Certifying Complexity Analysis

Clément Aubert[1], Thomas Rubiano[2],
Neea Rusch[1], Thomas Seiller[2,3]

[1] Augusta University
[2] LIPN - Laboratoire d'Informatique de Paris-Nord
[3] CNRS - Centre National de la Recherche Scientifique

CoqPL 2023 21 January 2023

In This Presentation

We present a plan to formally verify a **complexity analysis technique** — mwp-flow analysis — in Coq.

- This is a technique from *Implicit Computational Complexity* (ICC).
- It guarantees program input variable values have polynomial growth bounds.

Implicit Computational Complexity (ICC)

Let L be a programming language, C a complexity class, and $[\![p]\!]$ the function computed by program p.

Find a restriction $R \subseteq L$, such that the following equality holds:

$$\{[\![p]\!] \mid p \in R\} = C$$

The variables L, C, and R are the parameters that vary greatly between different ICC systems[1].

[1] Romain Péchoux. *Complexité implicite : bilan et perspectives*. Habilitation à Diriger des Recherches (HDR). 2020. URL: https://hal.univ-lorraine.fr/tel-02978986.

Analyzing Variable Value Growth

For a deterministic imperative program,
is the growth of input variable values polynomially bounded?

Example

```
C′ ≡ X1 := X2 + X3;
X1 := X1 + X1
```

$[\![C']\!](x_1, x_2, x_3 \rightsquigarrow x_1', x_2', x_3')$
implies $x_1' \leq 2x_2 + 2x_3$
and $x_2' \leq x_2$ and $x_3' \leq x_3$.

```
C″ ≡ X1 := 1;
loop X2 { X1 := X1 + X1 }
```

$[\![C'']\!](x_1, x_2 \rightsquigarrow x_1', x_2')$
implies $x_1' \leq 2^{x_2}$ and $x_2' \leq x_2$.

mwp-Flow Analysis[2]

- Tracks how each variable depends on other variables.
- Flows characterize dependencies:

  0 – no dependency

  m – maximal *weaker*

  w – weak polynomial

  p – polynomial *stronger*
- Apply inference rules to program statements.
- Collect analysis result in a matrix.

[2] Neil D. Jones and Lars Kristiansen. "A flow calculus of *mwp*-bounds for complexity analysis". In: *ACM Trans. Comput. Log.* 10.4 (Aug. 2009), 28:1–28:41. DOI: 10.1145/1555746.1555752.

mwp-Flow Analysis Inference Rules

$$\frac{}{\vdash \mathsf{Xi} : \{^m_{\mathsf{i}}\}}\ \mathsf{E1}$$

$$\frac{\vdash \mathsf{C1} : M_1 \quad \vdash \mathsf{C2} : M_2}{\vdash \textbf{if}\ \mathsf{b}\ \textbf{then}\ \mathsf{C1}\ \textbf{else}\ \mathsf{C2} : M_1 \oplus M_2}\ \mathsf{I}$$

$$\frac{}{\vdash \mathsf{e} : \{^w_{\mathsf{i}} \mid \mathsf{Xi} \in \mathrm{var}(\mathsf{e})\}}\ \mathsf{E2}$$

$$\frac{\vdash \mathsf{Xi} : V_1 \quad \vdash \mathsf{Xj} : V_2}{\vdash \mathsf{Xi} \star \mathsf{Xj} : pV_1 \oplus V_2}\ \mathsf{E3}$$

$$\frac{\vdash \mathsf{Xi} : V_1 \quad \vdash \mathsf{Xj} : V_2}{\vdash \mathsf{Xi} \star \mathsf{Xj} : V_1 \oplus pV_2}\ \mathsf{E4}$$

$$\frac{\vdash \mathsf{e} : V}{\vdash \mathsf{Xj} = \mathsf{e} : 1 \overset{\mathsf{j}}{\leftarrow} V}\ \mathsf{A}$$

$$\forall i, M^*_{ii} = m \quad \frac{\vdash \mathsf{C} : M}{\vdash \textbf{loop}\ \mathsf{Xl}\ \{\mathsf{C}\} : M^* \oplus \{^p_{\mathsf{l}} \to j \mid \exists i, M^*_{ij} = p\}}\ \mathsf{L}$$

$$\frac{\vdash \mathsf{C1} : M_1 \quad \vdash \mathsf{C2} : M_2}{\vdash \mathsf{C1};\ \mathsf{C2} : M_1 \otimes M_2}\ \mathsf{C}$$

$$\forall i, M^*_{ii} = m \text{ and } \forall i, j, M^*_{ij} \neq p \quad \frac{\vdash \mathsf{C} : M}{\vdash \textbf{while}\ \mathsf{b}\ \textbf{do}\ \{\mathsf{C}\} : M^*}\ \mathsf{W}$$

mwp-Analysis Example

```
void main(int X1, int X2, int X3){
    if (X1 < X2) {
        X3 = X1 + X1;
    }
    else {
        X3 = X3 + X2;
    }
    while (X1 < 0){
        X1 = X2 + X3;
    }
}
```

