Certifying polynomial time and linear/polynomial space

improvements and optimality

Karl-Heinz Niggl

(Join work with Henning Wunderlich and Jan Mehlher)

Geocal’06: Implicit Computational Complexity

February 13-17, 2006
Marseille-Luminy
No mixed datatypes

No vital instructions such as assignment statements $x = x$

Except for non-size-increasing BL's,

No user-friendly basic instructions (BL's)

From a programming perspective, this is not practically appealing:

Loop programs of $n$-measure $f(n) = \text{FLINSPACE}$

Stack programs of $n$-measure $f(n) = \text{PTIME}$

Earlier research [Kristiansen & N.]:

Starting point
Programs in this talk are built from arbitrary Bl's. Tmp's, \( (x_1', \ldots, x'_n, \ldots) \) by

Each variable \( x_i \) may represent any datatype, e.g.

- stacks, registers, trees, graphs, arrays, . . .

- each variable \( x_i \) might be the unary or binary length of the number stored in \( x_i \).

- \( x_i \) serves as a register and is implicitly equipped with a notion of size \( |x_i| \). e.g.

- \( |x_i| \) is the unary or binary length of the number stored in \( x_i \).
Novelty: A new method of certifying polynomial size boundedness (psb) for such imperative programs, provided that all BI’s are psb, too.

- During its execution, the contents of $X^h$ remain unchanged.
  - Loop $I^h [Q]$, the body $Q$ is executed $2 |X^h| - 1$ times.
  - Loop $I^h [Q]$, the body $Q$ is executed $|X^h|$ times.
  - Loop $I^h [Q]$, the body $Q$ is executed $|X^h|$ times.

- For each instance of a loop statement
- Each instance of (cond) be evaluated in polynomial time
- No specification on (cond) or loop statements, except that

General outline – 2
Theorem (Optimality).
The new method is optimal on core programs, i.e., programs built from honestly certified B1's by sequencing and loops.

\( \text{fptime} = \text{Certified string programs (stack programs built from polynomial-time computable B1's).} \)

\( \text{flinspace} = \text{Certified general loop programs (loop programs built from linear-space computable B1's).} \)

\( \text{fpspace} = \text{Certified power string programs (string programs built from polynomial-space computable B1's).} \)

Theorem (Characterizations).
Method – 1

A program $P$ in variables $X_1, \ldots, X_n$ is polynomially size bounded (psb) iff there are polynomials $p_1, \ldots, p_n$ such that:

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

Def. A program $P$ in variables $X_1, \ldots, X_n$ is polynomial bound (pb) on $P$.

Call $p_1, \ldots, p_n$ a polynomial bound (pb) on $P$. Call $p_1, \ldots, p_n$ a polynomial bound (pb) on $P$.

Def. A program $P$ in variables $X_1, \ldots, X_n$ is polynomially size bounded (psb) iff there are polynomials $p_1, \ldots, p_n$ such that:

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

Def. A program $P$ in variables $X_1, \ldots, X_n$ is polynomial bound (pb) on $P$. Call $p_1, \ldots, p_n$ a polynomial bound (pb) on $P$. Call $p_1, \ldots, p_n$ a polynomial bound (pb) on $P$.
Method – 2

Big Question: Given a pb on the body 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:

\[
\cdots + \frac{u}{\ell} x \cdots \frac{1}{\ell} x \cdot c + \cdots + c_0 = (x) d
\]

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

\( A \) ordered by \( \{0, 1, \infty\} \):
Example: \((\tilde{\mathbf{p}}, \tilde{\mathbf{p}})\) with rows \((\tilde{p}_1), \ldots, (\tilde{p}_n), 0^n 1\) is a certificate for \(P\).

For \(\tilde{a} \in A^{n+1}\), the class of polynomials of bound \(\tilde{a}\) is defined by:

\[
\begin{align*}
\langle p \rangle [j] &= \begin{cases} 
0 & \text{if } X_j \notin p_i \\
1 & \text{if } p_i = X_j + q(X / X_j) \\
\infty & \text{else}
\end{cases} \\
\langle p \rangle [n+1] &= \begin{cases} 
c_0 & \text{if } k(p_i) = c_0 \leq 1 \\
\infty & \text{else}
\end{cases}
\end{align*}
\]

