

Making Numerical Program Analysis Fast

Gagandeep Singh Markus Püschel Martin Vechev

Department of Computer Science
ETH Zürich



Static Program Analysis

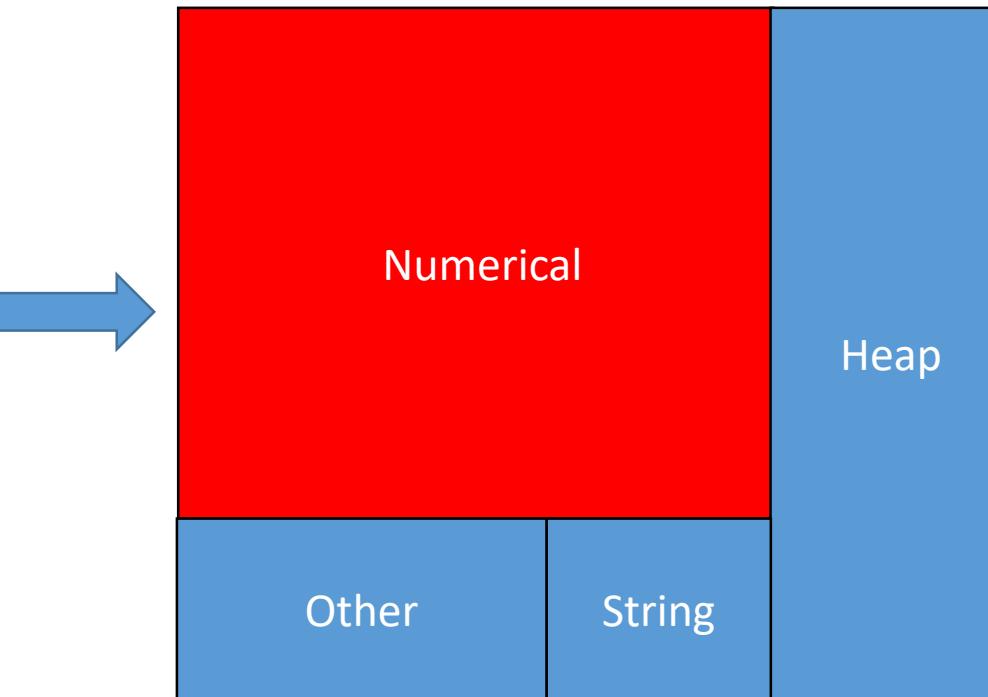
Static Program Analysis

```
public static void verify() {
    int[] ptr = new int[8];
    int start = 0;
    for(int i0 = 0; i0 < 8; ++i0) {
        int x1 = i0 | start;
        for(int x2 = 0; x2<100000;++x2) {
            int y3 = 2*x1;
            int index4 = 0;
            if (y3 == 0) { index4 = 1; }
            if (y3 == 49) { index4 = 8; }
            if (y3 == 36) { index4 = 8; }
            if (y3 == -1) { index4 = 0; }
            if (y3 == 50) { index4 = 9; }
            ptr[index4] = 1;
        }
    }
}
```

Static Program Analysis

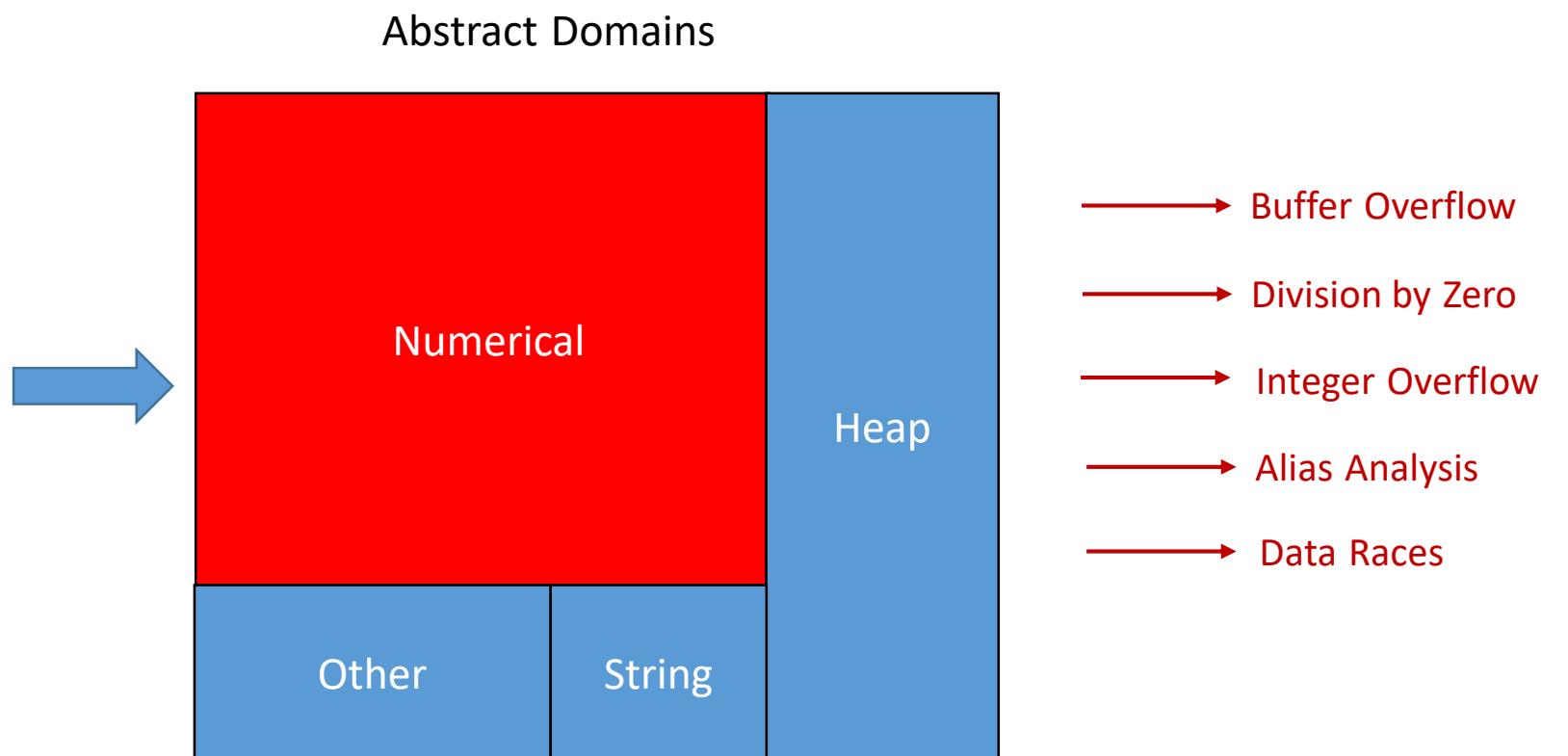
```
public static void verify() {  
    int[] ptr = new int[8];  
    int start = 0;  
    for(int i0 = 0; i0 < 8; ++i0) {  
        int x1 = i0 | start;  
        for(int x2 = 0; x2<100000;++x2) {  
            int y3 = 2*x1;  
            int index4 = 0;  
            if (y3 == 0) { index4 = 1; }  
            if (y3 == 49) { index4 = 8; }  
            if (y3 == 36) { index4 = 8; }  
            if (y3 == -1) { index4 = 0; }  
            if (y3 == 50) { index4 = 9; }  
            ptr[index4] = 1;  
        }  
    }  
}
```

Abstract Domains



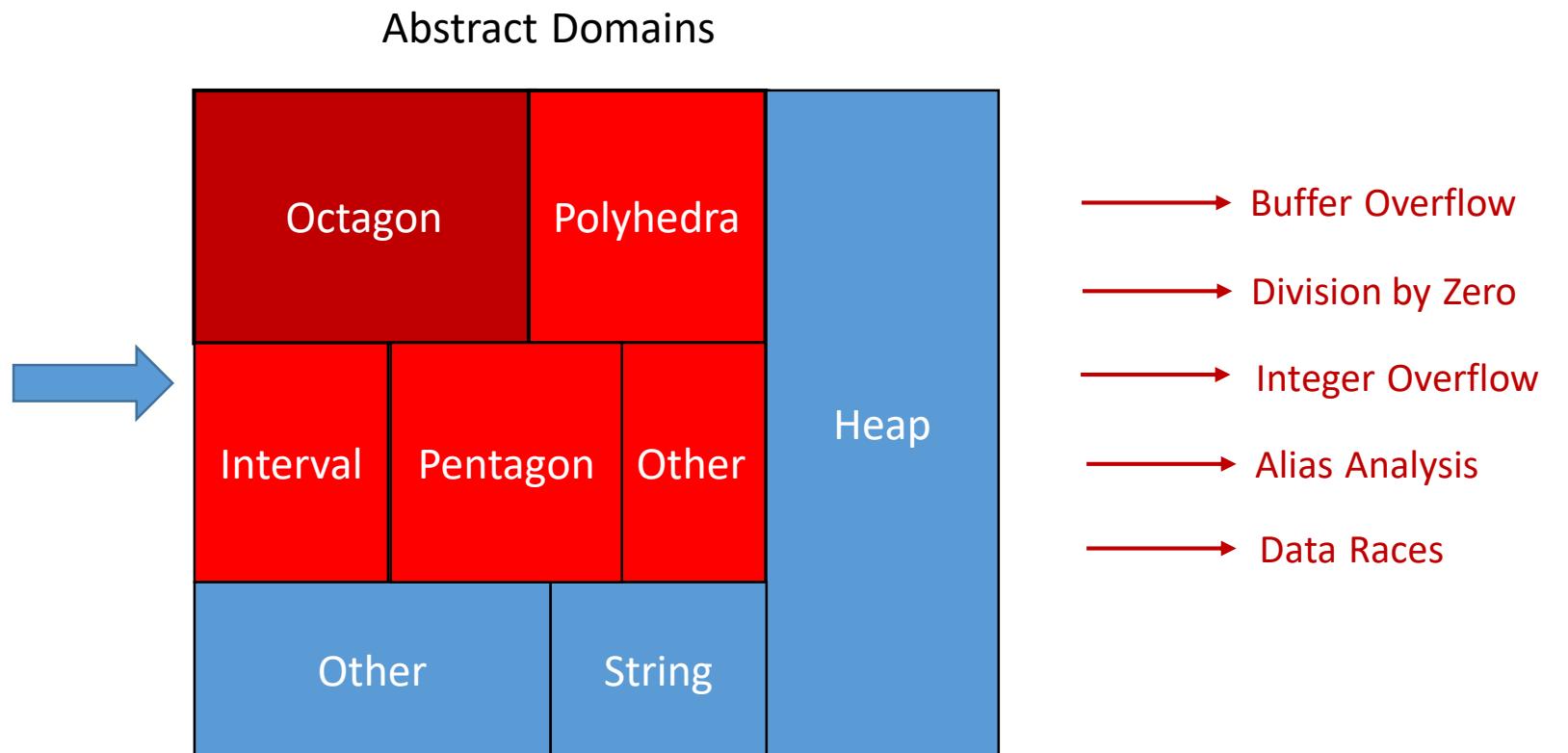
Static Program Analysis

```
public static void verify() {  
    int[] ptr = new int[8];  
    int start = 0;  
    for(int i0 = 0; i0 < 8; ++i0) {  
        int x1 = i0 | start;  
        for(int x2 = 0; x2<100000;++x2) {  
            int y3 = 2*x1;  
            int index4 = 0;  
            if (y3 == 0) { index4 = 1; }  
            if (y3 == 49) { index4 = 8; }  
            if (y3 == 36) { index4 = 8; }  
            if (y3 == -1) { index4 = 0; }  
            if (y3 == 50) { index4 = 9; }  
            ptr[index4] = 1;  
        }  
    }  
}
```



