

Interprocedurally Analyzing Polynomial Identities

Helmut Seidl

Markus Müller-Olm + Michael Petter

Münster + München

Marseille, STACS 2006

Questions:

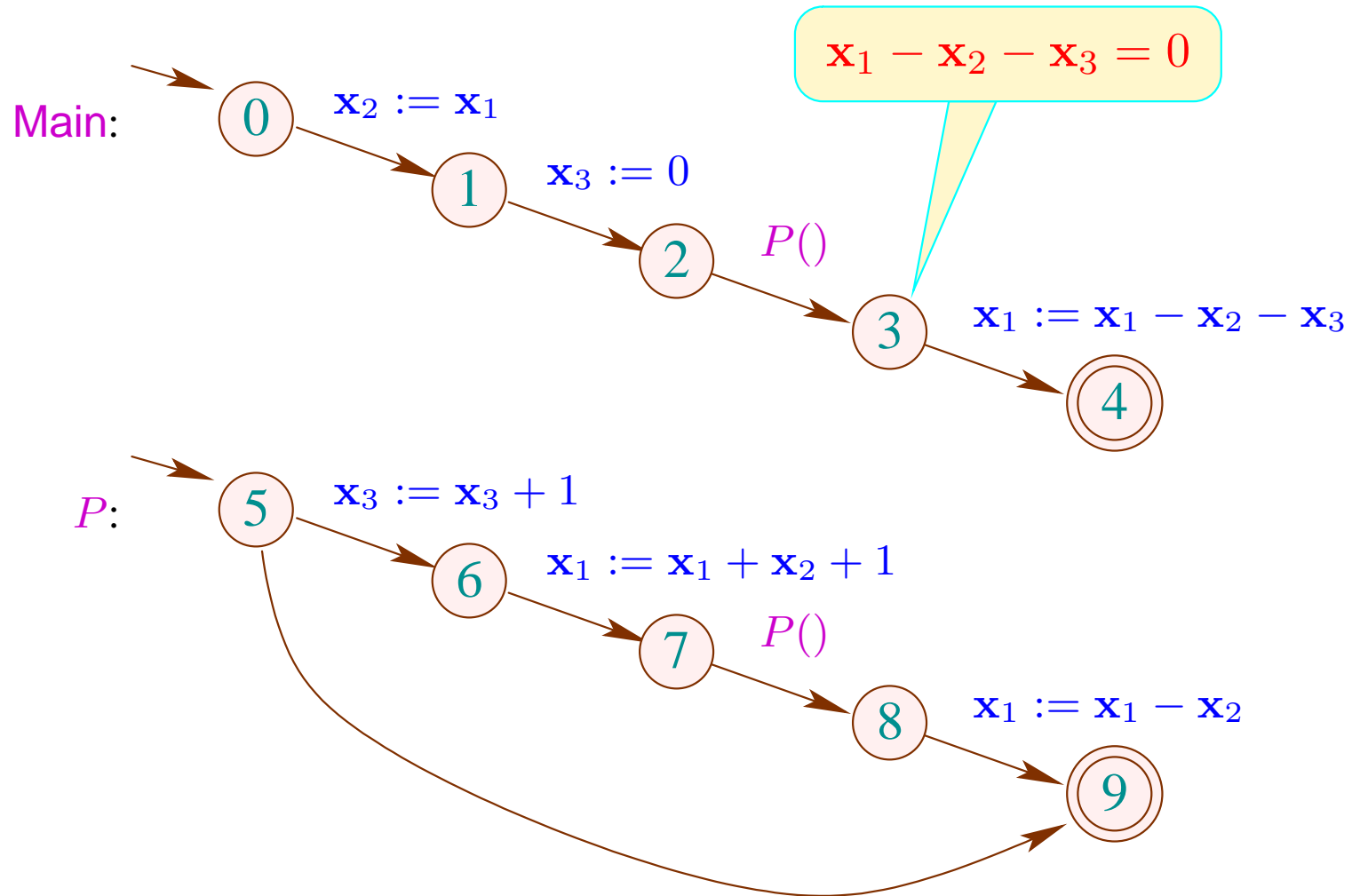
- What is the value of x_1 at program exit?
- Where is $x_1 - x_2 = 0$?
- What is the relationship between x_1, x_2 and x_3 at program point 3 ?

