Abstract Interpretation

Wei Le

September 14, 2023

Acknowledgement: this lecture used slides from Profs Alex Aiken, David Schmidt

Outline

- What is abstract intepretation?
- Abstract domain
- Galois connection
- Design an abstract interpretation system
- Certify neural networks: abstract interpretation for artificial intelligence (ai^2, ai4ai)

History: Patrick Cousot, Radhia Cousot 1977

- Abstract Interpretation: A Unified Lattice Model for Static Analysis of Programs by Construction or Approximation of Fixpoints, 1977
- Methods and Logics for Proving Programs, 1990
- Completeness in Abstract Interpretation, 1995
- Directions for Research in Approximate System Analysis, 1999
- Probabilistic Abstract Interpretation, 2012
- An abstract interpretation framework for termination, 2012
- Abstract interpretation: past, present and future, 2014

What is an abstract intepretation?

Abstract interpretation: interpret using abstract values

Purpose: using abstract interpretation to prove program property. Here are the steps:

1. Create an abstract domain and the mapping from concrete domain to abstract domain: e.g., abstract domain — positive int, 0, negative int
2. Create abstract semantics: how each type of statements perform computation on abstract domain, e.g., how to compute + among positive int, 0 and negative int
3. Use abstract semantics to perform computations on abstract domain for the program to get the *abstract value*
4. We design abstract interpretation in such a way that the abstract value computed is the property we want to prove in the concrete system

What is an abstract interpretation

An *abstract interpretation* consists of:

- An abstract domain A ($+$,-,0) and concrete domain D (Int)
- Concretization γ and abstraction functions σ, forming a *Galois connection*
- (sound) abstract semantic function (s)

What is Abstract Intepretation?

- A theoretical framework to formalize *approximation*
- A sound approximation: the conclusion proved in the abstract domain will be held in the conrete domain
- Abstract intepretation can lose information, meaning some conclusions that can be reached by the concrete executions but cannot be reached by abstract intepretation

An Example

See Prof. Alex Aiken's slide

Abstract Domain

- Partition: how to partition and create abstract sets of concrete inputs (D)
- Abstract domain construction: there are many ways of partition. what are the properties about D that I wish to calculate?
- Using math formula to specify constraints on input X, Y, Z ..
  - Interval domain: upper and lower bound
  - Congruence domain: measure density of its values

- Intervals (nonrelational):

  $x \Rightarrow [a, b],\ y \Rightarrow [a', b'], \ldots$

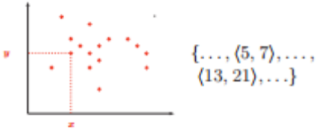
- Polyhedra (relational):

  $x + y - 2z \leq 10, \ldots$

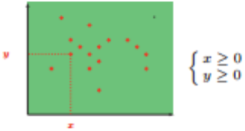
- Difference-bound matrices (weakly relational):

  $y - x \leq 5,\ z - y \leq 10, \ldots$

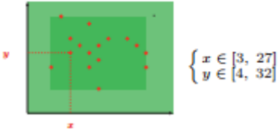
Abstract Domain



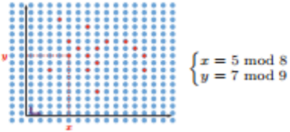
(a) [In]finite Set of Points



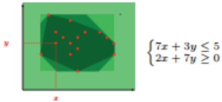
(b) Sign Abstraction



(c) Interval Abstraction

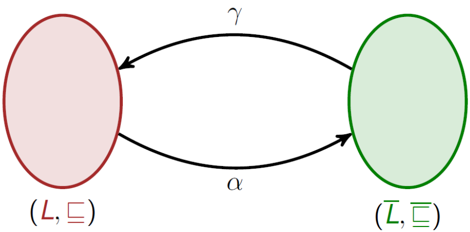


(d) Simple Congruence Abstraction



(b) Polyhedral Abstraction

Galois Connection: intuition



Concretization

γ is the concretization function.

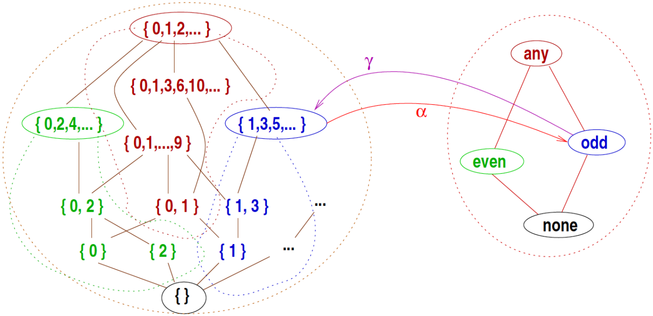
$\gamma(\overline{y})$ is the concrete value in L that is represented by $\overline{y}$.

Abstraction

α is the abstraction function.

