A Theory of Totally Correct Logic Program Transformations

Alberto Pettorossi

DISP - University of Roma Tor Vergata
Via del Politecnico 1
00133 Roma, Italy
pettorossi@info.uniroma2.it

Maurizio Proietti

IASI-CNR
Viale Manzoni 30
00185 Roma, Italy
proietti@iasi.rm.cnr.it

PEPM'04, Verona, Italy

August 24-25, 2004

Plan

1. Introduction: Correctness of rule-based transformations of logic programs

2. Clause Replacement: a general transformation rule

3. Partial Correctness of Clause Replacement

4. Total Correctness via Partial Correctness + Termination
   (Unique Fixpoint Principle [Mc Carthy 70])

5. Total Correctness via Well-Founded Annotations
   (our method)

6. Related work: Improvement [Tamaki-Sato 84, Sands 94, 97]
   - Counting [Tamaki-Sato 86, Kanamori-Fujita 87, Roychoudhury 04]
   - Simultaneous Replacement [Bossi-Cocco-Etalle 96]
   - Semantic delay [Bossi-Cocco-Etalle 92]
   - Post-hoc termination check [Cook-Gallagher 94]

1. Rule-Based Program Transformation

- Rule-based transformations of logic programs
  [Burstall-Darlington 77, Tamaki-Sato 84]

$$P_1 \overset{R_1}{\longmapsto} P_2 \overset{R_2}{\longmapsto} \cdots \overset{R_{n-1}}{\longmapsto} P_n$$

where $R_1, R_2, \ldots, R_{n-1}$ are transformation rules (unfolding, folding, goal replacement, ...)

- Correctness wrt the least Hebrand model semantics

Partial Correctness: $M(P_1) \supseteq M(P_n)$

Increasingness: $M(P_1) \subseteq M(P_n)$

Total Correctness: $M(P_1) = M(P_n)$

An Example of Program Transformation

- The Goal Replacement Rule

  if $M(P_1) \models \forall(G_1 \leftrightarrow G_2)$ goal equivalence

  P_1: H$\leftarrow$...$\boxed{G_1}$... $\longmapsto$ P_2: H$\leftarrow$...$\boxed{G_2}$...

- Example

  $M(P_1) \models q \leftrightarrow r$

  P_1: p $\leftarrow \boxed{q}$ $\longmapsto$ P_2: p $\leftarrow \boxed{r}$
  q $\leftarrow$ r q $\leftarrow$ r
  r $\leftarrow$ r $\leftarrow$

The transformation is totally correct: $M(P_1)=M(P_2)$

Goal Replacement is not Always Totally Correct

- Total correctness is not a consequence of goal equivalence

$M(P_1) \models q \leftrightarrow p$

P_1: $p \leftarrow \boxed{q}$ $\longmapsto$ P_2: $p \leftarrow \boxed{p}$

$q \leftarrow r$ $q \leftarrow r$

$r \leftarrow$ $r \leftarrow$

$M(P_1)=\{p,q\} \neq M(P_2)=\{q\}$

A Proof-Theoretic Solution

[Tamaki-Sato 84, Sands 94]

The Goal Replacement Rule revisited

if (1) $M(P_1) \models \forall(G_1 \leftrightarrow G_2)$

and (2) G_2 improves G_1:

for all ground instances $G_1\vartheta_1$ and $G_2\vartheta_2$,
the smallest proof of $G_1\vartheta_1$ using P_1 (if any)
is not smaller than
the smallest proof of $G_2\vartheta_2$ using P_1

P_1: H←... $\boxed{G_1}$ ... ⟼ P_2: H←... $\boxed{G_2}$ ...