Questions:

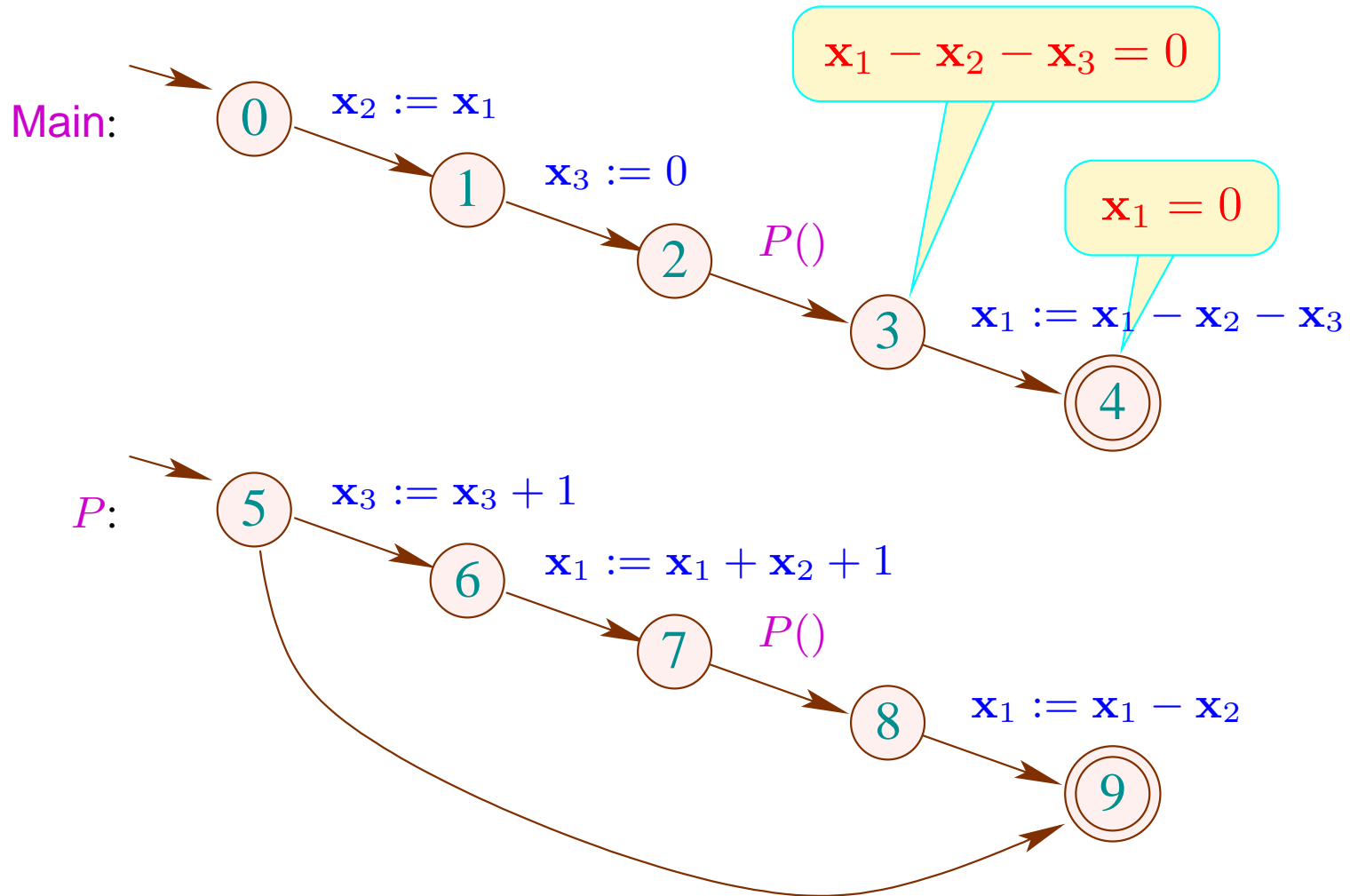
- What is the value of x_1 at program exit?
- Where is $x_1 - x_2 = 0$?
- What is the relationship between x_1, x_2 and x_3 at program point 3 ?

