Neha Maddali

HW4

1. Abstract Domain A (+, -, 0)

Concrete Domain D: INT

Variable ‘x’:

* If x is negative (x < 0 ): it belongs to the abstract domain ‘-‘
* If x is zero (x == 0): it belongs to the abstract domain ‘0’
* If x is positive (x > 0): it belongs to the abstract domain ‘+’

Variable y:

* If y is negative (y < 0): it belongs to the abstract domain ‘-‘
* If y is zero (y == 0): it belongs to the abstract domain ‘0’
* If y is positive (y > 0): it belongs to the abstract domain ‘+’

Variable a:

* If both x and y are in abstract domain ‘-‘, a will be in ‘-‘
* If either x or y (or both) are in abstract domain ‘+’, a will be in ‘+’
* When x and y are both in abstract domain ‘0’, a will be in ‘0’
* When x is in ‘0’ or when y is in ‘0’, a will be in ‘0’

1. e = i | e \* e | e + e | e < e

μ: Exp 🡪 Int

μ(i) = i

μ(e1 \* e2) = μ(e1) x μ(e2)

μ(e1 + e2) = μ(e1) + μ(e2)

σ: Exp 🡪 {+,-,0}

σ(i) = (+ if i > 0

0 if i = 0

- if i < 0)

σ(e1 \* e2) = σ(e1) x̄ σ(e2)

σ(e1 + e2) = σ(e1) ∓ σ(e2)

|  |  |  |  |
| --- | --- | --- | --- |
| x̄ | + | 0 | - |
| + | + | 0 | - |
| 0 | 0 | 0 | 0 |
| - | - | 0 | + |

γ(T) = Int

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| ∓ | + | 0 | - | T |
| + | + | + | T | T |
| 0 | + | 0 | - | T |
| - | T | - | - | T |
| T | T | T | T | T |

μ: Exp 🡪 Bool

μ(e1 < e2) = 1 if μ(e1) < μ(e2)

μ(e1 < e2) = 0 if μ(e1) >= μ(e2)

σ: Exp 🡪 {+,-,0}

σ(e1 < e2) = (+ if = 1, 0 if = 0)

1. The results will depend on the input value of x and y in the concrete domain.

1: int func(int x, int y){

2: int a

μ(a) = 0

σ(a) = 0

3: if (x < 0){

If σ(x) = '-', then x < 0 (negative).

If σ(x) = '0', then x == 0 (zero).

If σ(x) = '+', then x > 0 (positive).

4: if (y < 0){

If σ(y) = '-', then y < 0 (negative).

If σ(y) = '0', then y == 0 (zero).

If σ(y) = '+', then y > 0 (positive).

5: a = x

μ(a) = μ(x)

σ(a) = σ(x)

6: a = a \* y

μ(a) = μ(a) x μ(y)

σ(a) = σ(a) x̄ σ(y)

7: a = a + 1

μ(a) = μ(a) + 1

σ(a) = σ(a) ∓ σ(1)

8: } else {

σ(y) = '+'

9: a = 2

μ(a) = 2

σ(a) = '+'

10: }

11: } else {

σ(x) = '+'

12: a = 2

μ(a) = 2

σ(a) = '+'

13: }

14: return a

μ(return) = μ(a)

σ(return) = σ(a)

1. Based on the analysis of the abstract domains and abstract semantics applied to the function, the property appears to hold for all possible inputs of x and y. The property characterizes how the sign of a is determined based on the signs of x and y, and it correctly accounts for various input scenarios.