1 An Example of Construction and Usage of a Lattice of Guarded Literals

The code example in Fig. 1 describes the *greatest common divisor* (GCD) algorithm. We assume that both inputs are positive integers. The program is safe with respect to the assertion $g \le x$. However, with the LRA theory, an SMT solver cannot prove correctness of the program, as GCD is not expressible in LRA. The standard approach is to have $gcd(x, y)$ assume any real value; thus, attempting to verify this program with an SMT solver and the LRA theory results in an infinite number of spurious counterexamples.

```
int gcd(int x, int y)
{
  int tmp;
        while(y != 0) {
          tmp = x%y;
                x=y;
          y=tmp; }
        return x;
}
```

```
int main(void)
{
        int x=45;
        int y=18;
        int g = gcd(x,y);

        assert(g <= x);
}
```

Figure 1: *The GCD program using modulo function.*

In the example, we augment the solver with a set of *guarded literals* about the *modulo* function, arranged in a meet semilattice. These guarded literals are taken from an existing set of lemmas and theorems of the Coq proof assistant [1] for $a\%n$:

$$f_1 \equiv z\_mod\_mult \equiv$$

$$\equiv a \ mod \ n = 0 \text{ with the assumption } a == x * n \text{ for some positive integer } x;$$

$$f_2 \equiv z\_mod\_pos\_bound \land z\_mod\_unique \equiv$$

$$\equiv (0 \le a \ mod \ n < n) \land (0 \le r < n \implies a = n * q + r \implies r = a \ mod \ n)$$

$$\text{for some positive integers } r \text{ and } q, \text{ with the assumption } (n > 0) \land (a \ne x * n);$$

$$f_3 \equiv z\_mod\_remainder \land z\_mod\_unique\_full \equiv$$

$$\equiv (n \ne 0 \implies (0 \le a \ mod \ n < n \lor n < a \ mod \ n \le 0)) \land ((0 \le r < n \lor n < r \le 0)$$

$$\implies a = b * q + r \implies r = a \ mod \ n) \text{ with the assumption } \mathbf{true}.$$

The assumptions are different from the original guards in [1], as these are rewritten during the build of the meet semilattice. The original subset lattice consists of all subsets of the set $\{f_1, f_2, f_3\}$. It is analysed and reduced as described in Sec. **??** to remove contradicting guarded literals and equivalent elements. In this example, the set $\{f_3\}$ generalises $\{f_1\}\{f_2\}$. Fig. 2 shows the original subset lattice on the left, and the resulting meet semilattice of guarded literals on the right.

In the lattice traversal, we start from the bottom element $\emptyset$ and traverse the meet semilattice until we either prove that the program is safe or find a real counterexample (or show that a further theory refinement is needed). In this

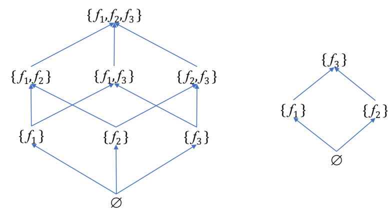

Figure 2: Original subset lattice of guarded literals and reduced meet semilattice for the *modulo* function in LRA.

example, we traverse the lattice until the element $\{f_3\}$, which is sufficient to prove that the program is safe. Specifically, the guarded literal f_1 is used to prove loop termination, and the guarded literal f_2 is used to prove the assert.

References

[1] The coq proof assistant. https://coq.inria.fr/