Chapter 2: Certifying FPTIME and FLINSPACE/FPSPACE for imperative programs

As seen, by earlier research [Kristiansen & N.] we know:

- Stack programs of $\mu$-measure $0 = \text{FPTIME}$
- Loop programs of $\mu$-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
§2.1 General outline

**Def.** (Restricted Imperative) **Programs** are built from

arbitrary basic instructions \( (\text{BI} \ imp(X_1,\ldots,X_n) ) \) by

- sequences \( P_1; P_2 \)
- conditionals \( \text{if (cond) then } P_1 \) and \( \text{if (cond) then } P_1 \text{ else } P_2 \)
- 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, …

provided \( X_i \) is implicitly equipped with a sensible notion of \( \text{size } |X_i| \), 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 \((\text{cond})\) or loop statements, except that

- each instance of \((\text{cond})\) be evaluated in polynomial time
  (as usual, in the size of the variables involved)

- for each instance of a loop statement
  - \(\text{loopI } X_h [Q]\), the body \(Q\) is executed \(|X_h|\) times
  - \(\text{loopII } X_h [Q]\), 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 [N. & Wunderlich]).**
- **FPTIME** = Certified string programs (stack programs built from polynomial-time computable BI’s).
- **FLINSPECE** = 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 loopII statements)

**Thm (Optimality [Mehler & N.]).**
The new method is optimal on core programs, i.e., programs built from honestly certified BI’s by sequencing and loops.
§2.2 Method

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

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

Call $\bar{p}$ a **polynomial bound (pb) on** $P$ and each $p_i$ a **pb on** $P$ w.r.t. $X_i$.

- **polynomials:** $p = c_0 + \ldots + c_i \cdot X_1^{i_1} \cdots X_n^{i_n} + \ldots$ (with $c_i \in \mathbb{N}$)
  
  Write $\kappa(p)$ for $c_0$ and $X_j \in p$ ($X_j \notin p$) for "$X_j$ does (not) occur in $p$".

- **maximum** $p \sqcup q$:
  
  $p = c_0 + \ldots + c_i \cdot X_1^{i_1} \cdots X_n^{i_n} + \ldots$
  
  $q = d_0 + \ldots + d_i \cdot X_1^{i_1} \cdots X_n^{i_n} + \ldots$
  
  $\Rightarrow p \sqcup q := \max(c_0, d_0) + \ldots + \max(c_i, d_i) \cdot X_1^{i_1} \cdots X_n^{i_n} + \ldots$
Method – 2

Examples for polynomial bounds (pb) on programs in $X_1, \ldots, X_n$:

- $X_1 + 1, X_2, \ldots, X_n$ is a pb on $\text{push}(a, X_1), \text{inc}(X_1)$.
- $X_2, X_2, \ldots, X_n$ is a pb on $X_1 = X_2, \text{sort}(X_2; X_1), \text{reverse}(X_2; X_1)$.
- $X_1, X_2 + X_1, X_3, \ldots, X_n$ is a pb on $\text{concat}(X_2, X_1), X_2 += X_1$.
- Let $p_1, \ldots, p_n$ be a pb on $P_1$, and let $q_1, \ldots, q_n$ be a pb on $P_2$.

Then

- $q_1(p_1, \ldots, p_n), \ldots, q_n(p_1, \ldots, p_n)$ is a pb on $P_1; P_2$
- $p_1 \sqcup q_1, \ldots, p_n \sqcup q_n$ is a pb on $\text{if} \ (\text{cond}) \ \text{then} \ P_1 \ \text{else} \ P_2$
- $p_1 \sqcup X_1, \ldots, p_n \sqcup X_n$ is a pb on $\text{if} \ (\text{cond}) \ \text{then} \ P_1$. 
Method – 3

