

**Certifying polynomial time and linear/polynomial space
for imperative programs: improvements and optimality**

Karl-Heinz Niggl

(Joint work with Henning Wunderlich and Jan Mehler)

Geocal'06: Implicit Computational Complexity

Marseille-Luminy

February 13–17, 2006

Starting point

Earlier research [Kristiansen & N.]:

- Stack programs of μ -measure $0 = \text{FP-TIME}$
- Loop programs of μ -measure $0 = \text{FLINSPACE}$

From a programming perspective, this is **not practically appealing**:

- no **user-friendly basic instructions (BI)**, except for nonsize-increasing BI's
- no vital instructions such as assignment statements $X_i = X_j$
- no **mixed datatypes**

General outline – 1

Programs in this talk are built from **arbitrary BI's** $\text{imp}(X_1, \dots, X_n)$ by

- **sequences** $P_1; P_2$
- **conditionals** **if** (cond) **then** P_1 **and if** (cond) **then** P_1 **else** P_2
- **loop statements** $\text{loopI } X_h [Q]$ and $\text{loopII } X_h [Q]$.

Each **variable** X_i may represent **any datatype**, e.g.

— stacks, registers, trees, graphs, arrays, ...

and is implicitly equipped with a notion of **size** $|X_i|$, e.g.

— X_i serves as a **register** $\implies |X_i|$ might be the **unary or binary length** of the number stored in X_i .

General outline – 2

No specification on (cond) or loop statements, except that

- each instance of (cond) be evaluated in polynomial time
- for each instance of a loop statement
 - loopI $X_h [Q]$, the body Q is executed $|X_h|$ times
 - loopII $X_h [Q]$, the body Q is executed $2^{|X_h|} - 1$ times
 - during its execution, the contents of X_h remain unchanged.

Novelty: A new method of certifying polynomial size boundedness (psb) for such imperative programs, provided that all BI's are psb, too.

General outline – 3

Thm (Characterisations).

- FP_{TIME} = Certified string programs (stack programs built from polynomial-time computable BI's).
- FLINSPACE = Certified general loop programs (loop programs built from linear-space computable BI's)
- FPSPACE = Certified power string programs (string programs built from poly-space computable BI's and extended by loop|| statements)

Thm (Optimality).

The new method is optimal on **core programs**, i.e., programs built from honestly certified BI's by sequencing and loops.

Method – 1

Def. A program P in variables X_1, \dots, X_n is **polynomially size bounded (psb)** iff there are polynomials $p_1, \dots, p_n \in \mathbb{N}[X_1, \dots, X_n]$ such that

$$\{|X_1| = s_1, \dots, |X_n| = s_n\} \subseteq \{ |X_i| \leq p_i(s_1, \dots, s_n) \}$$

Call p_1, \dots, p_n a **polynomial bound (pb)** on P.

- **polynomials:** $c_0 + \dots + c_i \cdot X_1^{i_1} \dots \dots \cdot X_n^{i_n} + \dots$
- **maximum $p \sqcup q$:**

$$p = c_0 + \dots + c_i \cdot X_1^{i_1} \dots \dots \cdot X_n^{i_n} + \dots$$
$$q = d_0 + \dots + d_i \cdot X_1^{i_1} \dots \dots \cdot X_n^{i_n} + \dots$$

$$\implies p \sqcup q := \max(c_0, d_0) + \max(c_i, d_i) \cdot X_1^{i_1} \dots \dots \cdot X_n^{i_n} + \dots$$

Method – 2

Big Question: Given a pb \vec{q} on the body \mathcal{Q} of a loop P ,

- what criterion on \vec{q} guarantees the existence of a pb on P ?
- how can one construct from \vec{q} a pb on P ?

Central to the certification method:

We store and process only a finite amount of information on the class of possible polynomial size bounds for programs.