Program Transformation Examples Revisited

P_1: p ← q
q ← r
r ←

$M(P_1) \models q \leftrightarrow r$
proof size: 2 > 1 ***ok***

P_2: p ← r
q ← r
r ←

$M(P_1) = M(P_2)$

P_1: p ← q
q ← r
r ←

$M(P_1) \models q \leftrightarrow p$
proof size: 2 < 3 ***not ok***

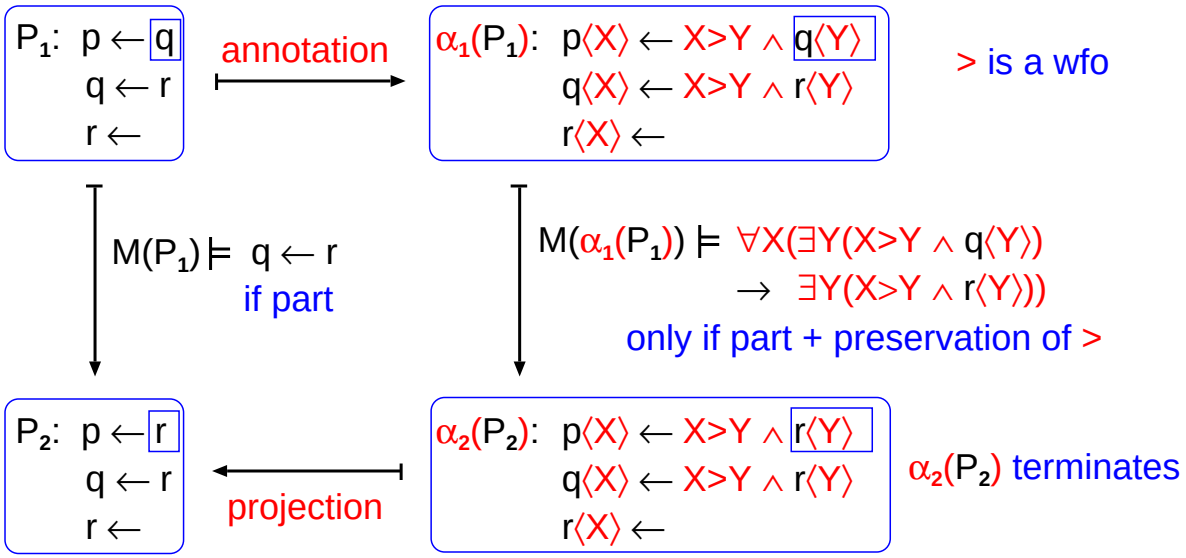
P_2: p ← p
q ← r
r ←

$M(P_1) \neq M(P_2)$

How to prove improvement?
- a proof system for functional programs [Sands 97]
- no first order logic method

Correctness via Well-Founded Annotations

Proving total correctness via first order logic



$M(P_1)=M(P_2)$ by Mc Carthy's Unique Fixpoint Principle + properties of annotation/projection

2. Clause Implication and Equivalence

Γ_1: $p(X) \leftarrow G_1$ ... $p(X) \leftarrow G_m$

Γ_2: $p(X) \leftarrow Q_1$ ... $p(X) \leftarrow Q_n$

- For an interpretation $\Im$,

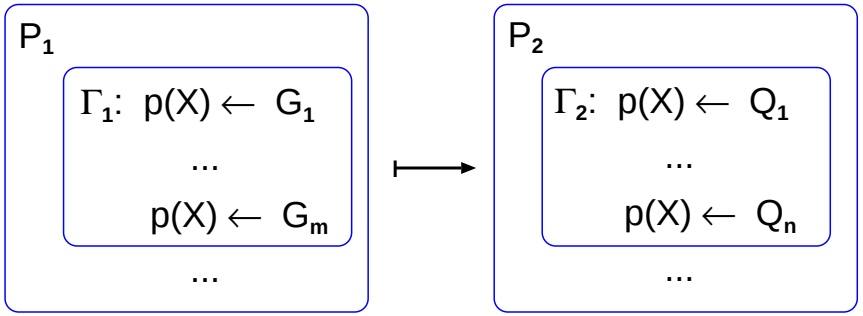
$\Im \models \Gamma_1 \Rightarrow \Gamma_2$ iff $\Im \models \forall X(\exists Y(G_1 \vee ... \vee G_m) \leftarrow \exists Z(Q_1 \vee ... \vee Q_n))$

where $Y=vars(G_1 \vee ... \vee G_m) - X$ $Z=vars(Q_1 \vee ... \vee Q_n) - X$

(X,Y,Z are tuples of distinct variables)

