Transformations of Logic Programs on Infinite Lists

Alberto Pettorossi (Univ. Tor Vergata, Rome, Italy),

Maurizio Proietti (IASI-CNR, Rome, Italy),

Valerio Senni (Univ. Tor Vergata, Rome, Italy)

ICLP, Edinburgh, UK

July 16-19, 2010

Motivations

- Transformations are often useful for theorem proving and program verification: from FOL to clause form, quantifier elimination, from Temporal Logics or Monadic Second Order Logics to Büchi automata.

- Unfold/fold transformations have been used for proving program properties [Kott 1982, P.P. 1993, Roychoudhury et al. 1999, Seki 2009].

- Goal of this work: Define a general framework for proving properties of reactive systems via unfold/fold transformations.

Plan of the Talk

- Specifying reactive systems by ω-programs.
- Proving properties of ω-programs via transformation rules.
- Correctness of the transformation rules.

Plan of the Talk

- Specifying reactive systems by ω-programs.
- Proving properties of ω-programs via transformation rules.
- Correctness of the transformation rules.

Reactive Systems

- Reactive systems: communication protocols, security protocols, hardware controllers, etc.

- Various models of reactive systems with infinite behaviour:
  ω-languages,
  Buchi automata,
  temporal and modal logics.

Logic Programming and Reactive Systems

- GHC [Ueda 86], CCP [Saraswat 89]: infinite executions, no infinite lists
- Temporal Logic Programming [Abadi-Manna 89]: temporal specifications, no infinite lists
- LP with rational trees [Colmerauer 82]: CLP with equality and inequality constraints on finite and infinite terms
- Coinductive LP [Gupta et al. 2006-2010]: rational trees, greatest fixed-points, coinductive hypothesis rule
- ω-programs: perfect model semantics, no new operational semantics
- Easy extension of program transformation techniques from LP on finite terms to ω-programs

ω-programs

- ω-programs are typed logic programs with three types:

  fterm (finite term), elem (element of an infinite list), ilist (infinite list).

- [_ | _]: elem x ilist $\rightarrow$ ilist is interpreted as the constructor of infinite lists.

- Each predicate has at most one argument of type ilist (to avoid unification between infinite lists).

Semantics of ω-programs

- The Herbrand base B_ω is defined on the Herbrand universe extended with infinite lists.

- A local stratification is a function $\sigma: B_\omega \rightarrow W$, where W is the set of countable ordinals. For $A \in B_\omega$, $\sigma(\neg A) = \sigma(A) + 1$.

- Given two literals L_1 and L_2, $L_1 \geq_\sigma L_2$ if for all groundings $v(L_1)$, $v(L_2)$ of L_1, L_2, $\sigma(v(L_1)) \geq \sigma(v(L_2))$. Similarly for $>_\sigma$ and $=_\sigma$.

- A clause $H \leftarrow L_1 \wedge \ldots \wedge L_n$ is locally stratified wrt σ if, for i=1,...,n, $H \geq_\sigma L_i$.

  An ω-program P is locally stratified if there exists σ s.t. every clause in P is locally stratified wrt σ.

- For a locally stratified ω-program P, the perfect model M(P) is constructed as a subset of B_ω, by induction on W.

Examples of ω-programs

Let $\{a,b\}$ be the set of constants of type elem.

Let L be a variable of type ilist (i.e., L ranges over $(a+b)^\omega$).

P:	$p([a\|L]) \leftarrow q(L)$	$p(L) \in M(P)$ iff $L \in a(a+b)^\omega$
	$q(L) \leftarrow$	$q(L) \in M(P)$ iff $L \in (a+b)^\omega$
P:	$p(L) \leftarrow \neg q(L)$	$p(L) \in M(P)$ iff $L \in a^\omega$
	$q([a\|L]) \leftarrow q(L)$	$q(L) \in M(P)$ iff $L \in a^*b(a+b)^\omega$
	$q([b\|L]) \leftarrow$	
P:	$p([a\|L]) \leftarrow p(L)$	$M(P) = \emptyset$.

Monadic ω-programs

- A monadic ω-program is a set of clauses of the form:

$$p_0([s|X_0]) \leftarrow p_1(X_1) \wedge \ldots \wedge p_k(X_k) \wedge \neg p_{k+1}(X_{k+1}) \wedge \ldots \wedge \neg p_m(X_m)$$

where:

- s is a constant of type elem,
- $X_0, X_1, \ldots, X_k, X_{k+1}, \ldots, X_m$ are variables of type ilist, and
- there exists a level mapping h: Pred $\rightarrow$ N such that:

  for $i=1,\ldots,k$, if $X_i=X_0$ then $h(p_i) \leq h(p_0)$ else $h(p_i) < h(p_0)$

  for $i=k+1,\ldots,m$, $h(p_i) < h(p_0)$

- Some of the predicates p_i's may be nullary.

- A monadic ω-program is stratified (hence locally stratified).

Decidability of Monadic ω-programs

There exists an algorithm MDec such that, given any monadic ω-program T and unary predicate p, terminates and checks whether or not

$$M(T) \models \exists X\; p(X)$$

[PPS LOPSTR'09]

Plan of the Talk

- Specifying reactive systems by ω-programs.
- Proving properties of ω-programs via transformation rules.
- Correctness of the transformation rules.

A Transformation-based Proof Method

Given any ω-program P and unary predicate p, in order to prove
$M(P) \models \exists X\, p(X)$

1. Try to transform P into a monadic ω-program T, such that
$$M(P) \models \exists X\, p(X) \quad \text{iff} \quad M(T) \models \exists X\, p(X)$$

2. Apply Mdec

Transformation Sequences

- The transformation from P to T is a transformation sequence:

$$\begin{array}{ccccccc} P & & & & & & T \\ \| & & & & & & \| \\ P_0 & \rightarrow & P_1 & \rightarrow & \ldots & \rightarrow & P_n \end{array}$$

  - $P_0, P_1, \ldots, P_n$ are locally stratified ω-programs;
  - $P_i \rightarrow P_{i+1}$ is an application of a transformation rule among: Definition Introduction, Instantiation, Positive Unfolding, Negative Unfolding, Positive Folding, Negative Folding, Subsumption.

- The transformation rules are similar to [Seki 91, Maher 93, Roychoudhury et al. 02, FPP 04, Seki 09], but with different applicability conditions, needed for the correctness of the proof technique.

