# Task5 - SCSR

Andrea Munarin (879607), Simone Jovon (882028)

March 2023

## 1 Exercise 1

Consider the domains of Parity, the domain of Sign extended with 0+ and 0-, and the domain of intervals. Using each of these abstract domains, compute the abstract interpretation of the following program:

```
input(x) //(where the expected values range from 0 to 5)
x=2*x
i=0
y=-3
while (i<100)
    x=x+2
    y=y-x
print(y)
```

### 1.1 SOLUTION

***Domain of Parity***

Before starting with the analysis it's important to state that the print(y) does not affect the domain of our variable, also this instruction is never reached in our piece of code (since i is never incremented). So we can ignore it.
In order to perform the abstract interpretation of this snippet of code, we apply the sigma function to our instructions. In particular, we apply the rule for sequential statements until we reach the while loop

$$\sigma'_{\texttt{x=2*x;i=0;y=-3}}\{x/T\} = \sigma'_{\texttt{i=0;y=-3}}(\sigma'_{\texttt{x=2*x}}\{x/T\})$$

Since every number multiplied by two became even, we can write:

$\sigma'_{\texttt{i=0;y=-3}}\{x/Even\} =$
$\sigma'_{\texttt{y=-3}}(\sigma'_{\texttt{i=0}}\{x/E\}) =$
$\sigma'_{\texttt{y=-3}}\{x/E, i/E\} =$
$\{x/E, i/E, y/O\}$

Now that we collect this information, we proceed with the while loop:

$\sigma'_{\texttt{while i<100 do x=x+2;y=y-x}}\{x/E, i/E, y/O\}$

To extract the information from the while loop, we need to get the least upper bound between the current input set (the case in which we don't enter the loop) and the sequence of instructions in the loop

$\texttt{lub}(\{x/E, i/E, y/O\}, \sigma'_{\texttt{x=x+2;y=y-x;while i<100 do x=x+2;y=y-x}}\{x/E, i/E, y/O\})$

The result from a sum between two and an even number is again an even number. An odd number subtracted by an even number will remain odd:

$\texttt{lub}(\{x/E, i/E, y/O\}, \sigma'_{\texttt{while i<100 do x=x+2;y=y-x}}\{x/E, i/E, y/O\})$

Now we obtain again a situation equal to the one before, this means that we reach a ***fixpoint***:

$\texttt{lub}(\{x/E, i/E, y/O\}, \texttt{lub}(\{x/E, i/E, y/O\}, \sigma'_{\texttt{x=x+2;y=y-x;while i<100 do x=x+2;y=y-x}}\{x/E, i/E, y/O\}))$

Since we have reached a fixpoint we can write:

$\texttt{lub}(\{x/E, i/E, y/O\}, \texttt{lub}(\{x/E, i/E, y/O\}, \{x/E, i/E, y/O\})) =$
$\texttt{lub}(\{x/E, i/E, y/O\}, \{x/E, i/E, y/O\})$

in the end, we obtain:

$$\{x/E, i/E, y/O\}$$

***Domain of Sign Extended***

We proceed with the same reasoning as the domain of parity:

$\sigma'_{\texttt{x=2*x;i=0;y=-3}}\{x/0+\} =$
$\sigma'_{\texttt{i=0;y=-3}}(\sigma'_{\texttt{x=2*x}}\{x/0+\}) =$
$\sigma'_{\texttt{i=0;y=-3}}\{x/0+\} =$
$\sigma'_{\texttt{y=-3}}(\sigma'_{\texttt{i=0}}\{x/0+\}) =$
$\sigma'_{\texttt{y=-3}}\{x/0+, i/0\} =$
$\{x/0+, i/0, y/-\}$

We proceed with the while loop:

$\sigma'_{\texttt{while i<100 do x=x+2;y=y-x}}\{x/0+, i/0, y/-\} =$
$\texttt{lub}(\{x/0+, i/0, y/-\}, \sigma'_{\texttt{x=x+2;y=y-x;while i<100 do x=x+2;y=y-x}}\{x/0+, i/0, y/-\})$
$\texttt{lub}(\{x/0+, i/0, y/-\}, \sigma'_{\texttt{while i<100 do x=x+2;y=y-x}}\{x/+, i/0, y/-\})$