- $\Leftarrow$ and $\Leftrightarrow$ are similarly defined

Clause Replacement

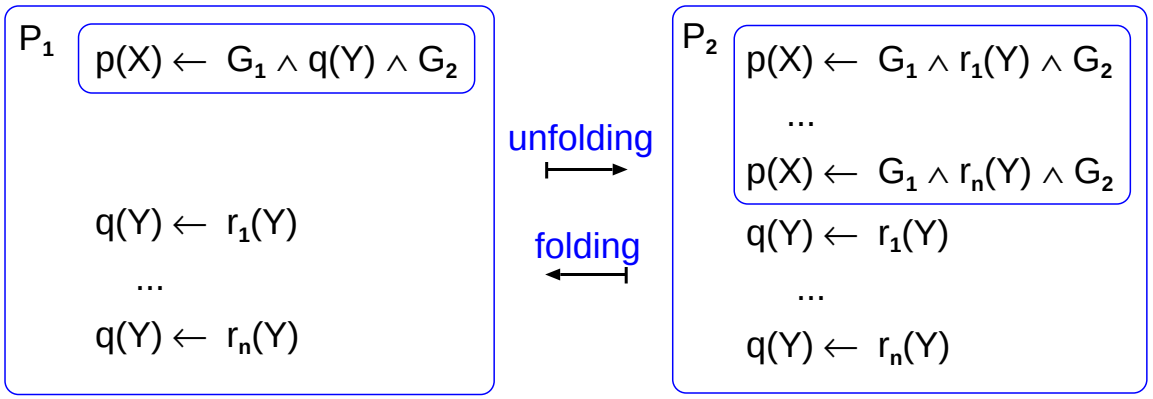


Clause replacement is:

- implication-based iff $M(P_1) \models \Gamma_1 \Rightarrow \Gamma_2$
- reverse-implication-based iff $M(P_1) \models \Gamma_1 \Leftarrow \Gamma_2$
- equivalence-based iff $M(P_1) \models \Gamma_1 \Leftrightarrow \Gamma_2$

Completeness: Every totally correct transformation can be realized by the clause replacement rule

Unfolding and Folding



Unfolding and folding are equivalence-based clause replacements

unfolding: $M(P_1) \models \forall Y\ (q(Y) \leftrightarrow r_1(Y) \vee ... \vee r_n(Y))$

folding: $M(P_2) \models \forall Y\ (r_1(Y) \vee ... \vee r_n(Y) \leftrightarrow q(Y))$

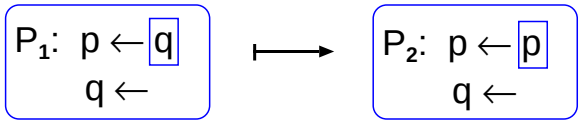
3. Partial Correctness

- Partial correctness is a consequence of clause implication:

$$P_1: \ldots \Gamma_1 \ldots \longmapsto P_2: \ldots \Gamma_2 \ldots$$

if $M(P_1) \models \Gamma_1 \Rightarrow \Gamma_2$ then $\boxed{M(P_1) \supseteq M(P_2)}$

- Increasingness is not a consequence of clause equivalence



$M(P_1) \models \{p \leftarrow q\} \Leftrightarrow \{p \leftarrow p\}$ (because $M(P_1) \models q \leftrightarrow p$)

$M(P_1)=\{p,q\} \not\supseteq M(P_2)=\{q\}$

Termination of p is not preserved

4. Unique Fixpoint Principle

- P terminates iff for all ground G, $P \cup \{\leftarrow G\}$ has a finite SLD tree

  strong termination [Bezem 93]

- If P terminates, then the immediate consequence operator T_P has a unique fixpoint

- Increasingness is a consequence of reverse clause implication and termination

$$P_1: \ldots \Gamma_1 \ldots \longmapsto P_2: \ldots \Gamma_2 \ldots$$

if $M(P_1) \models \Gamma_1 \Leftarrow \Gamma_2$ and P_2 terminates, then $\boxed{M(P_1) \subseteq M(P_2)}$