Let \vec{p} be a pb on a program P . Then we store and process only for each

$$p_i(\vec{X}) = c_0 + \dots + c_j \cdot X_1^{j_1} \dots \dots \cdot X_n^{j_n} + \dots$$

an $(n+1)$ -tuple $\langle p_i \rangle$ (the **representation of p_i**) over the **forgetting set** $A := \{0, 1, \infty\}$ ordered by $0 < 1 < \infty$:

Method – 3

$$\text{for } 1 \leq j \leq n, \quad \langle p_i \rangle [j] := \begin{cases} 0 & \text{if } \mathbf{X}_j \notin p_i \\ 1 & \text{if } p_i = \mathbf{X}_j + q(\vec{\mathbf{X}} \setminus \mathbf{X}_j) \\ \infty & \text{else} \end{cases}$$

$$\langle p_i \rangle [n+1] := \begin{cases} c_0 & \text{if } K(p_i) = c_0 \leq 1 \\ \infty & \text{else} \end{cases}$$

For $\vec{a} \in A^{n+1}$, the class of **polynomials of bound \vec{a}** is defined by:

$$\text{POLY}(\vec{a}) := \{p(\mathbf{X}_1, \dots, \mathbf{X}_n) \mid \langle p \rangle \leq \vec{a}\}$$

A **certificate for P** is any $(n+1) \times (n+1)$ matrix Y over A with last row $0^n 1$ such that $\exists p, b, q_1 \in \text{POLY}(Y[1]), \dots, q_n \in \text{POLY}(Y[n])$ on P .

Example: $\langle P, \vec{p} \rangle$ with rows $\langle p_1 \rangle, \dots, \langle p_n \rangle, 0^n 1$ is a certificate for P .

Method – 4 (Examples for $n = 2$)

imperative	bounding polynomial	certificate
$\text{push}(a, X_2), \text{inc}(X_2)$	$X_2 + 1$	$\begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 1 \\ 0 & 0 & 1 \end{pmatrix}$
$\text{pop}(X_1), \text{reverse}(X_1)$	X_i	1_3
$X_2 = X_1$	X_1	$\begin{pmatrix} 1 & 0 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 1 \end{pmatrix}$
$\text{concat}(X_2, X_1), X_2 += X_1$	$X_2 + X_1$	$\begin{pmatrix} 1 & 0 & 0 \\ 1 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix}$

Method – 5

Operations on $A = \{0, 1, \infty\}$.

	+	•	⊔
+	0 1 ∞	0 1 ∞	0 1 ∞
0	0 1 ∞	0 0 0	0 1 ∞
1	1 ∞ ∞	1 0 1 ∞	1 1 ∞
∞	∞ ∞ ∞	∞ 0 ∞ ∞	∞ ∞ ∞

$+$, \bullet , \sqcup are commutate, associative, with neutral elements $0, 1, 0$

$\mathcal{M}_n[A] :=$ all $(n+1) \times (n+1)$ matrices over A with last row $0^n 1$.

\rightsquigarrow **matrix multiplication** \otimes on $\mathcal{M}_n[A]$, which is associative, preserves the last row property, and with **neutral element** 1_n

$+$, \bullet , \sqcup , $<$ denote the componentwise extensions to A^m and $\mathcal{M}_n[A]$

Method – 6

Lemma (Base). If $\text{imp}(\vec{X})$ is a basic instruction with pb p_1, \dots, p_n , then $\langle \text{imp}(\vec{X}), \vec{p} \rangle$ with rows $\langle p_1 \rangle, \dots, \langle p_n \rangle, 0^n 1$ is a certificate for $\text{imp}(\vec{X})$.

Lemma (Structure). Let u, v be in A^{n+1} .

- (a) $u \leq v \implies \text{POLY}(u) \subseteq \text{POLY}(v)$
- (b) $p \in \text{POLY}(u), q \in \text{POLY}(v) \implies p \sqcup q \in \text{POLY}(u \sqcup v)$

Corollary (Conditional).

Z_1, Z_2 are certificates for $P_1, P_2 \implies$

$Z_1 \sqcup Z_2$ is a certificate for **if (cond) then P_1 else P_2**

$Z_1 \sqcup 1_{n+1}$ is a certificate of **if (cond) then P_1**

Method – 7

Lemma (Composition and Sequence).

