Learning Nonlinear Loop Invariants with Gated Continuous Logic Networks

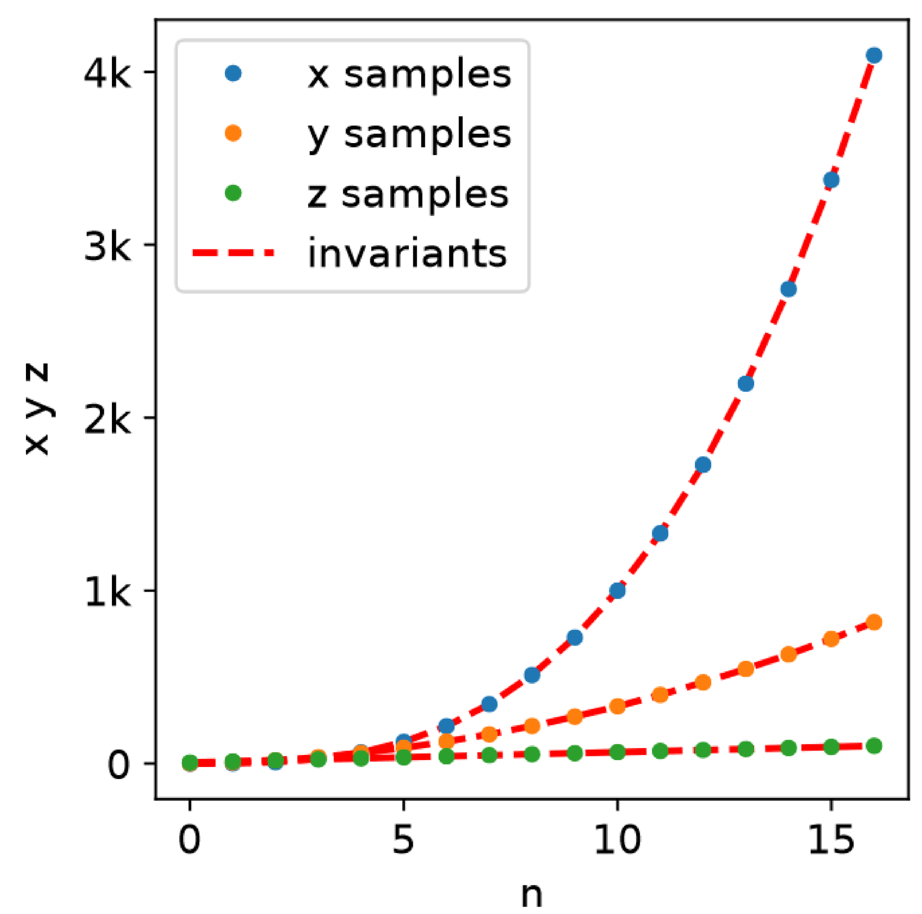
Jianan Yao*, Gabriel Ryan*, Justin Wong*, Suman Jana, Ronghui Gu
Columbia University

Background:
Loop invariant is critical for verification & requires intensive manual effort

Objective:
Automatically learn the invariant given a loop

Approach:
Learn the formula from execution traces

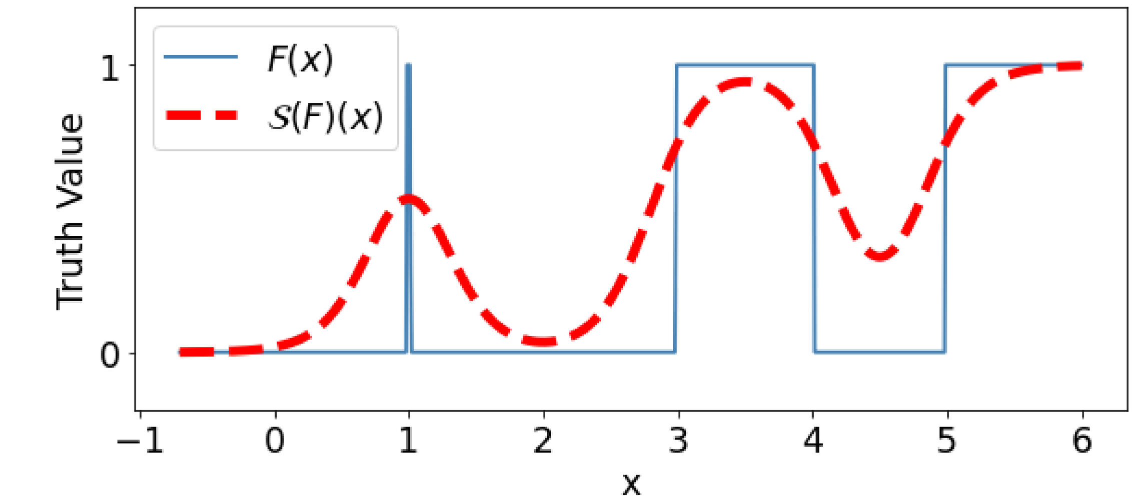
```
// pre: (a >= 0)
n=0; x=0;
y=1; z=6;
// compute cube:
while(n != a){
    n += 1;
    x += y;
    y += z;
    z += 6;
}
return x;
// post: x == a^3
```