- Total Correctness:

  if $M(P_1) \models \Gamma_1 \Leftrightarrow \Gamma_2$ and P_2 terminates, then $\boxed{M(P_1) = M(P_2)}$

A Bad Definition of Clause Equivalence

- $\Leftrightarrow$ is based on equivalence of disjunctions of bodies

- Logical equivalence of clauses + termination does not ensure total correctness.

$$P_1: \ldots \Gamma_1 \ldots \longmapsto P_2: \ldots \Gamma_2 \ldots$$

$M(P_1) \models \Gamma_1 \leftrightarrow \Gamma_2$ and P_2 terminates does not imply $M(P_1) = M(P_2)$

- Example.

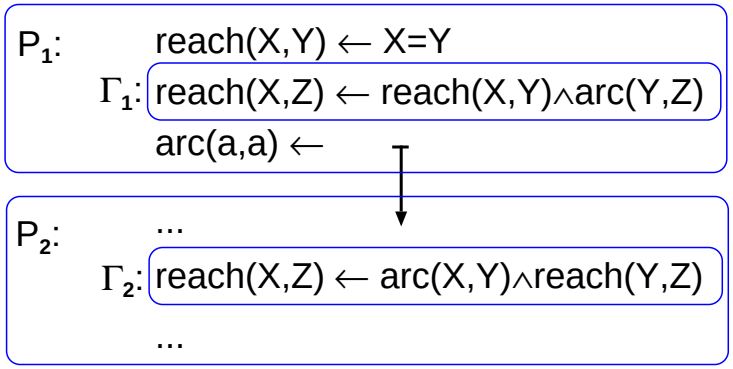
$$P_1: \; p \leftarrow \qquad \longmapsto \qquad P_2: \; p \leftarrow$$
$$q \leftarrow p$$

$M(P_1) \models (q \leftarrow p) \leftrightarrow \text{true}$ and P_2 terminates

$M(P_1)=\{p,q\} \neq M(P_2)=\{p\}$

A limitation of the Unique Fixpoint Principle

The total correctness of a transformation cannot be proved by the Unique Fixpoint Principle when the final program does not terminate



- $M(P_1) \models \Gamma_1 \Leftrightarrow \Gamma_2$
  because
  $M(P_1) \models \forall X \forall Z(\exists Y(reach(X,Y) \wedge arc(Y,Z)) \leftrightarrow \exists Y(arc(X,Y) \wedge reach(Y,Z)))$
- P_2 does not terminate for the query reach(a,b).
  $M(P_1)=M(P_2)$ cannot be proved by the Unique Fixpoint Principle. ☹

5. Well-Founded Annotations

Making programs terminate by adding suitable arguments and relations between them

P: $p \leftarrow q(b)$ $\quad\longmapsto\quad$ $\alpha(P)$: $p\langle X\rangle \leftarrow X>Y \wedge q(b)\langle Y\rangle$

$q(a) \leftarrow$ $\qquad$ $q(a)\langle X\rangle \leftarrow$

$q(b) \leftarrow q(b)\wedge r(b)$ $\qquad$ $q(b)\langle X\rangle \leftarrow X>Y+Z \wedge q(b)\langle Y\rangle \wedge r(b)\langle Z\rangle$

