

# A Method for Loop Verification Based on Fixed Points

Steffen Schlager

October 17th, 2006

# Motivation

So far

- ▶ Loop verification: induction or invariant-based proof
- ▶ Special induction rule for every data type

# Motivation

## So far

- ▶ Loop verification: induction or invariant-based proof
- ▶ Special induction rule for every data type

## What we want

- ▶ Unified rule for loop verification
- ▶ Understanding relation between inductive and invariant-based proofs

# Motivation

## So far

- ▶ Loop verification: induction or invariant-based proof
- ▶ Special induction rule for every data type

## What we want

- ▶ Unified rule for loop verification
- ▶ Understanding relation between inductive and invariant-based proofs

## Approach

- ▶ Understand loops as fixed points
- ▶ Successfully used in EVT (Erlang Verification Tool)
- ▶ Logic with fixed points is very expressive
- ▶ Fix point operators can be hidden for loop verification in KeY



## Signature

$\Sigma = (P, F, LVar, PVar, OrdVar, \alpha)$

## Formulas

- ▶ Basis: typed first-order logic
- ▶ If  $\phi \in Formulas$ , then  $[p]\phi, \langle p \rangle \phi \in Formulas$  for each sequential Java program  $p$

## Signature

$\Sigma = (P, F, LVar, PVar, OrdVar, \alpha)$

## Formulas

- ▶ Basis: typed first-order logic
- ▶ If  $\phi \in Formulas$ , then  $[p]\phi, \langle p \rangle \phi \in Formulas$  for each sequential Java program  $p$

Recursive definition of *Formulas* extended by:

- ▶ If  $X$  occurs only **positively** in  $\phi$ , then
$$\mu X.\phi \in Formulas, \quad (\mu X.\phi)^\kappa \in Formulas$$
$$\nu X.\phi \in Formulas, \quad (\nu X.\phi)^\kappa \in Formulas$$
for  $X \in PVar$ ,  $\phi \in Formulas$ , and  $\kappa \in OrdVar$

# Dynamic Logic with Fix-Point Operators – Semantics

- ▶ Semantics based on Kripke structures  $\mathcal{K} = (\mathcal{S}, \rho)$  and a variable assignment for predicate and ordinal variables
- ▶ State  $s$  is a first-order structure  $(I, D)$

# Dynamic Logic with Fix-Point Operators – Semantics

- ▶ Semantics based on Kripke structures  $\mathcal{K} = (S, \rho)$  and a variable assignment for predicate and ordinal variables
- ▶ State  $s$  is a first-order structure  $(I, D)$
- ▶ Valuation function for terms  $val_s : Terms \rightarrow D$ 
  - ▶  $val_s(f(t_1, \dots, t_n)) = I_s(f)(val_s(t_1), \dots, val_s(t_n))$
- ▶ Valuation function for formulas  $val : Formulas \rightarrow 2^S$ 
  - ▶  $val(true) = S, val(false) = \emptyset$
  - ▶  $val(P(t_1, \dots, t_n)) = \{s \in S \mid (val_s(t_1), \dots, val_s(t_n)) \in I_s(P)\}$
  - ▶  $val(\langle p \rangle \phi) = \{s \in S \mid \text{there is } s' \text{ with } s \llbracket p \rrbracket s' \text{ and } s' \in val(\phi)\}$
  - ▶  $val(X) = \beta(X) \subseteq S$

# Approximation of Fixed Points

- ▶ If  $f$  is monotonic on a lattice then the least fixed point can be obtained by iterating  $f$  on the bottom element  $\perp$  (here  $f = \text{val}$  and  $\perp = \emptyset$ ).