	X1	X2	X3
X1	m	0	0
X2	0	m	0
X3	0	0	m

mwp-Analysis Example

```
void main(int X1, int X2, int X3){
    if (X1 < X2) {
        X3 = X1 + X1;
    }
    else {
        X3 = X3 + X2;
    }
    while (X1 < 0){
        X1 = X2 + X3;
    }
}
```

	X1	X2	X3
X1	m	0	$\boxed{p}$
X2	0	m	0
X3	0	0	m

mwp-Analysis Example

```
void main(int X1, int X2, int X3){
    if (X1 < X2) {
        X3 = X1 + X1;
    }
    else {
        X3 = X3 + X2;
    }
    while (X1 < 0){
        X1 = X2 + X3;
    }
}
```

	X1	X2	X3
X1	m	0	0
X2	0	m	$\boxed{p}$
X3	0	0	$\boxed{m}$

mwp-Analysis Example

```
void main(int X1, int X2, int X3){
    if (X1 < X2) {
        X3 = X1 + X1;
    }
    else {
        X3 = X3 + X2;
    }
    while (X1 < 0){
        X1 = X2 + X3;
    }
}
```

	X1	X2	X3
X1	m	0	p
X2	0	m	p
X3	0	0	m

mwp-Analysis Example

```
void main(int X1, int X2, int X3){
    if (X1 < X2) {
        X3 = X1 + X1;
    }
    else {
        X3 = X3 + X2;
    }
    while (X1 < 0){
        X1 = X2 + X3;
    }
}
```

	X1	X2	X3
X1	m	0	0
X2	w	m	0
X3	w	0	m

mwp-Analysis Example

```
void main(int X1, int X2, int X3){
    if (X1 < X2) {
        X3 = X1 + X1;
    }
    else {
        X3 = X3 + X2;
    }
    while (X1 < 0){
        X1 = X2 + X3;
    }
}
```

$$\begin{array}{c|ccc} & \texttt{X1} & \texttt{X2} & \texttt{X3} \\ \hline \texttt{X1} & m & 0 & 0 \\ \texttt{X2} & w & m & 0 \\ \texttt{X3} & w & 0 & m \end{array}$$

$$= M^*$$

Side condition: $\forall i, M^*_{ii} = m$ and $\forall i, j, M^*_{ij} \neq p$

mwp-Analysis Example

```
void main(int X1, int X2, int X3){
    if (X1 < X2) {
        X3 = X1 + X1;
    }
    else {
        X3 = X3 + X2;
    }
    while (X1 < 0){
        X1 = X2 + X3;
    }
}
```

$$
\begin{array}{c|ccc}
 & \texttt{X1} & \texttt{X2} & \texttt{X3} \\
\hline
\texttt{X1} & p & 0 & p \\
\texttt{X2} & p & m & p \\
\texttt{X3} & w & 0 & m
\end{array}
$$

$= \texttt{C;C}$

mwp-Analysis Example

```
void main(int X1, int X2, int X3){
    if (X1 < X2) {
        X3 = X1 + X1;
    }
    else {
        X3 = X3 + X2;
    }
    while (X1 < 0){
        X1 = X2 + X3;
    }
}
```

	X1	X2	X3
X1	m	0	0
X2	m	m	0
X3	p	0	m

Side condition: $\forall i, M^*_{ii} = m$ and $\forall i, j, M^*_{ij} \neq p$ $\leftarrow$ **!**

Analysis Proofs

- The soundness theorem is the main achievement of the paper.
- The paper contains 8 lemmas and 7 theorems.

Theorem 1: Soundness

$\vdash \texttt{C} : M$ implies $\vDash \texttt{C} : M$.

The paper "proofs are long, technical and occasionally highly nontrivial."[3].

[3] Jones and Kristiansen, "A flow calculus of *mwp*-bounds for complexity analysis", p. 2.

Analysis Proofs

There are multiple proofs regarding the correctness of the inference rules, e.g., the loop rules.

Theorem 2: 7.18

If $\vDash \texttt{C} : M$ and $M^*_{ii} = m$ for all i, then

$$\vDash \texttt{loop X}_\ell\{\texttt{C}\} : M^* \oplus \{^p_\ell \to j \mid \exists i\, [M^*_{ij} = p]\}$$

Theorem 3: 7.19

If $\vDash \texttt{C} : M$ and $M^*_{ii} = m$ for all i, and $M^*_{ij} \neq p$ for all i, j, then $\vDash \texttt{while b}\{\texttt{C}\} : M^*$.

The loop rules require this lemma.