$r(b) \leftarrow$ $\qquad$ $r(b)\langle X\rangle \leftarrow$

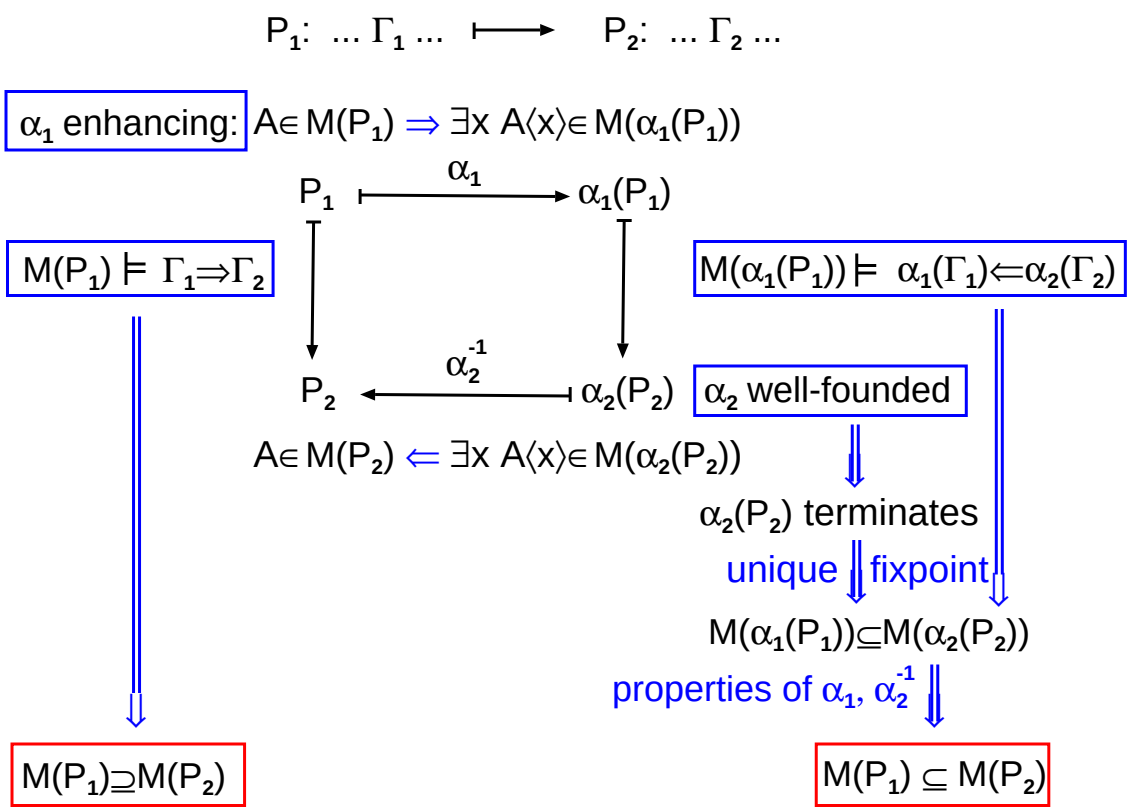
Definition. For any well-founded order $(S,>)$

α is enhancing: if $A \in M(P)$ then $\exists n \in S\ A\langle n\rangle \in M(\alpha(P))$

α is well-founded: for all annotated clauses $H\langle X\rangle \leftarrow c(X,Y) \ldots B\langle Y\rangle \ldots$ $\models \forall(c(X,Y) \rightarrow X>Y)$. Thus, $\alpha(P)$ terminates for all ground $A\langle n\rangle$

Note that P does not terminate for the query p

Total Correctness via Well-Founded Annotations



Reachability Revisited

$\alpha_1(P_1)$ $\alpha_1(\Gamma_1)$: reach(X,Z)⟨K⟩ ← K>N ∧ reach(X,Y)⟨N⟩ ∧ arc(Y,Z)

$\alpha_2(P_2)$ $\alpha_2(\Gamma_2)$: reach(X,Z)⟨K⟩ ← K>N ∧ arc(X,Y) ∧ reach(Y,Z)⟨N⟩

The transformation $P_1 \longmapsto P_2$ is totally correct because:

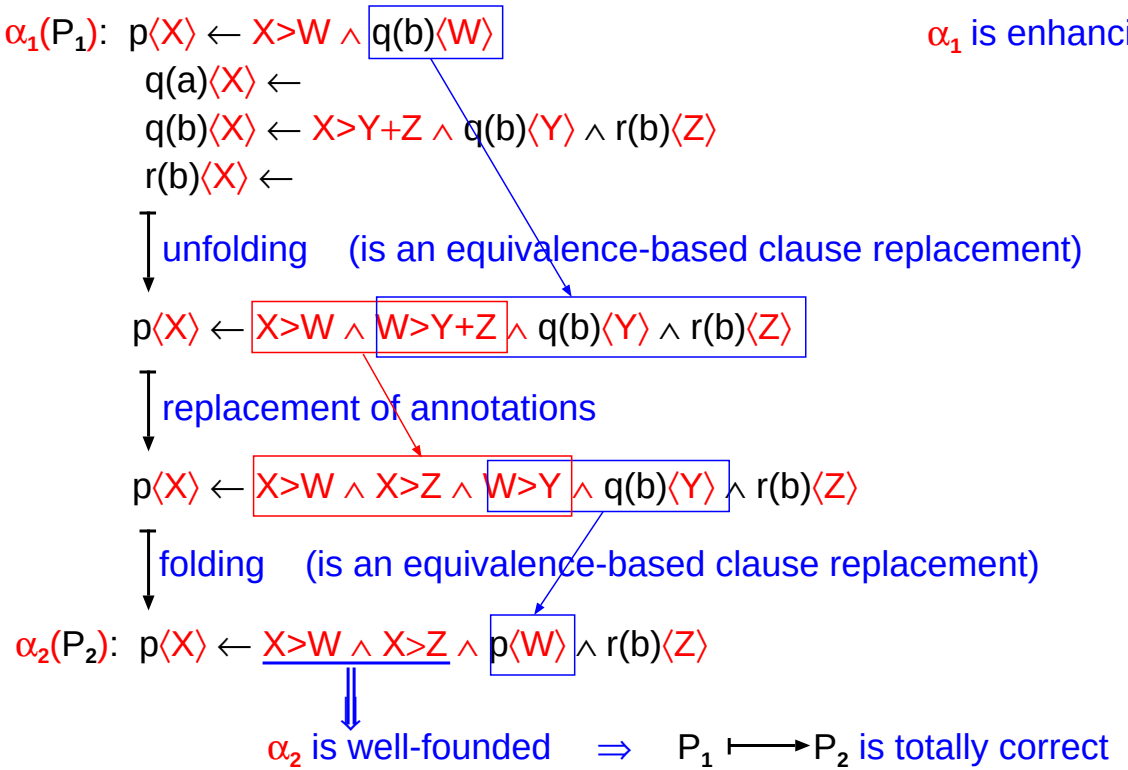
- α_1 is enhancing, α_2 is well-founded
- $M(P_1) \models \Gamma_1 \Rightarrow \Gamma_2$

  because $M(P_1) \models$ ∀X∀Z(∃Y(reach(X,Y) ∧ arc(Y,Z)) ← ∃Y(arc(X,Y) ∧ reach(Y,Z)))

- $M(\alpha_1(P_1)) \models \alpha_1(\Gamma_1) \Leftarrow \alpha_2(\Gamma_2)$

  because $M(\alpha_1(P_1)) \models$ ∀X∀Z∀K (∃N∃Y (K>N ∧ reach(X,Y)⟨N⟩ ∧ arc(Y,Z))
  → ∃N∃Y (K>N ∧ arc(X,Y) ∧ reach(Y,Z)⟨N⟩))

Unfold/Fold of Annotated Programs



6. Related Work

Improvement [Tamaki-Sato 84, Sands 94, 97]

Counting [Tamaki-Sato 86, Kanamori-Fujita 87, Roychoudhury et al. 04]

Simultaneous Replacement [Bossi-Cocco-Etalle 96]

Semantic delay [Bossi-Cocco-Etalle 92]

Post-hoc termination check [Cook-Gallagher 94]

Well-Founded Annotations vs. Improvement

[this work] [Tamaki-Sato 84, Sands 94]

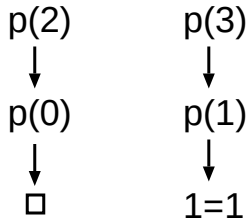
P_1: $p(0) \leftarrow$
$p(X) \leftarrow X=1$
$p(X) \leftarrow p(X-2)$
$p(X) \leftarrow p(X)$