# Approximation of Fixed Points

- ▶ If  $f$  is monotonic on a lattice then the least fixed point can be obtained by iterating  $f$  on the bottom element  $\perp$  (here  $f = val$  and  $\perp = \emptyset$ ).
- ▶ Length of iteration is in general transfinite but *bound* by the cardinality  $c$  of the lattice.

$$lfp(f) = \bigcup_{\alpha < c} f^\alpha(\emptyset)$$

# Approximation of Fixed Points

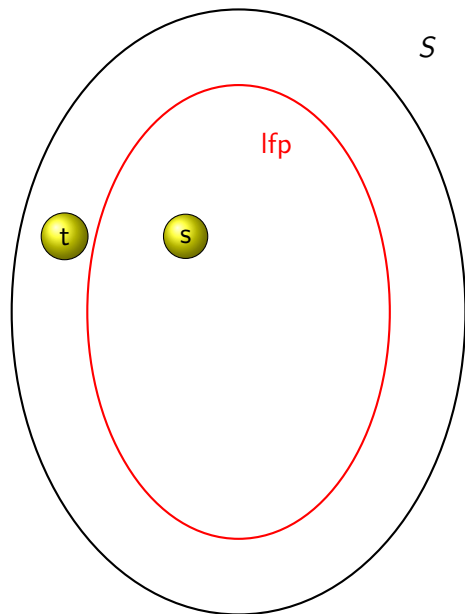
- ▶ If  $f$  is monotonic on a lattice then the least fixed point can be obtained by iterating  $f$  on the bottom element  $\perp$  (here  $f = val$  and  $\perp = \emptyset$ ).
- ▶ Length of iteration is in general transfinite but *bound* by the cardinality  $c$  of the lattice.

$$lfp(f) = \bigcup_{\alpha < c} f^\alpha(\emptyset)$$

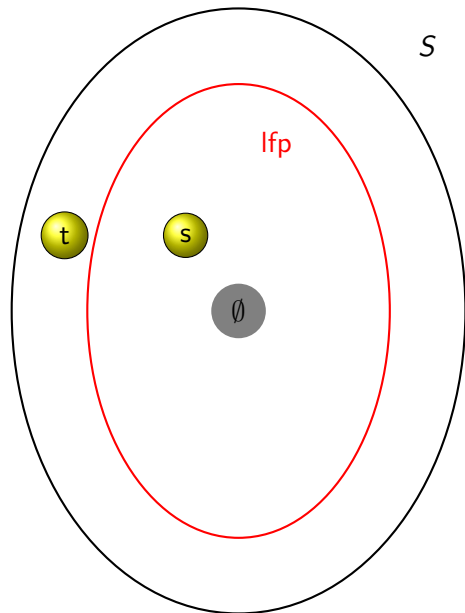
- ▶ Similar for the greatest fixed point