An Example of Transformation

Property: There exists an infinite list $L=[s_0, s_1, s_2, \ldots]$ in $\{a,b\}^\omega$ whose elements at even positions are all a's (that is, $a=s_0=s_2= \cdots$):

$$\exists L\ \boxed{\forall X\ (\mathrm{position}(X) \wedge \mathrm{even}(X) \rightarrow \mathrm{member}(X,L,a))}$$

prop(L)

EvenAs:

$\mathrm{prop}(L) \leftarrow \neg\ \mathrm{negprop}(L)$
$\mathrm{negprop}(L) \leftarrow \mathrm{position}(X) \wedge \mathrm{even}(X) \wedge \neg\ \mathrm{member}(X,L,a)$
$\mathrm{position}(0) \leftarrow$
$\mathrm{position}(s(X)) \leftarrow \mathrm{position}(X)$
$\mathrm{even}(0) \leftarrow$
$\mathrm{even}(s(X)) \leftarrow \neg\ \mathrm{even}(X)$
$\mathrm{member}(0,[H|T],H) \leftarrow$
$\mathrm{member}(s(X),[H|T],S) \leftarrow \mathrm{member}(X,T,S)$

Locally stratified w.r.t. a suitable stratification function σ

Not a monadic ω-program.

An Example of Transformation

Property: There exists an infinite list $L=[s_0, s_1, s_2, \ldots]$ in $\{a,b\}^\omega$ whose elements at even positions are all a's (that is, $a=s_0=s_2=\cdots$):

$$\exists L\ \underbrace{\boxed{\forall X\ (position(X) \wedge even(X) \rightarrow member(X,L,a))}}_{prop(L)}$$

EvenAs:

$prop(L) \leftarrow \neg\ negprop(L)$
$\boxed{negprop(L) \leftarrow position(X) \wedge even(X) \wedge \neg\ member(X,L,a)}$
$position(0) \leftarrow$
$position(s(X)) \leftarrow position(X)$
$even(0) \leftarrow$
$even(s(X)) \leftarrow \neg\ even(X)$
$member(0,[H|T],H) \leftarrow$
$member(s(X),[H|T],S) \leftarrow member(X,T,S)$

Locally stratified w.r.t. a suitable stratification function σ

Not a monadic ω-program.

Transformation into a Monadic ω-Program

$\boxed{\text{negprop}(L) \leftarrow \text{position}(X) \wedge \text{even}(X) \wedge \neg\ \text{member}(X,L,a)}$ non monadic

Transformation into a Monadic ω-Program

$\boxed{\text{negprop}(L) \leftarrow \text{position}(X) \wedge \text{even}(X) \wedge \neg\ \text{member}(X,L,a)}$ non monadic

Instantiation L / [a|T] ; L / [b|T]

negprop([a|T]) ← position(X) ∧ even(X) ∧ ¬ member(X,[a|T],a)
negprop([b|T]) ← position(X) ∧ even(X) ∧ ¬ member(X,[b|T],a)

Transformation into a Monadic ω-Program

negprop(L) ← position(X) ∧ even(X) ∧ ¬ member(X,L,a)

non monadic

Instantiation

negprop([a|T]) ← position(X') ∧ even(X') ∧ ¬ member(X',[a|T],a)
negprop([b|T]) ← position(X) ∧ even(X) ∧ ¬ member(X,[b|T],a)

X'/0

position(0) ←

X'/s(X)

position(s(X)) ← position(X)

Positive Unfolding

negprop([a|T]) ← even(0) ∧ ¬ member(0,[a|T],a)
negprop([a|T]) ← position(X) ∧ even(s(X)) ∧ ¬ member(s(X),[a|T],a)
negprop([b|T]) ← position(X) ∧ even(X) ∧ ¬ member(X,[b|T],a)

Transformation into a Monadic ω-Program

$\boxed{negprop(L) \leftarrow position(X) \wedge even(X) \wedge \neg\ member(X,L,a)}$ non monadic

$\downarrow$ *Instantiation*

negprop([a|T]) ← position(X) ∧ even(X) ∧ ¬ member(X,[a|T],a)
negprop([b|T]) ← position(X) ∧ even(X) ∧ ¬ member(X,[b|T],a)

$\downarrow$ *Positive Unfolding+*

negprop([a|T]) ← $\boxed{even(0)}$ ∧ ¬ member(0,[a|T],a)
negprop([a|T]) ← position(X) ∧ $\boxed{even(s(X))}$ ∧ ¬ member(s(X),[a|T],a)
negprop([b|T]) ← $\boxed{position(X)}$ ∧ $\boxed{even(X)}$ ∧ ¬ member(X,[b|T],a)

Transformation into a Monadic ω-Program

$\boxed{negprop(L) \leftarrow position(X) \wedge even(X) \wedge \neg\ member(X,L,a)}$ non monadic

Instantiation

$negprop([a|T]) \leftarrow position(X) \wedge even(X) \wedge \neg\ member(X,[a|T],a)$

$negprop([b|T]) \leftarrow position(X) \wedge even(X) \wedge \neg\ member(X,[b|T],a)$

Positive Unfolding+

$negprop([a|T]) \leftarrow \neg\ member(0,[a|T],a)$

$negprop([a|T]) \leftarrow position(X) \wedge \neg\ even(X) \wedge \neg\ member(s(X),[a|T],a)$

$negprop([b|T]) \leftarrow \neg\ member(0,[b|T],a)$

$negprop([b|T]) \leftarrow position(X) \wedge even(X) \wedge \neg\ member(s(X),[b|T],a)$

Transformation into a Monadic ω-Program

$\boxed{\mathit{negprop}(L) \leftarrow \mathit{position}(X) \land \mathit{even}(X) \land \neg\, \mathit{member}(X,L,a)}$ non monadic

↓ *Instantiation*

negprop([a|T]) ← position(X) ∧ even(X) ∧ ¬ member(X,[a|T],a)
negprop([b|T]) ← position(X) ∧ even(X) ∧ ¬ member(X,[b|T],a)

↓ *Positive Unfolding+ ; Negative Unfolding*

negprop([a|T]) ← ¬ member(0,[a|T],a)
negprop([a|T]) ← position(X) ∧ ¬ even(X) ∧ ¬ $\boxed{\text{member(s(X),[a|T],a)}}$
negprop([b|T]) ← ¬ member(0,[b|T],a)
negprop([b|T]) ← position(X) ∧ even(X) ∧ ¬ member(s(X),[b|T],a)

H/a, S/a

member(0,[H|T],H) ←
$\boxed{\text{member(s(X),[H|T],S)}}$ ← member(X,T,S)

Transformation into a Monadic ω-Program

negprop(L) ← position(X) ∧ even(X) ∧ ¬ member(X,L,a)   non monadic

Instantiation

negprop([a|T]) ← position(X) ∧ even(X) ∧ ¬ member(X,[a|T],a)
negprop([b|T]) ← position(X) ∧ even(X) ∧ ¬ member(X,[b|T],a)

Positive Unfolding+ ; *Negative Unfolding*

negprop([a|T]) ← ¬ member(0,[a|T],a)
negprop([a|T]) ← position(X) ∧ ¬ even(X) ∧ ¬ member(X,T,a)
negprop([b|T]) ← ¬ member(0,[b|T],a)
negprop([b|T]) ← position(X) ∧ even(X) ∧ ¬ member(s(X),[b|T],a)

S/a

member(0,[H|T],H) ←
member(s(X),[H|T],S) ← member(X,T,S)