A certificate for \(P\) is any \((n+1) \times (n+1)\) matrix \(Y\) over \(A\) with last row \(0^n 1\) such that \(\exists p b q \in \text{poly}(Y[1]), \ldots, q_n \in \text{poly}(Y[n])\) on \(P\).
<table>
<thead>
<tr>
<th>Method – 4 (Examples for ( n = 2 ))</th>
<th>Imperative certificate</th>
<th>Bounding polynomial</th>
<th>Certificate</th>
</tr>
</thead>
<tbody>
<tr>
<td>( \begin{pmatrix} 1 &amp; 0 &amp; 0 \ 0 &amp; 1 &amp; 1 \ 0 &amp; 0 &amp; 1 \end{pmatrix} )</td>
<td>( X + \mathcal{I} X )</td>
<td>( \begin{pmatrix} 1 &amp; 0 &amp; 0 \ 0 &amp; 1 &amp; 1 \ 0 &amp; 0 &amp; 1 \end{pmatrix} )</td>
<td>( X = \mathcal{I} X )</td>
</tr>
<tr>
<td>( \begin{pmatrix} 1 &amp; 0 &amp; 0 \ 0 &amp; 0 &amp; 1 \ 0 &amp; 0 &amp; 1 \end{pmatrix} )</td>
<td>( \mathcal{I} X )</td>
<td>( \begin{pmatrix} 1 &amp; 0 &amp; 0 \ 0 &amp; 0 &amp; 1 \ 0 &amp; 0 &amp; 1 \end{pmatrix} )</td>
<td>( X = \mathcal{I} X )</td>
</tr>
<tr>
<td>( \begin{pmatrix} 1 &amp; 0 &amp; 0 \ 1 &amp; 1 &amp; 0 \ 0 &amp; 0 &amp; 1 \end{pmatrix} )</td>
<td>( \mathcal{I} + \mathcal{I} X )</td>
<td>( \begin{pmatrix} 1 &amp; 0 &amp; 0 \ 1 &amp; 1 &amp; 0 \ 0 &amp; 0 &amp; 1 \end{pmatrix} )</td>
<td>( \mathcal{I} X = \mathcal{I} X )</td>
</tr>
<tr>
<td>( \begin{pmatrix} 1 &amp; 0 &amp; 0 \ 1 &amp; 0 &amp; 0 \ 0 &amp; 0 &amp; 1 \end{pmatrix} )</td>
<td>( \mathcal{I} X )</td>
<td>( \begin{pmatrix} 1 &amp; 0 &amp; 0 \ 1 &amp; 0 &amp; 0 \ 0 &amp; 0 &amp; 1 \end{pmatrix} )</td>
<td>( \mathcal{I} X = \mathcal{I} X )</td>
</tr>
</tbody>
</table>
Method – 5

[\mathcal{A}]^u \mathcal{M} \text{ and } \mathcal{A}^u \mathcal{M}

\text{denote the componentwise extensions to } \mathcal{A}^u \mathcal{M} \text{ and } \mathcal{A}^u \mathcal{M}

\text{preserves the last row property, and with neutral element } [I]^u,\text{ which is associative,}

\text{ matrix multiplication } \otimes \text{ on } [\mathcal{A}]^u \mathcal{M}\text{, which is associative,}

\text{denote the componentwise extensions to } \mathcal{A}^u \mathcal{M}

\text{are commutative, associative, with neutral elements } 0, [A], I, 0

\text{are commutative, associative, with neutral elements } 0, [A], I, 0

\text{with the last row property.}

\text{Operations on } \mathcal{A}^u \mathcal{M} = \{[\infty], 0, I\}
Lemma (Base). If \( \text{tmp}(\overline{x}) \) is a basic instruction with pb \( p_1 \) of \( \text{tmp}(\overline{x}) \), then \( Z \) is a certificate for \( \text{tmp}(\overline{x}) \).

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

(a) \( u \leq v \) \( \Rightarrow \) \( \text{poly}(u) \subseteq \text{poly}(v) \) (b) \( p \in \text{poly}(u), q \in \text{poly}(v) \) \( \Rightarrow \) \( p \sqcup q \in \text{poly}(u \sqcup v) \)

Corollary (Conditional). \( Z_1, Z_2 \) are certificates for \( P_1, P_2 \) \( \iff \) \( Z_1 \sqcap Z_2 \) is a certificate for \( if(\text{cond}) then P_1 else P_2 \).
Method–7

Lemma (Composition and Sequence).