Loop Invariant
`(x=n^3)&&(y=3n^2+3n+1)&&(z=6n+6)&&(n<=a)`

Continuous Logic Networks[1]

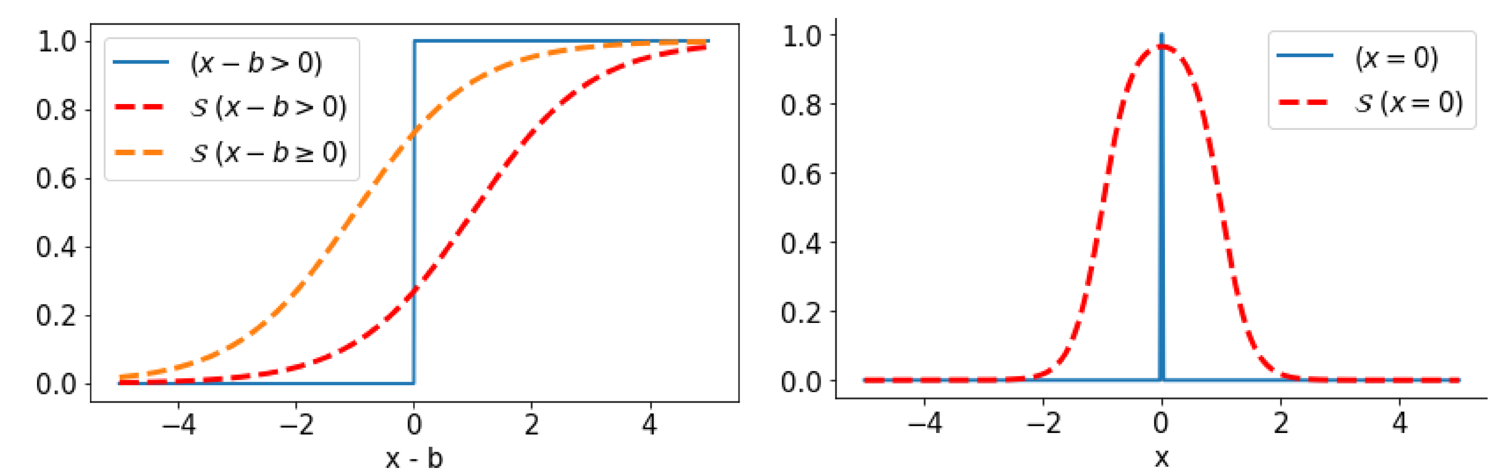
$$F(x) = (x = 1) \lor ((x > 3) \land (x < 4)) \lor (x > 5)$$



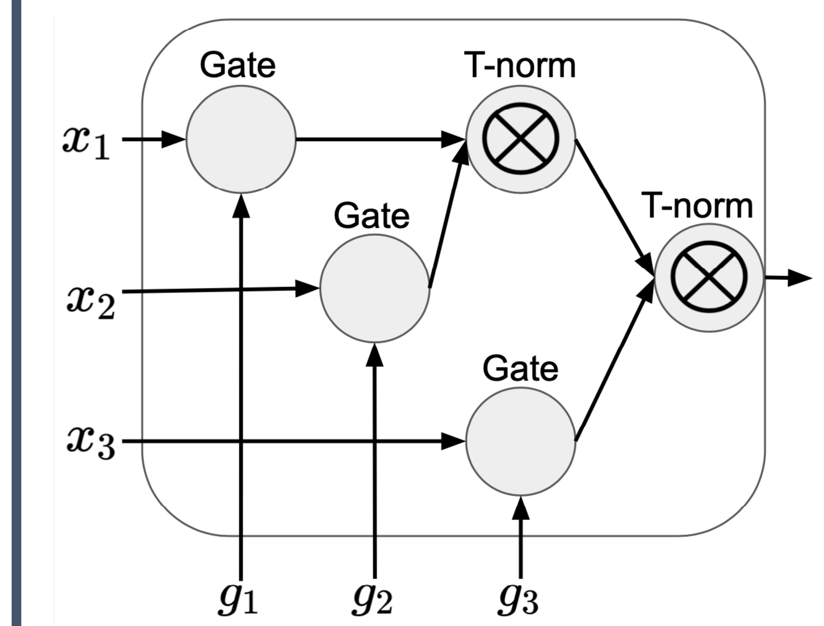
Parametric relaxation from discrete SMT formula to continuous and differentiable function.
Differentiable logical connectives (AND, OR):

