Status meeting Oct-Dec

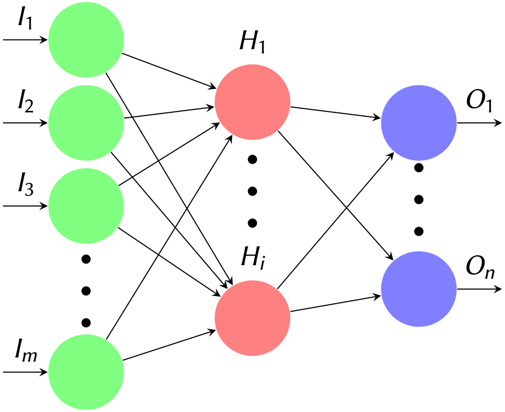
Mohammad Afzal

January 11, 2022

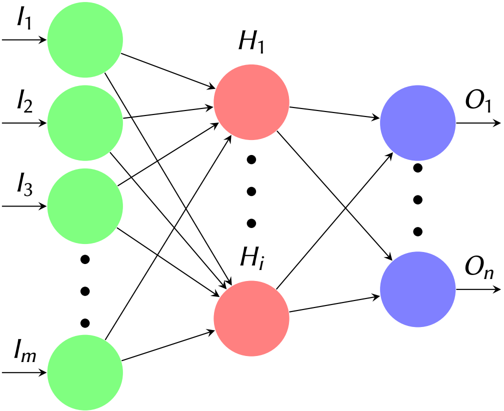
Introduction

- I am working on two problems:
  1. Quantitative learning of LTL-formulae.
     - Rejected from TACAS'22.
     - No progress after that.
  2. Formal verification of neural network.
     - Abstract refinement with progress guarantee.
     - Implementation is in progress.
- Attended SAT-SMT and iVerif workshops.

Formal Verification of Neural Network



Formal Verification of Neural Network



Neural Network
$f : \mathbb{R}^m \rightarrow \mathbb{R}^n$

Verification Problem:
$\{\phi(\overline{x})\}$ ... Pre-Cond
$\overline{y} \leftarrow f(\overline{x})$ ... NN
$\{\phi(\overline{y})\}$ ... Post-Cond

Figure: Neural network architecture

Example

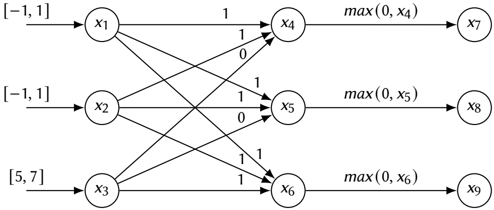


Figure: Hypothetical Example of Neural network

Pre-condition: $-1 \leq x_1 \leq 1 \wedge -1 \leq x_2 \leq 1 \wedge 5 \leq x_3 \leq 7$
Post-condition: $x_7 \leq x_8 \wedge x_7 \leq x_9$.

Challanges

Encoding of a neuron H_1 have ReLU activation:

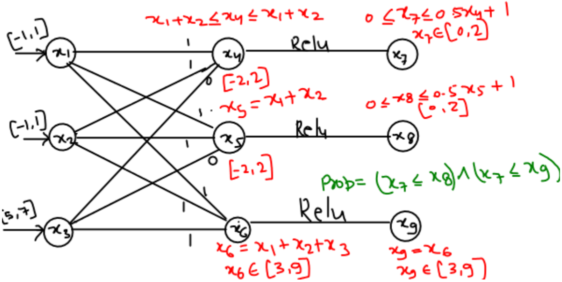
$$(z = w_1 * i_1 + w_2 * i_2 + ... + w_n * i_n) \wedge (y_{h_1} = max(0, z))$$

$$F \wedge ((y_{h_1} = z \wedge z > 0) \vee (y_{h_1} = 0 \wedge z \leq 0))$$

$$(F \wedge y_{h_1} = z \wedge z > 0) \vee (F \wedge y_{h_1} = 0 \wedge z \leq 0)$$