$$gfp(f) = \bigcap_{\alpha < c} f^\alpha(S)$$

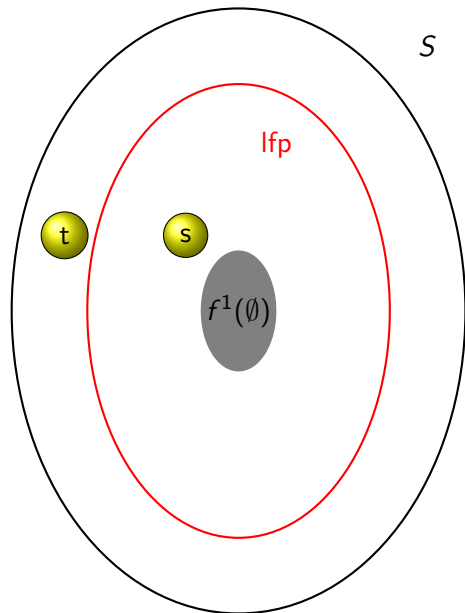
# Approximation of Least Fixed Points



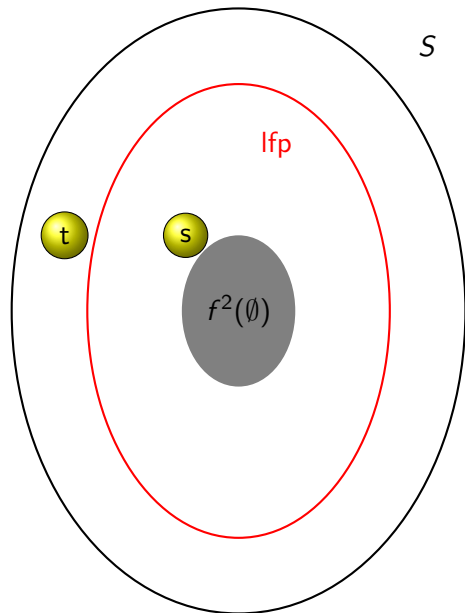
# Approximation of Least Fixed Points



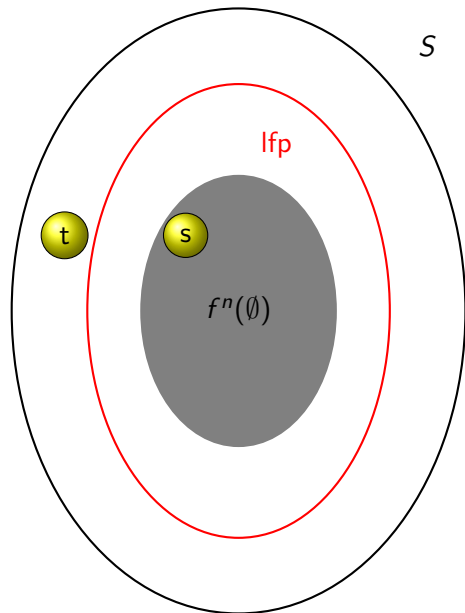
# Approximation of Least Fixed Points



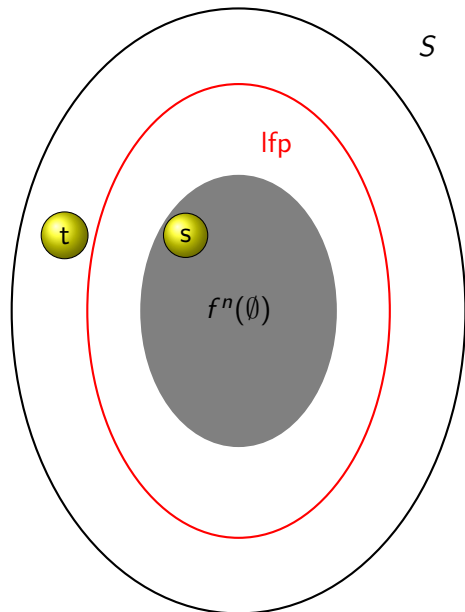
# Approximation of Least Fixed Points



# Approximation of Least Fixed Points



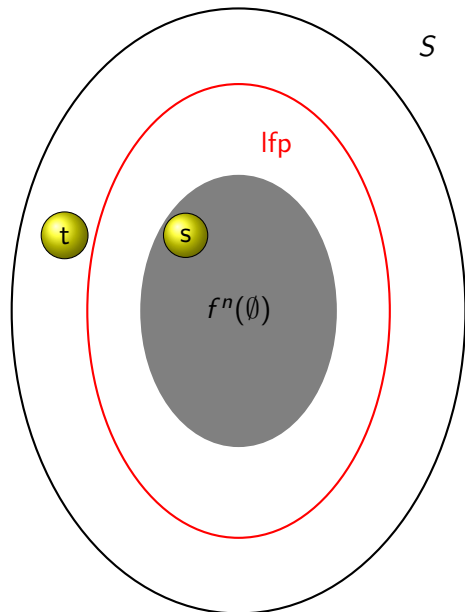
# Approximation of Least Fixed Points



Intuitive meaning of least fixed points:

**"finite looping"**

# Approximation of Least Fixed Points



Intuitive meaning of least fixed points:

“finite looping”

Analogously  
greatest fixed points:

“infinite looping”