Static Program Analysis

```
public static void verify() {  
    int[] ptr = new int[8];  
    int start = 0;  
    for(int i0 = 0; i0 < 8; ++i0) {  
        int x1 = i0 | start;  
        for(int x2 = 0; x2<100000;++x2) {  
            int y3 = 2*x1;  
            int index4 = 0;  
            if (y3 == 0) { index4 = 1; }  
            if (y3 == 49) { index4 = 8; }  
            if (y3 == 36) { index4 = 8; }  
            if (y3 == -1) { index4 = 0; }  
            if (y3 == 50) { index4 = 9; }  
            ptr[index4] = 1;  
        }  
    }  
}
```



Octagon Abstract Domain

(Miné, HOSC, 2006)

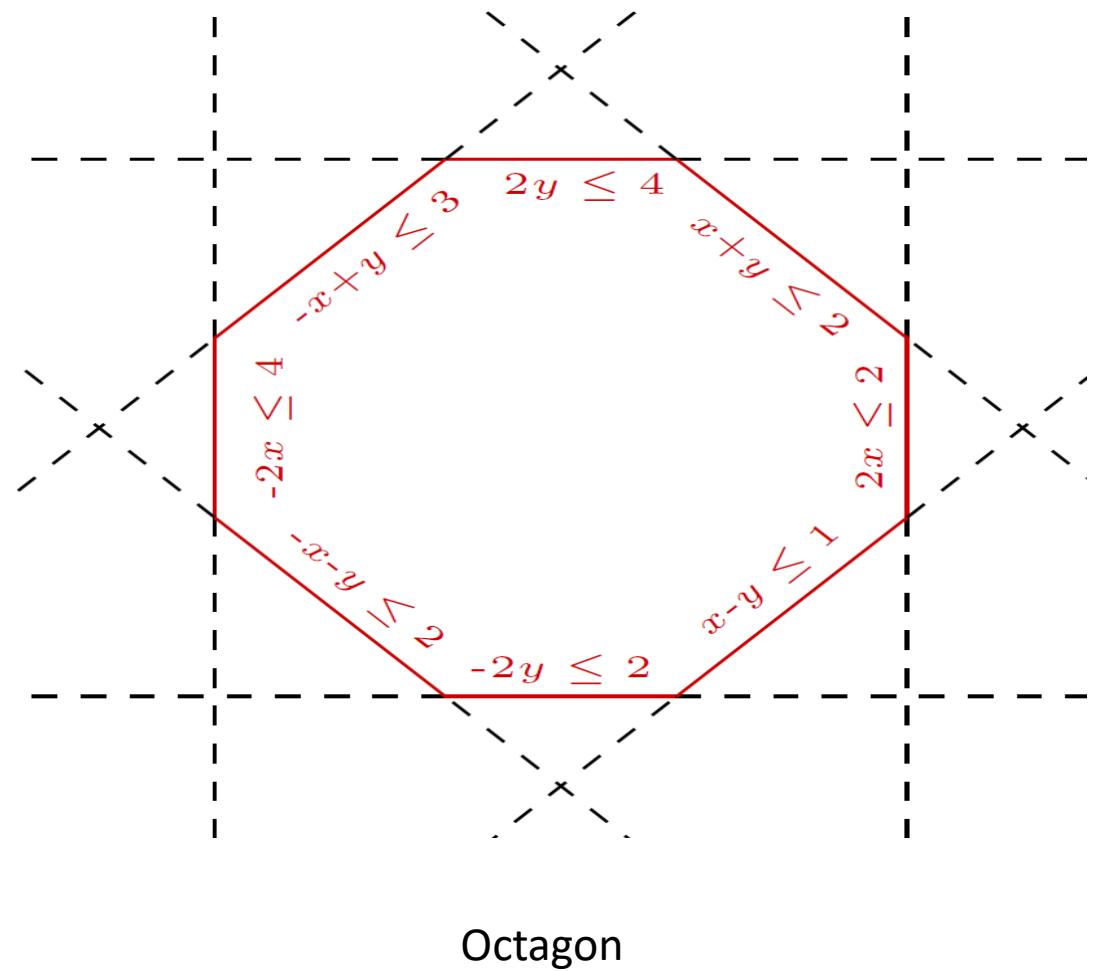
- Octagonal Inequalities:

- Binary: $\pm x \pm y \leq c, x \neq y$
- Unary: $\pm 2x \leq 2d$
- $c, d \in \mathbb{R} \cup \{\infty\}$

Octagon Abstract Domain

(Miné, HOSC, 2006)

- Octagonal Inequalities:
 - Binary: $\pm x \pm y \leq c$, $x \neq y$
 - Unary: $\pm 2x \leq d$
 - $c, d \in \mathbb{R} \cup \{\infty\}$



Octagon Abstract Domain

(Miné, HOSC, 2006)

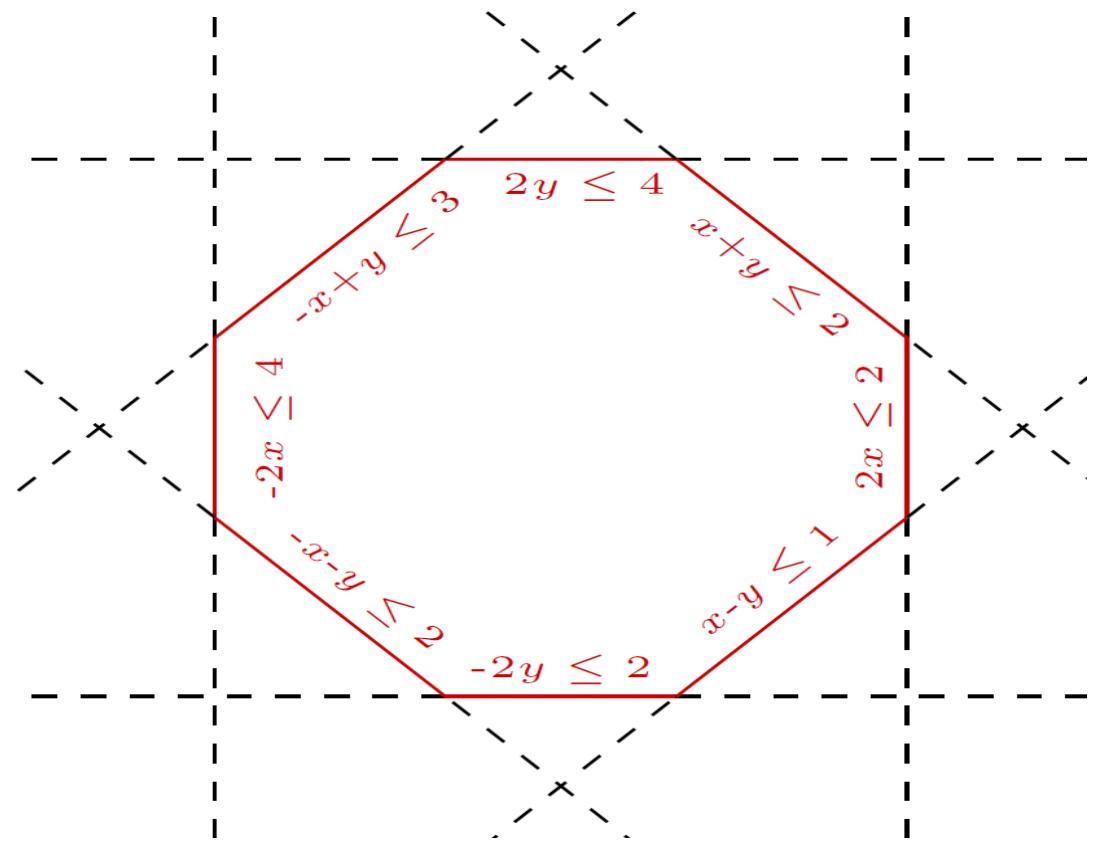
- Octagonal Inequalities:

- Binary: $\pm x \pm y \leq c$, $x \neq y$
- Unary: $\pm 2x \leq d$
- $c, d \in \mathbb{R} \cup \{\infty\}$

	x^+	x^-	y^+	y^-
x^+	0	4	3	2
x^-	2	0	2	1
y^+	1	2	0	2
y^-	2	3	4	0



Difference Bound Matrix (DBM)



Octagon

Octagon Analysis is Expensive

Example: Static analyzer for TouchDevelop
(Brutschy et al. OOPSLA, 2014)

Using APRON



Single Core

Octagon Analysis is Expensive

Example: Static analyzer for TouchDevelop
(Brutschy et al. OOPSLA, 2014)

Using APRON



Single Core

Using ELINA

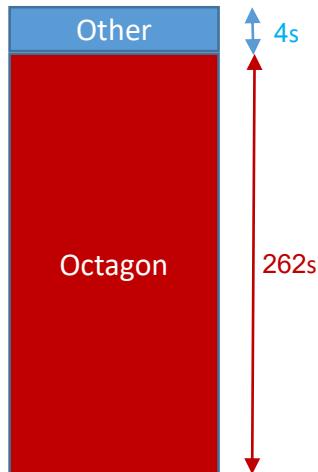


Single Core

Octagon Analysis is Expensive

Example: Static analyzer for TouchDevelop
(Brutschy et al. OOPSLA, 2014)

Using APRON



Single Core

Our Contribution: drop-in replacement for APRON

Using ELINA

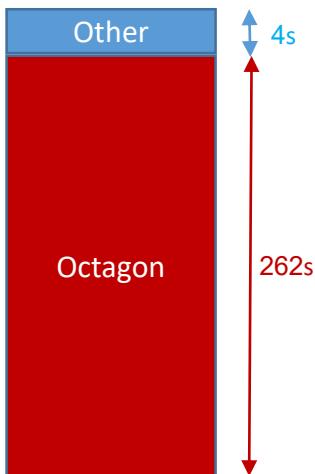


Single Core

Octagon Analysis is Expensive

Example: Static analyzer for TouchDevelop
(Brutschy et al. OOPSLA, 2014)

Using APRON



Single Core

Our Contribution: drop-in replacement for APRON

Using ELINA

- Octagon Speedup: 26x
- Overall Speedup: 19x
- No loss in precision



Single Core

Octagon Analysis

→**x = 1;**

y = x;