Lemma 1: 7.17

Let $\mathtt{C}^0 \equiv \mathtt{skip}$ and $\mathtt{C}^{t+1} \equiv \mathtt{C}^t;\mathtt{C}$. Assume that $\vDash \mathtt{C} : M$ and that $M^*_{ii} = m$ for all i. Then for any $j \in \{1, \ldots, n\}$ there exists a fixed number k and honest polynomials[4] p, q such that for any t we have

$$[\![C]\!](x_1, ..., x_n \rightsquigarrow x_1^{'}, ..., x_n^{'}) \Rightarrow x_j^{'} \leq \max(\vec{u}, q(\vec{y})) + (t+2)^k p(\vec{z}) \qquad (*_j)$$

where $\vec{u} = \{x_i \mid M^*_{ij} = m\}$ and $\vec{y} = \{x_i \mid M^*_{ij} = w\}$ and $\vec{z} = \{x_i \mid M^*_{ij} = p\}$. Moreover, neither the polynomial p nor the polynomial q depends on k or t; and if the list $\vec{z}$ is empty, then $p(\vec{z}) = 0$.

[4] An *honest polynomial* is a polynomial build up from constants in $\mathbb{N}$ and variables by applying operators $+$ (addition) and $\times$ (multiplication). It is needed to express mwp-bounds.

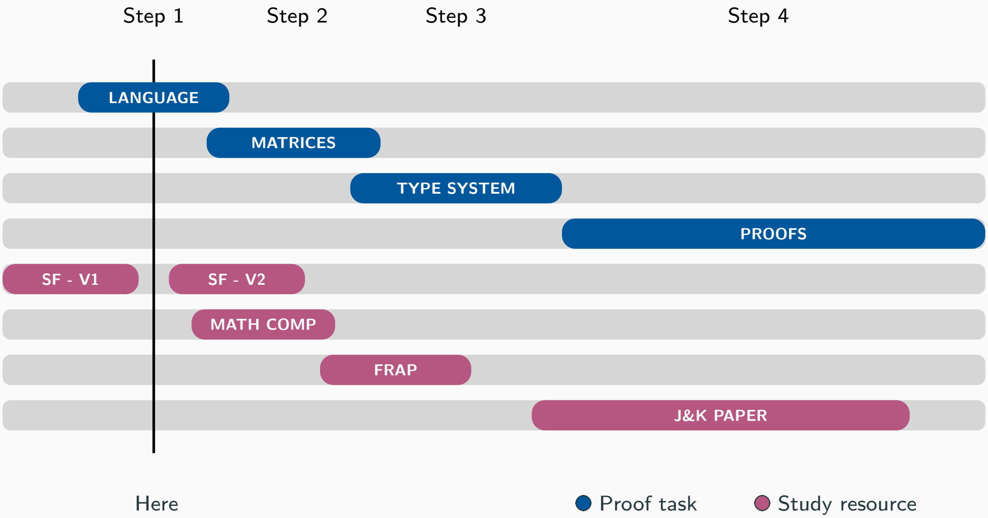
The Goal of This Work

`TODO:` ☐ a **mechanical proof** of the mwp analysis **technique**, as defined in the **original paper**, in Coq.

This will require, defining or proving:

1. programming language under analysis,
2. mathematical machinery (matrices, vectors, analysis terms, . . .),
3. typing system, and
4. the lemmas and proofs from the paper.

Timeline and Progress



Defining the Programming Language

It is `Imp` + added `loop` command.

Variable	$\mathtt{X}_1 \mid \mathtt{X}_2 \mid \mathtt{X}_3 \mid \ldots$
Expression	`X` | `e + e` | `e * e`
Boolean Exp.	`e = e`, `e < e`, etc.
Commands	`skip` | `X := e` | `C;C` | `loop X {C}` | `if b then C else C` | `while b do {C}`

Steps - 2 of 4

Define the mathematical machinery.

- Need e.g., (sparse) matrices, semi-ring.
- Other related mathematical concepts e.g., honest polynomial.

Steps - 3 of 4

Implementing the typing system.

- Define the flow calculus rules[5].
- Define a typing system.

[5] There is some non-determinism in these rules

Prove the paper lemmas and theorems.

- There are 8 lemmas and 7 theorems.
- The soundness theorem, $\vdash \texttt{C} : M$ implies $\vDash \texttt{C} : M$, is essential.
- "These proofs are long, technical and occasionally highly nontrivial."[6]

[6] Jones and Kristiansen, "A flow calculus of *mwp*-bounds for complexity analysis", p. 2.

Expected Main Result

A *certified* complexity analysis technique.

- Proves a positive result obtained by analysis is correct.
- Establishes certified “growth bound” on input variable values.

Conclusion

The plan is to formally verify the analysis technique

Many directions can follow from the correctness proof
e.g., a formally verified static analyzer.

- Our previous work: adjusting analysis makes it practical and fast[7]
- Proof would show the original technique is correct, but not fast.
- It should be possible to combine those two results.

[7] Clément Aubert et al. "mwp-Analysis Improvement and Implementation: Realizing Implicit Computational Complexity". In: *FSCD 2022*. Vol. 228. LIPIcs. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2022, 26:1–26:23. DOI: 10.4230/LIPIcs.FSCD.2022.26.