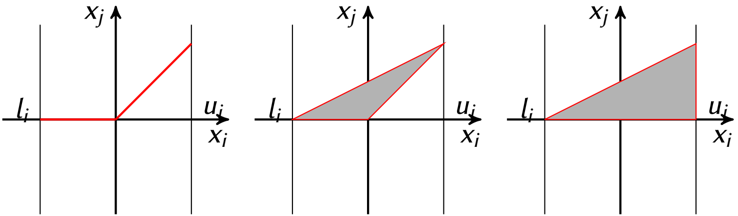
- Simplex call for each subfomula.
- Exponential simplex calls in terms of number of neuron.
- Researchers use abstraction based approach.

DeepPoly: which uses interval + polyhedral domain.



Relu Approximation

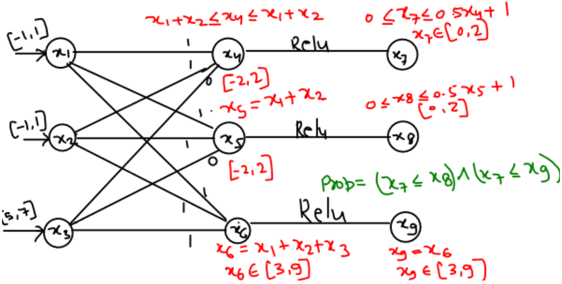
DeepPoly: Over-approximation of relu $x_j = max(0, x_i)$



(a) (b) (c)

Figure: (a) exact relu activation, (b) tightest approaximation, (c)DeepPoly's approaximation

DeepPoly: which uses interval + polyhedral domain.



Couses of imprecision:

- Triangle approximation.
- Analysing each neuron separately.

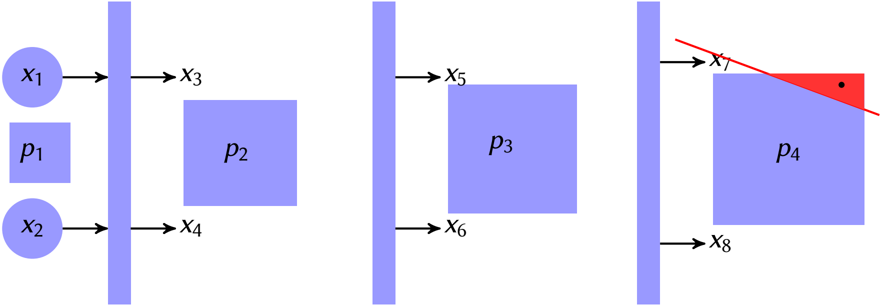
Our abstraction refinement process

Our abstraction refinement process has two part:

1. Find the causing point of spuriousness.
   - Backpropagation approach.
   - Optimization based approach.
2. How to utilize the above information(refinement).
   - MILP-based refinement.
   - Path-splitting based refinement.

Find the causing point of spuriousness

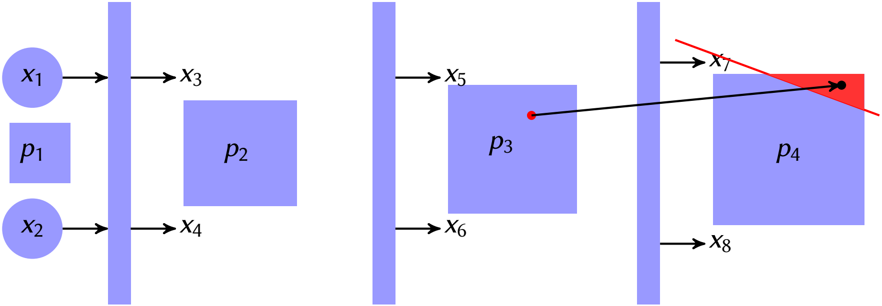
Backpropagation approach



Steps:

- $And(C, \neg prop)$, find the satisfying assignment.