$\texttt{lub}(\{x/0+, i/0, y/-\}, \texttt{lub}(\{x/+, i/0, y/-\}, \sigma'_{\texttt{x=x+2;y=y-x;while i<100 do x=x+2;y=y-x}}\{x/+, i/0, y/-\}))$

We have reached a fixpoint. So:

$\texttt{lub}(\{x/0+, i/0, y/-\}, \texttt{lub}(\{x/+, i/0, y/-\}, \{x/+, i/0, y/-\}))$
$\texttt{lub}(\{x/0+, i/0, y/-\}, \{x/+, i/0, y/-\})$

The final result:

$$\{x/0+, i/0, y/-\}$$

***Domain of Intervals***

With the intervals, in order to perform an efficient computation, we take advantage of the widening operator. Once we compute the analysis, we use the narrowing operator to reduce the results and obtain a more detailed information.

| Stored Values | statement | Out |
|---|---|---|
| $x_1 = \emptyset, y_1 = \emptyset, i_1 = \emptyset$ | $x_1 = [0,5]$ | $x_1 = [0,5], y_1 = \emptyset, i_1 = \emptyset$ |
| $x_2 = \emptyset, y_2 = \emptyset, i_2 = \emptyset$ | $x_2 = x_1 \otimes [2,2]$ | $x_2 = \emptyset, y_2 = \emptyset, i_2 = \emptyset$ |
| $x_3 = \emptyset, y_3 = \emptyset, i_3 = \emptyset$ | $i_3 = [0,0]$ | $x_3 = \emptyset, y_3 = \emptyset, i_3 = \emptyset$ |
| $x_4 = \emptyset, y_4 = \emptyset, i_4 = \emptyset$ | $y_4 = [-3,-3]$ | $x_4 = \emptyset, y_4 = \emptyset, i_4 = \emptyset$ |
| $x_5 = \emptyset, y_5 = \emptyset, i_5 = \emptyset$ | $x_5 = x_5 \nabla (x_4 \cup x_7)$<br>$y_5 = y_5 \nabla (y_4 \cup y_7)$<br>$i_5 = i_5 \nabla ((i_4 \cup i_7) \cap [0,99])$ | $x_5 = \emptyset, y_5 = \emptyset, i_5 = \emptyset$ |
| $x_6 = \emptyset, y_6 = \emptyset, i_6 = \emptyset$ | $x_6 = x_5 \oplus [2,2]$ | $x_6 = \emptyset, y_6 = \emptyset, i_6 = \emptyset$ |
| $x_7 = \emptyset, y_7 = \emptyset, i_7 = \emptyset$ | $y_7 = y_6 \ominus x_6$ | $x_7 = \emptyset, y_7 = \emptyset, i_7 = \emptyset$ |
| $x_8 = \emptyset, y_8 = \emptyset, i_8 = \emptyset$ | $x_8 = x_7 \cup x_4$<br>$y_8 = y_7 \cup y_4$<br>$i_8 = (i_7 \cup i_4) \cap [100,+\infty]$ | $x_8 = \emptyset, y_8 = \emptyset, i_8 = \emptyset$ |

We don't show each basic instruction because it's very intuitive. Let's start reasoning around the while loop and the instructions inside it:

| Stored Values | statement | Out |
|---|---|---|
| $x_1 = [0,5], y_1 = \emptyset, i_1 = \emptyset$ | $x_1 = [0,5]$ | $x_1 = [0,5], y_1 = \emptyset, i_1 = \emptyset$ |
| $x_2 = [0,10], y_2 = \emptyset, i_2 = \emptyset$ | $x_2 = x_1 \otimes [2,2]$ | $x_2 = [0,10], y_2 = \emptyset, i_2 = \emptyset$ |
| $x_3 = [0,10], y_3 = \emptyset, i_3 = [0,0]$ | $i_3 = [0,0]$ | $x_3 = [0,10], y_3 = \emptyset, i_3 = [0,0]$ |
| $x_4 = [0,10], y_4 = [-3,-3], i_4 = [0,0]$ | $y_4 = [-3,-3]$ | $x_4 = [0,10], y_4 = [-3,-3], i_4 = [0,0]$ |
| $x_5 = [0,10]$<br>$y_5 = [-3,-3]$<br>$i_5 = [0,0]$ | $x_5 = x_5 \nabla (x_4 \cup x_7)$<br>$y_5 = y_5 \nabla (y_4 \cup y_7)$<br>$i_5 = i_5 \nabla ((i_4 \cup i_7) \cap [0,99])$ | $x_5 = [0,10]$<br>$y_5 = [-3,-3]$<br>$i_5 = [0,0]$ |
| $x_6 = \emptyset, y_6 = \emptyset, i_6 = \emptyset$ | $x_6 = x_5 \oplus [2,2]$ | $x_6 = [2,12], y_6 = [-3,-3], i_6 = [0,0]$ |
| $x_7 = \emptyset, y_7 = \emptyset, i_7 = \emptyset$ | $y_7 = y_6 \ominus x_6$ | $x_7 = \emptyset, y_7 = \emptyset, i_7 = \emptyset$ |
| $x_8 = \emptyset, y_8 = \emptyset, i_8 = \emptyset$ | $x_8 = x_7 \cup x_4$<br>$y_8 = y_7 \cup y_4$<br>$i_8 = (i_7 \cup i_4) \cap [100,+\infty]$ | $x_8 = \emptyset, y_8 = \emptyset, i_8 = \emptyset$ |

| Stored Values | statement | Out |
|---|---|---|
| $x_1 = [0,5], y_1 = \emptyset, i_1 = \emptyset$ | $x_1 = [0,5]$ | $x_1 = [0,5], y_1 = \emptyset, i_1 = \emptyset$ |
| $x_2 = [0,10], y_2 = \emptyset, i_2 = \emptyset$ | $x_2 = x_1 \otimes [2,2]$ | $x_2 = [0,10], y_2 = \emptyset, i_2 = \emptyset$ |
| $x_3 = [0,10], y_3 = \emptyset, i_3 = [0,0]$ | $i_3 = [0,0]$ | $x_3 = [0,10], y_3 = \emptyset, i_3 = [0,0]$ |
| $x_4 = [0,10], y_4 = [-3,-3], i_4 = [0,0]$ | $y_4 = [-3,-3]$ | $x_4 = [0,10], y_4 = [-3,-3], i_4 = [0,0]$ |
| $x_5 = [0,10]$<br>$y_5 = [-3,-3]$<br>$i_5 = [0,0]$ | $x_5 = x_5 \nabla (x_4 \cup x_7)$<br>$y_5 = y_5 \nabla (y_4 \cup y_7)$<br>$i_5 = i_5 \nabla ((i_4 \cup i_7) \cap [0,99])$ | $x_5 = [0,10]$<br>$y_5 = [-3,-3]$<br>$i_5 = [0,0]$ |
| $x_6 = [2,12], y_6 = [-3,-3], i_6 = [0,0]$ | $x_6 = x_5 \oplus [2,2]$ | $x_6 = [2,12], y_6 = [-3,-3], i_6 = [0,0]$ |
| $x_7 = \emptyset, y_7 = \emptyset, i_7 = \emptyset$ | $y_7 = y_6 \ominus x_6$ | $x_7 = [2,12], y_7 = [-15,-5], i_7 = [0,0]$ |
| $x_8 = \emptyset, y_8 = \emptyset, i_8 = \emptyset$ | $x_8 = x_7 \cup x_4$<br>$y_8 = y_7 \cup y_4$<br>$i_8 = (i_7 \cup i_4) \cap [100,+\infty]$ | $x_8 = \emptyset, y_8 = \emptyset, i_8 = \emptyset$ |