**Big Question:** Given a pb \( \vec{q} \) on the body \( Q \) 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.

**Def.** Let \( \vec{p} \) be a pb on a prog. \( P \) in \( x_1, \ldots, x_n \). Then for each polynomial

\[
p_i(\vec{x}) = c_0 + \ldots + c_j \cdot x_1^{j_1} \ldots x_n^{j_n} + \ldots
\]

we store and process only an \( (n+1) \)-tuple \( \langle p_i \rangle \) (representation of \( p_i \)) over the **forgetting set** \( A := \{0, 1, \infty\} \) ordered by \( 0 < 1 < \infty \):
**Method – 4**

For $1 \leq j \leq n$, 

$$\langle p_i \rangle[j] := \begin{cases} 
0 & \text{if } X_j \notin p_i \\
1 & \text{if } p_i = X_j + q_i (\vec{X} \backslash X_j) \\
\infty & \text{else}
\end{cases}$$

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

**Def.** For $\vec{a} \in \mathcal{A}^{n+1}$, the class of **polynomials of bound** $\vec{a}$ is defined as

$$\mathrm{POLY}(\vec{a}) := \{ p(X_1, \ldots, X_n) \mid \langle p \rangle \leq \vec{a} \}.$$

**Def.** A **certificate for** $P$ is any $(n+1) \times (n+1)$ matrix $Y$ over $\mathcal{A}$ with last row $0^n1$ such that $\exists$ pb $q_1 \in \mathrm{POLY}(Y[1]), \ldots, q_n \in \mathrm{POLY}(Y[n])$ on $P$.

**Example:** $\langle P, \vec{p} \rangle$ with rows $\langle p_1 \rangle, \ldots, \langle p_n \rangle, 0^n1$ is a certificate for $P$. 
Method – 5 (Examples for \( n = 2 \))

<table>
<thead>
<tr>
<th>imperative</th>
<th>polynomial bound</th>
<th>certificate</th>
</tr>
</thead>
</table>
| `push(a, X_2), inc(X_2)` | \( X_1, X_2 + 1 \) | \[
\begin{pmatrix}
1 & 0 & 0 \\
0 & 1 & 1 \\
0 & 0 & 1
\end{pmatrix}
\] |
| `pop(X_2), reverse(X_2)` | \( X_1, X_2 \) | \( 1_3 \) |
| \( X_2 = X_1 \) | \( X_1, X_1 \) | \[
\begin{pmatrix}
1 & 0 & 0 \\
1 & 0 & 0 \\
0 & 0 & 1
\end{pmatrix}
\] |
| `concat(X_2, X_1), X_2 += X_1` | \( X_1, X_2 + X_1 \) | \[
\begin{pmatrix}
1 & 0 & 0 \\
1 & 1 & 0 \\
0 & 0 & 1
\end{pmatrix}
\] |
Method – 6

Idea for $\mathcal{A}$. Let $\vec{q}$ be a pb on the body $Q$ of a loop $P$, say

$$q_i = c_0 + \ldots + c_j \cdot x_1^{j_1} \cdot \ldots \cdot x_j^{j_s} \cdot \ldots \cdot x_n^{j_n} + \ldots$$

What is so critical about coefficients $c_j \geq 2$ or exponents $j_s \geq 2$ of $x_j$ in monomials or about constant coefficients $c_0 \geq 2$?

Fact: Through iteration one might exceed the realm of polynomials!

(D1) $f(x) \geq 2 \cdot x \implies$ iteration $f(y)(x) \geq 2^y \cdot x$

(D2) $f(x) \geq x^2 \implies$ iteration $f(y)(x) \geq x^{2^y}$

(D3) $q_i \geq 2 + r \implies$ composition $q_i(q_1, \ldots, q_n)$ plus iteration can lead to (D1) or (D2).
Method – 7

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

<table>
<thead>
<tr>
<th></th>
<th>0</th>
<th>1</th>
<th>$\infty$</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>1</td>
<td>$\infty$</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>$\infty$</td>
<td>$\infty$</td>
</tr>
<tr>
<td>$\infty$</td>
<td>$\infty$</td>
<td>$\infty$</td>
<td>$\infty$</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th></th>
<th>0</th>
<th>1</th>
<th>$\infty$</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>1</td>
<td>$\infty$</td>
</tr>
<tr>
<td>$\infty$</td>
<td>$\infty$</td>
<td>$\infty$</td>
<td>$\infty$</td>
</tr>
</tbody>
</table>

**Note.** $+$, $\bullet$, $\sqcup$ are commutative, associative, with neutral elements 0, 1, 0. Furthermore, $(\bullet, \sqcup)$ is distributive, and $a + (b \bullet c) \leq (a+b) \bullet (a+c)$, only.