(a) $q \in \text{POLY}(v), p_1 \in \text{POLY}(w_1), \dots, p_n \in \text{POLY}(w_n) \implies$

$$q(p_1, \dots, p_n) \in \text{POLY}(v \otimes M) \text{ with } M := \begin{pmatrix} w_1 \\ \vdots \\ w_n \\ 0^{n1} \end{pmatrix}.$$

(b) Z_1, Z_2 certificates for $P_1, P_2 \implies Z_2 \otimes Z_1$ certificate for $P_1; P_2$

Clearly, (a) implies (b). Idea for (a): for each $i = 1, \dots, n$,

$$(v \otimes M)[i] = v \otimes M[\cdot][i] = \sum_{j=1}^n v[j] \bullet M[j][i]$$

gives full information on the usage of X_i in $q(p_1, \dots, p_n)$ w.r.t. A .

Method – 8

Big Question. Given a **certificate** Y for the body Q of a loop P with variables among X_1, \dots, X_n ,

- what **criterion on** Y guarantees a certificate Z for P ?
- how can one **construct** Z from Y ?

Know: For each $m \geq 0$, Y^m is a **certificate for** Q^m ($=Q$; \dots ; Q , m times)

\rightsquigarrow It is natural to investigate the following two (finite) **limit forms**:

- $Y^+ := \bigsqcup_{k=1}^{\infty} Y^k$

non-monotonic

- $\hat{Y}^* := \bigsqcup_{k=0}^{\infty} \hat{Y}^k$ where $\hat{Y} := 1_{n+1} \sqcup Y$

monotonic ($\hat{Y}^i \leq \hat{Y}^{i+1}$)

Control graphs and Certificates – 1

Intuition. An entry $Y[i][j] > 0$ with $i \in \{1, \dots, n\}$ represents

- a possible data-flow in P from X_j to X_i denoted by $j \rightarrow_Y i$ (read j controls i in Y), for $j = 1, \dots, n$, or
- a possible influence of a constant on X_i , ($* \rightarrow_Y i$), for $j = n+1$.

Let \rightarrow_Y^+ (\rightarrow_Y^*) denote the transitive (transitive, reflexive) closure of \rightarrow_Y . The **control graph** \mathcal{C}_Y of Y is then defined by

$$\mathcal{C}_Y := (\{1, \dots, n, *\}, \rightarrow_Y^+).$$

Lemma (Control Graph).

- (a) \mathcal{C}_Y is the reflexive closure of \mathcal{C}_Y .
- (b) \mathcal{C}_{Y^+} is the transitive closure of \mathcal{C}_Y .

Control graphs and Certificates – 2

Example. Consider the following stack program Q .

$X_5 = X_1$;

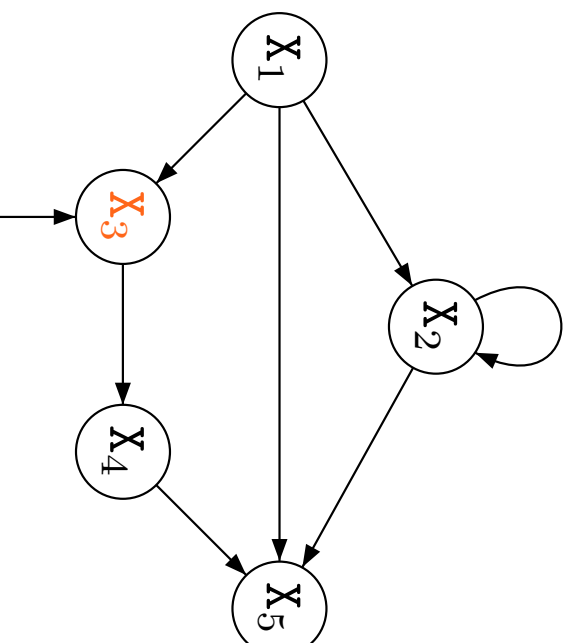
concat(X_5, X_2); concat(X_5, X_4);

concat(X_2, X_1);

$X_4 = X_3$; $X_3 = X_1$;