Now that we reach the end of the while loop we need to reapply the condition and apply the nabla operator. Here we get something interesting. The intervals of x and y seem to be growing in one direction. Let's continue:

| Stored Values | statement | Out |
|---|---|---|
| $x_1 = [0,5], y_1 = \emptyset, i_1 = \emptyset$ | $x_1 = [0,5]$ | $x_1 = [0,5], y_1 = \emptyset, i_1 = \emptyset$ |
| $x_2 = [0,10], y_2 = \emptyset, i_2 = \emptyset$ | $x_2 = x_1 \otimes [2,2]$ | $x_2 = [0,10], y_2 = \emptyset, i_2 = \emptyset$ |
| $x_3 = [0,10], y_3 = \emptyset, i_3 = [0,0]$ | $i_3 = [0,0]$ | $x_3 = [0,10], y_3 = \emptyset, i_3 = [0,0]$ |
| $x_4 = [0,10], y_4 = [-3,-3], i_4 = [0,0]$ | $y_4 = [-3,-3]$ | $x_4 = [0,10], y_4 = [-3,-3], i_4 = [0,0]$ |
| $x_5 = [0,12]$<br>$y_5 = [-15,-3]$<br>$i_5 = [0,0]$ | $x_5 = x_5 \nabla (x_4 \cup x_7)$<br>$y_5 = y_5 \nabla (y_4 \cup y_7)$<br>$i_5 = i_5 \nabla ((i_4 \cup i_7) \cap [0,99])$ | $x_5 = [0,12]$<br>$y_5 = [-15,-3]$<br>$i_5 = [0,0]$ |
| $x_6 = [2,14], y_6 = [-15,-3], i_6 = [0,0]$ | $x_6 = x_5 \oplus [2,2]$ | $x_6 = [2,14], y_6 = [-15,-3], i_6 = [0,0]$ |
| $[2,12], y_7 = [-15,-5], i_7 = [0,0]$ | $\underline{y_7 = y_6 \ominus x_6}$ | $x_7 = [2,14], y_7 = [-29,-5], i_7 = [0,0]$ |
| $x_8 = \emptyset, y_8 = \emptyset, i_8 = \emptyset$ | $x_8 = x_7 \cup x_4$<br>$y_8 = y_7 \cup y_4$<br>$i_8 = (i_7 \cup i_4) \cap [100,+\infty]$ | $x_8 = \emptyset, y_8 = \emptyset, i_8 = \emptyset$ |

As we saw there is a convergence issue. So we applied a threshold widening operator on intervals. For x $k_x = 14$, while for y $k_y = -29$

| Stored Values | statement | Out |
|---|---|---|
| $x_1 = [0,5], y_1 = \emptyset, i_1 = \emptyset$ | $x_1 = [0,5]$ | $x_1 = [0,5], y_1 = \emptyset, i_1 = \emptyset$ |
| $x_2 = [0,10], y_2 = \emptyset, i_2 = \emptyset$ | $x_2 = x_1 \otimes [2,2]$ | $x_2 = [0,10], y_2 = \emptyset, i_2 = \emptyset$ |
| $x_3 = [0,10], y_3 = \emptyset, i_3 = [0,0]$ | $i_3 = [0,0]$ | $x_3 = [0,10], y_3 = \emptyset, i_3 = [0,0]$ |
| $x_4 = [0,10], y_4 = [-3,-3], i_4 = [0,0]$ | $y_4 = [-3,-3]$ | $x_4 = [0,10], y_4 = [-3,-3], i_4 = [0,0]$ |
| $x_5 = [0,12]$<br>$y_5 = [-15,-3]$<br>$i_5 = [0,0]$ | $x_5 = x_5 \nabla (x_4 \cup x_7)$<br>$y_5 = y_5 \nabla (y_4 \cup y_7)$<br>$i_5 = i_5 \nabla ((i_4 \cup i_7) \cap [0,99])$ | $x_5 = [0,12] \nabla [0,14] = [0,+\infty]$<br>$y_5 = [-15,-3] \nabla [-29,-3] = [-\infty,-3]$<br>$i_5 = [0,0]$ |
| $x_6 = [0,14], y_6 = [-15,-3], i_6 = [0,0]$ | $x_6 = x_5 \oplus [2,2]$ | $x_6 = [2,14], y_6 = [-15,-3], i_6 = [0,0]$ |
| $x_7 = [2,14], y_7 = [-29,-5], i_7 = [0,0]$ | $y_7 = y_6 \ominus x_6$ | $x_7 = [2,14], y_7 = [-29,-5], i_7 = [0,0]$ |
| $x_8 = \emptyset, y_8 = \emptyset, i_8 = \emptyset$ | $x_8 = x_7 \cup x_4$<br>$y_8 = y_7 \cup y_4$<br>$i_8 = (i_7 \cup i_4) \cap [100,+\infty]$ | $x_8 = \emptyset, y_8 = \emptyset, i_8 = \emptyset$ |