# Dynamic Logic with Fix-Point Operators – Semantics

- ▶  $val(\mu X.\phi) = \bigcup_{n \in Ord} (val(\beta_\kappa^n, (\mu X.\phi)^\kappa))$
- ▶  $val(\beta, (\mu X.\phi)^\kappa) = \begin{cases} \emptyset & \beta(\kappa) = 0 \\ val(\beta_X^P, \phi) & \beta(\kappa) = n + 1 \\ \text{where } P = val(\beta_\kappa^n, (\mu X.\phi)^\kappa) \\ \bigcup_{n \in Ord} \{val(\beta_\kappa^n, (\mu X.\phi)^\kappa) \mid n < \beta(\kappa)\} & \beta(\kappa) \text{ is limit ordinal} \end{cases}$

# Least or Greatest Fixed Points?

## Data types

- ▶ finite data types (e.g. natural numbers, lists, trees):  $\mu$
- ▶ infinite data types (e.g. streams, infinite lists):  $\nu$

# Least or Greatest Fixed Points?

## Data types

- ▶ finite data types (e.g. natural numbers, lists, trees):  $\mu$
- ▶ infinite data types (e.g. streams, infinite lists):  $\nu$

## Example

- ▶  $\mu X. \lambda n. (n \doteq 0 \vee \exists n'. n' \doteq n - 1 \wedge X(n'))$

# Least or Greatest Fixed Points?

## Data types

- ▶ finite data types (e.g. natural numbers, lists, trees):  $\mu$
- ▶ infinite data types (e.g. streams, infinite lists):  $\nu$

## Example

- ▶  $\mu X. \lambda n. (n \doteq 0 \vee \exists n'. n' \doteq n - 1 \wedge X(n'))$

## Temporal properties

- ▶ Liveness properties:  $\mu$
- ▶ Safety properties:  $\nu$

# Least or Greatest Fixed Points?

## Data types

- ▶ finite data types (e.g. natural numbers, lists, trees):  $\mu$
- ▶ infinite data types (e.g. streams, infinite lists):  $\nu$

## Example

- ▶  $\mu X. \lambda n. (n \doteq 0 \vee \exists n'. n' \doteq n - 1 \wedge X(n'))$

## Temporal properties

- ▶ Liveness properties:  $\mu$
- ▶ Safety properties:  $\nu$

## Example

- ▶  $\mu X. \phi \vee \langle \alpha \rangle X$

# Sequent Rules for $\mu$

$$\frac{(\mu X.\phi)^{\kappa} \vdash \Delta}{\mu X.\phi \vdash \Delta} \textit{ApproxL}$$

# Sequent Rules for $\mu$

$$\frac{(\mu X.\phi)^\kappa \vdash \Delta}{\mu X.\phi \vdash \Delta} \textit{ApproxL}$$

Soundness:

Let  $(\mu X.\phi)^\kappa \vdash \Delta$  be valid, i.e. for any valuation  $\beta$  of  $\kappa$   
 $val(\beta, (\mu X.\phi)^\kappa) \subseteq val(\Delta)$ .

# Sequent Rules for $\mu$

$$\frac{(\mu X.\phi)^\kappa \vdash \Delta}{\mu X.\phi \vdash \Delta} \text{ApproxL}$$

Soundness:

Let  $(\mu X.\phi)^\kappa \vdash \Delta$  be valid, i.e. for any valuation  $\beta$  of  $\kappa$   
 $val(\beta, (\mu X.\phi)^\kappa) \subseteq val(\Delta)$ .

We now assume  $\mu X.\phi \vdash \Delta$  is not valid, i.e. there exists  $s$  with  
 $s \in val(\beta, \mu X.\phi)$  and  $s \notin val(\Delta)$ .

# Sequent Rules for $\mu$

$$\frac{(\mu X.\phi)^\kappa \vdash \Delta}{\mu X.\phi \vdash \Delta} \text{ApproxL}$$

Soundness:

Let  $(\mu X.\phi)^\kappa \vdash \Delta$  be valid, i.e. for any valuation  $\beta$  of  $\kappa$   
 $val(\beta, (\mu X.\phi)^\kappa) \subseteq val(\Delta)$ .

