MACHINE-ASSISTED CONCURRENT PROGRAMMING (EXERCISES FOR LECTURE 1)

Martin Vechev

ETH Zürich

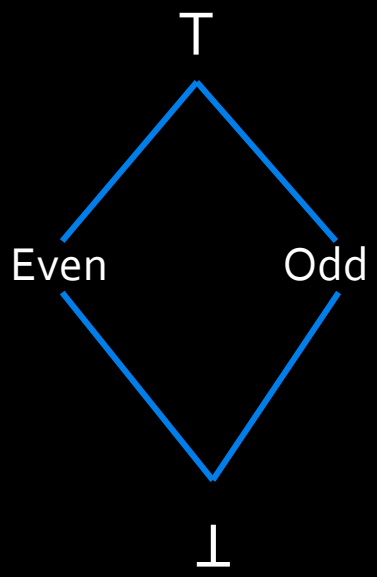
1. Suppose we have the Parity domain

- Even represents all even numbers including 0. Odd represents all other numbers. Iterate the example below (using the program structure, following each statement) on this domain and show the fixed point. Which of the assertions can you verify ?

```
foo (int i)

  1: int x :=5;
  2: int y :=7;

  3: if (i ≥ 0) {
  4:  y := y + 1;
  5:  i := i - 1;
  6:  goto 3;
   }
  7: assert 0 ≤ y + x;
     assert 0 ≤ y - x;
     assert 0 ≤ x - y;
```

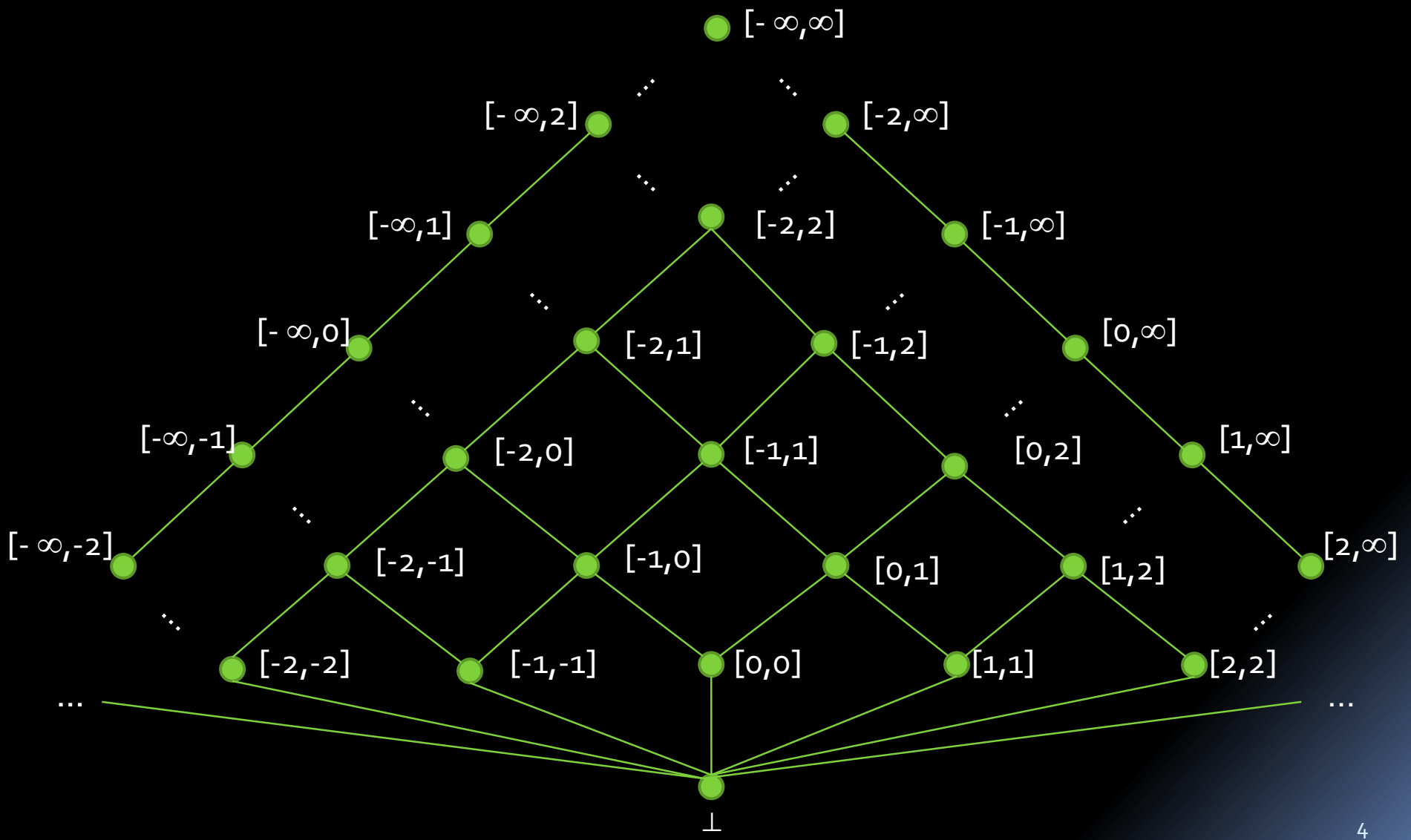


2. Which domain is more precise: Sign or Parity ? Are they comparable ? If no, show an example where it verifies with Parity but not Sign and vice versa. If yes, why ?

3. Why did we need state abstraction if we are using the Sign domain later ? Isn't the set of traces bounded with Sign ?

4. For the synthesis, can you find an example program that contains a loop such that only the first iteration needs to be repaired (be atomic), yet the fixing mechanism with atomic sections requires each iteration to be performed atomically ? You can consider the problem without abstraction (that is, in the concrete)

The interval domain



5. Using the interval domain, how would the algorithm from the lecture fix the program below ? List the trace(s) that you fixed. How many solutions are there ?

```
T1              ||   T2              ||   T3                    f(x)  {
                 ||                   ||                          if (x == 1) return 3
 1: x += z       ||    1: z++         ||    1: y1 = f(x)          else if (x == 2) return 6
 2: x += z       ||    2: z++         ||    2: y2 = x             else return 5
                 ||                   ||    3: assert(y1 != y2)   }
```