Transformation into a Monadic ω-Program

$\boxed{negprop(L) \leftarrow position(X) \wedge even(X) \wedge \neg\ member(X,L,a)}$ non monadic

Instantiation

negprop([a|T]) ← position(X) ∧ even(X) ∧ ¬ member(X,[a|T],a)
negprop([b|T]) ← position(X) ∧ even(X) ∧ ¬ member(X,[b|T],a)

Positive Unfolding+ ; Negative Unfolding+

negprop([a|T]) ← ¬ $\boxed{member(0,[a|T],a)}$
negprop([a|T]) ← position(X) ∧ ¬ even(X) ∧ ¬ member(X,T,a)
negprop([b|T]) ← ¬ $\boxed{member(0,[b|T],a)}$
negprop([b|T]) ← position(X) ∧ even(X) ∧ ¬ member(s(X),[b|T],a)

Transformation into a Monadic ω-Program

$\boxed{negprop(L) \leftarrow position(X) \land even(X) \land \neg\ member(X,L,a)}$ non monadic

$\downarrow$ ***Instantiation***

negprop([a|T]) ← position(X) ∧ even(X) ∧ ¬ member(X,[a|T],a)
negprop([b|T]) ← position(X) ∧ even(X) ∧ ¬ member(X,[b|T],a)

$\downarrow$ ***Positive Unfolding+ ; Negative Unfolding+***

negprop([a|T]) ← position(X) ∧ ¬ even(X) ∧ ¬ member(X,T,a)
negprop([b|T]) ←
negprop([b|T]) ← position(X) ∧ even(X) ∧ ¬ member(s(X),[b|T],a)

Transformation into a Monadic ω-Program

$negprop(L) \leftarrow position(X) \wedge even(X) \wedge \neg\ member(X,L,a)$ non monadic

Instantiation

$negprop([a|T]) \leftarrow position(X) \wedge even(X) \wedge \neg\ member(X,[a|T],a)$

$negprop([b|T]) \leftarrow position(X) \wedge even(X) \wedge \neg\ member(X,[b|T],a)$

Positive Unfolding+ ; Negative Unfolding+ ; Subsumption

$negprop([a|T]) \leftarrow position(X) \wedge \neg\ even(X) \wedge \neg\ member(X,T,a)$

$negprop([b|T]) \leftarrow$

subsumed by

$negprop([b|T]) \leftarrow position(X) \wedge even(X) \wedge \neg\ member(s(X),[b|T],a)$

Transformation into a Monadic ω-Program

$\boxed{negprop(L) \leftarrow position(X) \wedge even(X) \wedge \neg\ member(X,L,a)}$ non monadic

Instantiation

$negprop([a|T]) \leftarrow position(X) \wedge even(X) \wedge \neg\ member(X,[a|T],a)$
$negprop([b|T]) \leftarrow position(X) \wedge even(X) \wedge \neg\ member(X,[b|T],a)$

Positive Unfolding^{+} ; *Negative Unfolding*$^{+}$; *Subsumption*

$negprop([a|T]) \leftarrow position(X) \wedge \neg\ even(X) \wedge \neg\ member(X,T,a)$
$negprop([b|T]) \leftarrow$

Transformation into a Monadic ω-Program

negprop(L) $\leftarrow$ position(X) $\wedge$ even(X) $\wedge$ $\neg$ member(X,L,a) non monadic

Instantiation

negprop([a|T]) $\leftarrow$ position(X) $\wedge$ even(X) $\wedge$ $\neg$ member(X,[a|T],a)
negprop([b|T]) $\leftarrow$ position(X) $\wedge$ even(X) $\wedge$ $\neg$ member(X,[b|T],a)

Positive Unfolding+ ; *Negative Unfolding*+ ; *Subsumption*

negprop([a|T]) $\leftarrow$ position(X) $\wedge$ $\neg$ even(X) $\wedge$ $\neg$ member(X,T,a) non monadic
negprop([b|T]) $\leftarrow$ monadic

Transformation into a Monadic ω-Program

$\boxed{negprop(L) \leftarrow position(X) \wedge even(X) \wedge \neg\ member(X,L,a)}$ non monadic

$\downarrow$ ***Instantiation***

negprop([a|T]) ← position(X) ∧ even(X) ∧ ¬ member(X,[a|T],a)
negprop([b|T]) ← position(X) ∧ even(X) ∧ ¬ member(X,[b|T],a)

$\downarrow$ ***Positive Unfolding+ ; Negative Unfolding+ ; Subsumption***

negprop([a|T]) ← $\boxed{position(X) \wedge \neg\ even(X) \wedge \neg\ member(X,T,a)}$ non monadic
negprop([b|T]) ← monadic

$\downarrow$ ***Definition Introduction***

newp(T) ← $\boxed{position(X) \wedge \neg\ even(X) \wedge \neg\ member(X,T,a)}$

Transformation into a Monadic ω-Program

$\boxed{\text{negprop(L)} \leftarrow \text{position(X)} \wedge \text{even(X)} \wedge \neg\ \text{member(X,L,a)}}$ non monadic

Instantiation

negprop([a|T]) ← position(X) ∧ even(X) ∧ ¬ member(X,[a|T],a)
negprop([b|T]) ← position(X) ∧ even(X) ∧ ¬ member(X,[b|T],a)

Positive Unfolding+ ; Negative Unfolding+ ; Subsumption

negprop([a|T]) ← position(X) ∧ ¬ even(X) ∧ ¬ member(X,T,a) non monadic
negprop([b|T]) ← monadic

Definition Introduction

$\boxed{\text{newp(T)}}$ ← position(X) ∧ ¬ even(X) ∧ ¬ member(X,T,a)

Positive Folding

negprop([a|T]) ← $\boxed{\text{newp(T)}}$ monadic
negprop([b|T]) ← monadic

Transformation into a Monadic ω-Program

$\boxed{negprop(L) \leftarrow position(X) \wedge even(X) \wedge \neg\ member(X,L,a)}$ non monadic

Instantiation

negprop([a|T]) ← position(X) ∧ even(X) ∧ ¬ member(X,[a|T],a)
negprop([b|T]) ← position(X) ∧ even(X) ∧ ¬ member(X,[b|T],a)

Positive Unfolding+ ; Negative Unfolding+ ; Subsumption

negprop([a|T]) ← position(X) ∧ ¬ even(X) ∧ ¬ member(X,T,a)
negprop([b|T]) ←

Definition Introduction

$\boxed{newp(T) \leftarrow position(X) \wedge \neg\ even(X) \wedge \neg\ member(X,T,a)}$ non monadic

Positive Folding

negprop([a|T]) ← newp(T) monadic
negprop([b|T]) ← monadic

Transformation into a Monadic ω-Program

$\boxed{newp(L) \leftarrow position(X) \land \neg\ even(X) \land \neg\ member(X,L,a)}$ non monadic

Instantiation L / [a|T] ; L / [b|T]

$newp([a|T]) \leftarrow position(X) \land \neg\ even(X) \land \neg\ member(X,[a|T],a)$