We now assume  $\mu X.\phi \vdash \Delta$  is not valid, i.e. there exists  $s$  with  
 $s \in val(\beta, \mu X.\phi)$  and  $s \notin val(\Delta)$ .

From the semantics of  $\mu$  follows that there is some  $\beta'$  such that  
 $s \in val(\beta', (\mu X.\phi)^\kappa)$  which is a contradiction to the assumption.

# Sequent Rules for $\mu$

$$\frac{\kappa' < \kappa, \phi[(\mu X.\phi)^{\kappa'} / X] \vdash \Delta}{(\mu X.\phi)^{\kappa} \vdash \Delta} \text{UnfoldL}$$

$$\frac{\vdash \phi[(\mu X.\phi) / X], \Delta}{\vdash \mu X.\phi, \Delta} \text{UnfoldR}$$

# Sequent Rules for $\mu$

$$\frac{\kappa' < \kappa, \phi[(\mu X.\phi)^{\kappa'} / X] \vdash \Delta}{(\mu X.\phi)^{\kappa} \vdash \Delta} \text{UnfoldL}$$

$$\frac{\vdash \phi[(\mu X.\phi) / X], \Delta}{\vdash \mu X.\phi, \Delta} \text{UnfoldR}$$

Rules for  $\nu$  are analogous.

# Global Discharge Rule – Basic Idea

$$\frac{\frac{\frac{\frac{\frac{\frac{\frac{\kappa' < \kappa \vdash \Delta'_1, (\nu X.\phi)^{\kappa'}, \Delta_n}{\vdash \Delta_5, (\nu X.\phi)^\kappa} \text{Unfold}}{\vdash \Delta_4} \text{*}}{\vdash \Delta_2} \text{*}}{\vdash \Delta_3, (\nu X.\phi)^\kappa} \text{*}}{\vdash \Delta_1, (\nu X.\phi)^\kappa} \text{*}}{\vdash \Delta_1, \nu X.\phi} \text{Approx}}{\vdash \Delta_1, \nu X.\phi} \text{Approx}$$

# Global Discharge Rule – Basic Idea

$$\begin{array}{c}
 \frac{}{\kappa' < \kappa \vdash \Delta'_1, (\nu X.\phi)^{\kappa'}, \Delta_n} \\
 \vdots \\
 \frac{}{\vdash \Delta_5, (\nu X.\phi)^\kappa} \text{Unfold} \\
 \frac{\frac{}{\vdash \Delta_4} *}{\vdash \Delta_3, (\nu X.\phi)^\kappa} * \\
 \frac{\frac{}{\vdash \Delta_2} *}{\vdash \Delta_1, (\nu X.\phi)^\kappa} * \\
 \frac{\vdash \Delta_1, (\nu X.\phi)^\kappa}{\vdash \Delta_1, \nu X.\phi} \text{Approx}
 \end{array}$$

# Global Discharge Rule – Basic Idea

$$\begin{array}{c}
 \text{* global discharge} \\
 \hline
 \kappa' < \kappa \vdash \Delta'_1, (\nu X.\phi)^{\kappa'}, \Delta_n \\
 \hline
 \vdots \\
 \text{*} \\
 \hline
 \vdash \Delta_4 \qquad \vdash \Delta_5, (\nu X.\phi)^\kappa \quad \text{Unfold} \\
 \hline
 \text{*} \\
 \hline
 \vdash \Delta_2 \qquad \vdash \Delta_3, (\nu X.\phi)^\kappa \\
 \hline
 \vdash \Delta_1, (\nu X.\phi)^\kappa \\
 \hline
 \vdash \Delta_1, \nu X.\phi \quad \text{Approx}
 \end{array}$$









# Soundness of Global Discharge

$\{\mathcal{U}\}(\sigma(\Delta_1 \vee (\nu X.\phi)^\kappa)) \vdash \Delta'_1 \vee (\nu X.\phi)^{\kappa'}$ , i.e. (discharge cond.)

