


ATVA 2023 · 25 October 2023

pymwp: A Static Analyzer Determining Polynomial Growth Bounds

Clément Aubert, Thomas Rubiano, **Neea Rusch** and Thomas Seiller




```
void main(int X1, int X2, int X3)
{
    while(X1 < 10){
        X1 = X2 + X3;
    }
    // X1'  X2'  X3'
}
```

For input variables of an imperative program:

Does there exist a polynomially bounded data-flow relation between variables' **initial** values and **final** values?

That is, $\forall n$, is $X_n \rightsquigarrow X'_n$ polynomially bounded?



```
void main(int X1, int X2, int X3)
{
    while(X1 < 10){
        X1 = X2 + X3;
    }

    // X1'  X2'  X3'

}
```

Yes. Here is a bound:

$$X1' \leq \max(X1, X2+X3)$$

$$\wedge X2' \leq X2$$

$$\wedge X3' \leq X3$$

mwp-flow analysis¹

Calculus for resource analysis of imperative programs.

```
while (X1 < 10)
  X1 = X2 + X3
```

0 – no dependency

m – maximal (of linear)

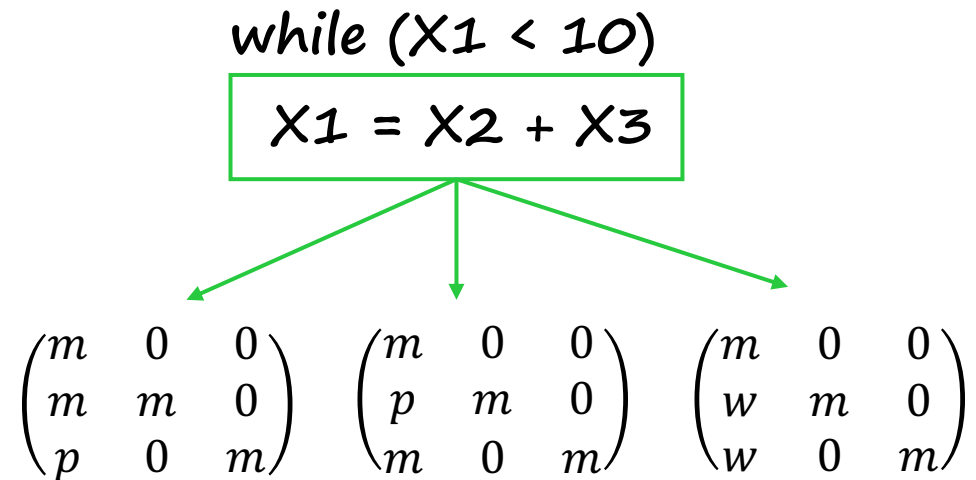
w – weak polynomial

p – polynomial

¹Neil 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](https://doi.org/10.1145/1555746.1555752).

mwp-flow analysis¹

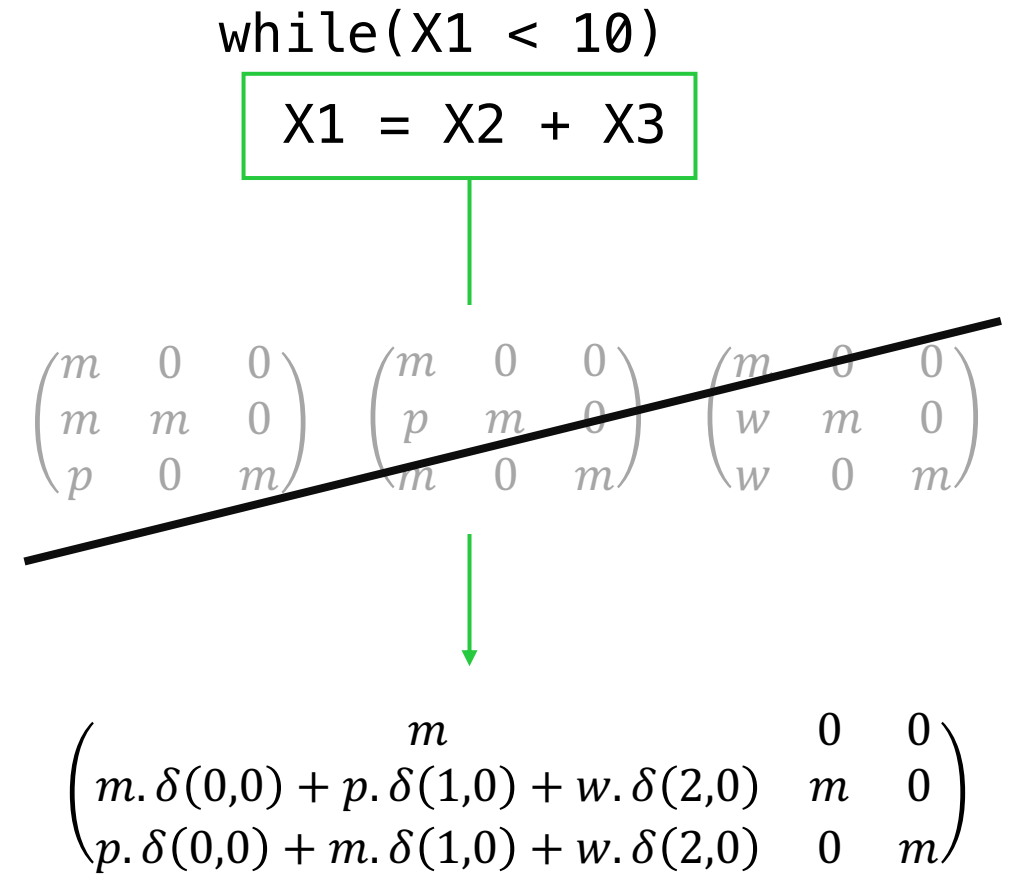
- 3^n derivation paths
- Some may fail



¹Neil 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](https://doi.org/10.1145/1555746.1555752).