(a) $q \in \text{poly}(v)$, $p_1 \in \text{poly}(w_1), \ldots, p_n \in \text{poly}(w_n) \Rightarrow q(p_1, \ldots, p_n) \in \text{poly}(v \otimes M)$ with $M := \begin{bmatrix} w_1 & \ldots & w_n \\ 0 & \ldots & 0 \end{bmatrix}$.

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

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

$$\left( \begin{array}{c} w_0 \\ u_1 \\ \vdots \\ u_n \end{array} \right) = \begin{array}{c} v_1 \\ \vdots \\ v_n \end{array} \text{ with } (\begin{array}{c} M \\ \vdots \\ M \end{array} \otimes \begin{array}{c} Z_1 \\ Z_2 \end{array})$$


\[ \text{gives full information on the usage of } X_i \text{ in } q \text{ w.r.t. } A. \]
Given a certificate $Y$ for the body $Q$ of a loop $P$ with variables among $X_1, \ldots, X_n$:

1. **What criterion on $Y$ guarantees a certificate $Z$ for $P$?**
2. **How can one construct $Z$ from $Y$?**

**Know**: For each $m \geq 0$, $Y_m$ is a certificate for $Q_m$ (= $Q$; $\ldots$; $Q$, $m$ times).

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

- **Non-monotonic**:
  
  $$ \lim_{\gamma \to \infty} Y \prod_{\gamma} =: +Y $$

- **Monotonic**:
  
  $$ \lim_{\gamma \to \infty} Y \prod_{\gamma} =: \Lambda $$

**Method**

Big Question: Given a certificate $\Lambda$ for the body $Q$ of a loop $P$ with variables among $X_1, \ldots, X_n$. 

- Method 8
Intuition

An entry \( Y_{ij} \) represents:

- A possible data-flow in \( P \) from \( X_j \) to \( X_i \) denoted by \( j \rightarrow Y_{ij} \) (read "\( j \) controls \( i \) in \( Y_{ij} \)), for \( j = 1, \ldots, n \), or
- A possible influence of a constant on \( X_i \) denoted by \( \ast \rightarrow Y_{ij} \) (\( \ast \) controls \( \ast \) in \( Y_{ij} \)), for \( j = n + 1 \).

The control graph \( C_Y \) of \( Y \) is then defined by

\[
C_Y := \left( \{ 1, \ldots, n, \ast \}, \rightarrow_Y \right)
\]

Let \( C_Y^\ast \) denote the transitive (transitive, reflexive) closure of \( \rightarrow_Y \) (\( \ast \rightarrow \) (\( \ast \rightarrow \) (\( \ast \rightarrow \) (\( \ast \rightarrow \) ...))).

Lemma (Control Graph)

(a) \( C_Y^\ast \) is the reflexive closure of \( C_Y \).
(b) \( C_Y^+ \) is the transitive closure of \( C_Y \).

Intuition

An entry \( Y_{ij} \) with \( i \neq 0 \) \( \leq \) \( \lfloor \lfloor 0 \rfloor \rfloor \) \( \neq \) \( \lfloor \rfloor \) represents \( \{ u, \ldots, i \} \).
Example. Consider the following stack program $Q$.

Control Graphs and Certificates – 2
Since $Y_3 = Y_4 = Y_5 = \ldots$, we obtain

$Y_\infty = Y_1$.

As $C^*_Y$ is the reflexive, transitive closure of $C^*_Y$, this suggests the following criterion on $X: Y \in \neq \Delta_{Diag}(Y^*)$.

If the body $Q$ of a loop $P$ has no "control circle", then $P$ causes no blowup in complexity.

[Kristiansen & N.]

As $C^*_Y$ is the reflexive, transitive closure of $C^*_Y$, this suggests the following criterion on $Y$:

\[ \frac{\infty}{\infty} \notin \text{Diag}(\hat{Y}^*) \]

That is:

\[ -Y \cap Y \cap Y = +Y \]

Since $Y_\infty = Y_1 = Y_2 = \ldots$, we obtain

$Y_\infty = Y_1 = Y_2 = \ldots = +Y$.