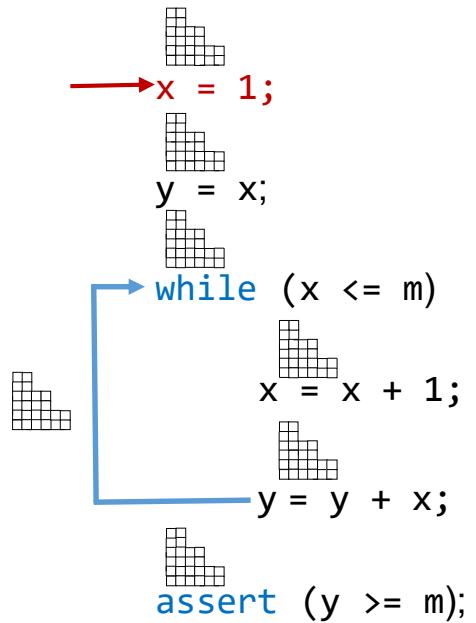
→**while (x <= m)**

x = x + 1;

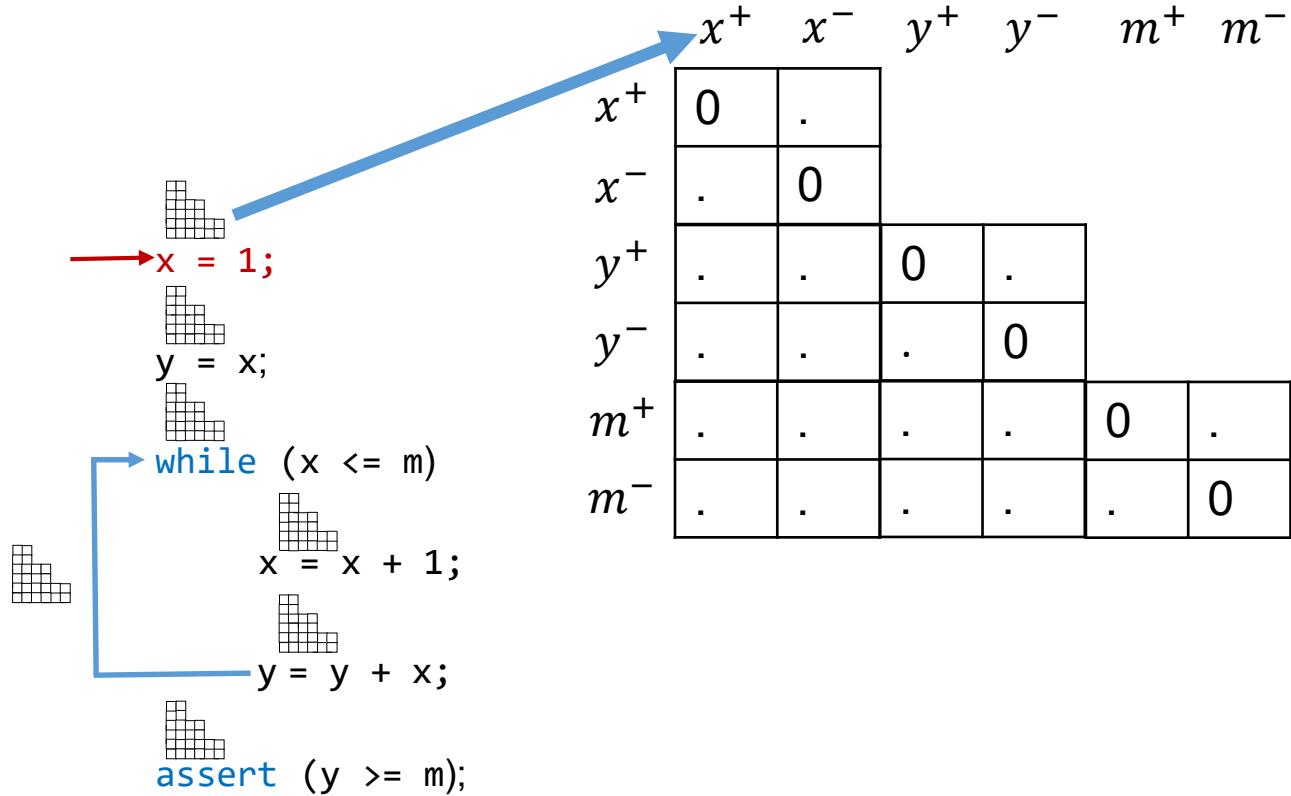
y = y + x;

assert (y >= m);

Octagon Analysis

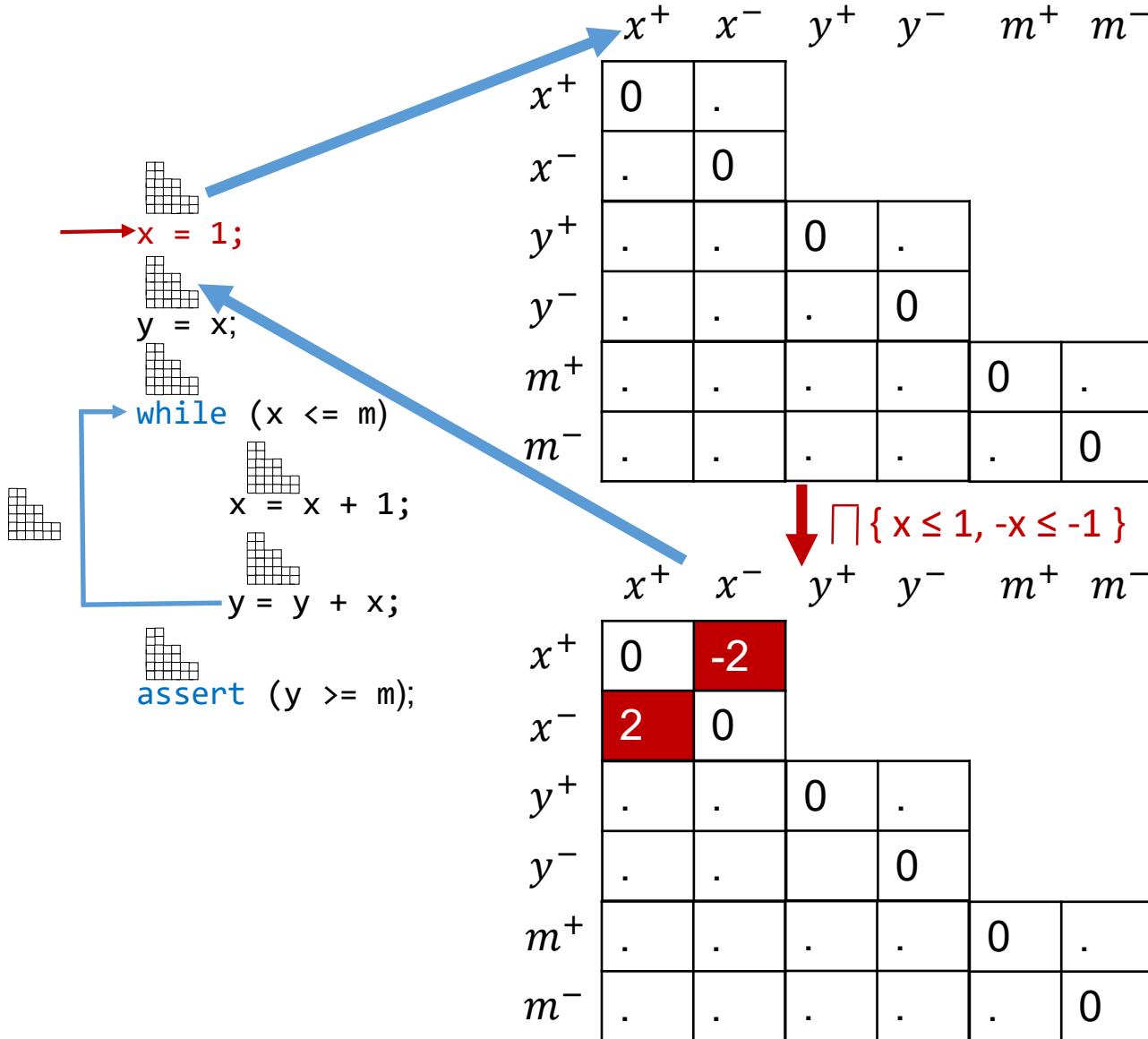


Octagon Analysis



{}

Octagon Analysis

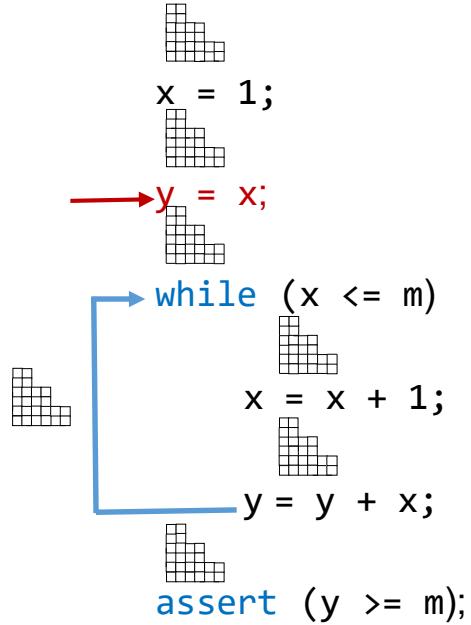


{}

{ $2x \leq 2, -2x \leq -2$ }

Octagon Analysis

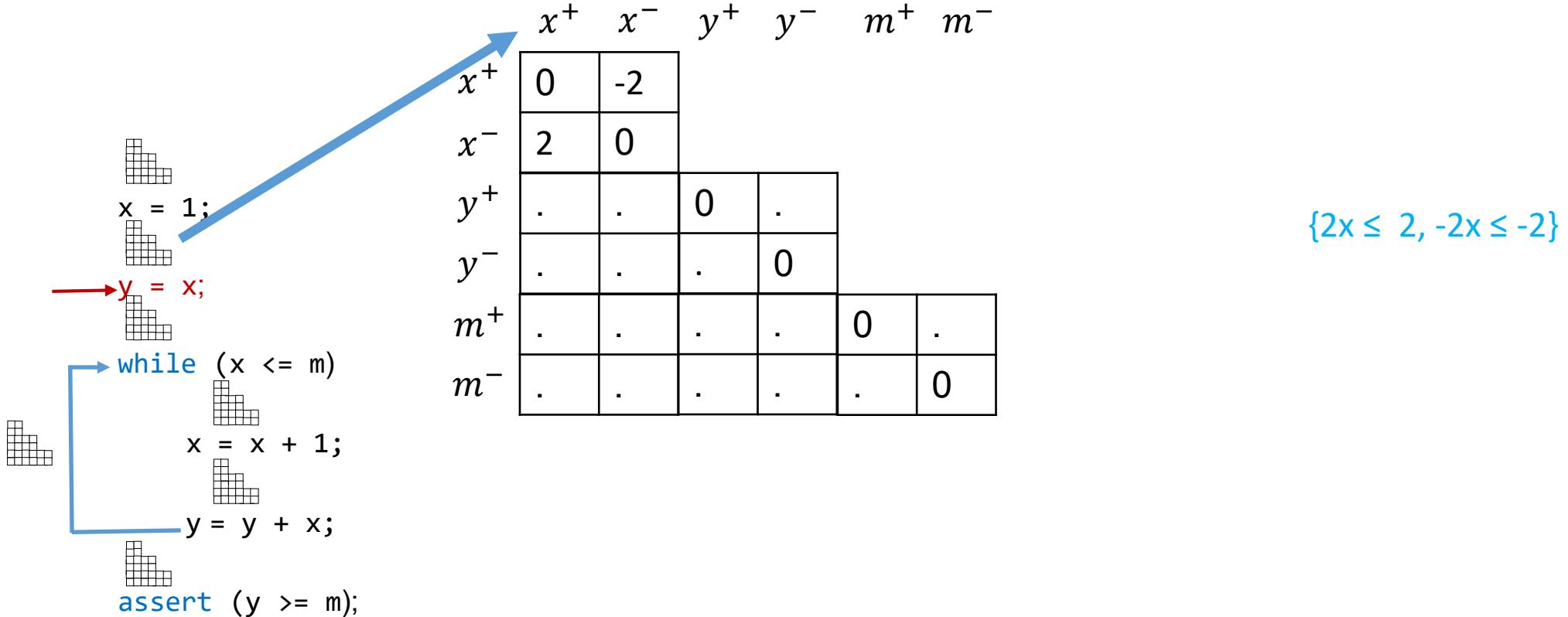
```
x = 1;  
y = x;  
while (x <= m)  
    x = x + 1;  
    y = y + x;  
assert (y >= m);
```