| Stored Values | statement | Out |
|---|---|---|
| $x_1 = [0,5], y_1 = \emptyset, i_1 = \emptyset$ | $x_1 = [0,5]$ | $x_1 = [0,5], y_1 = \emptyset, i_1 = \emptyset$ |
| $x_2 = [0,10], y_2 = \emptyset, i_2 = \emptyset$ | $x_2 = x_1 \otimes [2,2]$ | $x_2 = [0,10], y_2 = \emptyset, i_2 = \emptyset$ |
| $x_3 = [0,10], y_3 = \emptyset, i_3 = [0,0]$ | $i_3 = [0,0]$ | $x_3 = [0,10], y_3 = \emptyset, i_3 = [0,0]$ |
| $x_4 = [0,10], y_4 = [-3,-3], i_4 = [0,0]$ | $y_4 = [-3,-3]$ | $x_4 = [0,10], y_4 = [-3,-3], i_4 = [0,0]$ |
| $x_5 = [0,+\infty]$<br>$y_5 = [-\infty,-3]$<br>$i_5 = [0,0]$ | $x_5 = x_5 \nabla (x_4 \cup x_7)$<br>$y_5 = y_5 \nabla (y_4 \cup y_7)$<br>$i_5 = i_5 \nabla ((i_4 \cup i_7) \cap [0,99])$ | $x_5 = [0,12] \nabla [0,14] = [0,+\infty]$<br>$y_5 = [-15,-3] \nabla [-29,-3] = [-\infty,-3]$<br>$i_5 = [0,0]$ |
| $x_6 = [2,+\infty], y_6 = [-\infty,-3], i_6 = [0,0]$ | $x_6 = x_5 \oplus [2,2]$ | $x_6 = [2,+\infty], y_6 = [-\infty,-3], i_6 = [0,0]$ |
| $x_7 = [2,14], y_7 = [-29,-5], i_7 = [0,0]$ | $\underline{y_7 = y_6 \ominus x_6}$ | $x_7 = [2,+\infty], y_7 = [-\infty,-5], i_7 = [0,0]$ |
| $x_8 = \emptyset, y_8 = \emptyset, i_8 = \emptyset$ | $x_8 = x_7 \cup x_4$<br>$y_8 = y_7 \cup y_4$<br>$i_8 = (i_7 \cup i_4) \cap [100,+\infty]$ | $x_8 = \emptyset, y_8 = \emptyset, i_8 = \emptyset$ |

If we keep going we can easily understand that we reach a fixpoint. It's also important to understand that at the end we cannot rely on narrowing because we have no limit intervals (only on i that is never incremented). So our final result is the following (notice that we cannot exit from the cycle since i is never incremented, and does not break the loop condition).