Control Graphs and Certificates - 4
Method–9

Lemma (Certificate for Loop I)

Let $P \equiv \text{loop IX} [\mathcal{Q}]$ be a program in $X_1, \ldots, X_n$. If $Y$ is a certificate for $Q$ such that $\infty / \in Diag(\hat{\mathcal{Y}} \odot \mathcal{Y})$ and $\not \in ADD(\mathcal{Y})$, then $P$ has a certificate of the form $Z$ where $Z_1, \ldots, Z_n$ fall into two groups:

$$
\begin{pmatrix}
I_u0 \\
\vdots \\
I_z
\end{pmatrix}
$$

The $P \equiv \text{loop IX} [\mathcal{Q}]$ has a certificate for Loop I.
The *ADD-case* – 1

Def. An $i \in \{1, \ldots, n\}$ is additive in $Y$, if $Y + [i] \leq 1^\infty$.

For $i \in \text{ADD}(Y)$ and $j \in \{1, \ldots, n\}$, one has $j \rightarrow Y$ a bp on $X_i$ w.r.t. $X_j$.

Examples for $Y + [i] \leq 1^\infty$ and each $p \in \text{poly}(Y + [i])$ satisfies $p \leq c_i + \sum_{j \rightarrow Y} X_j$.

For $i \in \text{ADD}(Y)$, one can read from $Y + [i]$ a bp on $X_i$, except for the coefficient for $X_h$, and the constant $c_i$.

<table>
<thead>
<tr>
<th>Examples for $Y + [i] \leq 1^\infty$</th>
<th>Bounding polynomial</th>
</tr>
</thead>
<tbody>
<tr>
<td>$Y + [i] = 0^i \cdot 1^{n-i}$ (e.g. const. assign.)</td>
<td>$Y + [i][j] = 0$</td>
</tr>
<tr>
<td>$Y + [i] = 10^n \cdot 0^i$ (e.g. push/inc)</td>
<td>$Y + [i][j] = 1$</td>
</tr>
<tr>
<td>$Y + [i] = 10^n$ (e.g. variable assign.)</td>
<td>$Y + [i][j] = 1$</td>
</tr>
<tr>
<td>$X_i + c_i \cdot X_h$</td>
<td>$Y + [i][j] = 1$</td>
</tr>
</tbody>
</table>

Note: For $i \in \text{ADD}(Y)$, one can read from $Y + [i]$ a bp on $X_i$ w.r.t. $X_j$. Each $p \in \text{poly}(Y + [i])$ satisfies $p \leq c_i + \sum_{j \rightarrow Y} X_j$.
In the general situation of an additive case, $X_i$ will depend on other variables $X_j$ for which $f$ is also additive in $Y$. Then $X_i$ w.r.t. a program $Q$ is polynomially bounded by $(X_j + 1) + 1$.

Example. Consider the following program:

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

Hence $X_i$ w.r.t. loop $Q$ is polynomially bounded by $(X_j + 1) + 1$.
The ADD-case – 3

Thus, one can proceed by induction on \( \mathcal{A} \), and the ADD-case can be treated separately.

\[ \forall \mathcal{A} \in \mathcal{M}_n \left( \begin{array}{c} \mathcal{A} \in \text{ADD}(\mathcal{Y}) \wedge \mathcal{Y} \in \mathcal{M}_n \end{array} \right) \]

\[ \mathcal{A} \in \text{ADD}(\mathcal{Y}) \Leftrightarrow \mathcal{Y} \in \text{ADD}(\mathcal{A}) \]

Lemma (Partial Ordering and ADD-Closure).

If \( \mathcal{Y} \in \mathcal{M}_n \) is a partial ordering (po) of \( \{1, \ldots, n\} \), then the following holds:

\[ \mathcal{Y} \in \text{ADD}(\mathcal{Y}) \]

Let \( i \in \mathcal{M}_n \) be any possible certificate.

\[ \forall \mathcal{Y} \in \mathcal{M}_n \]

The ADD-case – 3
The \text{ADD-case} – 4

For $i \in \text{ADD}(Y)$ onedefines $\text{add}(z_i) := (y_1, \ldots, y_h-1, Y^*[i][h]+k_i, y_{h+1}, \ldots, y_n, c_i)$ where $y_j := Y^*[i][j] \text{ for } j = \{1, \ldots, n\} \backslash \{h\}$, and $c_i := \begin{cases} 0 & \text{if } Y^*[i][i] = 1 \\ Y^*[i][n+1]+ \sum_{j \in V} Y^*(i) C_j & \text{if } Y^*[i][i] = 0 \end{cases}$