$\{\mathcal{U}\}(\sigma(\Delta_1 \vee (\nu X.\phi)^\kappa)) \vdash_{s',\beta'} \Delta'_1 \vee (\nu X.\phi)^{\kappa'}$

Since  $\kappa' < \kappa \Vdash_{s',\beta'} \Delta'_1, (\nu X.\phi)^{\kappa'}, \Delta_n$  we have

$\Vdash_{s',\beta'} \{\mathcal{U}\}(\sigma(\Delta_1 \vee (\nu X.\phi)^\kappa))$  and

$\Vdash_{\mathcal{U}(s'),\beta''} (\Delta_1 \vee (\nu X.\phi)^\kappa)$  (appl. of update and subst. theorem)

# Soundness of Global Discharge

$\{\mathcal{U}\}(\sigma(\Delta_1 \vee (\nu X.\phi)^\kappa)) \vdash \Delta'_1 \vee (\nu X.\phi)^{\kappa'}$ , i.e. (discharge cond.)

$\{\mathcal{U}\}(\sigma(\Delta_1 \vee (\nu X.\phi)^\kappa)) \vdash_{s',\beta'} \Delta'_1 \vee (\nu X.\phi)^{\kappa'}$

Since  $\kappa' < \kappa \not\vdash_{s',\beta'} \Delta'_1, (\nu X.\phi)^{\kappa'}, \Delta_n$  we have

$\not\vdash_{s',\beta'} \{\mathcal{U}\}(\sigma(\Delta_1 \vee (\nu X.\phi)^\kappa))$  and

$\not\vdash_{\mathcal{U}(s'),\beta''} (\Delta_1 \vee (\nu X.\phi)^\kappa)$  (appl. of update and subst. theorem)

$\kappa' < \kappa \vdash \sigma(\kappa) < \kappa$ , i.e.,

$\kappa' < \kappa \vdash_{s',\beta'} \sigma(\kappa) < \kappa$

Since  $\kappa' < \kappa \not\vdash_{s',\beta'} \Delta'_1, (\nu X.\phi)^{\kappa'}, \Delta_n$  we have

$\vdash_{s',\beta'} \sigma(\kappa) < \kappa$ , and, as a consequence

$\beta'(\sigma(\kappa)) < \beta'(\kappa)$  and

$\beta''(\kappa) < \beta'(\kappa)$

# Soundness of Global Discharge

We now apply the same argument inductively and obtain an infinite chain

$$\beta'(\kappa) > \beta''(\kappa) > \beta'''(\kappa) > \dots$$

of ordinals which is a **contradiction to the well-foundedness of the ordinals**.

# Loop Verification Using Global Discharge in KeY

## Idea – Understanding loops as fixed points

Assumption: No abrupt termination

$$\langle \text{while } (\epsilon) \{ \alpha \} \rangle \phi \leftrightarrow \mu X. (\neg \epsilon \rightarrow \phi) \wedge (\epsilon \rightarrow \langle \alpha \rangle X)$$

$$[\text{while } (\epsilon) \{ \alpha \}] \phi \leftrightarrow \nu X. (\neg \epsilon \rightarrow \phi) \wedge (\epsilon \rightarrow [\alpha] X)$$

Demo

## Summary

- ▶ One rule for inductive and invariant (co-inductive) proofs
- ▶ Invariant rule and induction rule are derivable
- ▶ Works for recursive methods

# Summary & Future Work

## Summary

- ▶ One rule for inductive and invariant (co-inductive) proofs
- ▶ Invariant rule and induction rule are derivable
- ▶ Works for recursive methods

## Future Work

- ▶ Implementation
- ▶ Verification of temporal properties

$$\vdash \alpha : \nu X. \phi \wedge \langle - \rangle X$$

there is a path such that  $\phi$  always holds