$newp([b|T]) \leftarrow position(X) \land \neg\ even(X) \land \neg\ member(X,[b|T],a)$

Transformation into a Monadic ω-Program

$\boxed{newp(L) \leftarrow position(X) \wedge \neg\ even(X) \wedge \neg\ member(X,L,a)}$ non monadic

Instantiation L / [a|T] ; L / [b|T]

$newp([a|T]) \leftarrow \boxed{position(X)} \wedge \neg\ even(X) \wedge \neg\ member(X,[a|T],a)$

$newp([b|T]) \leftarrow \boxed{position(X)} \wedge \neg\ even(X) \wedge \neg\ member(X,[b|T],a)$

Transformation into a Monadic ω-Program

$\boxed{newp(L) \leftarrow position(X) \land \neg\ even(X) \land \neg\ member(X,L,a)}$ non monadic

$\downarrow$ ***Instantiation*** L / [a|T] ; L / [b|T]

$newp([a|T]) \leftarrow \boxed{position(X)} \land \neg\ even(X) \land \neg\ member(X,[a|T],a)$

$newp([b|T]) \leftarrow \boxed{position(X)} \land \neg\ even(X) \land \neg\ member(X,[b|T],a)$

$\downarrow$ ***Positive Unfolding*** +

$newp([a|T]) \leftarrow \neg\ even(0) \land \neg\ member(0,[a|T],a)$

$newp([a|T]) \leftarrow position(X) \land \neg\ even(s(X)) \land \neg\ member(s(X),[a|T],a)$

$newp([b|T]) \leftarrow \neg\ even(0) \land \neg\ member(0,[b|T],a)$

$newp([b|T]) \leftarrow position(X) \land \neg\ even(s(X)) \land \neg\ member(s(X),[b|T],a)$

Transformation into a Monadic ω-Program

$\boxed{newp(L) \leftarrow position(X) \wedge \neg\, even(X) \wedge \neg\, member(X,L,a)}$ non monadic

Instantiation L / [a|T] ; L / [b|T]

$newp([a|T]) \leftarrow position(X) \wedge \neg\, even(X) \wedge \neg\, member(X,[a|T],a)$
$newp([b|T]) \leftarrow position(X) \wedge \neg\, even(X) \wedge \neg\, member(X,[b|T],a)$

Positive Unfolding +

$newp([a|T]) \leftarrow \neg\, \boxed{even(0)} \wedge \neg\, \boxed{member(0,[a|T],a)}$
$newp([a|T]) \leftarrow position(X) \wedge \neg\, \boxed{even(s(X))} \wedge \neg\, \boxed{member(s(X),[a|T],a)}$
$newp([b|T]) \leftarrow \neg\, \boxed{even(0)} \wedge \neg\, \boxed{member(0,[b|T],a)}$
$newp([b|T]) \leftarrow position(X) \wedge \neg\, \boxed{even(s(X))} \wedge \neg\, \boxed{member(s(X),[b|T],a)}$

Transformation into a Monadic ω-Program

$$\boxed{newp(L) \leftarrow position(X) \wedge \neg\, even(X) \wedge \neg\, member(X,L,a)}$$

non monadic

Instantiation L / [a|T] ; L / [b|T]

$$newp([a|T]) \leftarrow position(X) \wedge \neg\, even(X) \wedge \neg\, member(X,[a|T],a)$$
$$newp([b|T]) \leftarrow position(X) \wedge \neg\, even(X) \wedge \neg\, member(X,[b|T],a)$$

Positive Unfolding **+**

$$newp([a|T]) \leftarrow \neg\, \boxed{even(0)} \wedge \neg\, \boxed{member(0,[a|T],a)}$$
$$newp([a|T]) \leftarrow position(X) \wedge \neg\, \boxed{even(s(X))} \wedge \neg\, \boxed{member(s(X),[a|T],a)}$$
$$newp([b|T]) \leftarrow \neg\, \boxed{even(0)} \wedge \neg\, \boxed{member(0,[b|T],a)}$$
$$newp([b|T]) \leftarrow position(X) \wedge \neg\, \boxed{even(s(X))} \wedge \neg\, \boxed{member(s(X),[b|T],a)}$$

Negative Unfolding **+**

$$newp([a|T]) \leftarrow position(X) \wedge even(X) \wedge \neg\, member(X,T,a)$$
$$newp([b|T]) \leftarrow position(X) \wedge even(X) \wedge \neg\, member(X,T,a)$$

Transformation into a Monadic ω-Program

$\boxed{newp(L) \leftarrow position(X) \wedge \neg\ even(X) \wedge \neg\ member(X,L,a)}$ non monadic

Instantiation L / [a|T] ; L / [b|T]

newp([a|T]) ← position(X) ∧ ¬ even(X) ∧ ¬ member(X,[a|T],a)
newp([b|T]) ← position(X) ∧ ¬ even(X) ∧ ¬ member(X,[b|T],a)

Positive Unfolding +

newp([a|T]) ← ¬ even(0) ∧ ¬ member(0,[a|T],a)
newp([a|T]) ← position(X) ∧ ¬ even(s(X)) ∧ ¬ member(s(X),[a|T],a)
newp([b|T]) ← ¬ even(0) ∧ ¬ member(0,[b|T],a)
newp([b|T]) ← position(X) ∧ ¬ even(s(X)) ∧ ¬ member(s(X),[b|T],a)

Negative Unfolding +

newp([a|T]) ← $\boxed{position(X) \wedge even(X) \wedge \neg\ member(X,T,a)}$
newp([b|T]) ← position(X) ∧ even(X) ∧ ¬ member(X,T,a)

Transformation into a Monadic ω-Program

$\boxed{newp(L) \leftarrow position(X) \land \neg\ even(X) \land \neg\ member(X,L,a)}$ non monadic

Instantiation L / [a|T] ; L / [b|T]

$newp([a|T]) \leftarrow position(X) \land \neg\ even(X) \land \neg\ member(X,[a|T],a)$

$newp([b|T]) \leftarrow position(X) \land \neg\ even(X) \land \neg\ member(X,[b|T],a)$

Positive Unfolding **+**

$newp([a|T]) \leftarrow \neg\ even(0) \land \neg\ member(0,[a|T],a)$

$newp([a|T]) \leftarrow position(X) \land \neg\ even(s(X)) \land \neg\ member(s(X),[a|T],a)$

$newp([b|T]) \leftarrow \neg\ even(0) \land \neg\ member(0,[b|T],a)$

$newp([b|T]) \leftarrow position(X) \land \neg\ even(s(X)) \land \neg\ member(s(X),[b|T],a)$

Negative Unfolding **+**

$newp([a|T]) \leftarrow \boxed{position(X) \land even(X) \land \neg\ member(X,T,a)}$

$newp([b|T]) \leftarrow position(X) \land even(X) \land \neg\ member(X,T,a)$

Positive Folding **+**

$newp([a|T]) \leftarrow \boxed{negprop(T)}$ $\boxed{negprop(T)} \leftarrow \boxed{position(X) \land even(X) \land \neg member(X,T,a)}$