\[ \begin{array}{c}
0 = [e][i] Y \parallel \underbrace{\ell X}_{\forall \ldots \in \ell} + [I+u][i] Y \\
1 = [e][i] Y \parallel \underbrace{\ell C}_{\forall \ldots \in \ell} + [I+u][i] Y \\
0 = [e][i] Y \parallel \underbrace{\ell C}_{\forall \ldots \in \ell} + [I+u][i] Y \\
1 = [e][i] Y \parallel \end{array} \]

where $\{y\} \backslash \{u, \ldots\} = \ell$ for $[\ell][i] \star Y = : \ell Y$

and defines \((\ell C, u_{hi} \cdot \ldots \cdot h_{hi} + 1, Y + [y][i] \star Y, t_{hi} \cdot \ldots \cdot t_{hi}) = : (\ell z) \text{add} \text{ADD}(Y) \) one defines

\text{The ADD-case} – 4
The ADD-case

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

\[
\begin{align*}
\forall i \in \text{ADD}(Y), \quad \text{and for each round } m \geq 0: \quad & \quad \text{define } q_{i,m}(\vec{X}) := n \sum_{j=1}^{Y[m][i][j]} X_j + K_i \cdot m + C_i.
\end{align*}
\]

By induction and calling $\text{add}(\vec{x})$ is obtained by setting $w.r.t. \text{loop } j$ an

\[
\{ (\vec{s})_{w \vec{i},b} \geq |\vec{x}| \} \quad \text{and} \quad \{ s = |x| \}
\]

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

\[
\forall i \in \text{ADD}(Y), \quad \text{and for each round } m \geq 0:
\]

\[
\begin{align*}
\forall i \in \text{ADD}(Y), \quad \text{and for each round } m \geq 0: \quad & \quad \text{define } q_{i,m}(\vec{X}) := n \sum_{j=1}^{Y[m][i][j]} X_j + K_i \cdot m + C_i.
\end{align*}
\]
For the "else" case, we know:

\[ [\mathcal{L}][i] * \mathcal{X} \]

\[
\begin{cases}
1 + u = \mathcal{L} & \text{if } 1 \\
\eta = \mathcal{L} & \text{if } 0 \\
\gamma = \mathcal{L} & \text{if } \infty \\
\end{cases}
\]

\[ =: [\mathcal{L}]_{(z) (\text{else})} \]

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

\[ \{ \gamma X \} \cap \{ (\gamma) \mathcal{X} \in \mathcal{L} \mid \gamma X \} \]

for some polynomial \( \gamma \) in \( \mathcal{L} \). The polynomial \( \gamma \) of the form

\[ \gamma b \cdot \gamma X + \gamma X = \gamma d \]

w.r.t. \( \{ \gamma \} \) is provably in \( \{ 0, \infty \} \). In the style of Kristiansen & N. a polynomial bound on \( \gamma \) constitutes in the style of Kristiansen & N. a polynomial bound on \( \gamma \). Since \( \gamma \notin \text{ADD}(\mathcal{L}) \), we know \( \gamma \neq i \) and we have

The Else-case
Recall that for loop \( I \) \( \text{body} \) \( g \), the body \( g \) is executed \( 2^{|x_n| - 1} \) times.

**Lemma (Certificate for Loop II).** Let \( P \equiv \text{loop} \{ x \} \mid h \) \( \text{be a program in} \ X_1, \ldots, X_n \).

For any \( Y \in M \) \( \{ x \} \), if
\[
Y[i] = 1 + u \text{ or } Y[i] = 0 \quad \text{for} \ i = 1, \ldots, n
\]
then \( Y \) is a certificate for \( P \).

**Lemma (Partial Ordering II).**
\[
(\forall \lambda \in \text{I} \equiv \text{\text{I}}^u \lambda = : \lambda^u \text{I})
\]
\[
\{ u, \ldots, \} \quad \text{is a partial ordering of} \ \text{I} \text{.}
\]

\[
\text{for } \forall \lambda \in \text{I} \equiv \text{I}^u \lambda = : \lambda^u \text{I}
\]

For any \( \lambda \in \text{I} \equiv \text{I}^u \lambda = : \lambda^u \text{I} \) \( \text{II} \).

Recall that for loop \( \text{II} \) \( \text{body} \) \( g \), the body \( g \) is executed \( 2^{|x_n| - 1} \) times.