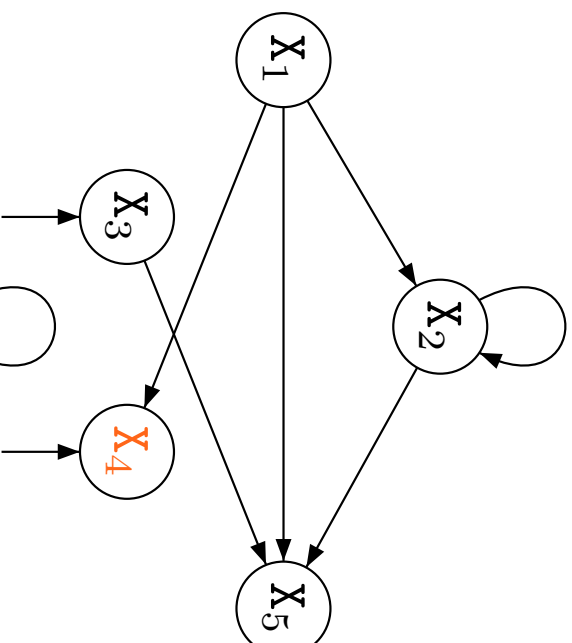
push(a, X_3); nil(X_1)

$$Y := \begin{pmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{pmatrix}$$

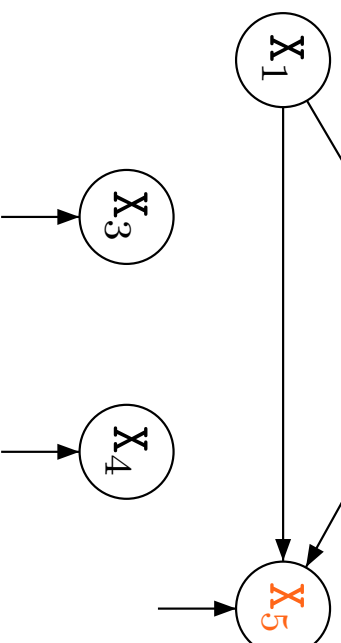


Control graphs and Certificates – 3

$$Y^2 := \begin{pmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 1 & 1 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{pmatrix}$$



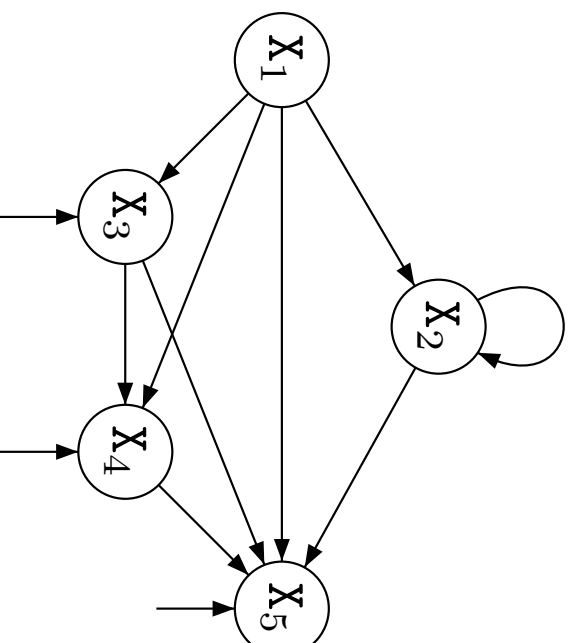
$$Y^3 := \begin{pmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & \infty & 1 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{pmatrix}$$



Control graphs and Certificates – 4

Since $Y^3 = Y^4 = Y^5 = \dots$, we obtain $Y^+ = Y \sqcup Y^2 \sqcup Y^3$, that is:

$$Y^+ = \begin{pmatrix} 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 1 \\ \infty & 1 & 1 & 1 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 1 \end{pmatrix}$$



[Kristiansen & N.] If the body Q of a loop P has no “control circle”, then P causes no blow up in complexity.

As $\mathcal{C}_{\hat{Y}^*}$ is the reflexive, transitive closure of \mathcal{C}_Y , that suggests the following criterion on Y : $\infty \notin \text{Diag}(\hat{Y}^*)$

Method – 9

Lemma (Certificate for loopI).

Let $P := \text{LoopI } X_h [Q]$ be a program in X_1, \dots, X_n .

