularity

## Definition

We write $(\mathcal{R}, G) \vdash^{\forall} \varphi \Rightarrow \varphi'$ iff there is a proof tree of $\varphi \Rightarrow \varphi'$ under $\mathrm{DCC}(\mathcal{R}, G)$ that is guarded. If $F$ is a set of reachability formulae, we write $(\mathcal{R}, G) \vdash^{\forall} F$ iff $(\mathcal{R}, G) \vdash^{\forall} \varphi \Rightarrow \varphi'$ for all $\varphi \Rightarrow \varphi' \in F$.

[Circularity Principle]
Let $\mathcal{R}$ be a constrained rule system and $G$ a set of goals. If $(\mathcal{R}, G) \vdash^{\forall} G$ then $\mathcal{R} \vDash^{\forall} G$.

# Demo

http://github.com/ciobaca/rmt/

# Conclusions

1. cleaner semantics of constrained rewrite systems;
2. coinductive approach to reachability properties;
3. coinductive proof system for reachability formulae inspired from our previous approaches to partial program correctness [1, 2];
4. implementation.


Ştefănescu, A., Ciobâcă, Ş., Mereuţă, R., Moore, B.M., Şerbănuţă, T.F., Roşu, G.: All-path reachability logic. In: RTA-TLCA 2014. LNCS, vol. 8560, pp. 425–440 (2014)

Lucanu, D., Rusu, V., Arusoaie, A.: A generic framework for symbolic execution: A coinductive approach. J. Symb. Comput. 80, 125–163 (2017)