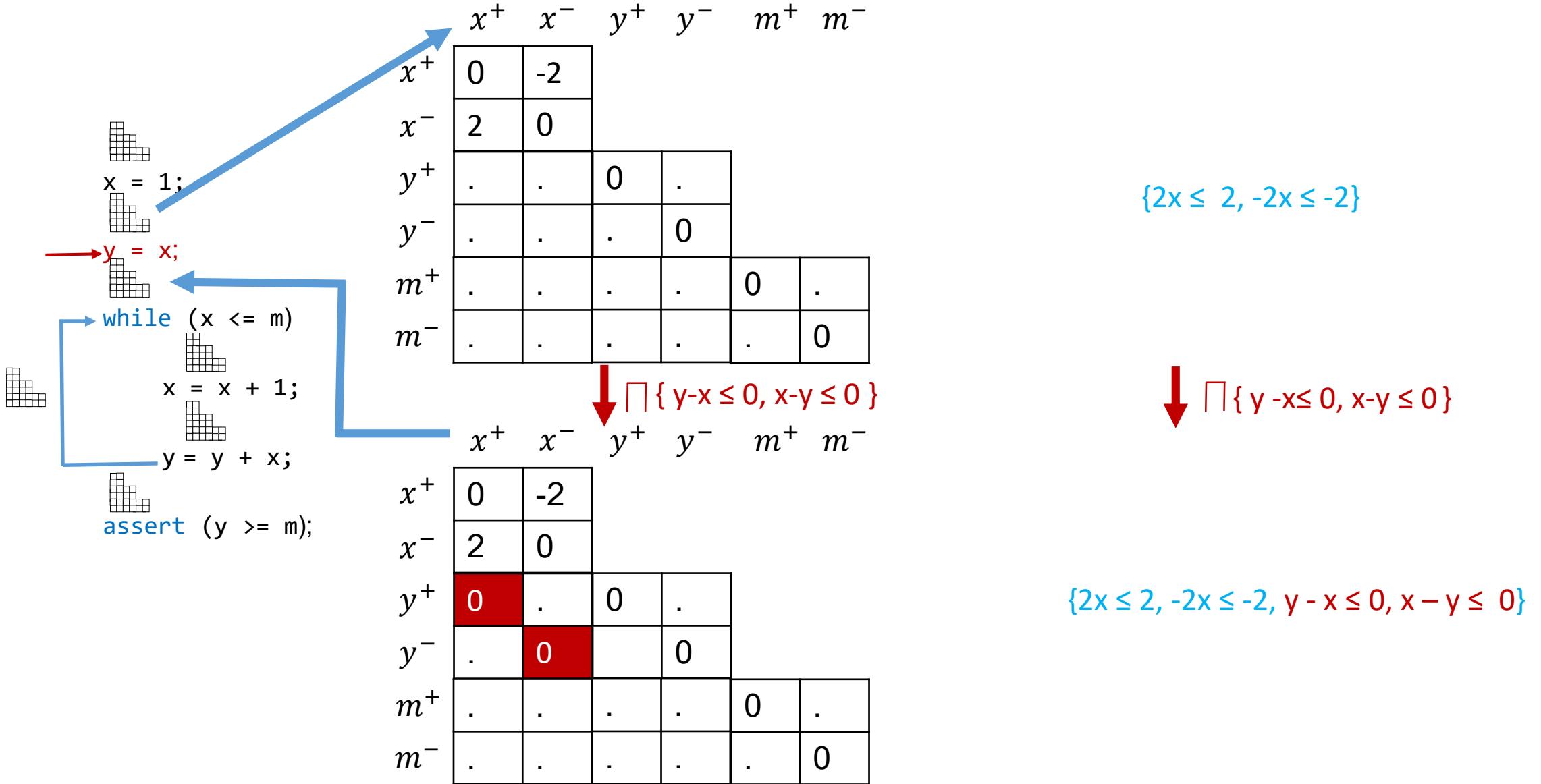
The diagram illustrates the analysis of two variables, x and y, using octagonal regions. The variable x is represented by a triangular region starting at (0,0) and ending at (m, m). The variable y is represented by a triangular region starting at (0,0) and ending at (m, m). A red arrow points to the assignment statement `y = x;`. A blue bracket encloses the `while` loop, indicating its scope. The code itself is as follows:

```
x = 1;  
y = x;  
while (x <= m)  
    x = x + 1;  
    y = y + x;  
assert (y >= m);
```

Octagon Analysis



Octagon Analysis



Octagon Analysis

```
x = 1;  
y = x;  
while (x <= m)  
    x = x + 1;  
    y = y + x;  
assert (y >= m);
```

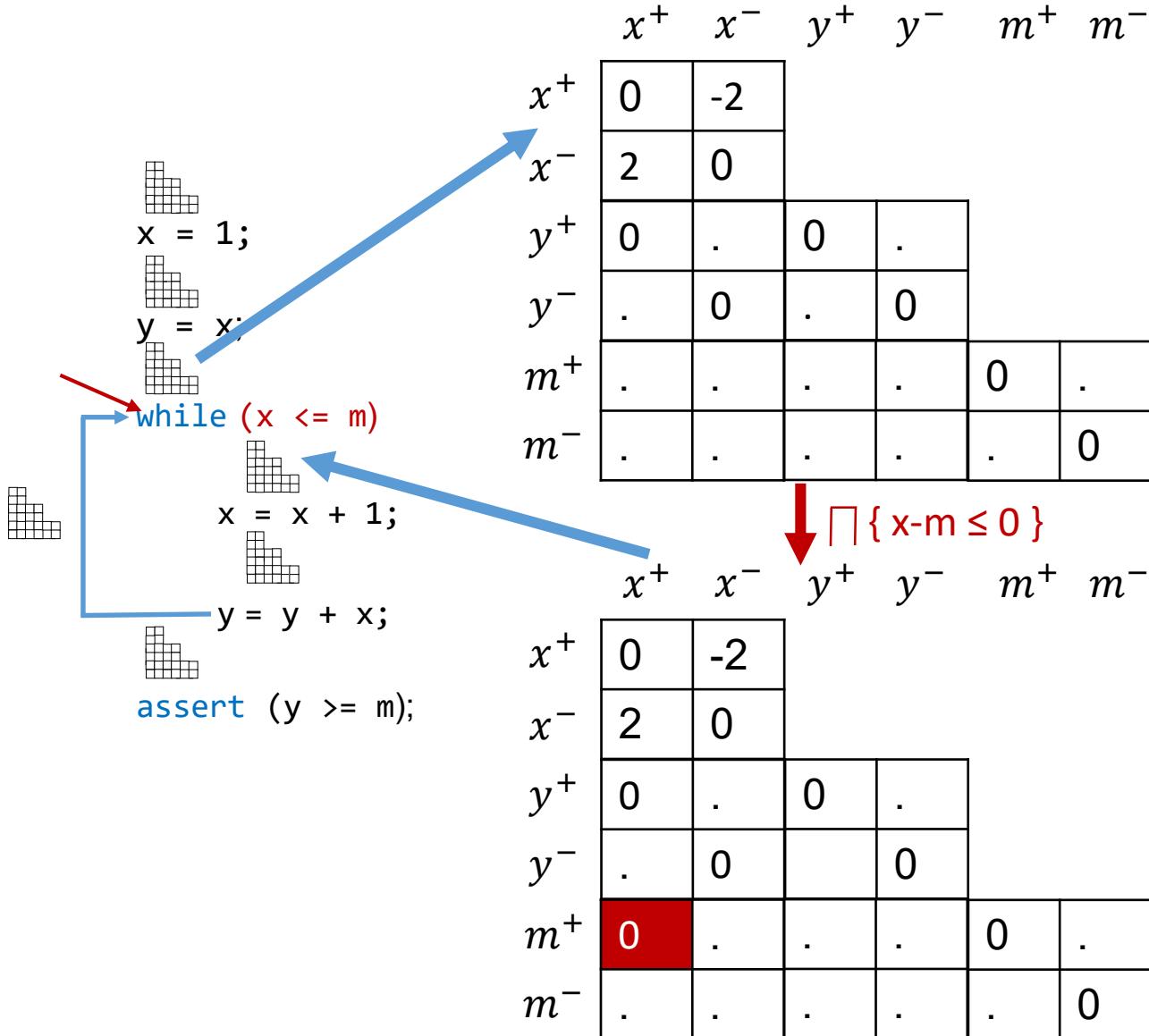
Octagon Analysis

```
x = 1;  
y = x;  
while (x <= m)  
    x = x + 1;  
    y = y + x;  
assert (y >= m);
```

	x^+	x^-	y^+	y^-	m^+	m^-
x^+	0	-2				
x^-	2	0				
y^+	0	.	0	.		
y^-	.	0	.	0		
m^+	0	.
m^-	0

$$\{2x \leq 2, -2x \leq -2, y - x \leq 0, x - y \leq 0\}$$

Octagon Analysis



$$\{2x \leq 2, -2x \leq -2, y - x \leq 0, x - y \leq 0\}$$

$$\downarrow \sqcap \{x - m \leq 0\}$$

$$\{2x \leq 2, -2x \leq -2, y - x \leq 0, x - y \leq 0, x - m \leq 0\}$$

Closure (*) increases precision of Join (\sqcup operator)

```
x = 1;
y = x;
while (x <= m)
    x = x + 1;
    y = y + x;
assert (y >= m);
```

Closure (*) increases precision of Join (\sqcup operator)

Diagram illustrating the closure of a while loop using the join (\sqcup) operator.