$$p \land q \to p \otimes q = p * q$$
$$p \lor q \to p \oplus q = p + q - p * q$$

Differentiable predicates (>=, >, <=, <, =):



Gated T-norm

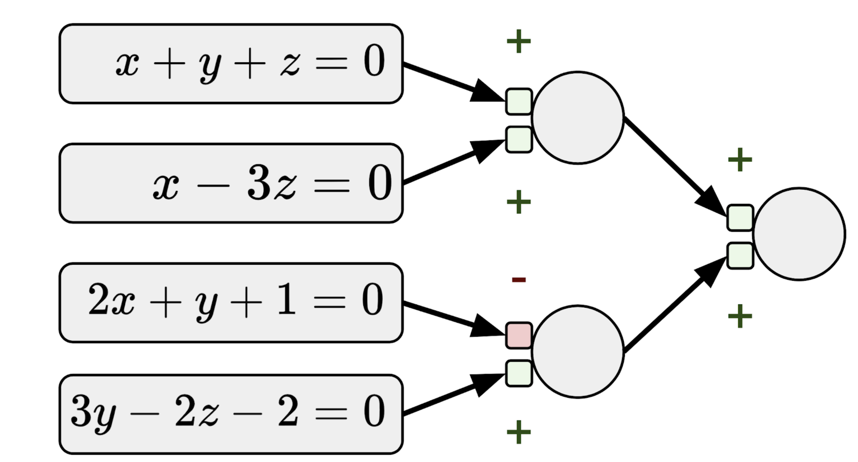


$T_G(x, y; g_1, g_2) =$
$(1 + g_1(x-1)) \otimes (1 + g_2(y-1))$

$x, y, g_1, g_2 \in [0, 1]$

x, y : gate input
g_1, g_2 : gate parameters

g_1, g_2 determine whether input x or y is activated

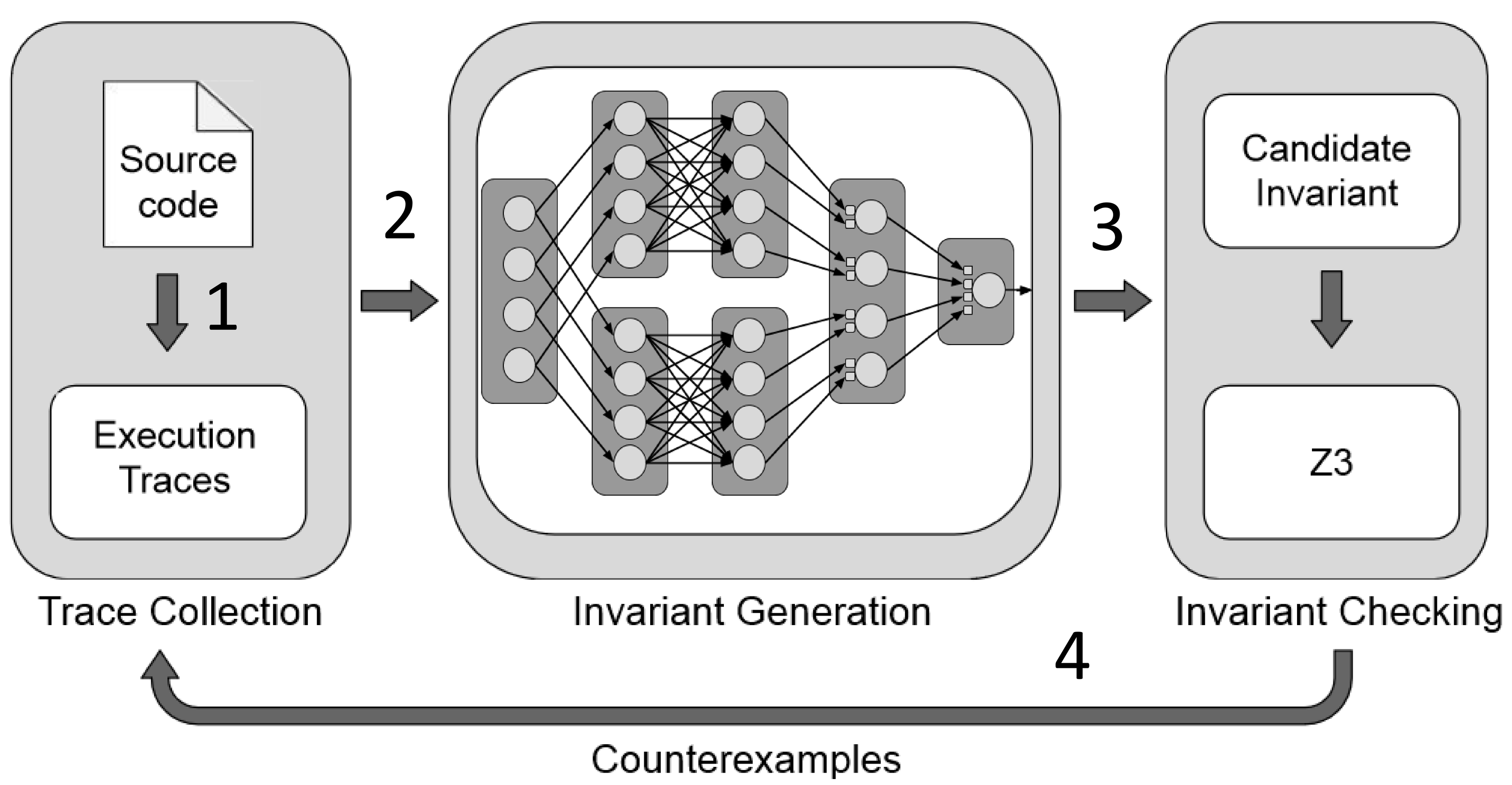


(+ indicates gating parameter g=1; - indicates g=0)