⇒ polynomial identities

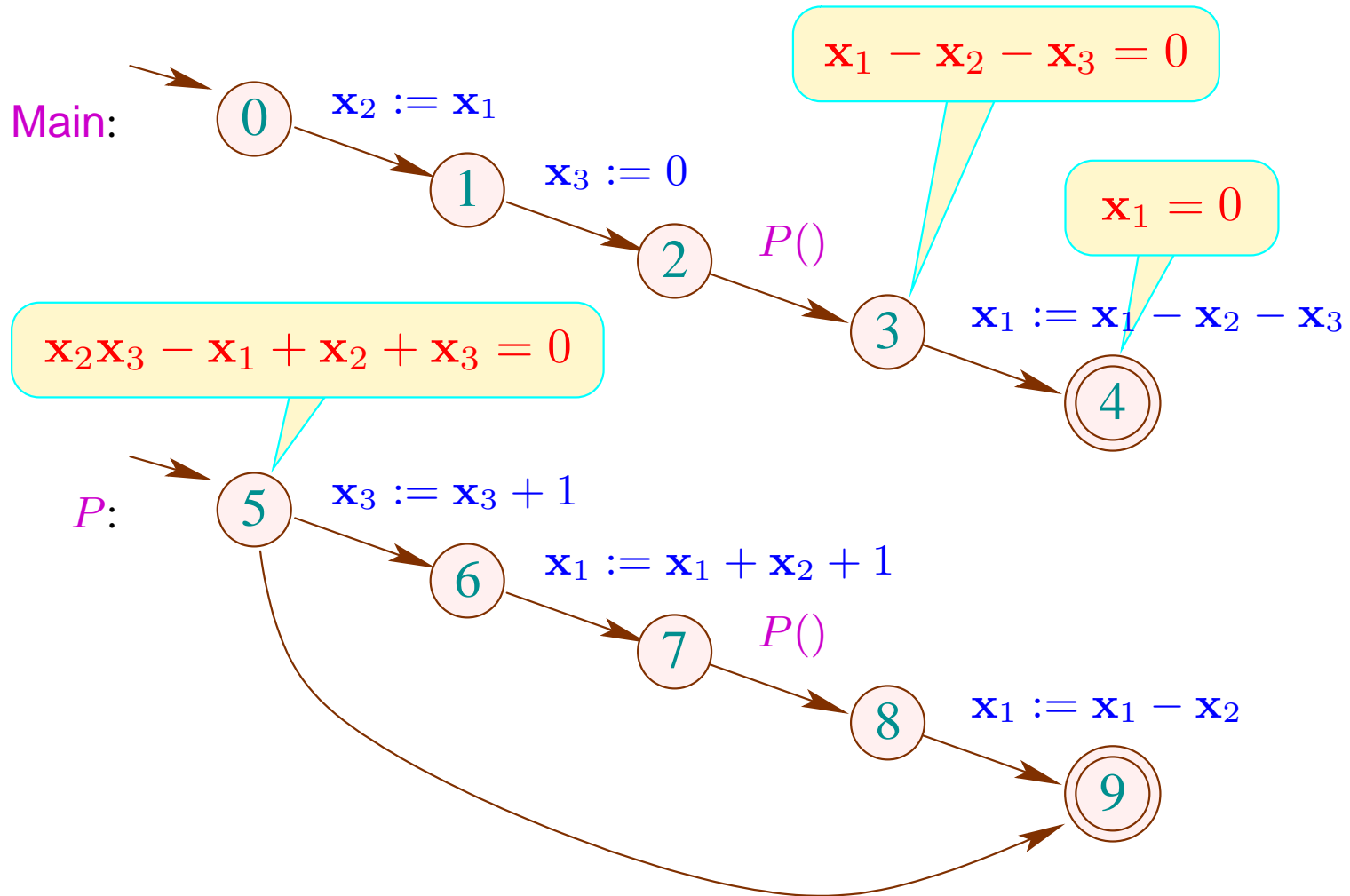
Example:



Example:



Example:



Applications:

- aggressive program optimizations;
- program verification :-))

Simplification:

Polynomial programs ...

Simplification:

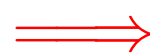
Polynomial programs ... consist of

- polynomial assignments: $\mathbf{x}_1 := \mathbf{x}_1 \mathbf{x}_2 - \mathbf{x}_3$
- unknown assignments: $\mathbf{x}_i := ?$
for too complex assignments :-)
- non-deterministic instead of conditional choice :-)

Goal:

Given this class of programs,

- Determine **all** valid polynomial identities; or
- Determine **all** valid polynomial identities **up to a given degree**; or
- Determine **some** valid polynomial identities.



Abstract Interpretation

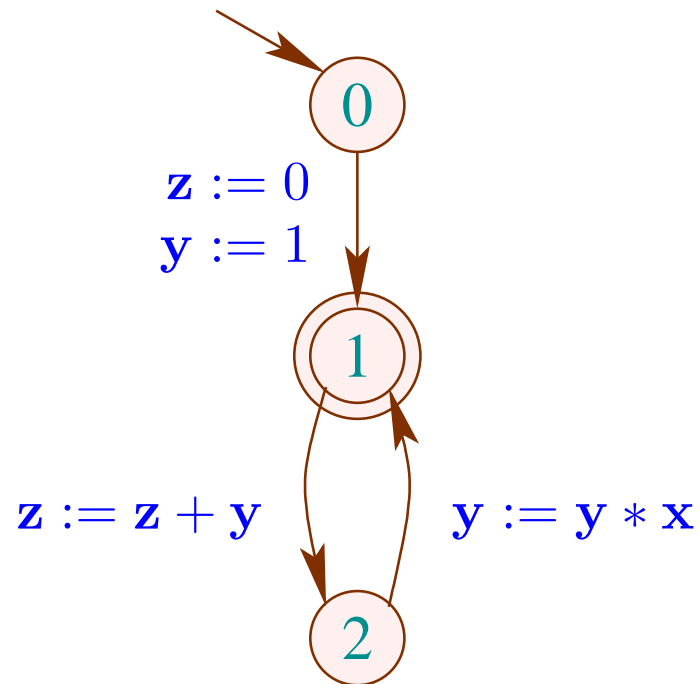
Intraprocedural Background:

Karr	affine programs and identities	1976
Gulwani, Necula	affine programs and identities: probabilistic	2003
MMO., S.	polynomial programs and identities: backward propagation	2002, 2004
Carbonell, Kapur	polynomial programs and identities: forward propagation	2004
Manna et al.	polynomial programs and identities: parametric	2004

Interprocedural Background:

Horwitz et al.	linear constants	1996
MMO., S.	affine programs and polynomial id.	2004
Gulwani, Necula	affine programs and identities: probabilistic	2005
Colon	polynomial programs and identities: Transition Invariants	2004

1. Intraprocedural Analysis:



Observation:

The set of valid polynomial identities at a program point form an **ideal** ...

An ideal $I \subseteq \mathbb{Q}[x_1, \dots, x_k]$ is a set of polynomials with:

$q_1, q_2 \in I$ implies $q_1 + q_2 \in I$;

$q \in I$ implies $r \cdot q \in I$ for all polynomials r .

Recall:

- Every polynomial ideal is **finitely generated**.



Every ascending chain of polynomial ideals is ultimately stable :-)

- Membership: “ $p \in I$ ”? is decidable :-)
- Emptiness: “ $I = \{0\}$ ”? is decidable :-)

Idea 1:

Carbonell, Kapur, S. 2004

Put up a constraint system for all valid identities ...

$$\mathcal{I}(\text{start}) \subseteq \{0\}$$

$$\mathcal{I}(v) \subseteq \llbracket \mathbf{x}_i := q \rrbracket (\mathcal{I}(u)) \quad (u, v) \text{ assignment } \mathbf{x}_i := q$$

where:

$$\llbracket \mathbf{x}_i := q \rrbracket (\mathcal{I}) = \{p \mid p[q/\mathbf{x}_i] \in \mathcal{I}\}$$

Theorem:

The greatest solution of the constraint system characterizes the sets of valid polynomial identities **precisely**.

Theorem:

The greatest solution of the constraint system characterizes the sets of valid polynomial identities **precisely**.

... **but:**

There might be **infinite** decreasing chains of ideals :-)

Idea 2:

MMO., S. 2002

- Check only a **given assertion** for validity !
- Compute **weakest pre-conditions** at all edges.
- Check that the weakest precondition for program start equals $0 = 0$:-)

A Constraint System for WP:

$$\begin{aligned} \mathcal{I}(v_0) &\supseteq \langle p \rangle && p \text{ identity to be checked} \\ \mathcal{I}(u) &\supseteq \llbracket \mathbf{x}_i := q \rrbracket^{\mathsf{T}}(\mathcal{I}(v)) && (u, v) \text{ assignment } \mathbf{x}_i := q \end{aligned}$$

where:

$$\llbracket \mathbf{x}_i := q \rrbracket^{\mathsf{T}}(p) = \langle p[q/\mathbf{x}_i] \rangle$$

Theorem

Let \mathcal{I} be the least solution of the constraint system.