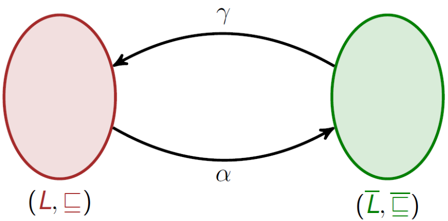
$\alpha(x)$ is the most precise abstract value in $\overline{L}$ whose concretization approximates x.

Galois connection: example



Element: potential values we know about the program.

Galois Connection: Definition



Definition

A Galois connection between a lattice $(L, \sqsubseteq)$ and a lattice $(\overline{L}, \overline{\sqsubseteq})$ is a pair of functions (α, γ), with $\alpha : L \to \overline{L}$ and $\gamma : \overline{L} \to L$, satisfying:

$$\alpha(x) \mathrel{\overline{\sqsubseteq}} \overline{y} \quad \text{iff} \quad x \sqsubseteq \gamma(\overline{y}) \qquad \text{(for all } x \in L, \overline{y} \in \overline{L})$$

Notation for Galois connections: $(L, \sqsubseteq) \underset{\alpha}{\overset{\gamma}{\leftrightarrows}} (\overline{L}, \overline{\sqsubseteq})$

The order is preserved. You do not lose too much information during approximation

Designing an abstract interpretation system

Property: what is the parity of succ(n):

Example: We have concrete domain, Nat, and concrete operation, $\mathrm{succ} : \mathrm{Nat} \to \mathrm{Nat}$, defined as $\mathrm{succ}(n) = n + 1$.

We have abstract domain, Parity, and abstract operation, $\mathrm{succ}^{\#} : \mathrm{Parity} \to \mathrm{Parity}$, defined as

$$\mathrm{succ}^{\#}(\mathrm{even}) = \mathrm{odd}, \quad \mathrm{succ}^{\#}(\mathrm{odd}) = \mathrm{even}$$

$$\mathrm{succ}^{\#}(\mathrm{any}) = \mathrm{any}, \quad \mathrm{succ}^{\#}(\mathrm{none}) = \mathrm{none}$$

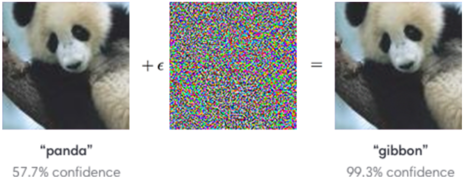
$\mathrm{succ}^{\#}$ must be consistent (sound) with respect to succ:

if $n \,\mathcal{R}_{\mathrm{Nat}}\, a$, then $\mathrm{succ}(n) \,\mathcal{R}_{\mathrm{Nat}}\, \mathrm{succ}^{\#}(a)$

where $\mathcal{R} \subseteq \mathrm{Nat} \times \mathrm{Parity}$ relates numbers to their parities (e.g., $2 \,\mathcal{R}_{\mathrm{Nat}}\, \mathrm{even}$, $5 \,\mathcal{R}_{\mathrm{Nat}}\, \mathrm{odd}$, etc.).

Abstract intepretation for robust neural networks (optional)

What does it mean to prove the robustness of a neural network?



Attack	Original	Perturbed	Diff
FGSM [12], $\epsilon = 0.3$			
Brightening, $\delta = 0.085$			

Fig. 1: Attacks applied to MNIST images [25].

Abstract intepretation for robust neural networks

Why can we use abstract intepretation?

- Deep Neural Nets: Concrete semantics —- Affine transforms + Restricted nonlinearity
- Abstract Interpretation: Scalable and precise numerical domains
- Abstract semantics should be defined on: Affine transformation (multiplictaion and addition), ReLu

Abstract intepretation for robust neural networks

What is the abstract domain?

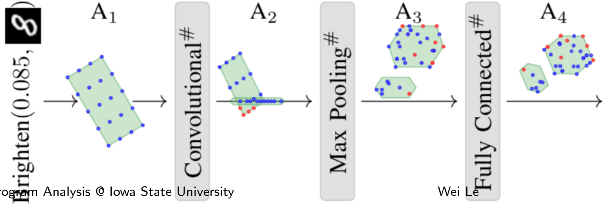
- Abstract domain: shapes expressible as a set of logical constraints
- Zonotope: a center-symmetric convex closed polyhedron [CAV09]

Abstract intepretation for robust neural networks

High level ideas:

- abstract element: A1 is an abstract element (represent a group of inputs) that captured all perbuted inputs
- abstract layer: process abstract element
- abstract transformer: design abstract semantics for each concrete transformation available in the neural network
- A4 is an overappoximation computed from A1
- verify A4 will generate the same classification

* In particular, we can capture the entire set of brightening perturbations exactly with a single zonotope. However, in general, this step may result in an abstract element that contains additional inputs (that is, red points).



Abstract intepretation for robust neural networks

Further reading:
AI2: Safety and Robustness Certification of Neural Networks with Abstract Interpretation