SMT formula extracted:

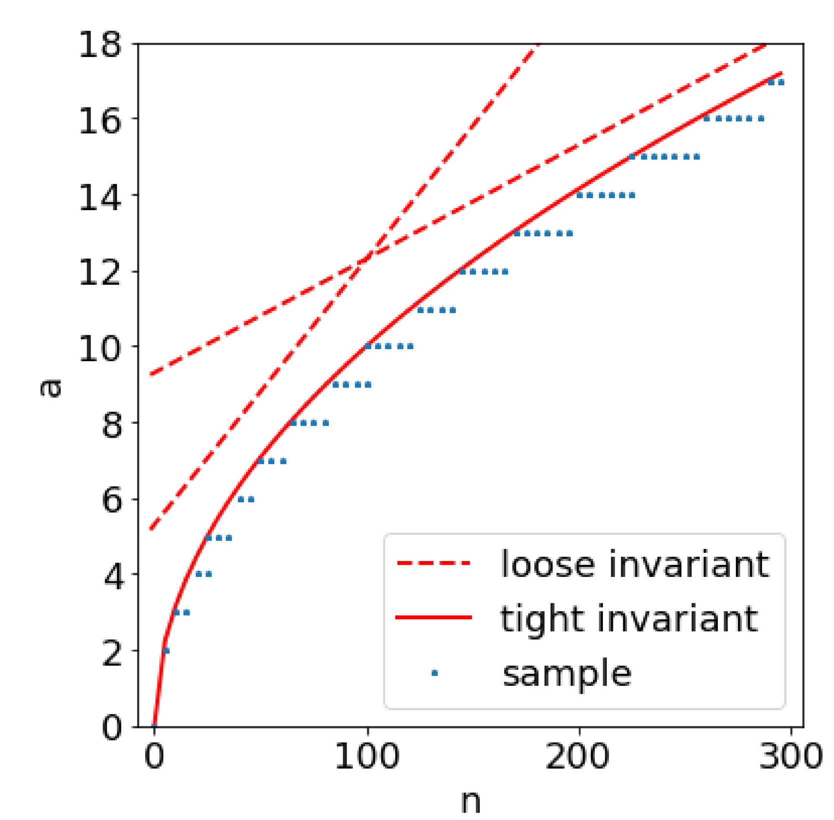
$$(3y - 3z - 2 = 0) \land ((x - 3z = 0) \lor (x + y + z = 0))$$

System Overview



1. Record execution traces from initial states satisfying the precondition.
2. Build and train a G-CLN model with the traces.
3. Extract the formula from G-CLN as the invariant.
4. If the invariant is rejected by SMT solver (Z3), enlarge sampling space and retrain G-CLN model.