If Y is a certificate for Q such that $\infty \notin \text{Diag}(\hat{Y}^*)$, then P has a certificate Z of the form

$$Z = \begin{pmatrix} z_1 \\ \vdots \\ z_n \\ 0^n 1 \end{pmatrix} \sqcup 1_{n+1}$$

where z_1, \dots, z_n fall into two groups:

$$z_i := \begin{cases} \text{add}(z_i) & \text{if } i \in \text{ADD}(Y) \\ \text{else}(z_i) & \text{else} \end{cases}$$

The ADD-case – 1

Def. An $i \in \{1, \dots, n\}$ is **additive in Y** , $i \in \text{ADD}(Y)$, if $Y^+[i] \leq 1^n \infty$.

For $i \in \text{ADD}(Y)$ and $j \in \{1, \dots, n\}$, one has $j \rightarrow_Y i \iff Y[i][j] = 1$, and each $p \in \text{POLY}(Y^+[i])$ satisfies $p \leq c_i + \sum_{j \rightarrow_Y i} X_j$.

\rightsquigarrow For $i \in \text{ADD}(Y)$, one can read from $Y^+[i]$ a bp on X_i w.r.t.

loopI $X_h[\mathbf{Q}]$, except for the **coefficient for X_h** , and the **constant c_i** .

examples for $Y^+[i] \leq 1^n \infty$

$\infty \notin Y^+[i]$ (e.g. variable assign.)	bounding polynomial
$Y^+[i] = 0^n \infty$ (e.g. const. assign.)	$Y^+[i][n+1] + \sum_j Y^+[i][j] \cdot X_j$
$Y^+[i] = 0^{i-1} 1 0^{n-i} \infty$ (e.g. push/inc)	$c_i \geq 0$
	$X_i + c_i \cdot X_h$

The ADD-case – 2

In the general situation of an additive i in Y ,

X_i will depend on other variables X_j for which j is also additive in Y .

In those cases, there will be an accumulation of values for c_h or c_i .

Example. Consider the following program

$Q ::= \text{push}(a, X_j); X_i = X_j; \text{push}(a, X_i)$.

Then X_i w.r.t. Q is polynomially bounded by $(X_j + 1) + 1$.

Hence X_i w.r.t. $\text{loopI } X_h [Q]$ is polynomially bounded by

$X_i + (X_j + 1) + 1$.

The ADD-case – 3

Def. Let $Y \in \mathcal{M}_n[A]$ be any possible certificate.

$V_Y(i) := \{j \in \{1, \dots, n\} \setminus \{i\} \mid j \rightarrow_Y i\}$ (imme. predecessors of i in Y)

$V_Y^+(i) := \{j \in \{1, \dots, n\} \setminus \{i\} \mid j \xrightarrow{+}_Y i\}$ (predecessors of i in Y)

Lemma (Partial Ordering I and ADD-Closure).

If $\infty \notin \text{Diag}(\hat{Y}^*)$, then the following holds:

- $\xrightarrow{*}_Y$ is a partial ordering (po) of $\{1, \dots, n\}$
- $i \in \text{ADD}(Y)$ and $j \xrightarrow{+}_Y i \implies j \in \text{ADD}(Y)$
- $i \in \text{ADD}(Y)$ and $i \rightarrow_Y i \implies \forall j \in V_Y^+(i) : j \not\rightarrow_Y i$

Thus, one can proceed by induction on $\xrightarrow{*}_Y$, and the ADD-case can be treated separately.

The ADD-case – 4

For $i \in \text{ADD}(Y)$ one defines

$$\text{add}(z_i) := (y_1, \dots, y_{h-1}, Y^*[i][h] + K_i, y_{h+1}, \dots, y_n, C_i)$$

where $y_j := Y^*[i][j]$ for $j = \{1, \dots, n\} \setminus \{h\}$, and

$$C_i := \begin{cases} 0 & \text{if } Y[i][i] = 1 \\ Y[i][n+1] + \sum_{j \in V_Y(i)} C_j & \text{if } Y[i][i] = 0 \end{cases}$$
$$K_i := \begin{cases} Y[i][n+1] + \sum_{j \in V_Y(i)} C_j & \text{if } Y[i][i] = 1 \\ \sum_{j \in V_Y(i)} K_j & \text{if } Y[i][i] = 0 \end{cases}$$