The code snippet shows a while loop that increments x and updates y until $y \geq m$:

```
x = 1;  
y = x;  
while (x <= m)  
    x = x + 1;  
    y = y + x;  
assert (y >= m);
```

A blue arrow points from the x^+ column of the state matrix to the `while` loop, indicating the flow of information.

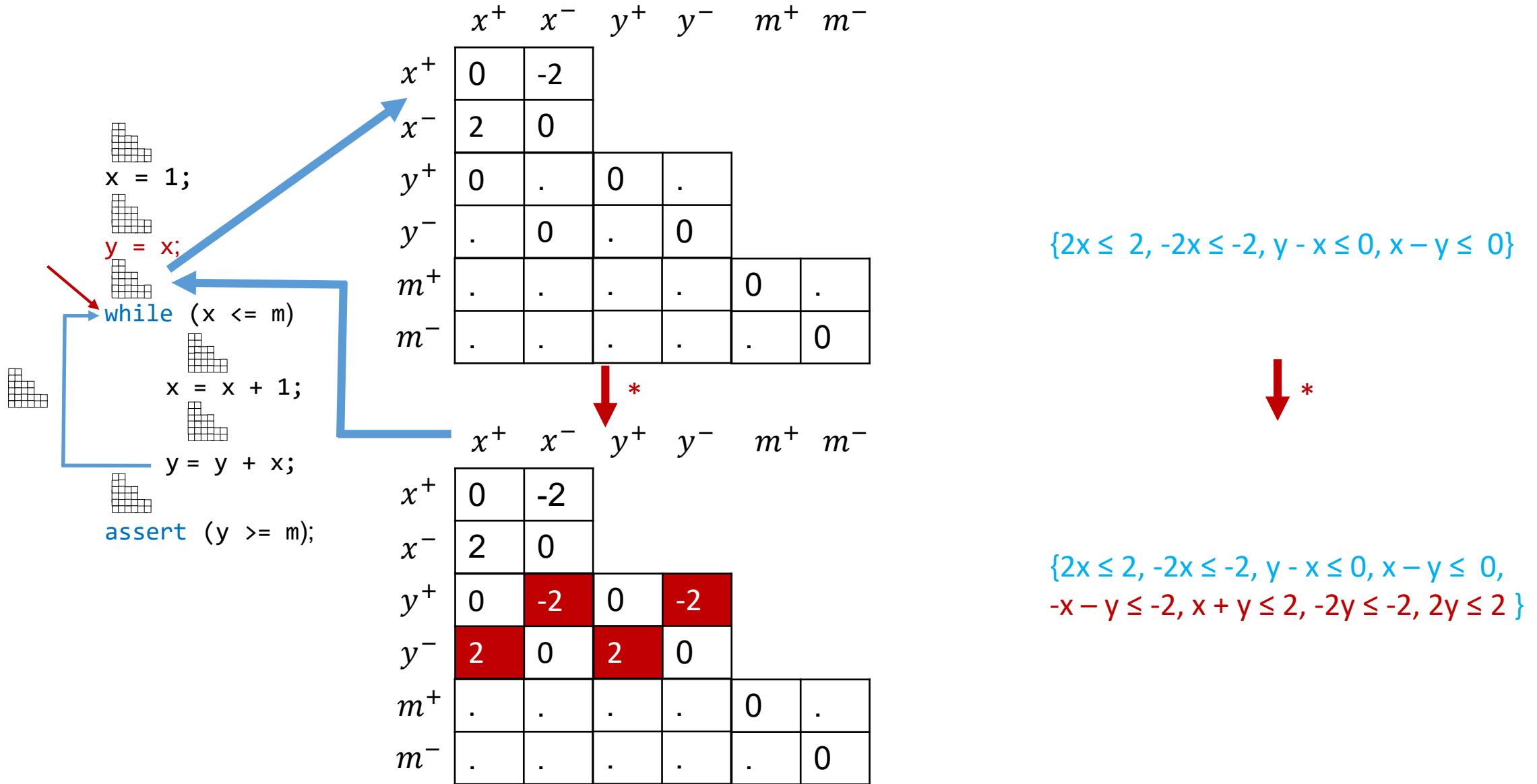
The state matrix has columns labeled $x^+, x^-, y^+, y^-, m^+, m^-$ and rows labeled $x^+, x^-, y^+, y^-, m^+, m^-$.

	x^+	x^-	y^+	y^-	m^+	m^-
x^+	0	-2				
x^-	2	0				
y^+	0	.	0	.		
y^-	.	0	.	0		
m^+	0	.
m^-	0

Annotations in cyan text show the constraints derived from the code:

$$\{2x \leq 2, -2x \leq -2, y - x \leq 0, x - y \leq 0\}$$

Closure (*) increases precision of Join (\sqcup operator)

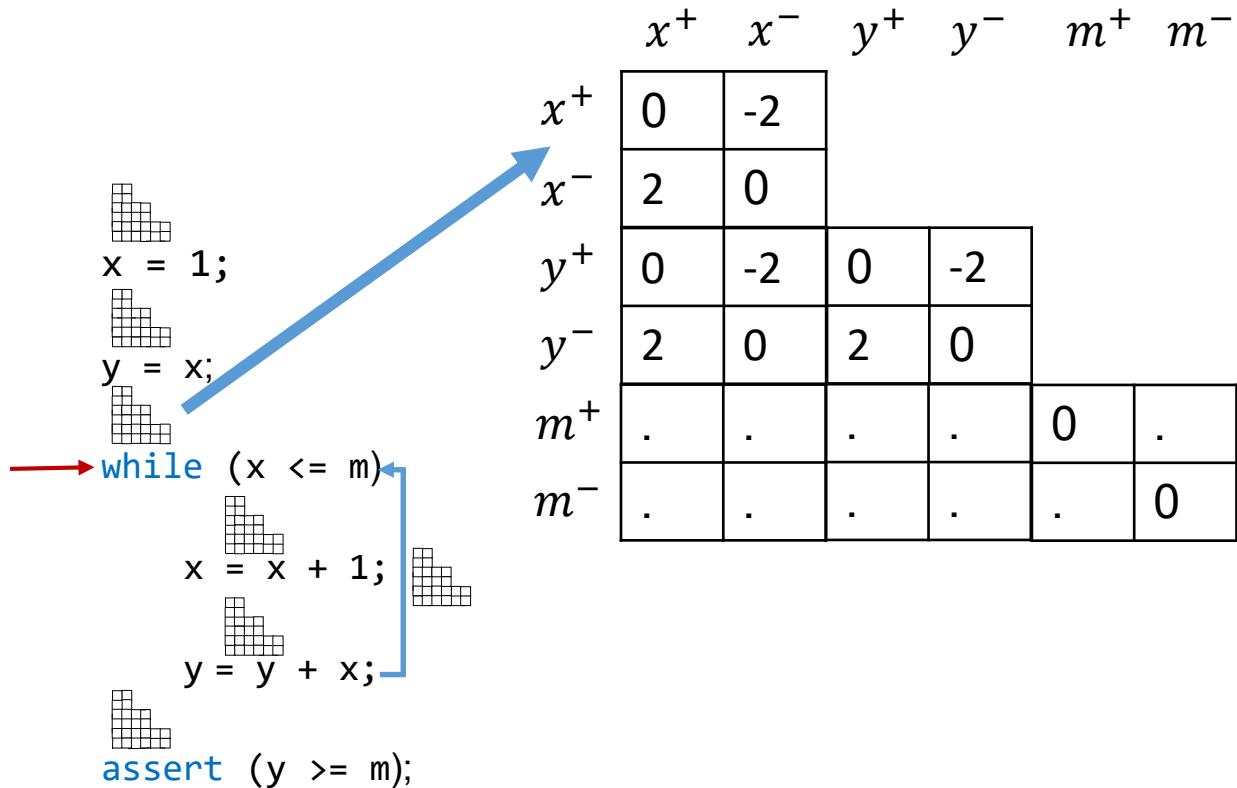


Join (\sqcup) of two closed Octagons

```
x = 1;  
y = x;  
while (x <= m)  
    x = x + 1;  
    y = y + x;  
assert (y >= m);
```