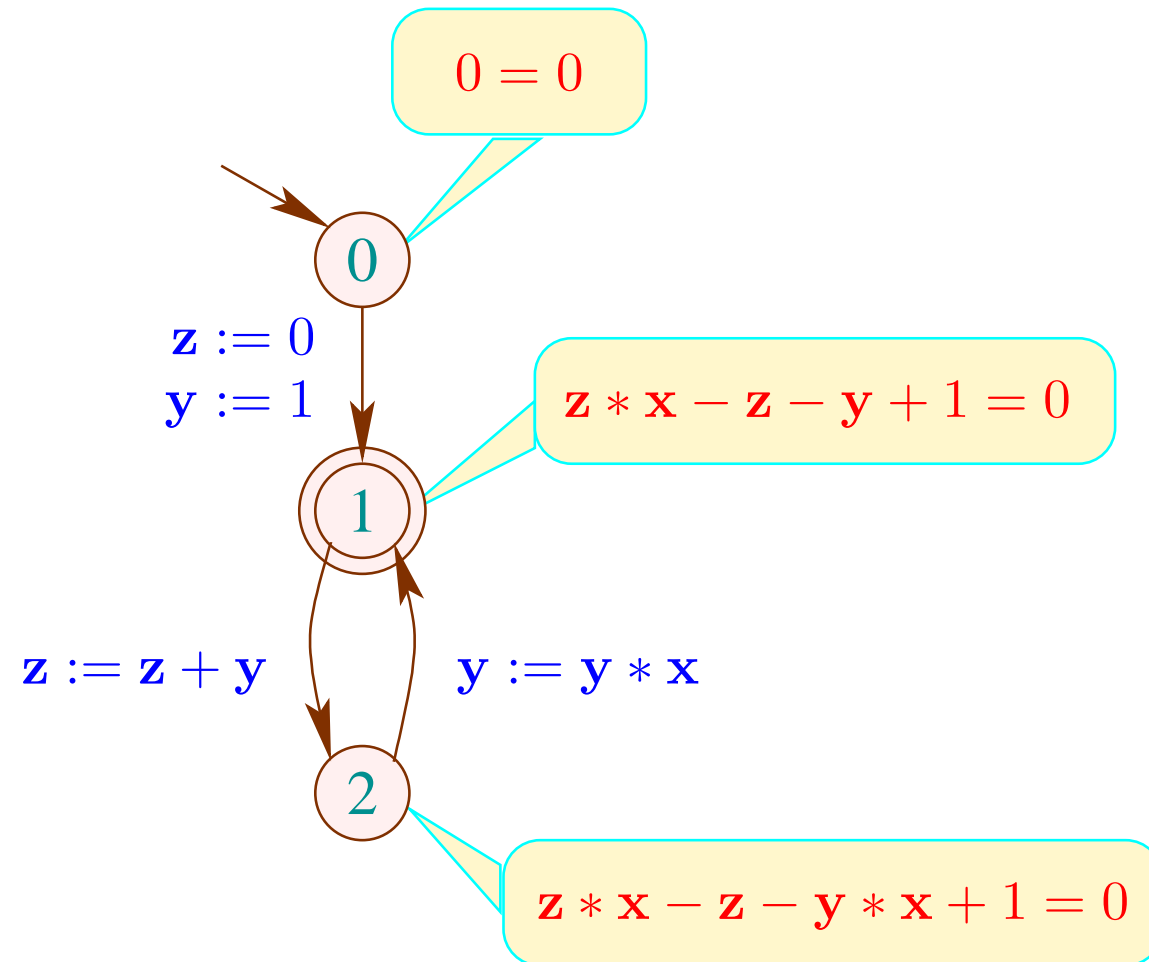
Then

- $p = 0$ holds at point v_0 iff

$$\langle 0 \rangle \supseteq I(\text{start})$$

- The sets $\mathcal{I}(v)$ can be effectively computed.

Our Example:



Problem:

How to infer an unknown identity??

Problem:

How to infer an unknown identity??

Idea:

Consider the generic polynomial w.r.t. some set D of tuples of exponents:

$$p_D = \sum_{j_1 \dots j_k \in D} a_{j_1, \dots, j_k} \cdot x_1^{j_1} \dots x_k^{j_k}$$

(a_{j_1, \dots, j_k} fresh variables)

Observation:

The generic weakest pre-condition at **start** is an ideal $\langle p_1, \dots, p_n \rangle$ where

$$p_i = \sum_{j_1, \dots, j_k} b_{j_1, \dots, j_k}^{(i)} \cdot x_1^{j_1} \cdot \dots \cdot x_k^{j_k}$$

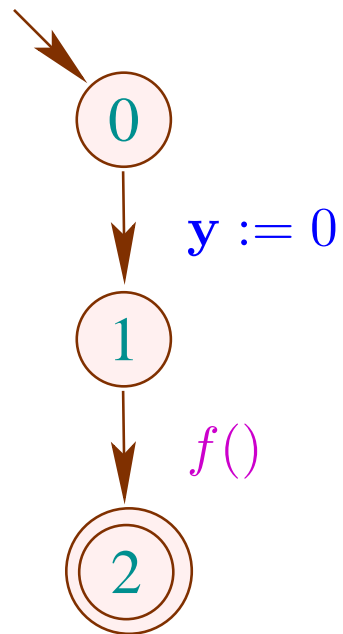
and the $b_{j_1, \dots, j_k}^{(i)}$ are linear combinations of the $a_{j'_1, \dots, j'_k}$:-)

Theorem

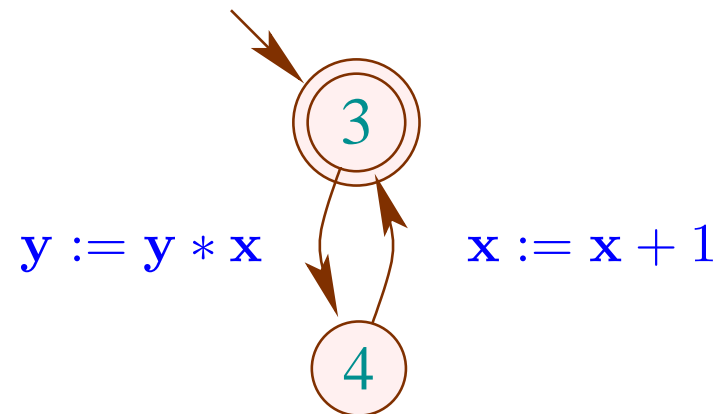
- The following two statements are equivalent:
 - (1) $p_D = 0$ holds at v_0 ;
 - (2) $b_{j_1, \dots, j_k}^{(i)} = 0$ for all i, j_1, \dots, j_k .
- The set of **all valid polynomial identities** (with exponents from D) at v_0 are computable :-)

2. Interprocedural Analysis

Main :



`f()` :



Problem: How to specify effects?