The ADD-case – 5

Proof sketch. For $i \in \text{ADD}(Y)$, and for each round $m \geq 0$ define

$$q_{i,m}(\vec{\mathbf{X}}) := \sum_{j=1}^n Y^m[i][j] \cdot \mathbf{X}_j + K_i \cdot m + C_i.$$

By induction on m , simultaneously for all $i \in \text{ADD}(Y)$, one obtains:

$$\{|\vec{\mathbf{X}}| = \vec{s}\} \mathbf{Q}^m \{|\mathbf{X}_i| \leq q_{i,m}(\vec{s})\}$$

As $Y^{s_h}[i][j] \leq Y^*[i][j]$, a polynomial bound $p_i \in \text{POLY}(\text{add}(z_i))$ on $|\mathbf{X}_i|$ w.r.t. $\text{loopI } \mathbf{X}_h \mathbf{Q}$ is obtained by setting

$$p_i(\vec{\mathbf{X}}) ::= \sum_{j=1}^n Y^*[i][j] \cdot \mathbf{X}_j + K_i \cdot \mathbf{X}_h + C_i$$

recalling $\text{add}(z_i) = (y_1, \dots, y_{h-1}, Y^*[i][h] + K_i, y_{h+1}, \dots, y_n, C_i)$. \square

The Else-case

For $i \notin \text{ADD}(Y)$, we know $i \neq h$ (since $h \in \text{ADD}(Y)$), and we construct (in the style of [Kristiansen & N.] a polynomial bound p_i on $|\mathbf{X}_i|$ w.r.t. $\text{LoopI } \mathbf{X}_h [\mathbf{Q}]$ of the form

$$p_i = \mathbf{X}_i + \mathbf{X}_h \cdot q_i$$

for some polynomial q_i in $\{\mathbf{X}_j \mid j \in V_Y(i)\} \cup \{\mathbf{X}_h\}$.

Therefore, we define $\text{else}(z_i)$ componentwise as follows:

$$\text{else}(z_i)[j] := \begin{cases} 1 & \text{if } j = i \\ \infty & \text{if } j = h \\ 0 & \text{if } j = n + 1 \\ \hat{Y}^*[i][j] & \text{else.} \end{cases}$$

For the “else” case, $\hat{Y}^*[i][j]$ is provably in $\{\infty, 0\}$.

□

Method – 10

Recall that for $\text{loopII } X_h [Q]$, the body Q is executed $2^{|X_h|} - 1$ times.

Lemma (Partial Ordering II).

For any $Y \in \mathcal{M}_n[A]$, if

$$(II) \quad Y^+[i] = 1_{n+1}[i] \text{ or } Y^+[i][i] = 0 \text{ for } i=1, \dots, n$$

then \rightarrow_{Y^*} is a partial ordering of $\{1, \dots, n\}$. $(Y^* := 1_n \sqcup M^+)$

Lemma (Certificate for loopII).

Let $P := \text{loopII } X_h [Q]$ be a program in X_1, \dots, X_n .

If Y is a certificate for Q with (II), then Y^* is a certificate for P .

Method – 11

Proof. For $m \geq 0$ define $q_{1,m}(\vec{x}), \dots, q_{n,m}(\vec{x})$ such that for $i=1, \dots, n$:

- (A) $\{|\vec{x}| = \vec{s}\} \cap \{|\mathbf{X}_i| \leq q_{i,m}(\vec{s})\}$ for all $m \geq 0$
- (B) $q_{i,m} \in \text{POLY}(Y^m[i])$ for all $m > 0$
- (C) $\exists m_i \geq 0$ such that $q_{i,m} = q_{i,m_i}$ for all $m \geq m_i$.

One obtains a pb $p_1 \in \text{POLY}(Y^*[1]), \dots, p_n \in \text{POLY}(Y^*[n])$ on P by

$$p_i := \bigsqcup_{m \leq m_i} q_{i,m}$$

By (B) and Structure: $p_i \in \text{POLY}(\bigsqcup_{m \leq m_i} Y^m[i]) \subseteq \text{POLY}(Y^*[i])$

By (A) and (C): $\{|\vec{x}| = \vec{s}\} \cap \{|\mathbf{X}_i| \leq q_{i,2^{s_h}-1}(\vec{s})\} \subseteq p_i(\vec{s})$.