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

**Note.** As usual, define matrix multiplication $\otimes$ on $\mathcal{M}_n[\mathcal{A}]$, which is associative, preserves the last row property, and has neutral element $1_n$.

**Def.** Let $+, \sqcup, <$ denote the componentwise extensions to $\mathcal{A}^m$, $\mathcal{M}_n[\mathcal{A}]$. 
**Method – 8**

**Def.** For matrices $A, B \in \mathcal{M}_n[A]$, the product $A \otimes B$ is defined as usual (for $i, j \in \{1, \ldots, n+1\}$):

$$(A \otimes B)[i][j] := A[i] \otimes B[\cdot][j] := \sum_{k=1}^{n+1} A[i][k] \cdot B[k][j]$$

In particular, $(A \otimes B)[i] = A[i] \otimes B$

**Corollary (Last row property).**

$\mathcal{M}_n[A]$ is closed under $\otimes$.

**Lemma (POLY).** Writing $X_i \in S(p)$ for $p = X_i + q(\vec{X}\setminus X_i)$, we have:

$p \in \text{POLY}(\vec{a}) \iff \langle p \rangle \leq \vec{a}$

$\iff \forall i \in \{1, \ldots, n\}: (a_i = 0 \Rightarrow X_i \notin p) \land$

$(a_i = 1 \Rightarrow X_i \in S(p) \lor X_i \notin p) \land$

$(a_{n+1} = 0 \Rightarrow K(p) = 0) \land$

$(a_{n+1} = 1 \Rightarrow K(p) \leq 1)$
Lemma (Base). For a basic instruction \(\text{imp}(\vec{X})\) with \(pb\ p_1, \ldots, p_n\), \(\langle \text{imp}(\vec{X}), \vec{p} \rangle\) with rows \(\langle p_1 \rangle, \ldots, \langle p_n \rangle\), \(0^n 1\) 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) \(p \in \text{POLY}(u), q \in \text{POLY}(v) \implies p \sqcup q \in \text{POLY}(u \sqcup v)\)

(c) \(\infty \notin u \implies \text{POLY}(u) \leq q_u := u[n+1] + \sum_{j=1}^{n} u[j] \cdot X_j, \ q_u \in \text{POLY}(u)\)

Corollary (Conditional).

\(Z_1, Z_2\) are certificates for \(P_1, P_2\) \(\implies\)

\(Z_1 \sqcup Z_2\) is a certificate for \(\text{if (cond) then } P_1 \text{ else } P_2\)

\(Z_1 \sqcup 1_{n+1}\) is a certificate of \(\text{if (cond) then } P_1\) □
Method – 10

Lemma (Sequence).

(a) \( Y \in \mathcal{M}_n[A], \, q \in \text{POLY}(u), \, p_1 \in \text{POLY}(Y[1]), \ldots, \, p_n \in \text{POLY}(Y[n]) \)

\[ \implies r := q(p_1, \ldots, p_n) \in \text{POLY}(u \otimes Y) \]

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

Proof. Clearly, (a) implies (b), as \((Z_2 \otimes Z_1)[i] = Z_2[i] \otimes Z_1\).

Idea for (a): For each \( j = 1, \ldots n \),

\[ (u \otimes Y)[j] = u \otimes Y[\cdot][j] = \sum_{k=1}^{n+1} u[k] \bullet Y[k][j] \]

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

Formally, check it by using Lemma \text{POLY}. Blackboard!