Problem: How to specify effects?

Idea 1: Colon, 2004

- Introduce a copy $\mathbf{x}'_1, \dots, \mathbf{x}'_k$ of the variables $\mathbf{x}_1, \dots, \mathbf{x}_k$;
- Express effects by **transition invariants**, here:
polynomials $p \in \mathbb{Q}[\mathbf{X} \cup \mathbf{X}']$ relating pre-state with post-state ...

Problem: How to specify effects?

Idea 1: Colon, 2004

- Introduce a copy $\mathbf{x}'_1, \dots, \mathbf{x}'_k$ of the variables $\mathbf{x}_1, \dots, \mathbf{x}_k$;
- Express effects by **transition invariants**, here:
polynomials $p \in \mathbb{Q}[\mathbf{X} \cup \mathbf{X}']$ relating pre-state with post-state ...

Note: These form an **ideal** :-)

Transformation of Assignments:

$$\llbracket \mathbf{x}_i := p \rrbracket^\# = \langle \{ \mathbf{x}_j - \mathbf{x}'_j \mid j \neq i \} \cup \{ \mathbf{x}_i - p[\mathbf{X}'/\mathbf{X}] \} \rangle$$

Composition:

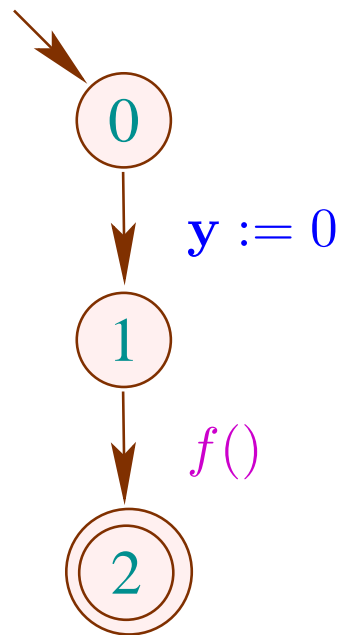
$$I_1 \circ I_2 = (I_1[\mathbf{Y}/\mathbf{X}'] \oplus I_2[\mathbf{Y}/\mathbf{X}]) \cap \mathbb{Q}[\mathbf{X} \cup \mathbf{X}']$$

A Constraint System for Transition Invariants:

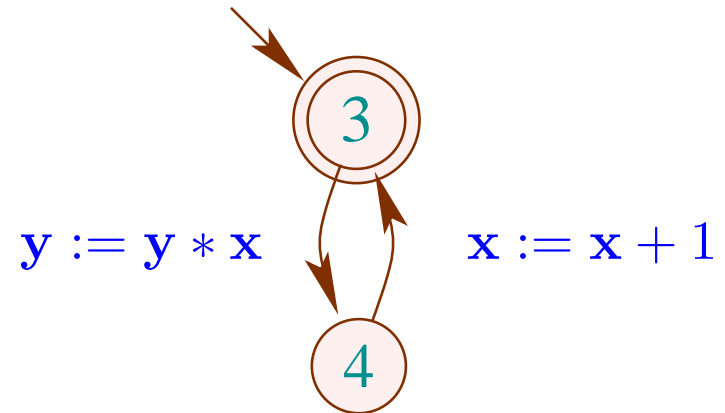
$$\begin{array}{ll} I(v) \subseteq \langle \mathbf{x}_i - \mathbf{x}'_i \mid i = 1, \dots, k \rangle & v \text{ is entry point} \\ I(v) \subseteq \llbracket \mathbf{x}_i := p \rrbracket^\# \circ I(v) & (u, v) \text{ assignment } \mathbf{x}_i := p \\ I(v) \subseteq I(f) \circ I(u) & (u, v) \text{ calls } f \\ I(f) \subseteq I(v) & v \text{ exit point of } f \end{array}$$

Our Example:

Main :



$f()$:



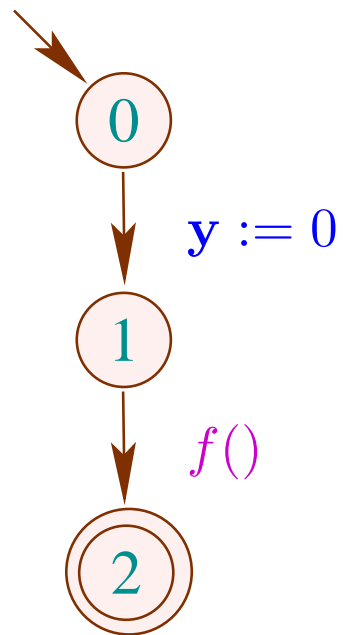
Calculation:

$$\begin{aligned} I(f) &= \langle \mathbf{x} - \mathbf{x}', y - y' \rangle \\ &\cap \langle \mathbf{x} - \mathbf{x}' - 1, y - y' * \mathbf{x}' \rangle \\ &\cap \langle \mathbf{x} - \mathbf{x}' - 2, y - y' * \mathbf{x}' * (\mathbf{x}' + 1) \rangle \\ &\dots \\ &= \langle 0 \rangle \end{aligned}$$

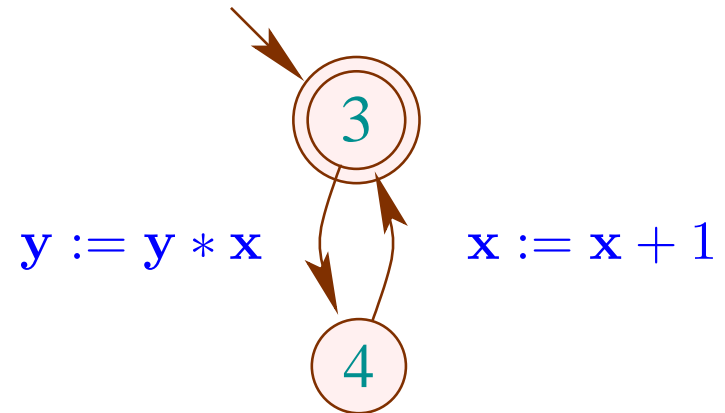
On the other hand ...

Our Example (continued):

Main :

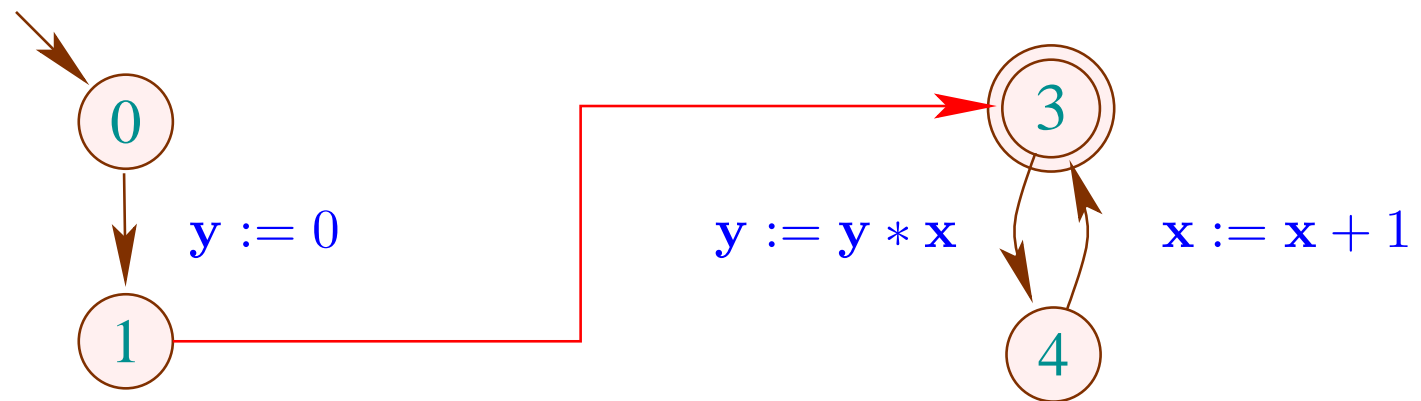


$f()$:



Our Example (continued):

Main :



Calculation:

$$\begin{aligned} I(\text{main}) &= \langle \mathbf{x} - \mathbf{x}', \mathbf{y} \rangle \\ &\cap \langle \mathbf{x} - \mathbf{x}' - 1, \mathbf{y} \rangle \\ &\cap \langle \mathbf{x} - \mathbf{x}' - 2, \mathbf{y} \rangle \\ &\dots \\ &= \langle \mathbf{y} \rangle \end{aligned}$$

We conclude:

- Fixpoint iteration may create decreasing chains of ideals of infinite length :-(
 $\text{:-}(\text{---})$
- Composition of transition invariant ideals is not continuous :-((
 $\text{:-}((\text{---})$

\implies loss in precision inevitable.

Idea 2:

- Represent effects by weakest pre-conditions of generic polynomials :-)
- Function calls are dealt with through instantiation of generic coefficients ...