Automating mwp

- ✓ Internalize non-determinism



Automating mwp

- ✓ Internalize non-determinism
- ✓ Handle derivation failure

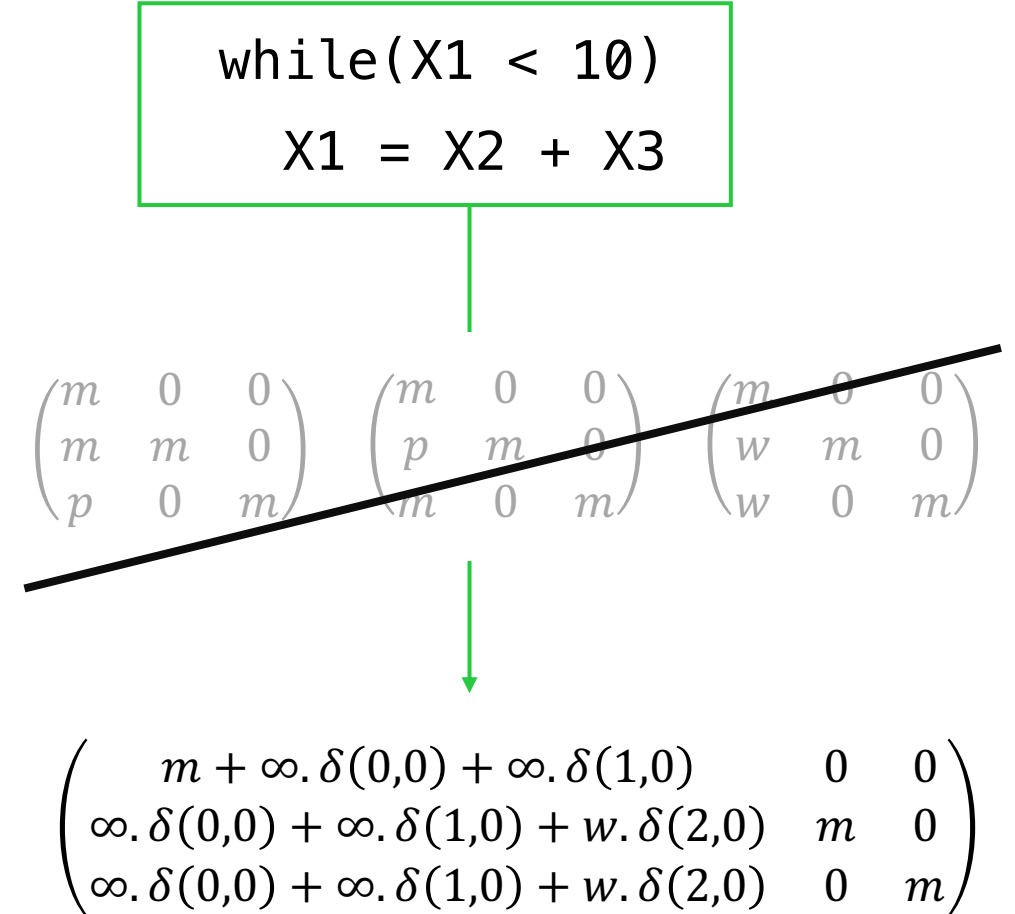
0 – no dependency


m – maximal (of linear)

w – weak polynomial

p - polynomial

∞ - **non-polynomial / failure**





```
void main(int X1, int X2, int X3)
{
    while(X1 < 10){
        X1 = X2 + X3;
    }

    // X1'   X2'   X3'

}
```

We were here

$$\begin{pmatrix} m + \infty \cdot \delta(0,0) + \infty \cdot \delta(1,0) & 0 & 0 \\ \infty \cdot \delta(0,0) + \infty \cdot \delta(1,0) + w \cdot \delta(2,0) & m & 0 \\ \infty \cdot \delta(0,0) + \infty \cdot \delta(1,0) + w \cdot \delta(2,0) & 0 & m \end{pmatrix}$$

But we *actually* want

$$\begin{aligned} X1' &\leq \max(X1, X2+X3) \\ \wedge X2' &\leq X2 \quad \wedge X3' \leq X3 \end{aligned}$$

Obtaining bounds compactly and efficiently

$$\begin{pmatrix} m + \infty.\delta(0,0) + \infty.\delta(1,0) & 0 & 0 \\ \infty.\delta(0,0) + \infty.\delta(1,0) + w.\delta(2,0) & m & 0 \\ \infty.\delta(0,0) + \infty.\delta(1,0) + w.\delta(2,0) & 0 & m \end{pmatrix}$$

Find the derivation choices that lead to ∞ then simplify.

$$[\{\cancel{0}, \cancel{1}, 2\}] \rightarrow [\{2\}]$$

$$\begin{pmatrix} m & 0 & 0 \\ w & m & 0 \\ w & 0 & m \end{pmatrix}$$



```
void main(int X1, int X2)
{
    X1 = X2 + X2;
    while(X1 < 10){
        X1 = X1 * X1;
    }
}
```


When derivation fails (`--fin`)

Problematic flows:

$X1 \rightarrow X1 \parallel X2 \rightarrow X1$

pymwp is an automatic static analyzer for (subset of) C code, to determine if inputs' value growth is polynomially bounded.

run in terminal



```
pip install pymwp  
pymwp file.c [ARGS]
```

run in browser

statycc.github.io/pymwp/demo



source code + docs: **statycc/pymwp**