Transformation into a Monadic ω-Program

$newp(L) \leftarrow position(X) \land \neg\ even(X) \land \neg\ member(X,L,a)$ non monadic

Instantiation L / [a|T] ; L / [b|T]

$newp([a|T]) \leftarrow position(X) \land \neg\ even(X) \land \neg\ member(X,[a|T],a)$

$newp([b|T]) \leftarrow position(X) \land \neg\ even(X) \land \neg\ member(X,[b|T],a)$

Positive Unfolding +

$newp([a|T]) \leftarrow \neg\ even(0) \land \neg\ member(0,[a|T],a)$

$newp([a|T]) \leftarrow position(X) \land \neg\ even(s(X)) \land \neg\ member(s(X),[a|T],a)$

$newp([b|T]) \leftarrow \neg\ even(0) \land \neg\ member(0,[b|T],a)$

$newp([b|T]) \leftarrow position(X) \land \neg\ even(s(X)) \land \neg\ member(s(X),[b|T],a)$

Negative Unfolding +

$newp([a|T]) \leftarrow position(X) \land even(X) \land \neg\ member(X,T,a)$

$newp([b|T]) \leftarrow \boxed{position(X) \land even(X) \land \neg\ member(X,T,a)}$

Positive Folding +

$\boxed{negprop(T)} \leftarrow \boxed{position(X) \land \neg even(X) \land \neg member(X,T,a)}$

$newp([a|T]) \leftarrow negprop(T)$

$newp([b|T]) \leftarrow \boxed{negprop(T)}$

Transformation into a Monadic ω-Program

$\boxed{newp(L) \leftarrow position(X) \wedge \neg\ even(X) \wedge \neg\ member(X,L,a)}$ non monadic

Instantiation L / [a|T] ; L / [b|T]

newp([a|T]) ← position(X) ∧ ¬ even(X) ∧ ¬ member(X,[a|T],a)
newp([b|T]) ← position(X) ∧ ¬ even(X) ∧ ¬ member(X,[b|T],a)

Positive Unfolding **+**

newp([a|T]) ← ¬ even(0) ∧ ¬ member(0,[a|T],a)
newp([a|T]) ← position(X) ∧ ¬ even(s(X)) ∧ ¬ member(s(X),[a|T],a)
newp([b|T]) ← ¬ even(0) ∧ ¬ member(0,[b|T],a)
newp([b|T]) ← position(X) ∧ ¬ even(s(X)) ∧ ¬ member(s(X),[b|T],a)

Negative Unfolding **+**

newp([a|T]) ← position(X) ∧ even(X) ∧ ¬ member(X,T,a)
newp([b|T]) ← position(X) ∧ even(X) ∧ ¬ member(X,T,a)

Positive Folding **+**

newp([a|T]) ← negprop(T) monadic
newp([b|T]) ← negprop(T) monadic

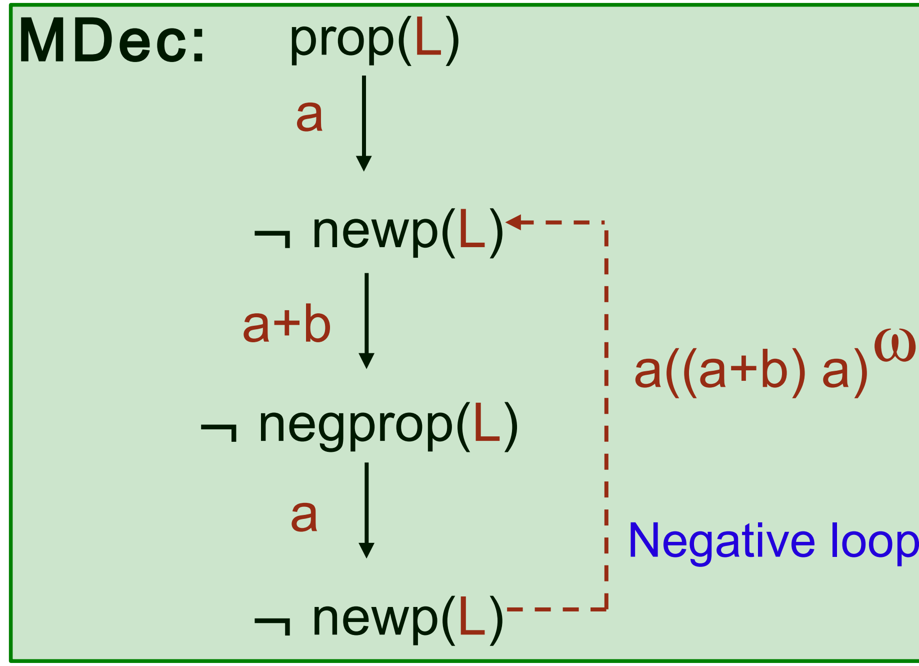
Monadic ω-Program T

T:

prop([a|T]) ← ¬ newp(T)

newp([a|T]) ← negprop(T)
newp([b|T]) ← negprop(T)

negprop([a|T]) ← newp(T)
negprop([b|T]) ←



By the soundness of Mdec: M(T) |= ∃L prop(L)

By the correctness of the transformation: M(EvenAs) |= ∃L prop(L)

Correctness of Transformations

Given:

- a transformation sequence $P_0 \to P_1 \to \dots \to P_n$ and
- the set $Defs_n$ of clauses introduced by the definition introduction rule

The transformation sequence is correct if

1. $P_0 \cup Defs_n$ and P_n are locally stratified
2. $M(P_0 \cup Defs_n) = M(P_n)$

Sufficient Conditions for Correctness

Three kinds of conditions guarantee correctness:

1. Conditions for the preservation of local stratification;

2. Conditions on single transformation steps;

3. Conditions on the transformation sequence.

Preservation of Local Stratification

- Consider a transformation sequence $P_0 \rightarrow P_1 \rightarrow . . . \rightarrow P_n$ and a stratification σ.
  - A definition introduction preserves σ if it introduces a set of clauses that are locally stratified w.r.t. σ.
  - A positive folding $(H \leftarrow ... G ...) \rightarrow (H \leftarrow ... K ...)$ preserves σ if $H \geq_\sigma K$.
  - A negative folding $(H \leftarrow ... G ...) \rightarrow (H \leftarrow ... \neg K ...)$ preserves σ if $H >_\sigma K$.
- Lemma: If P_0 is locally stratified w.r.t. σ and every application of definition introduction, positive folding, and negative folding preserves σ, then $P_0 \cup Defs_n$ and P_n are locally stratified w.r.t. σ.

Conditions on Single Transformation Steps

1. When applying the positive or negative unfolding, at most one argument of type ilist per predicate.

$$P_k:\ p \leftarrow \boxed{q(X,[a|X])} \qquad \xrightarrow{\text{Positive Unfolding}} \qquad P_{k+1}:\ \cancel{p \leftarrow q(X,[a|X])} \quad ??$$
$$q(Y,Y) \leftarrow \qquad\qquad\qquad\qquad q(Y,Y) \leftarrow$$

q(X,[a|X]) is not unifiable with q(Y,Y) by the standard unification algorithm.