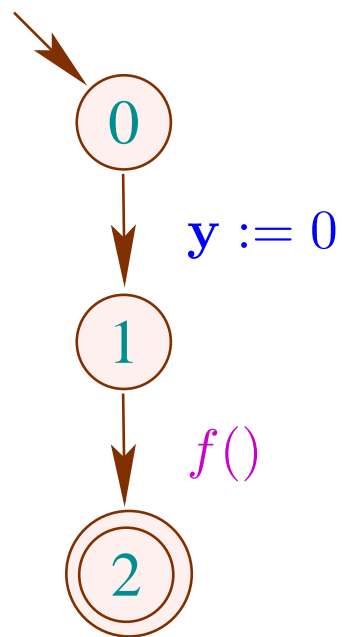
Note: Effects again are described by ideals —
though over **more variables !**

A Constraint System for Interprocedural WP:

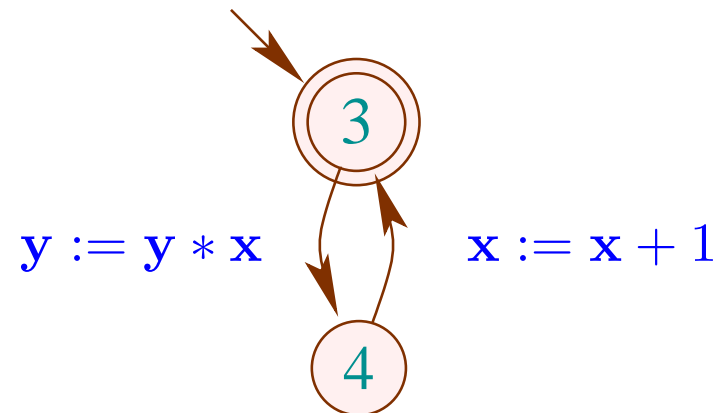
$$\begin{array}{ll} I(u) \supseteq \langle p_D \rangle & u \text{ is exit point} \\ I(u) \supseteq \llbracket \mathbf{x}_i := q \rrbracket^T(I(v)) & (u, v) \text{ assignment } \mathbf{x}_i := q \\ I(u) \supseteq \llbracket f \rrbracket^T(I(v)) & (u, v) \text{ calls } f \\ \llbracket f \rrbracket^T \supseteq I(u) & u \text{ entry point of } f \end{array}$$

Our Example (continued):

Main :



$f()$:



Calculation: $(p_d = ay + bx + c)$

$$\begin{aligned} \llbracket f \rrbracket^T &= \langle ay + bx + c \rangle \\ &\oplus \langle ayx + b(\mathbf{x} + 1) + c \rangle \\ &\oplus \langle ayx(\mathbf{x} + 1) + b(\mathbf{x} + 2) + c \rangle \\ &\oplus \langle ayx(\mathbf{x} + 1)(\mathbf{x} + 2) + b(\mathbf{x} + 3) + c \rangle \\ &\dots \\ &= \langle ay, b, c \rangle \end{aligned}$$

Discussion:

- The computed ideal $\llbracket f \rrbracket^T$ describes the weakest pre-condition transformation of f **precisely** — but only for polynomials of the right shape.
- If the exponents of polynomials in Gröbner bases at **all calls** are from D , the analysis is **exact**:

It computes **all valid** polynomial invariants with exponents from D :-)

What, if the degrees grow ???

Idea for Widening:

- Decompose polynomial p into polynomials of small degrees ...

$$2x^2y^2 - 7xy^3 + 5x^2 - 36 = xy(2xy - 7y^2) + (5x^2 - 36)$$

Idea for Widening:

- Decompose polynomial p into polynomials of small degrees ...

$$2x^2y^2 - 7xy^3 + 5x^2 - 36 = xy(2xy - 7y^2) + (5x^2 - 36)$$

- Apply the precondition transformation to each of the small polynomials :-)

Idea for Widening:

- Decompose polynomial p into polynomials of small degrees ...

$$2x^2y^2 - 7xy^3 + 5x^2 - 36 = xy(2xy - 7y^2) + (5x^2 - 36)$$

- Apply the precondition transformation to each of the small polynomials :-)



The algorithm always terminates !!!

If it doesn't apply the widening, it is precise.

Otherwise, it is at least sound.

Further Issues:

- disequality guards can easily be handled :-)
- equality guards \implies Langrange Multipliers
- dealing with local variables and return values :-)
- going from \mathbb{Q} to integers modulo 2^w :-)
- playing around with variable orderings for efficiency of Gröbner basis computations ...

Open Problems

- Can the restriction on the shape of polynomials be lifted?
- ... at least in some (useful) cases?
- Can the analysis be extended to parallel programs?
- What is the precise complexity?
- ... at least of some (useful) cases?