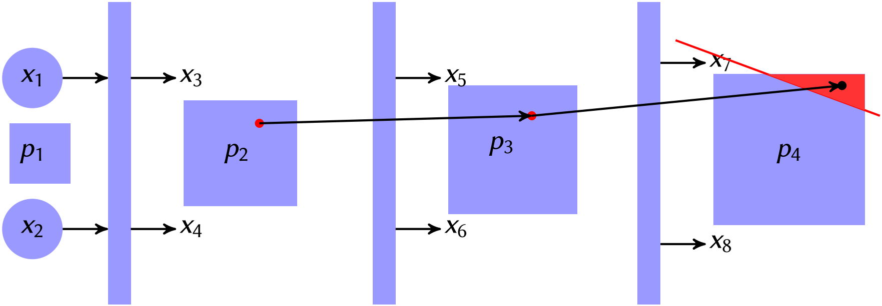
Backpropagation approach



Steps:

- $And(C, \neg prop)$, find the satisfying assignment.
- Find the corresponding point in the previous layer.

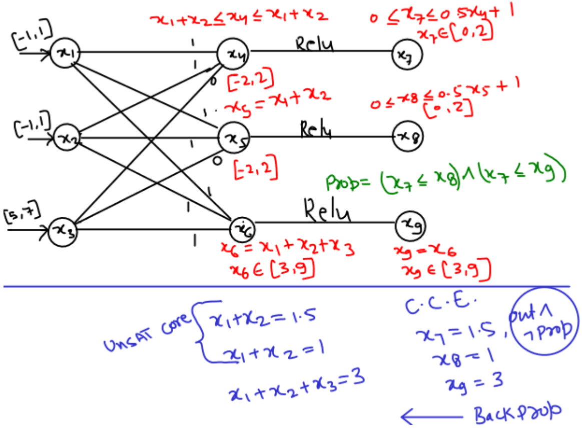
Backpropagation approach



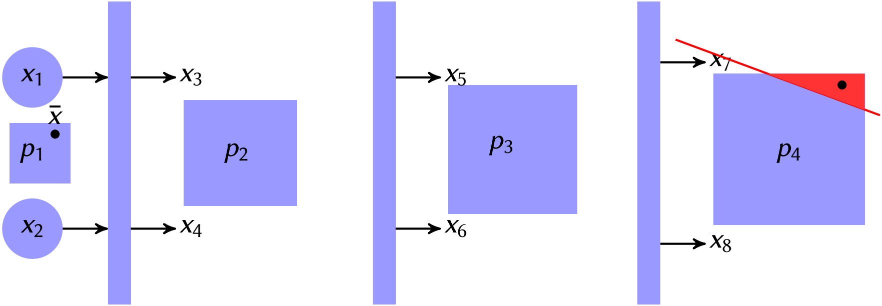
Steps:

- $And(C, \neg prop)$, find the satisfying assignment.
- Find the corresponding point in the previous layer.
- If stuck in some layer, find the causing neurons.

Example: Backpropagation approach



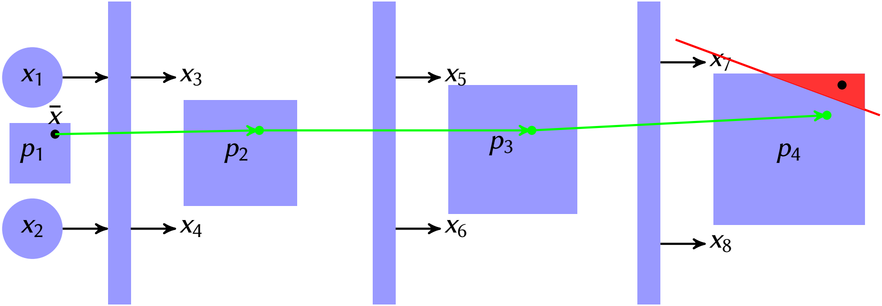
Optimization based approach



Steps:

- $And(C, \neg prop)$, find the satisfying assignment.