<img alt="Diagram showing the generation of two closed octagonal patterns. The first pattern is a 4x4 square of dots. The second pattern is a 5x5 square of dots. The third pattern is a 6x6 square of dots. The fourth pattern is a 7x7 square of dots. The fifth pattern is a 8x8 square of dots. The sixth pattern is a 9x9 square of dots. The seventh pattern is a 10x10 square of dots. The eighth pattern is a 11x11 square of dots. The ninth pattern is a 12x12 square of dots. The tenth pattern is a 13x13 square of dots. The eleventh pattern is a 14x14 square of dots. The twelfth pattern is a 15x15 square of dots. The thirteenth pattern is a 16x16 square of dots. The fourteenth pattern is a 17x17 square of dots. The fifteenth pattern is a 18x18 square of dots. The sixteenth pattern is a 19x19 square of dots. The seventeenth pattern is a 20x20 square of dots. The eighteenth pattern is a 21x21 square of dots. The nineteenth pattern is a 22x22 square of dots. The twentieth pattern is a 23x23 square of dots. The twenty-first pattern is a 24x24 square of dots. The twenty-second pattern is a 25x25 square of dots. The twenty-third pattern is a 26x26 square of dots. The twenty-fourth pattern is a 27x27 square of dots. The twenty-fifth pattern is a 28x28 square of dots. The twenty-sixth pattern is a 29x29 square of dots. The twenty-seventh pattern is a 30x30 square of dots. The twenty-eighth pattern is a 31x31 square of dots. The twenty-ninth pattern is a 32x32 square of dots. The thirtieth pattern is a 33x33 square of dots. The thirty-first pattern is a 34x34 square of dots. The thirty-second pattern is a 35x35 square of dots. The thirty-third pattern is a 36x36 square of dots. The thirty-fourth pattern is a 37x37 square of dots. The thirty-fifth pattern is a 38x38 square of dots. The thirty-sixth pattern is a 39x39 square of dots. The thirty-seventh pattern is a 40x40 square of dots. The thirty-eighth pattern is a 41x41 square of dots. The thirty-ninth pattern is a 42x42 square of dots. The forty-pattern is a 43x43 square of dots. The forty-one-pattern is a 44x44 square of dots. The forty-two-pattern is a 45x45 square of dots. The forty-three-pattern is a 46x46 square of dots. The forty-four-pattern is a 47x47 square of dots. The forty-five-pattern is a 48x48 square of dots. The forty-six-pattern is a 49x49 square of dots. The forty-seven-pattern is a 50x50 square of dots. The forty-eight-pattern is a 51x51 square of dots. The forty-nine-pattern is a 52x52 square of dots. The五十-pattern is a 53x53 square of dots. The五十-one-pattern is a 54x54 square of dots. The五十-two-pattern is a 55x55 square of dots. The五十-three-pattern is a 56x56 square of dots. The五十-four-pattern is a 57x57 square of dots. The五十-five-pattern is a 58x58 square of dots. The五十-six-pattern is a 59x59 square of dots. The五十-seven-pattern is a 60x60 square of dots. The五十-eight-pattern is a 61x61 square of dots. The五十-nine-pattern is a 62x62 square of dots. The六十-pattern is a 63x63 square of dots. The六十-one-pattern is a 64x64 square of dots. The六十-two-pattern is a 65x65 square of dots. The六十-three-pattern is a 66x66 square of dots. The六十-four-pattern is a 67x67 square of dots. The六十-five-pattern is a 68x68 square of dots. The六十六-pattern is a 69x69 square of dots. The六十七-pattern is a 70x70 square of dots. The六十八-pattern is a 71x71 square of dots. The六十九-pattern is a 72x72 square of dots. The七十-pattern is a 73x73 square of dots. The七十-one-pattern is a 74x74 square of dots. The七十-two-pattern is a 75x75 square of dots. The七十三-pattern is a 76x76 square of dots. The七十四-pattern is a 77x77 square of dots. The七十五-pattern is a 78x78 square of dots. The七十六-pattern is a 79x79 square of dots. The七十七-pattern is a 80x80 square of dots. The七十八-pattern is a 81x81 square of dots. The七十九-pattern is a 82x82 square of dots. The八十-pattern is a 83x83 square of dots. The八十-one-pattern is a 84x84 square of dots. The八十二-pattern is a 85x85 square of dots. The八十三-pattern is a 86x86 square of dots. The八十四-pattern is a 87x87 square of dots. The八十五-pattern is a 88x88 square of dots. The八十六-pattern is a 89x89 square of dots. The八十七-pattern is a 90x90 square of dots. The八十八-pattern is a 91x91 square of dots. The八十九-pattern is a 92x92 square of dots. The九十-pattern is a 93x93 square of dots. The九十-one-pattern is a 94x94 square of dots. The九十二-pattern is a 95x95 square of dots. The九十三-pattern is a 96x96 square of dots. The九十四-pattern is a 97x97 square of dots. The九十五-pattern is a 98x98 square of dots. The九十六-pattern is a 99x99 square of dots. The九十七-pattern is a 100x100 square of dots. The九十八-pattern is a 101x101 square of dots. The九十九-pattern is a 102x102 square of dots. The一百-pattern is a 103x103 square of dots. The一百一-pattern is a 104x104 square of dots. The一百二-pattern is a 105x105 square of dots. The一百三-pattern is a 106x106 square of dots. The一百四-pattern is a 107x107 square of dots. The一百五-pattern is a 108x108 square of dots. The一百六-pattern is a 109x109 square of dots. The一百七-pattern is a 110x110 square of dots. The一百八-pattern is a 111x111 square of dots. The一百九-pattern is a 112x112 square of dots. The一百十-pattern is a 113x113 square of dots. The一百十一-pattern is a 114x114 square of dots. The一百十二-pattern is a 115x115 square of dots. The一百十三-pattern is a 116x116 square of dots. The一百十四-pattern is a 117x117 square of dots. The一百十五-pattern is a 118x118 square of dots. The一百十六-pattern is a 119x119 square of dots. The一百十七-pattern is a 120x120 square of dots. The一百十八-pattern is a 121x121 square of dots. The一百十九-pattern is a 122x122 square of dots. The一百二十-pattern is a 123x123 square of dots. The一百二十一-pattern is a 124x124 square of dots. The一百二十二-pattern is a 125x125 square of dots. The一百二十三-pattern is a 126x126 square of dots. The一百二十四-pattern is a 127x127 square of dots. The一百二十五-pattern is a 128x128 square of dots. The一百二十六-pattern is a 129x129 square of dots. The一百二十七-pattern is a 130x130 square of dots. The一百二十八-pattern is a 131x131 square of dots. The一百二十九-pattern is a 132x132 square of dots. The一百三十-pattern is a 133x133 square of dots. The一百三十一-pattern is a 134x134 square of dots. The一百三十二-pattern is a 135x135 square of dots. The一百三十三-pattern is a 136x136 square of dots. The一百三十四-pattern is a 137x137 square of dots. The一百三十五-pattern is a 138x138 square of dots. The一百三十六-pattern is a 139x139 square of dots. The一百三十七-pattern is a 140x140 square of dots. The一百三十八-pattern is a 141x141 square of dots. The一百三十九-pattern is a 142x142 square of dots. The一百四十-pattern is a 143x143 square of dots. The一百四十一-pattern is a 144x144 square of dots. The一百四十二-pattern is a 145x145 square of dots. The一百四十三-pattern is a 146x146 square of dots. The一百四十四-pattern is a 147x147 square of dots. The一百四十五-pattern is a 148x148 square of dots. The一百四十六-pattern is a 149x149 square of dots. The一百四十七-pattern is a 150x150 square of dots. The一百四十八-pattern is a 151x151 square of dots. The一百四十九-pattern is a 152x152 square of dots. The一百五十-pattern is a 153x153 square of dots. The一百五十一-pattern is a 154x154 square of dots. The一百五十二-pattern is a 155x155 square of dots. The一百五十三-pattern is a 156x156 square of dots. The一百五十四-pattern is a 157x157 square of dots. The一百五十五-pattern is a 158x158 square of dots. The一百五十六-pattern is a 159x159 square of dots. The一百五十七-pattern is a 160x160 square of dots. The一百五十八-pattern is a 161x161 square of dots. The一百五十九-pattern is a 162x162 square of dots. The一百六十-pattern is a 163x163 square of dots. The一百六十一-pattern is a 164x164 square of dots. The一百六十二-pattern is a 165x165 square of dots. The一百六十三-pattern is a 166x166 square of dots. The一百六十四-pattern is a 167x167 square of dots. The一百六十五-pattern is a 168x168 square of dots. The一百六十六-pattern is a 169x169 square of dots. The一百六十七-pattern is a 170x170 square of dots. The一百六十八-pattern is a 171x171 square of dots. The一百六十九-pattern is a 172x172 square of dots. The一百七十-pattern is a 173x173 square of dots. The一百七十一-pattern is a 174x174 square of dots. The一百七十二-pattern is a 175x175 square of dots. The一百七十三-pattern is a 176x176 square of dots. The一百七十四-pattern is a 177x177 square of dots. The一百七十五-pattern is a 178x178 square of dots. The一百七十六-pattern is a 179x179 square of dots. The一百七十七-pattern is a 180x180 square of dots. The一百七十八-pattern is a 181x181 square of dots. The一百七十九-pattern is a 182x182 square of dots. The一百八十-pattern is a 183x183 square of dots. The一百八十一-pattern is a 184x184 square of dots. The一百八十二-pattern is a 185x185 square of dots. The一百八十三-pattern is a 186x186 square of dots. The一百八十四-pattern is a 187x187 square of dots. The一百八十五-pattern is a 188x188 square of dots. The一百八十六-pattern is a 189x189 square of dots. The一百八十七-pattern is a 190x190 square of dots. The一百八十八-pattern is a 191x191 square of dots. The一百八十九-pattern is a 192x192 square of dots. The一百九十-pattern is a 193x193 square of dots. The一百九十一-pattern is a 194x194 square of dots. The一百九十二-pattern is a 195x195 square of dots. The一百九十三-pattern is a 196x196 square of dots. The一百九十四-pattern is a 197x197 square of dots. The一百九十五-pattern is a 198x198 square of dots. The一百九十六-pattern is a 199x199 square of dots. The一百九十七-pattern is a 200x200 square of dots.</div>

Join (\sqcup) of two closed Octagons



Join (\sqcup) of two closed Octagons

Diagram illustrating the join (\sqcup) of two closed octagons, showing the state evolution of variables x , y , and m over time steps x^+ , x^- , y^+ , y^- , m^+ , and m^- .

The code snippet is as follows:

```

x = 1;
y = x;
while (x <= m) {
    x = x + 1;
    y = y + x;
}
assert (y >= m);

```

Initial state (top octagon):

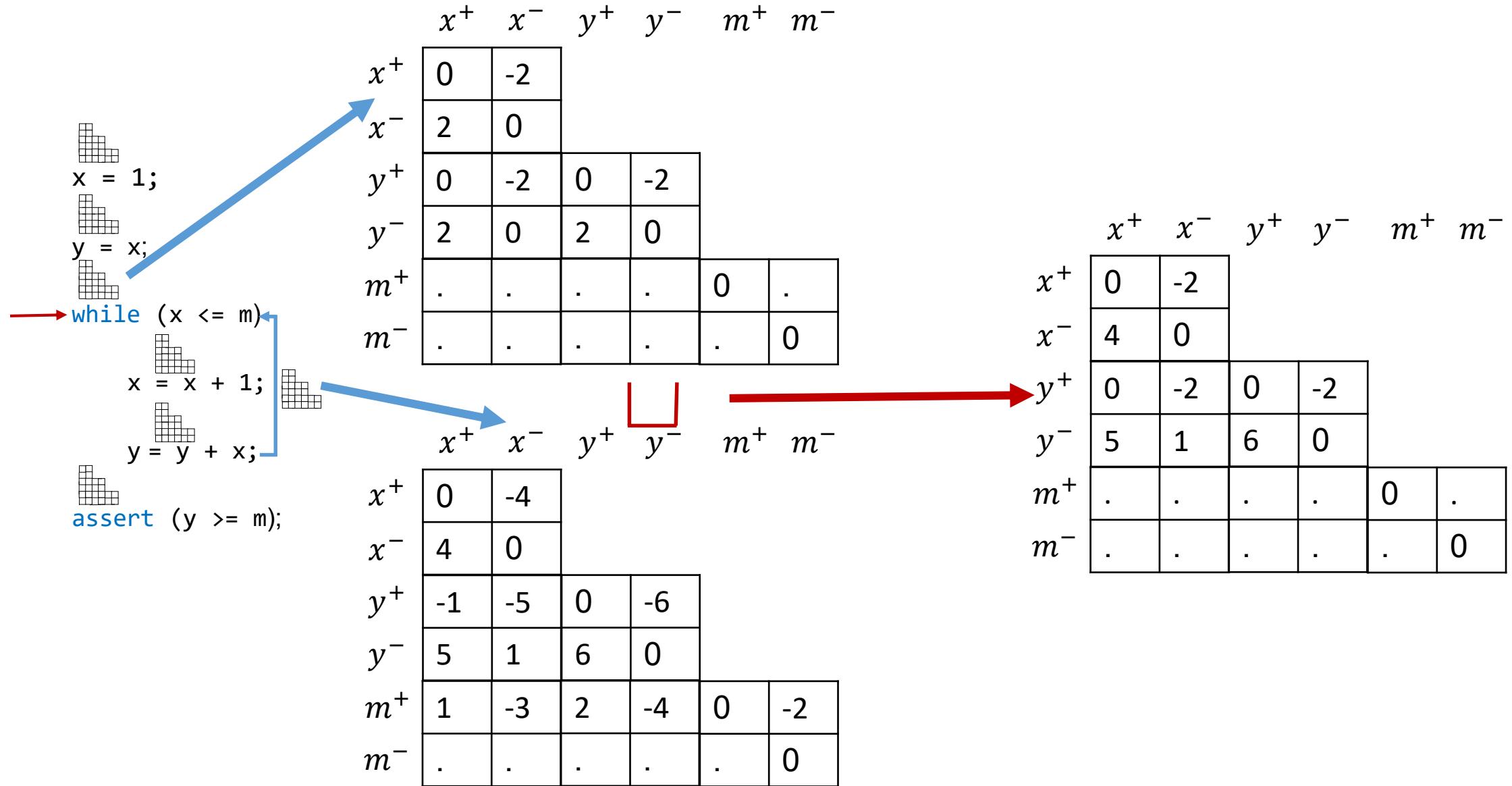
	x^+	x^-	y^+	y^-	m^+	m^-
x^+	0	-2				
x^-	2	0				
y^+	0	-2	0	-2		
y^-	2	0	2	0		
m^+	0	.
m^-	0

Final state (bottom octagon):

	x^+	x^-	y^+	y^-	m^+	m^-
x^+	0	-4				
x^-	4	0				
y^+	-1	-5	0	-6		
y^-	5	1	6	0		
m^+	1	-3	2	-4	0	-2
m^-	0

Arrows indicate the flow of state from the initial state to the final state.