**Method - IO**
For $m \geq 0$ define $q_{1,m}(\vec{X}), \ldots, q_{n,m}(\vec{X})$ such that for $i = 1, \ldots, n$:

(A) \{ |\vec{X}| = \vec{s} \} \cap \{ s = |X| \} 
(B) (\mathbb{P} \cdot X) \trianglerighteq ((\mathbb{P} \cdot X) \wedge (\mathbb{P} \cdot X)) \trianglerighteq m 

\therefore \prod_{m \geq m} = \trianglerighteq m

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

$$(\forall m \geq m) \exists m \geq m$$ such that $0 \sim 0 \in \trianglerighteq m (C)$

$$(\forall m \geq m) \exists m \geq m$$ such that $0 \leq m \in \trianglerighteq m (B)$$

$$(\forall m \geq m) \exists m \geq m$$ such that $0 \sim 0 \in \trianglerighteq m (A)$$

Method - II
Method 12

\[ \vdash \] follows by easy induction on \( \star \).

\[(\mathcal{L}(\text{poly}(Y)) \land \mathcal{X}) \iff (\mathcal{L}(\text{poly}(Y)) \land \mathcal{X}) \uplus \text{poly}(\mathcal{X}) \]

\( \forall \mathcal{A}, \mathcal{B}, \mathcal{X} \)

\( \{ \mathcal{A} \} \land (\mathcal{A} \land \mathcal{A}) \quad \mathcal{B} \quad \mathcal{A} \in \mathcal{L}(\text{poly}(Y)) \quad \mathcal{X} \in \mathcal{X} \)

We define the required polynomials \( \mathcal{A}, \mathcal{B}, \mathcal{X} \) inductively by:

\( \{ (s) \mathcal{B} \geq |s| \} \quad \& \quad \{ s = |x| \} \)

\( \vdash \text{Construction of Certificate for } \mathcal{A}, \mathcal{B}, \mathcal{X} \) since \( \mathcal{A} \) is a

\( \text{Construction of Certificate for } \mathcal{A}, \mathcal{B}, \mathcal{X} \) since \( \mathcal{A} \) is a

\( \vdash \)
Every loop/stack program of $\mu$-measure 0 has a certificate.

**Theorem (Embedding of $\mu$-measure 0).**

Let $P$ be any loop/stack program, and let $Z \in \mathcal{M}$ be a certificate.

**Note.** (a) $(\exists)$ are distinct cases, e.g.: $P \equiv \forall X : \mathcal{E}$.

As $\mathcal{M}(\mathcal{E}) = 1$, we obtain $\mathcal{M}(P) = 1 + u$.

Let $P$ be any loop/stack program, and let $Z \in \mathcal{M}$ be a certificate.

**Lemma (Embedding of Control).**
There is no certification method for the psb-property that certifies all psb programs. Core programs are built from honestly certified basic instructions by sequencing and loop statements.

Core programs. For core programs $P$, the following holds true:

\[ \begin{align*}
\{ 1 + \hat{s} \preceq 1 \} & \quad \text{and} \quad \{ \mu \preceq s = |X| \} \quad \iff \quad 1 \preceq [1+\mu][\hat{s}] Z \\
\{ \hat{s} \cdot \mathcal{Z} \preceq 1 \} & \quad \text{and} \quad \{ \mu \preceq s = |X| \} \quad \iff \quad \infty = [\hat{s}][\mu] Z \\
\{ \hat{s} + \hat{s} \preceq 1 \} & \quad \text{and} \quad \{ \mu \preceq s = |X| \} \quad \iff \quad 1 \preceq [\hat{s}][\hat{s}] Z \\
\{ \hat{s} \preceq 1 \} & \quad \text{and} \quad \{ \mu \preceq s = |X| \} \quad \iff \quad 1 \preceq [\hat{s}][\mu] Z
\end{align*} \]

\{ \mu \neq i \} \text{ such that for all } i \in \{ 1, \ldots, n \} \text{ a constant } m \preceq \mu \text{ not exist, if there exist a certificate } \forall \mathcal{Z} \in \mathcal{Z} \text{ that certifies } \forall \text{ psb programs.}

\text{Fact ([Kristiansen & N.]): There is no certification method for the Optimality.}
Examples of certified algorithms

• elementary school algorithms like binary addition
  binary multiplication
  binary addition

• benchmark algorithms like insertion sort

Everybody is invited to a demonstration

Jan Mehler (TU Ilmenau)

There exists a Java applet of the present certification method by

\[ \text{Example of certified algorithms} \]