Method – 12

Construction of $q_{1,m}(\vec{X}), \dots, q_{n,m}(\vec{X})$ satisfying (A),(B),(C): Since Y is a **certificate** for Q , there exist $q'_1 \in \text{POLY}(Y[1]), \dots, q'_n \in \text{POLY}(Y[n])$ s.t.:

$$(*) \quad \{|\vec{X}| = \vec{s}\} \mathbf{Q} \{|\mathbf{X}_j| \leq q'_j(\vec{s})\}$$

We define the required polynomials $q_{i,0}, q_{i,1}, q_{i,2}, \dots$ inductively by:

$$q_{i,0} := \mathbf{X}_i$$

$$q_{i,m+1} := q'_i(r_1, \dots, r_n), \text{ where } r_j := \begin{cases} q_{j,m} & \text{if } j \in V_Y(i) \cup \{i\} \\ 0 & \text{else.} \end{cases}$$

(A), (B) follow by an easy induction on $m \geq 0$, using (*) and **Sequence**:

$$q'_i \in \text{POLY}(Y[i]), r_1, \dots, r_n \in \text{POLY}(Y^m[i]) \implies q_{i,m+1} \in \text{POLY}(Y \otimes Y^m[i])$$

(C) follows by easy induction on $\xrightarrow{*} Y$. \square

Embedding of μ -measure 0 programs

Lemma (Embedding of Control).

Let P be any loop/stack program, and let $Z \in \mathcal{M}_n[A]$ be a certificate for P . Then for all $i, j \in \{1, \dots, n\}$, one has:

- (a) $Z[i][j] \geq 1$ and $i \neq j \implies X_j \not\rightarrow_P X_i$
- (b) $Z[i][i] = \infty \implies X_i \not\rightarrow_P X_i$ (top circle!)

Note. (a), (b) are distinct cases, e.g. $P := \text{foreach } X_n [\text{pop}(X_1)]$. As $M(\text{pop}(X_1)) = 1_{n+1}$, we obtain $M(P) = 1_{n+1}$; hence $M(P)[i][i] = 1$, but $X_i \not\rightarrow_P X_i$.

Theorem (Embedding of μ -measure 0).

Every loop/stack program of μ -measure 0 has a certificate.

Optimality

Fact ([Kristiansen & N.]). There is no certification method for the psb-property that certifies all psb programs.

Def. A certificate $Z \in \mathcal{M}_n[A]$ for a program P is **honest**, if there exist constants $\vec{m}_i \geq \vec{1}$ such that for all $i \neq j$ in $\{1, \dots, n\}$,

$$\begin{aligned} Z[i][i] \geq 1 &\implies \{|\vec{X}| = \vec{s} \geq \vec{m}_i\} P \{|\mathbf{X}_i| \geq s_i\} \\ Z[i][j] \geq 1 &\implies \{|\vec{X}| = \vec{s} \geq \vec{m}_i\} P \{|\mathbf{X}_i| \geq s_i + s_j\} \\ Z[i][i] = \infty &\implies \{|\vec{X}| = \vec{s} \geq \vec{m}_i\} P \{|\mathbf{X}_i| \geq 2 \cdot s_i\} \\ Z[i][n+1] \geq 1 &\implies \{|\vec{X}| = \vec{s} \geq \vec{m}_i\} P \{|\mathbf{X}_i| \geq s_i + 1\}. \end{aligned}$$

Core programs are built from honestly certified basic instructions by sequencing and loop statements.

Thm (Optimality). For core programs P, the following holds true:

$$P \text{ has a certificate} \iff P \text{ is psb}$$

Examples of certified algorithms

- elementary school method algorithms like
BINARY ADDITION
BINARY MULTIPLICATION
- benchmark algorithms like INSERTION SORT

There exists a **Java applet** of the present certification method by

Jan Mehler (TU-Ilmenau)

Everybody is invited to a **demonstration**
at any time !