Join (\sqcup) of two closed Octagons



Time Complexity of Octagon Operators

Octagon Operator	Time Complexity
Meet (\sqcap)	$O(n^2)$
Join (\sqcup)	$O(n^2)$
Inclusion (\sqsubseteq)	$O(n^2)$
Equality (=)	$O(n^2)$
Widening (\bar{v})	$O(n^2)$
Closure (*)	$O(n^3)$

Key Idea: Online Decomposition

- The set of program variables can be partitioned into disjoint subsets called independent components.
- Each independent component corresponds to a smaller octagon.
- Transitive closure can be applied independently on smaller octagons.
- Maintain the decomposition dynamically throughout the analysis.

	x^+	x^-	y^+	y^-	m^+	m^-
x^+	0	-2				
x^-	4	0				
y^+	0	-2	0	-2		
y^-	5	1	6	0		
m^+	0	2
m^-	0



	x^+	x^-	y^+	y^-	m^+	m^-
x^+	0	-2				
x^-	4	0				
y^+	0	-2	0	-2		
y^-	5	1	6	0		
m^+	0	2				
m^-	.	0				

Other Improvements

- We reduced operation count of closure by half.
- We designed sparse closure for very sparse matrices that runs in $O(n^2)$ time.
- We performed cache optimizations and vectorization for all octagon operators.
- If the matrix becomes dense, keeping decomposition is not feasible.
 - We designed different octagon types and their corresponding operators.
 - We keep track of sparsity and switch dynamically between different types.

Implementation

- ELINA is implemented in C using double precision.
- Provides interface for analyzing program written in C++ and Java.
- Supports SSE and AVX intrinsics.
- Can be directly plugged into any existing static analyzer using APRON.

Experimental Evaluation

- CPAchecker (Beyer et al. CAV, 2011)
 - participates in software verification competitions.
- TOUCHBOOST (Brutschy et al. OOPSLA, 2014)
 - analyzes eventdriven TouchDevelop applications.
- DPS (Raychev et al. SAS, 2013)
 - analyzes parallel programs and introduces synchronization for determinism.
- DIZY (Partush et al. SAS, 2013)
 - computes semantic differences between a program and its patched version.

Experimental Results: CPAchecker

(Beyer et al., CAV, 2011)

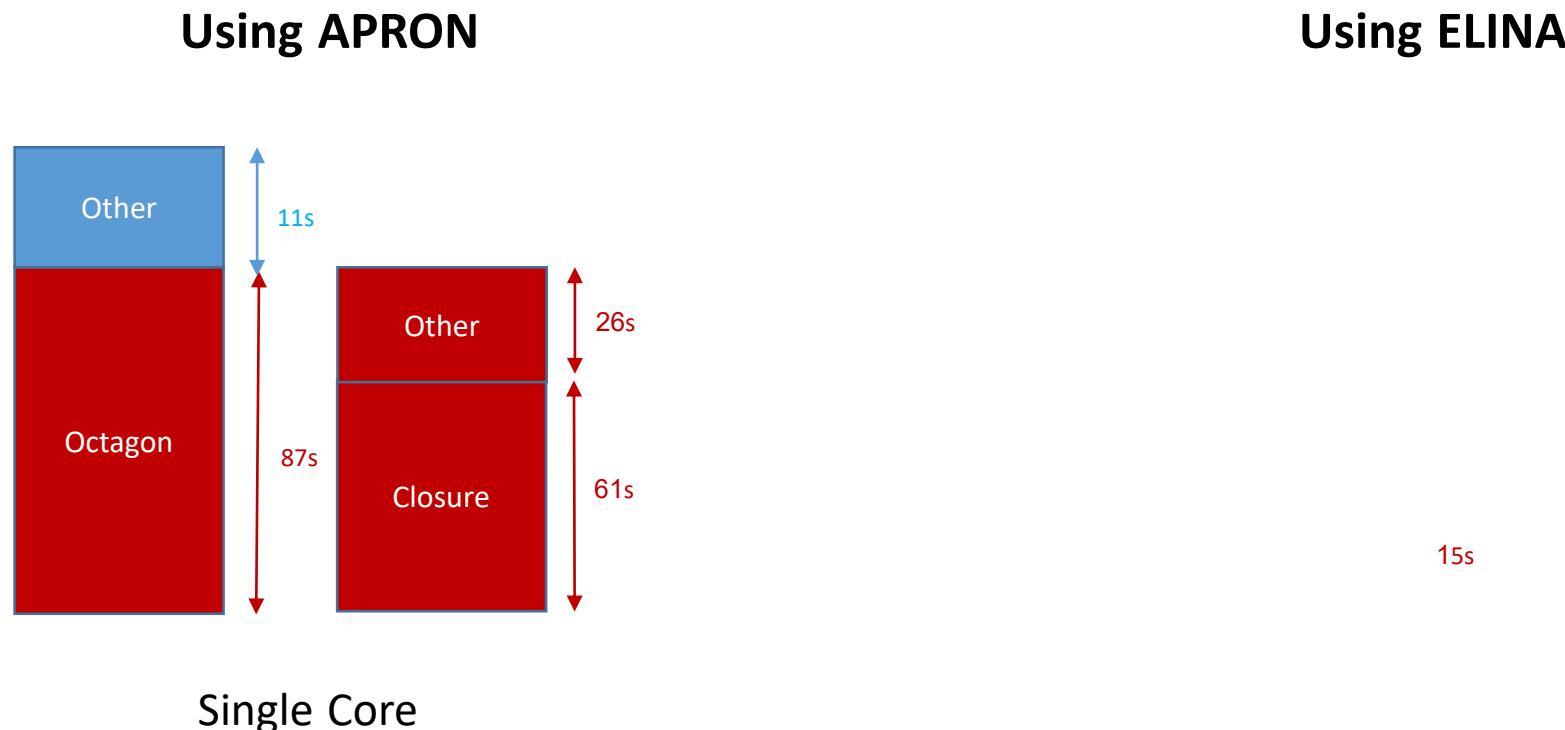
Using APRON

Using ELINA

15s

Experimental Results: CPAchecker

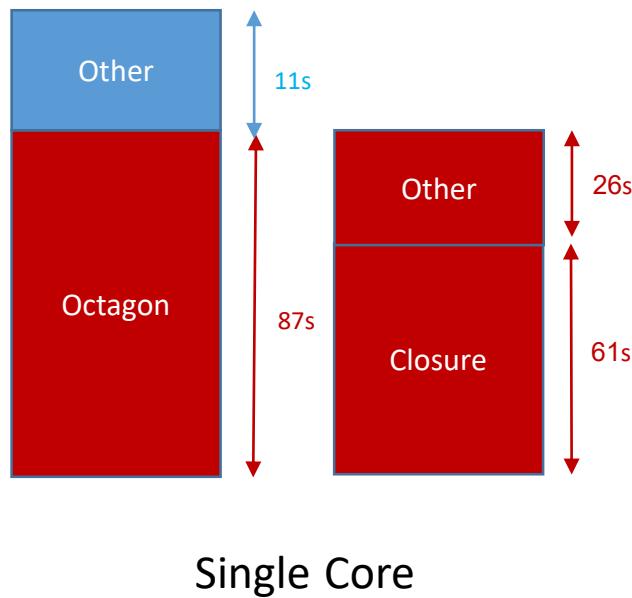
(Beyer et al., CAV, 2011)



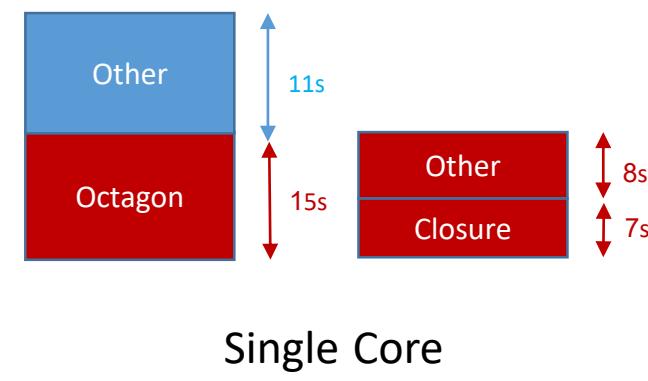
Experimental Results: CPAchecker

(Beyer et al., CAV, 2011)

Using APRON



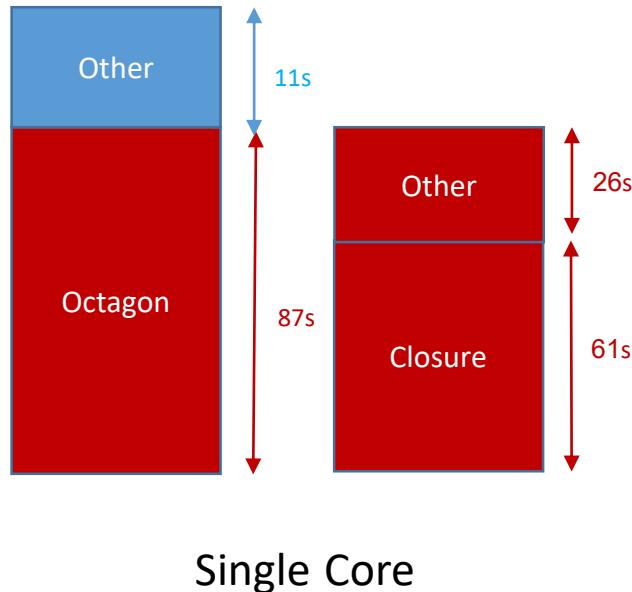
Using ELINA



Experimental Results: CPAchecker

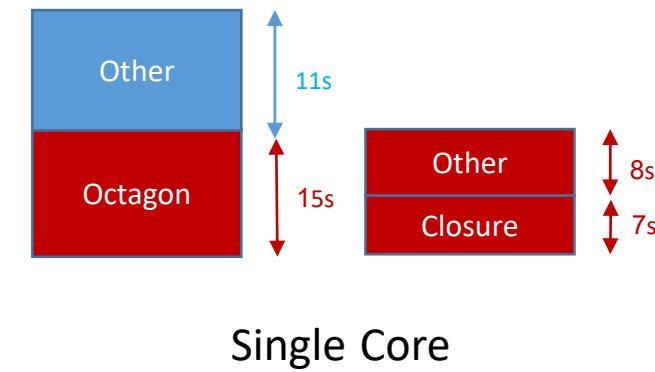
(Beyer et al., CAV, 2011)