This condition can be dropped by using unification of rational trees [Colmerauer 82].

2. When applying negative unfolding, no existential variables must occur in the program clauses used for unfolding.

$$P_k:\ p \leftarrow \neg\, \boxed{q} \qquad \xrightarrow{\text{Negative Unfolding}} \qquad P_{k+1}:\ p \leftarrow \neg\, \boxed{r(X)} \quad ??$$
$$q \leftarrow r(X) \qquad\qquad\qquad\qquad q \leftarrow r(X)$$
$$r(a) \leftarrow \qquad\qquad\qquad\qquad r(a) \leftarrow$$

. . . more local conditions are needed for correctness

An Incorrect Transformation Sequence [Seki 09]

P_0: $f \leftarrow \boxed{m} \land \neg e$ $\quad \boxed{m} \leftarrow$ $\quad e \leftarrow k$ $\quad e \leftarrow \neg m$ $\quad e \leftarrow e$

Positive Unfolding

P_1: $f \leftarrow \neg e$ $\quad m \leftarrow$ $\quad e \leftarrow k$ $\quad e \leftarrow \neg m$ $\quad e \leftarrow e$

An Incorrect Transformation Sequence [Seki 09]

P_0: $f \leftarrow m \wedge \neg e \qquad m \leftarrow \qquad e \leftarrow k \qquad e \leftarrow \neg m \qquad e \leftarrow e$

Positive Unfolding

P_1: $f \leftarrow \neg \boxed{e} \qquad m \leftarrow \qquad \boxed{e} \leftarrow \boxed{k} \qquad \boxed{e} \leftarrow \boxed{\neg m} \qquad \boxed{e} \leftarrow \boxed{e}$

Negative Unfolding

P_2: $f \leftarrow \neg \boxed{k} \wedge \cancel{\neg}\boxed{\cancel{\neg} m} \wedge \neg \boxed{e} \qquad m \leftarrow \qquad e \leftarrow k \qquad e \leftarrow \neg m \qquad e \leftarrow e$

An Incorrect Transformation Sequence [Seki 09]

P_0: $f \leftarrow m \wedge \neg e \qquad m \leftarrow \qquad e \leftarrow k \qquad e \leftarrow \neg m \qquad e \leftarrow e$

$\Big\downarrow$ Positive Unfolding

P_1: $f \leftarrow \neg e \qquad m \leftarrow \qquad e \leftarrow k \qquad e \leftarrow \neg m \qquad e \leftarrow e$

$\Big\downarrow$ Negative Unfolding

P_2: $f \leftarrow \neg k \wedge m \wedge \neg e \qquad m \leftarrow \qquad e \leftarrow k \qquad e \leftarrow \neg m \qquad e \leftarrow e$

An Incorrect Transformation Sequence [Seki 09]

P_0:	$\boxed{f} \leftarrow \boxed{m \wedge \neg e}$	$m \leftarrow$	$e \leftarrow k$	$e \leftarrow \neg m$	$e \leftarrow e$
Positive Unfolding $\downarrow$					
P_1:	$f \leftarrow \neg e$	$m \leftarrow$	$e \leftarrow k$	$e \leftarrow \neg m$	$e \leftarrow e$
Negative Unfolding $\downarrow$					
P_2:	$f \leftarrow \neg k \wedge \boxed{m \wedge \neg e}$	$m \leftarrow$	$e \leftarrow k$	$e \leftarrow \neg m$	$e \leftarrow e$
Positive Folding $\downarrow$					
P_3:	$f \leftarrow \neg k \wedge \boxed{f}$	$m \leftarrow$	$e \leftarrow k$	$e \leftarrow \neg m$	$e \leftarrow e$

An Incorrect Transformation Sequence [Seki 09]

P_0: $f \leftarrow m \wedge \neg e \qquad m \leftarrow \qquad e \leftarrow k \qquad e \leftarrow \neg m \qquad e \leftarrow e$

$\Big\downarrow$ Positive Unfolding

P_1: $f \leftarrow \neg e \qquad m \leftarrow \qquad e \leftarrow k \qquad e \leftarrow \neg m \qquad e \leftarrow e$

$\Big\downarrow$ Negative Unfolding

P_2: $f \leftarrow \neg k \wedge m \wedge \neg e \qquad m \leftarrow \qquad e \leftarrow k \qquad e \leftarrow \neg m \qquad e \leftarrow e$

$\Big\downarrow$ Positive Folding

P_3: $f \leftarrow \neg k \wedge f \qquad m \leftarrow \qquad e \leftarrow k \qquad e \leftarrow \neg m \qquad e \leftarrow e$

$f \in M(P_0)$ and $f \notin M(P_3)$

A Condition on Transformation Sequences (1)

- Condition (NU): The negative unfolding rule can be applied only if it does not increase the number of positive occurrences of atoms in bodies of clauses. [Seki 2009]

- Condition (NU) rules out the incorrect transformation sequence

P_0: $f \leftarrow m \wedge \neg e \qquad m \leftarrow \qquad e \leftarrow k \qquad e \leftarrow \neg m \qquad e \leftarrow e$

Positive Unfolding

P_1: $f \leftarrow \neg e \qquad m \leftarrow \qquad e \leftarrow k \qquad e \leftarrow \neg m \qquad e \leftarrow e$

~~Negative Unfolding~~

positive occurrences of atoms increased

P_2: $f \leftarrow \neg k \wedge \boxed{m} \wedge \neg e \qquad m \leftarrow \qquad e \leftarrow k \qquad e \leftarrow \neg m \qquad e \leftarrow e$

Positive Folding

P_3: $f \leftarrow \neg k \wedge f \qquad m \leftarrow \qquad e \leftarrow k \qquad e \leftarrow \neg m \qquad e \leftarrow e$

A Condition on Transformation Sequences (2)

Condition (NU) prevents proving prop in the EvenAs example.

$$\boxed{newp(L) \leftarrow position(X) \wedge \neg\ even(X) \wedge \neg\ member(X,L,a)}$$

Instantiation L / [a|T] ; L / [b|T]

$$newp([a|T]) \leftarrow position(X) \wedge \neg\ even(X) \wedge \neg\ member(X,[a|T],a)$$
$$newp([b|T]) \leftarrow position(X) \wedge \neg\ even(X) \wedge \neg\ member(X,[b|T],a)$$

Positive Unfolding $^{+}$

$$newp([a|T]) \leftarrow \neg\ even(0) \wedge \neg\ member(0,[a|T],a)$$
$$newp([a|T]) \leftarrow position(X) \wedge \neg\ \boxed{even(s(X))} \wedge \neg\ member(s(X),[a|T],a)$$
$$newp([b|T]) \leftarrow \neg\ even(0) \wedge \neg\ member(0,[b|T],a)$$
$$newp([b|T]) \leftarrow position(X) \wedge \neg\ even(s(X)) \wedge \neg\ member(s(X),[b|T],a)$$