Piecewise Biased Quadratic Unit

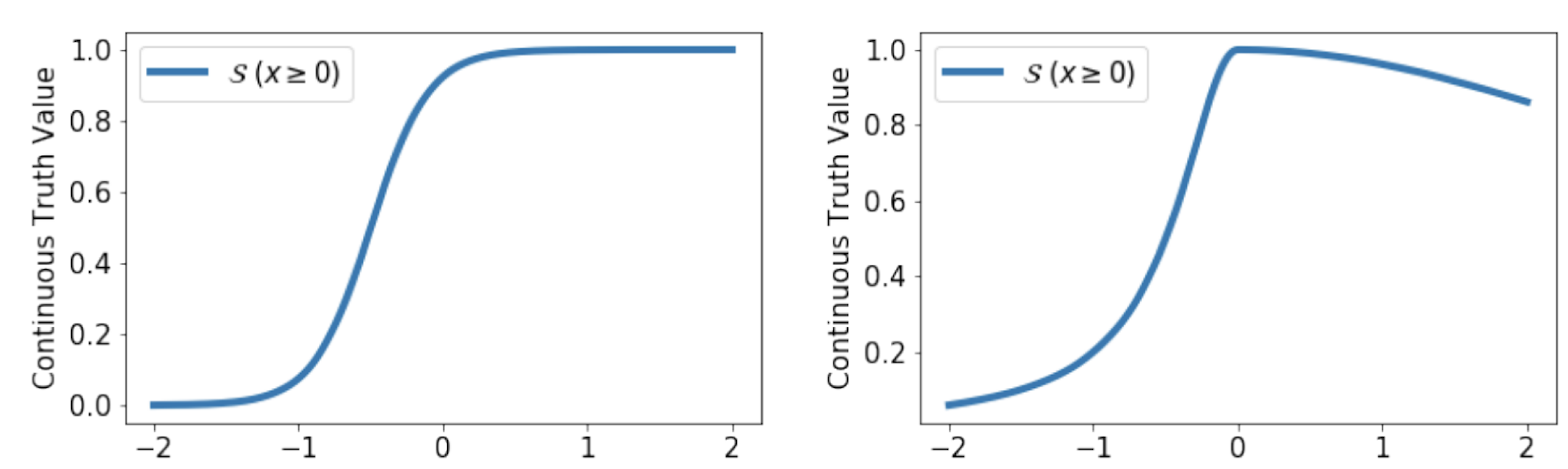


Blue dots: Execution traces of a loop calculating square root

Red solid line: Tight inequality invariant: $a^2 \le n$

Challenge: Sigmoid activation encourage loose bounds.

Solution: New activation with highest truth value at zero



Fractional Sampling

```
//pre: x = y = 0
//     /\ k >= 0
while (y < k) {
  y++;
  x += y * y * y;
}
//post: 4x == k^2
//      * (k + 1)^2
```

x	y	y^2	y^3	y^4
0	0	0	0	0
1	1	1	1	1
9	2	4	8	16
36	3	9	27	81
100	4	16	64	256
225	5	25	125	625

Challenge:
Low quality training samples x and y^4 too large and dominate model training.

Solution: Incorporate training samples from other initial states, by learning a more general loop invariant.

Loop invariant I: predicate over variables V initialized with V_0

Normal invariant: $\forall V,\ V_0 \mapsto^* V \implies I(V)$

Generalized invariant: $\forall V_I\ V,\ V_I \mapsto^* V \implies I'(V_I \cup V)$

Instantiated invariant: $\forall V,\ V_0 \mapsto^* V \implies I'(V_0 \cup V)$

[1] Ryan, Gabriel, et al. "CLN2INV: Learning Loop Invariants with Continuous Logic Networks." *ICLR 2020.*