Using APRON



Using ELINA

- Closure Speedup: 8.4x
- Octagon Speedup: 6x
- Overall Speedup: 3.7x



Experimental Results: DPS

(Raychev et al, SAS, 2013)

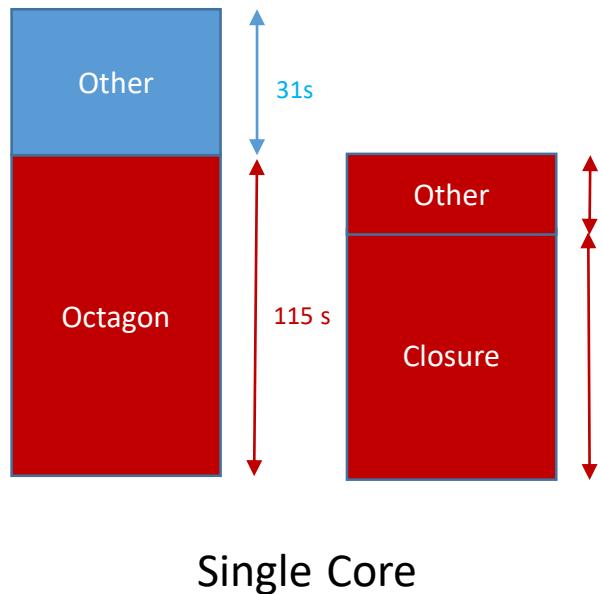
Using APRON

Using ELINA

Experimental Results: DPS

(Raychev et al, SAS, 2013)

Using APRON

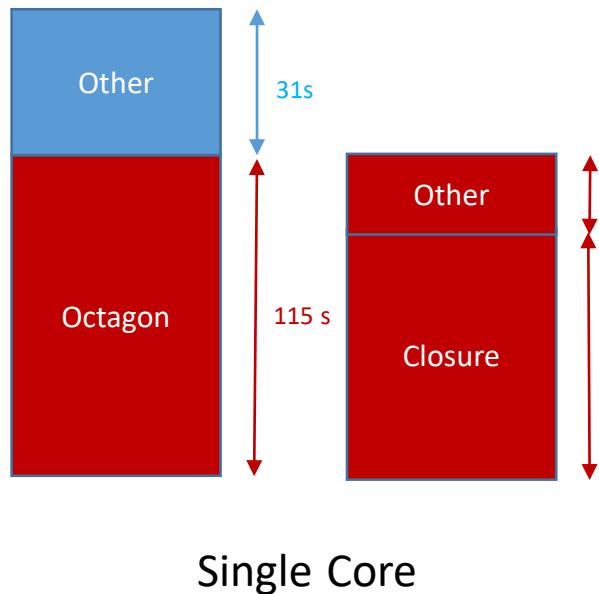


Using ELINA

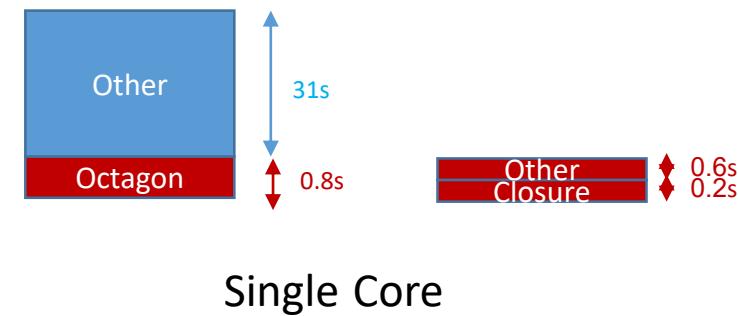
Experimental Results: DPS

(Raychev et al, SAS, 2013)

Using APRON



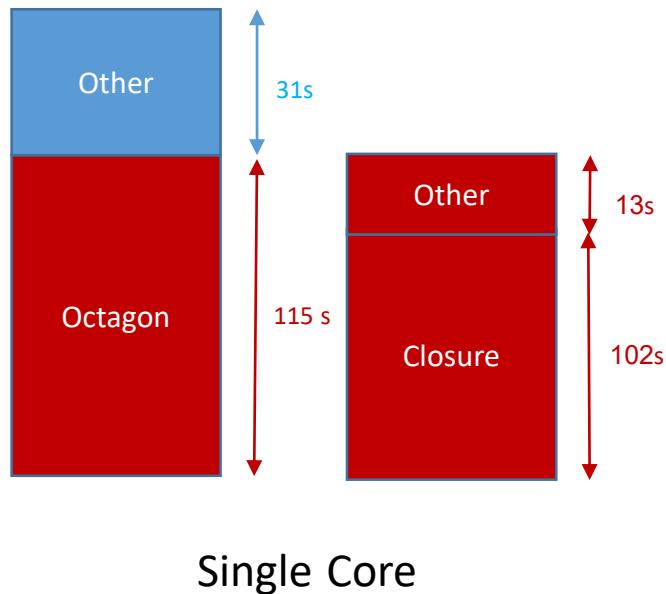
Using ELINA



Experimental Results: DPS

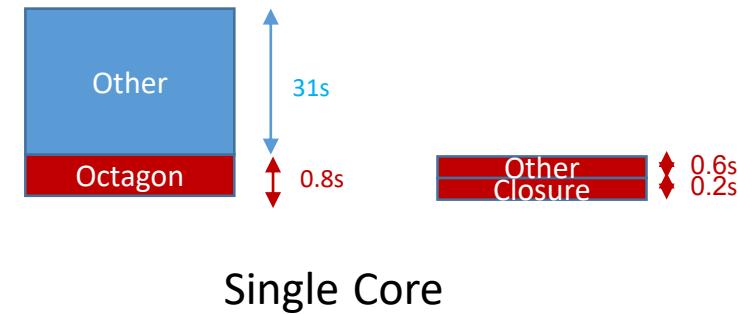
(Raychev et al, SAS, 2013)

Using APRON



Using ELINA

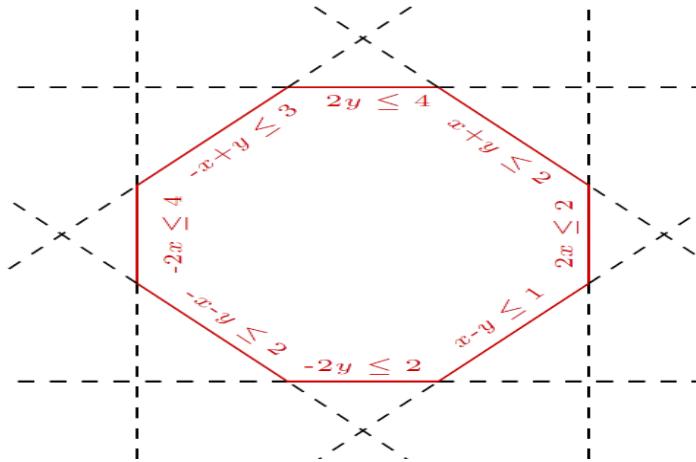
- Closure Speedup: 665x
- Octagon Speedup: 146x
- Overall Speedup: 4.2x



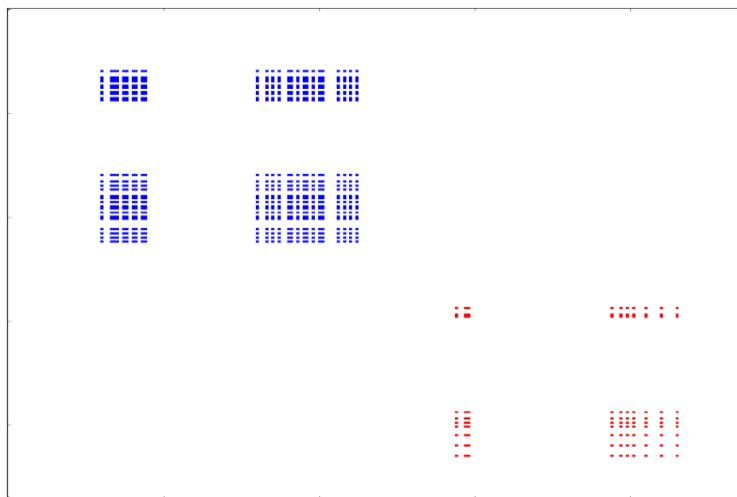
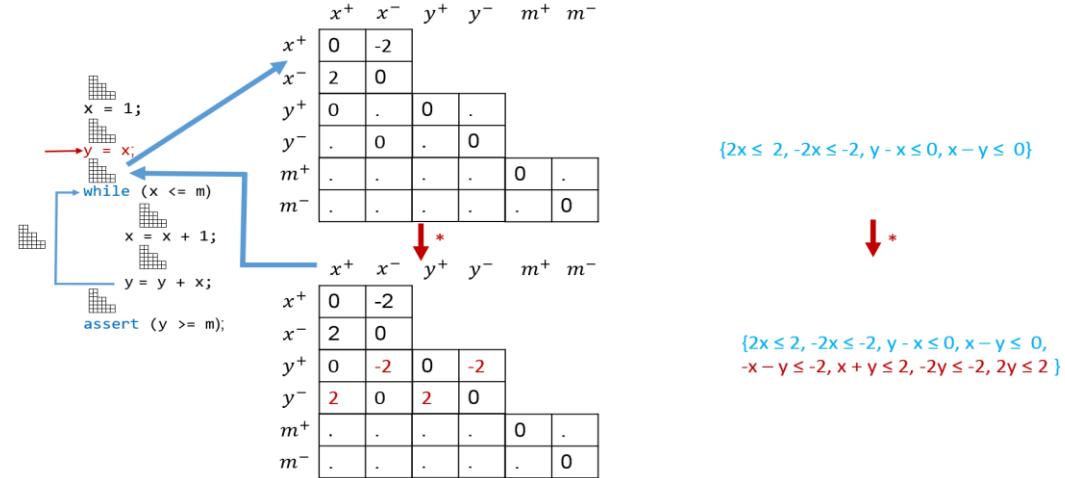
Related Work

- Variable Packing (Venet et al. PLDI, 2004)
 - Loses precision, may take more iterations to converge.
- Octagon operators on GPUs (Banterle et al. SAS, 2007)
 - Our optimized library will run much faster on GPUs.

Conclusion



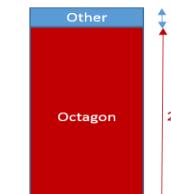
Closure (*) increases precision of Join (\sqcup) operator



Octagon Analysis is Expensive

Example: Static analyzer for TouchDevelop
(Brutschy et al. OOPSLA, 2014)

Using APRON



Single Core

Using ELINA



Single Core