Negative Unfolding $^{+}$

positive occurrences of atoms increased

$$newp([a|T]) \leftarrow position(X) \wedge \boxed{even(X)} \wedge \neg\ member(X,T,a)$$
$$newp([b|T]) \leftarrow position(X) \wedge even(X) \wedge \neg\ member(X,T,a)$$

Relaxing the Condition (NU)

- Given $H \leftarrow G$, an atom A in G is σ-maximal if, for all literals L in G, $A \geq_\sigma L$
- $H \leftarrow G$ is σ-tight if there is a σ-maximal atom A in G such that s.t. $H =_\sigma A$
- Positive folding of a clause γ using a clause δ is admissible if
  (3.1) δ is σ-tight and
  (3.2) γ has been derived by a transformation sequence containing at least one positive unfolding wrt a σ-maximal atom

P_0

$\downarrow^*$

P_i

$\downarrow$ Definition introduction

P_{i+1} δ: $K \leftarrow B$ σ-tight

$\downarrow^*$

P_j α: $H_1 \leftarrow A \wedge G$ A is σ-maximal

$\downarrow$ Positive Unfolding wrt A

P_{j+1} β

$\downarrow^*$

P_k γ: $H_2 \leftarrow B\theta \wedge G$

$\downarrow$ Positive Folding using δ

P_{k+1} η: $H_2 \leftarrow K\theta \wedge G$

Correctness of Transformation Sequences

- A transformation sequences is admissible if:

  1. local stratification is preserved
  2. every application of positive folding is admissible

- Theorem: Every admissible transformation sequence is correct.

- Proof: A suitable measure on proof trees does not increase and, thus, finiteness of proof trees is preserved

Seki's Example Revisited

not σ-tight

not σ-maximal

$e >_\sigma m$

P_0:	$\boxed{f \leftarrow \boxed{m} \wedge \neg e}$	$m \leftarrow$	$e \leftarrow k$	$e \leftarrow \neg m$	$e \leftarrow e$
Positive Unfolding ↓					
P_1:	$f \leftarrow \neg e$	$m \leftarrow$	$e \leftarrow k$	$e \leftarrow \neg m$	$e \leftarrow e$
Negative Unfolding ↓					
P_2:	$f \leftarrow \neg k \wedge m \wedge \neg e$	$m \leftarrow$	$e \leftarrow k$	$e \leftarrow \neg m$	$e \leftarrow e$
Positive Folding ↓	not admissible				
P_3:	$f \leftarrow \neg k \wedge f$	$m \leftarrow$	$e \leftarrow k$	$e \leftarrow \neg m$	$e \leftarrow e$

EvenAs Revisited

Take: newp(L) $=_\sigma$ position(X), position(X) $>_\sigma$ even(X), position(X) $>_\sigma$ member(X,L,a)

newp(L) $\leftarrow$ position(X) $\wedge$ $\neg$ even(X) $\wedge$ $\neg$ member(X,L,a) $\boxed{\sigma\text{-tight}}$

↓ ***Instantiation*** L / [a|T] ; L / [b|T]

newp([a|T]) $\leftarrow$ position(X) $\wedge$ $\neg$ even(X) $\wedge$ $\neg$ member(X,[a|T],a)
newp([b|T]) $\leftarrow$ position(X) $\wedge$ $\neg$ even(X) $\wedge$ $\neg$ member(X,[b|T],a)

↓ ***Positive Unfolding*** **+** $\boxed{\text{wrt } \sigma\text{-maximal atom position(X)}}$

newp([a|T]) $\leftarrow$ $\neg$ even(0) $\wedge$ $\neg$ member(0,[a|T],a)
newp([a|T]) $\leftarrow$ position(X) $\wedge$ $\neg$ even(s(X)) $\wedge$ $\neg$ member(s(X),[a|T],a)
newp([b|T]) $\leftarrow$ $\neg$ even(0) $\wedge$ $\neg$ member(0,[b|T],a)
newp([b|T]) $\leftarrow$ position(X) $\wedge$ $\neg$ even(s(X)) $\wedge$ $\neg$ member(s(X),[b|T],a)

↓ ***Negative Unfolding*** **+**

newp([a|T]) $\leftarrow$ position(X) $\wedge$ even(X) $\wedge$ $\neg$ member(X,T,a)
newp([b|T]) $\leftarrow$ position(X) $\wedge$ even(X) $\wedge$ $\neg$ member(X,T,a)

↓ ***Positive Folding*** **+** $\boxed{\text{admissible}}$

newp([a|T]) $\leftarrow$ negprop(T)
newp([b|T]) $\leftarrow$ negprop(T)

More Complex Examples

- Emptiness of Büchi Automata
- Containment between regular languages
- CTL* [LOPSTR'09]
- All examples have been worked out interactively by the MAP system http://www.iasi.cnr.it/~proietti/system.html

ω-regular Languages

$e ::= a \mid e_1 e_2 \mid e_1 + e_2 \mid e^*$ (regular expressions over Σ)

$f ::= e^\omega \mid e_1 e_2^\omega \mid f_1 + f_2$ (ω-regular expressions over Σ)

P:

$acc(E,[E]) \leftarrow symbol(E)$

$acc(E_1E_2,X) \leftarrow append(X_1,X_2,X) \wedge acc(E_1,X_1) \wedge acc(E_2,X_2)$

$acc(E_1+E_2,X) \leftarrow acc(E_1,X)$

$acc(E_1+E_2,X) \leftarrow acc(E_2,X)$

$acc(E^*,[\,]) \leftarrow$

$acc(E^*,X) \leftarrow append(X_1,X_2,X) \wedge acc(E,X_1) \wedge acc(E^*,X_2)$

$\omega\text{-}acc(E^\omega,X) \leftarrow \neg\, p_1(E,X)$

$p_1(E,X) \leftarrow nat(M) \wedge \neg\, p_2(E,M,X)$

$p_2(E,M,X) \leftarrow geq(N,M) \wedge prefix(X,N,V) \wedge acc(E^*,V)$

. . .

where:

$\omega\text{-}acc(E^\omega, X) \leftrightarrow \forall M(nat(M) \rightarrow \exists N \exists V(geq(N,M) \wedge prefix(X,N,V) \wedge acc(E^*,V)))$

Containment of ω-regular Languages (1)

$\mathrm{expr}_1(X) \leftarrow \omega\text{-acc}(a^\omega, X)$

$\mathrm{expr}_2(X) \leftarrow \omega\text{-acc}((b^*a)^\omega, X)$

$\mathrm{prop}(X) \leftarrow \mathrm{expr}_1(X) \wedge \neg\, \mathrm{expr}_2(X)$ $\quad a^\omega \nsubseteq (b^*a)^\omega$

Containment of ω-regular Languages (2)