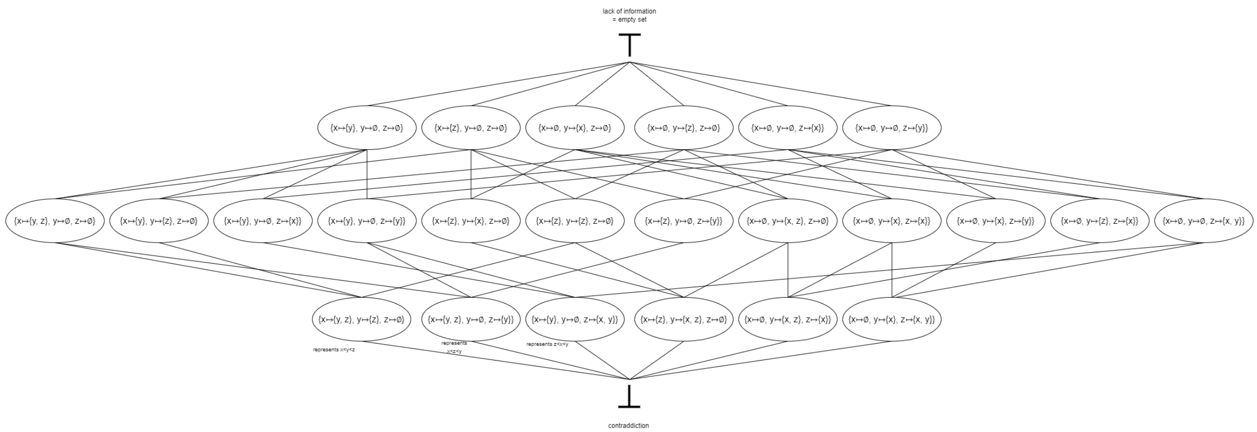
| Stored Values | statement | Out |
|---|---|---|
| $x_1 = [0,5], y_1 = \emptyset, i_1 = \emptyset$ | $x_1 = [0,5]$ | $x_1 = [0,5], y_1 = \emptyset, i_1 = \emptyset$ |
| $x_2 = [0,10], y_2 = \emptyset, i_2 = \emptyset$ | $x_2 = x_1 \otimes [2,2]$ | $x_2 = [0,10], y_2 = \emptyset, i_2 = \emptyset$ |
| $x_3 = [0,10], y_3 = \emptyset, i_3 = [0,0]$ | $i_3 = [0,0]$ | $x_3 = [0,10], y_3 = \emptyset, i_3 = [0,0]$ |
| $x_4 = [0,10], y_4 = [-3,-3], i_4 = [0,0]$ | $y_4 = [-3,-3]$ | $x_4 = [0,10], y_4 = [-3,-3], i_4 = [0,0]$ |
| $x_5 = [0,+\infty]$<br>$y_5 = [-\infty,-3]$<br>$i_5 = [0,0]$ | $x_5 = x_5 \nabla (x_4 \cup x_7)$<br>$y_5 = y_5 \nabla (y_4 \cup y_7)$<br>$i_5 = i_5 \nabla ((i_4 \cup i_7) \cap [0,99])$ | $x_5 = [0,+\infty]$<br>$y_5 = [-\infty,-3]$<br>$i_5 = [0,0]$ |
| $x_6 = [2,+\infty], y_6 = [-\infty,-3], i_6 = [0,0]$ | $x_6 = x_5 \oplus [2,2]$ | $x_6 = [2,+\infty], y_6 = [-\infty,-3], i_6 = [0,0]$ |
| $x_7 = [2,+\infty], y_7 = [-\infty,-5], i_7 = [0,0]$ | $y_7 = y_6 \ominus x_6$ | $x_7 = [2,+\infty], y_7 = [-\infty,-5], i_7 = [0,0]$ |
| $x_8 = \emptyset, y_8 = \emptyset, i_8 = \emptyset$ | $x_8 = x_7 \cup x_4$<br>$y_8 = y_7 \cup y_4$<br>$i_8 = (i_7 \cup i_4) \cap [100,+\infty]$ | $x_8 = \emptyset, y_8 = \emptyset, i_8 = \emptyset$ |

# 2 Exercise 2

Consider the domain of strict upper bounds as defined in the paper "Pentagons: A Weakly Relational Abstract Domain for the Efficient Validation of Array Accesses" by Logozzo and Fahndrich.
Depict the domain when the set of program variables is {x, y, z}
Compute the abstract interpretation of a small program snippet (at your choice) using such domain.

## 2.1 SOLUTION

The domain when the set of program variable is {x, y, z} can be represented by the following picture.




Notice that the information that collect is related to each variable. The set {x↦{x,y,z}} for example is not admitted because does not exist a variable that is less than itself (so this case is included in the bottom element).