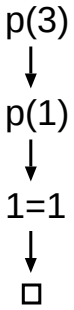
$\longmapsto$

P_2: $p(0) \leftarrow$
$p(X) \leftarrow p(X-1)$
$p(X) \leftarrow p(X)$

- $M(P_1) \models \forall X\ (X=1 \vee p(X-2) \leftrightarrow p(X-1))$
- Not an improvement: e.g., for X=4,

size(4=1 ∨ p(2)) =
= min(size(4=1), size(p(2)))
= 2
size(p(3)) = 3





$P_1 \longmapsto P_2$ totally correct ?

... Well-Founded Annotations vs. Improvement

$\alpha_1(P_1)$: $p(0)\langle 0\rangle \leftarrow$
$p(X)\langle 1\rangle \leftarrow X=1$
$p(X)\langle X\rangle \leftarrow p(X-2)\langle X-2\rangle$
$p(X)\langle N\rangle \leftarrow p(X)\langle N-1\rangle$

$\longmapsto$

$\alpha_2(P_2)$: $p(0)\langle 0\rangle \leftarrow$
$p(X)\langle X\rangle \leftarrow p(X-1)\langle X-1\rangle$
$p(X)\langle N\rangle \leftarrow p(X)\langle N-1\rangle$

- α_1 is enhancing, α_2 is well-founded
- $M(P_1) \models \forall X\,(X=1 \vee p(X-2) \leftarrow p(X-1))$
- $M(\alpha_1(P_1)) \models \forall X\,(X=1 \vee p(X-2)\langle X-2\rangle \rightarrow p(X-1)\langle X-1\rangle)$

$P_1 \longmapsto P_2$ is totally correct

Well-Founded Annotations vs. Counting

Transformation rules with counters: each clause has an associated counter, which is incremented upon unfolding and decremented upon folding. [Tamaki-Sato 86, Kanamori-Fujita 87, Roychoudhury et al. 04]

Transformation rules with counters can be proved totally correct by the Well-Founded Annotation method.

Well-Founded Annotations vs. Counting

Example.

stands for

[1] $p \leftarrow q \wedge r$ — $p\langle X\rangle \leftarrow X \geq Y+Z+ 1 \wedge q\langle Y\rangle \wedge r\langle Z\rangle$

[1] $q \leftarrow s \wedge t$ — $q\langle X\rangle \leftarrow X \geq Y+Z+ 1 \wedge s\langle Y\rangle \wedge t\langle Z\rangle$

[1] $q \leftarrow p$ — $q\langle X\rangle \leftarrow X \geq Y+ 1 \wedge p\langle Y\rangle$

[1] $u \leftarrow t \wedge r$ — $u\langle X\rangle \leftarrow X \geq Y+Z+ 1 \wedge t\langle Y\rangle \wedge r\langle Z\rangle$

unfolding

unfolding + replacement of annotations

[2] $p \leftarrow s \wedge t \wedge r$ — $p\langle X\rangle \leftarrow X \geq Y_1+Y_2+Z+ 2 \wedge s\langle Y_1\rangle \wedge t\langle Y_2\rangle \wedge r\langle Z\rangle$

[2] $p \leftarrow p \wedge r$ — $p\langle X\rangle \leftarrow X \geq Y_1+Z+ 2 \wedge p\langle Y_1\rangle \wedge r\langle Z\rangle$

folding

unfolding + replacement of annotations

[1] $p \leftarrow s \wedge u$ — $p\langle X\rangle \leftarrow X \geq Y_1+Z_1+ 1 \wedge s\langle Y_1\rangle \wedge u\langle Z_1\rangle$

Positive counters $\Rightarrow$ annotation is well-founded $\Rightarrow$ total correctness

Future Work

- Completeness of Proof Methods.
  The unique fixpoint and the improvement methods are incomplete: there exist totally correct transformations that cannot be proved by these methods.
  Is the well-founded annotation method complete?

- Find useful annotations that cover most cases that occur in practice.

- Extension of the Well-Founded Annotation method to general programs (with negation).