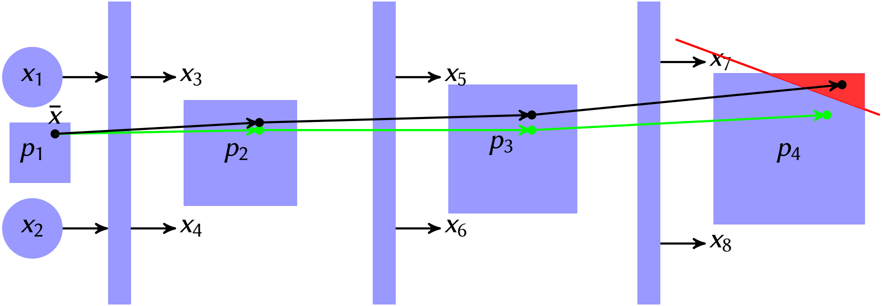
Optimization based approach



Steps:

- $And(C, \neg prop)$, find the satisfying assignment.
- Execute the neural network on $\bar{x}$

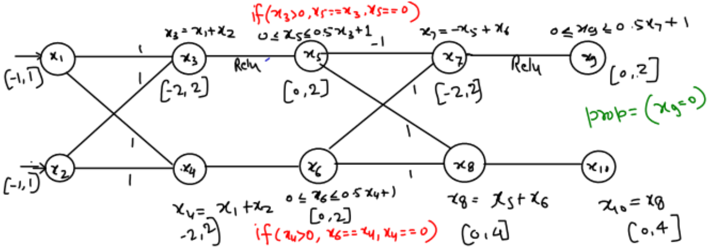
Optimization based approach

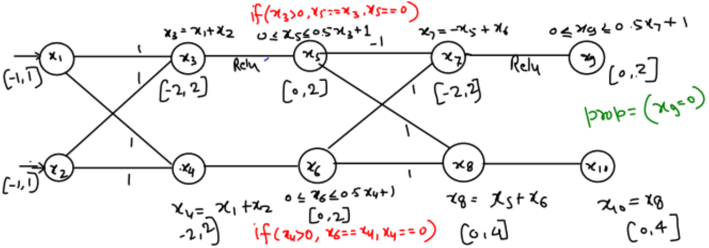


Steps:

- $And(C, \neg prop)$, find the satisfying assignment.
- Execute the neural network on $\bar{x}$
- Maximize the equality of neurons of black and green points.
- Mark neurons which have different values.

MILP-based refinement

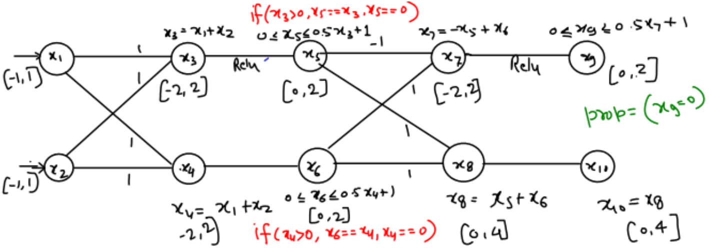




Iter-1:

- $x_9 = 2, x_{10} = 2$
- $x_1{+}x_2 = 2, -2 \le x_1{+}x_2 \le 0$

MILP-based refinement



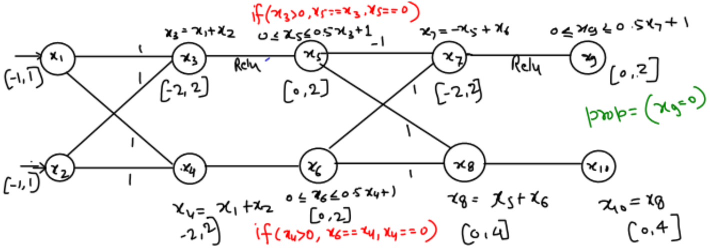
Iter-1:

- $x_9 = 2, x_{10} = 2$
- $x_1 + x_2 = 2, -2 \leq x_1 + x_2 \leq 0$

Iter-2:

- $x_9 = 1, x_{10} = 3$
- $x_1 + x_2 = 2, x_1 + x_2 = 1$

MILP-based refinement



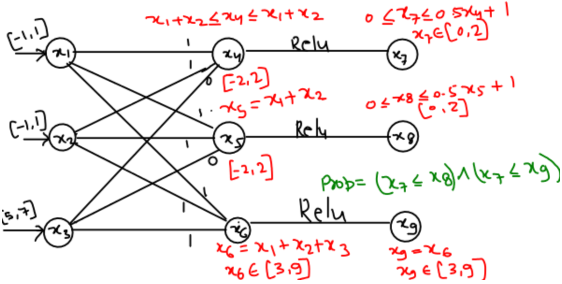
Iter-1:

- $x_9 = 2, x_{10} = 2$
- $x_1{+}x_2 = 2, -2 \leq x_1{+}x_2 \leq 0$

Iter-2:

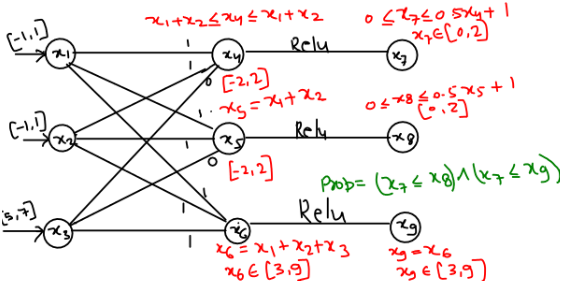
- $x_9 = 1, x_{10} = 3$
- $x_1 + x_2 = 2, x_1 + x_2 = 1$

Could not progress



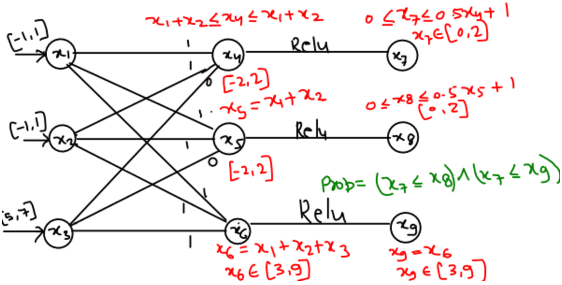
- Suppose marked neuron is x_4.

Path splitting based refinement



- Suppose marked neuron is x_4.
- We split at x_4, i.e. $x_4 > 0$ and $x_4 \le 0$, run DeepPoly on each copy.

Path splitting based refinement



- Suppose marked neuron is x_4.
- We split at x_4, i.e. $x_4 > 0$ and $x_4 \le 0$, run DeepPoly on each copy.
- if $x_4 > 0$, add constraint $x_1 + x_2 > 0$ on each node of the same layer.
- Run DeepPoly (Not exactly).

Implementation design

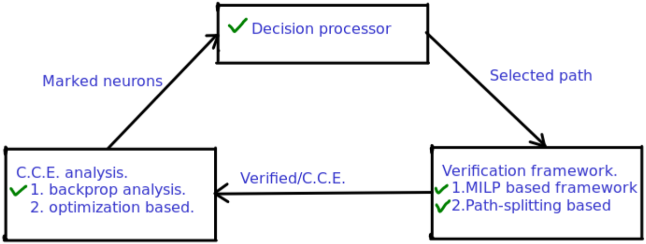


Figure: Tool overview

Plan for Jan-Mar

- Complete the implementation.
- Testing.
- Resume the LTL work.

Plan for Jan-Mar

- Complete the implementation.
- Testing.
- Resume the LTL work.

Questions/Suggestions?