After transformation:

T:

$$\begin{aligned}
&prop([a|X]) \leftarrow \neg\ new_1(X) \wedge new_2(X)\\
&new_1([a|X]) \leftarrow new_1(X)\\
&new_1([b|X]) \leftarrow\\
&new_2([a|X]) \leftarrow new_2(X)\\
&new_2([b|X]) \leftarrow new_3(X)\\
&new_3([a|X]) \leftarrow new_2(X)\\
&new_3([b|X]) \leftarrow new_3(X)\\
&new_3([b|X]) \leftarrow \neg\ new_4(X)\\
&new_4([a|X]) \leftarrow\\
&new_4([b|X]) \leftarrow new_4(X)
\end{aligned}$$

Future Work

- Strategies for automating the transformation from ω-programs to monadic ω-programs
- Use of constraints to avoid explicit state representation

  For instance,

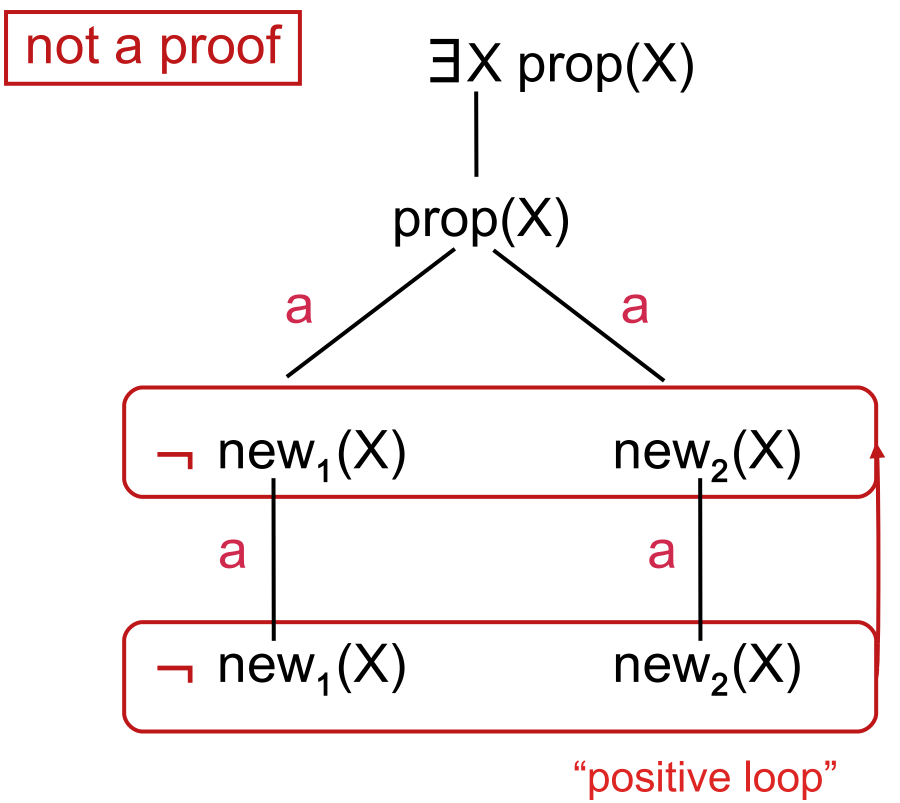
  $$p([H|T]) \leftarrow H \geq 1 \wedge q(T)$$

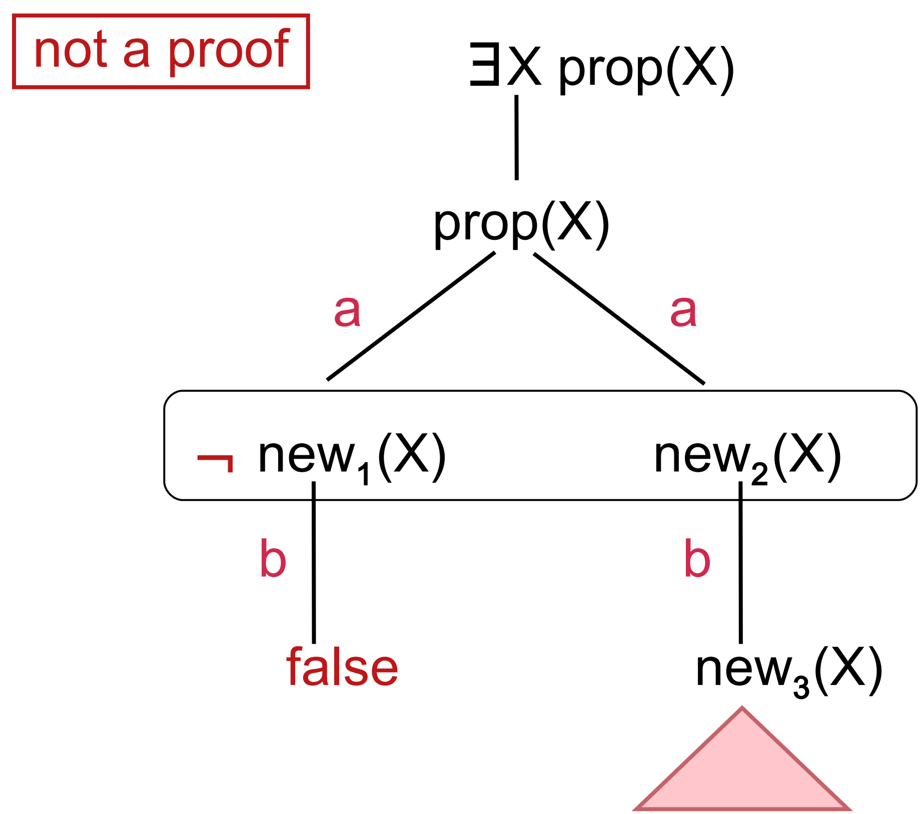
- Infinite state model checking

Containment of ω-regular Languages (3)

T:

$$\mathrm{prop}([a|X]) \leftarrow \neg\, \mathrm{new}_1(X) \wedge \mathrm{new}_2(X)$$
$$\mathrm{new}_1([a|X]) \leftarrow \mathrm{new}_1(X)$$
$$\mathrm{new}_1([b|X]) \leftarrow$$
$$\mathrm{new}_2([a|X]) \leftarrow \mathrm{new}_2(X)$$
$$\mathrm{new}_2([b|X]) \leftarrow \mathrm{new}_3(X)$$
$$\cdots$$





Containment of ω-regular Languages (4)

T:

$prop([a|X]) \leftarrow \neg\ new_1(X) \wedge new_2(X)$

$new_1([a|X]) \leftarrow new_1(X)$

$new_1([b|X]) \leftarrow$

$new_2([a|X]) \leftarrow new_2(X)$

$new_2([b|X]) \leftarrow new_3(X)$

...

$\boxed{\text{not a proof}}$

$\exists X\ prop(X)$

$prop(X)$

b

false

Thus, $a^\omega \subseteq (b^*a)^\omega$.