In the picture and in the abstract interpretation we use the following notation: `x` $\mapsto\{y_1$, ... $y_n\}$ with the meaning that `x` is strictly smaller than each of the $y_i$.
The code in which we want to work is the following

```
input(x) //(where the expected values range from 0 to 5)
z=x+2
y=x-1
while(y<=z)
    x=x-1
    y=y+x
print(y)
```

Let's start the analysis using sequential composition of instructions:

$$\sigma'_{\texttt{z=x+2;y=x-1}}\{x \mapsto \emptyset, y \mapsto \emptyset, z \mapsto \emptyset\} = \sigma'_{\texttt{y=x-1}}(\sigma'_{\texttt{z=x+2}}\{x \mapsto \emptyset, y \mapsto \emptyset, z \mapsto \emptyset\})$$

Since z is strictly greater than x we can write:

$\sigma'_{\texttt{y=x-1}}\{x \mapsto \{z\}, y \mapsto \emptyset, z \mapsto \emptyset\} =$
$\{x \mapsto \{z\}, y \mapsto \{x\}, z \mapsto \emptyset\}$

Now that we collect this information, we proceed with the while loop:

$\sigma'_{\texttt{while y<=z do x=x-1;y=y+x}}\{x \mapsto \{z\}, y \mapsto \{x\}, z \mapsto \emptyset\}$

To extract the information from the while loop, we need to get the least upper bound between the current input set (the case in which we don't enter the loop) and the sequence of instructions in the loop

$\texttt{lub}(\{x \mapsto \{z\}, y \mapsto \{x\}, z \mapsto \emptyset\}, \sigma'_{\texttt{x=x-1;y=y+x; while y<=z do x=x-1;y=y+x}}\{x \mapsto \{z\}, y \mapsto \{x\}, z \mapsto \emptyset\})$

When we decrease x we can still rely on the fact that x is smaller than z, while increasing y does not guarantee that y remains smaller than x. We cannot also write that x is smaller than y, because y can be zero (y = 0 + x so y = x). Notice also that since the condition of the while loop consider also equality, we cannot gain any information:

$\texttt{lub}(\{x \mapsto \{z\}, y \mapsto \{x\}, z \mapsto \emptyset\}, \sigma'_{\texttt{while y<=z do x=x-1;y=y+x}}\{x \mapsto \{z\}, y \mapsto \emptyset, z \mapsto \emptyset\})$

Now we proceed with the iteration:

$\texttt{lub}(\{x \mapsto \{z\}, y \mapsto \{x\}, z \mapsto \emptyset\}, \texttt{lub}(\{x \mapsto \{z\}, y \mapsto \emptyset, z \mapsto \emptyset\}, \sigma'_{\texttt{x=x-1;y=y+x; while y<=z do x=x-1;y=y+x}}\{x \mapsto \{z\}, y \mapsto \emptyset, z \mapsto \emptyset\}))$

We have reached a fixpoint:

$\texttt{lub}(\{x \mapsto \{z\}, y \mapsto \{x\}, z \mapsto \emptyset\}, \texttt{lub}(\{x \mapsto \{z\}, y \mapsto \emptyset, z \mapsto \emptyset\}, \{x \mapsto \{z\}, y \mapsto \emptyset, z \mapsto \emptyset\})) =$
$\texttt{lub}(\{x \mapsto \{z\}, y \mapsto \{x\}, z \mapsto \emptyset\}, \{x \mapsto \{z\}, y \mapsto \emptyset, z \mapsto \emptyset\})$

The `lub` between this two set is $\{x \mapsto \{z\}, y \mapsto \emptyset, z \mapsto \emptyset\}$, but, since we have to consider also the comparisons in this analysis, when we reach the print instruction, we are sure that y¿z (otherwise we will be inside the loop). So in the end, we obtain:

$$\{x \mapsto \{z\}, y \mapsto \emptyset, z \mapsto \{y\}\}$$

So we are sure, at the end of our program, that x is strictly smaller than z and z is smaller than y: $x < z < y$.