Static Analysis Methods for Neural Networks

Dagstuhl Seminar 25061 “Logic and Neural Networks”

Caterina Urban
Inria & École Normale Supérieure | Université PSL

Static Analysis Methods for Neural Networks

= Neural Network-Based Air Transportation Software

Runway Excursions during Landing

~20% of Air Transportation Accidents*

Jacksonville, Florida, USA (May 3rd, 2019)

https://www.flickr.com/photos/ntsb/46857358255

Montpellier, France (September 23rd, 2022)

https://x.com/BEA_Aero/status/1573588715552866305

*https://www.airbus.com/en/newsroom/stories/2022-10-safety-innovation-5-runway-overrun-prevention-system-rops-and-runway

Runway Excursions during Landing

~20% of Air Transportation Accidents*

Jeju Air Crash (December 29th, 2024)

https://www.newsweek.com/

Wikipedia
The Free Encyclopedia

Wikipedia

Jeju Air Flight 2216

Jeju Air Flight 2216 was a scheduled international passenger flight operated by Jeju Air from Suvarnabhumi Airport in Bangkok, Thailand, to Muan International Airport in Muan County, South Korea. On 29 December 2024, the Boeing 737-800 operating the flight was approaching Muan, when a bird strike occurred. The pilots issued a mayday alert, performed a go-around, and on the second landing attempt, the landing gear did not deploy and the airplane belly landed well beyond the normal touchdown zone. It overran the runway and crashed into a berm encasing a concrete structure that supported an antenna array for the instrument landing system.

Jeju Air Flight 2216

HL8088, the aircraft involved in the accident, pictured in 2023

Accident

Date	29 December 2024

*https://www.airbus.com/en/newsroom/stories/2022-10-safety-innovation-5-runway-overrun-prevention-system-rops-and-runway

Regulation (EU) 2020/1159

August 5th, 2020

L 257/14 EN Official Journal of the European Union 6.8.2020

COMMISSION IMPLEMENTING REGULATION (EU) 2020/1159

of 5 August 2020

amending Regulations (EU) No 1321/2014 and (EU) No 2015/640 as regards the introduction of new additional airworthiness requirements

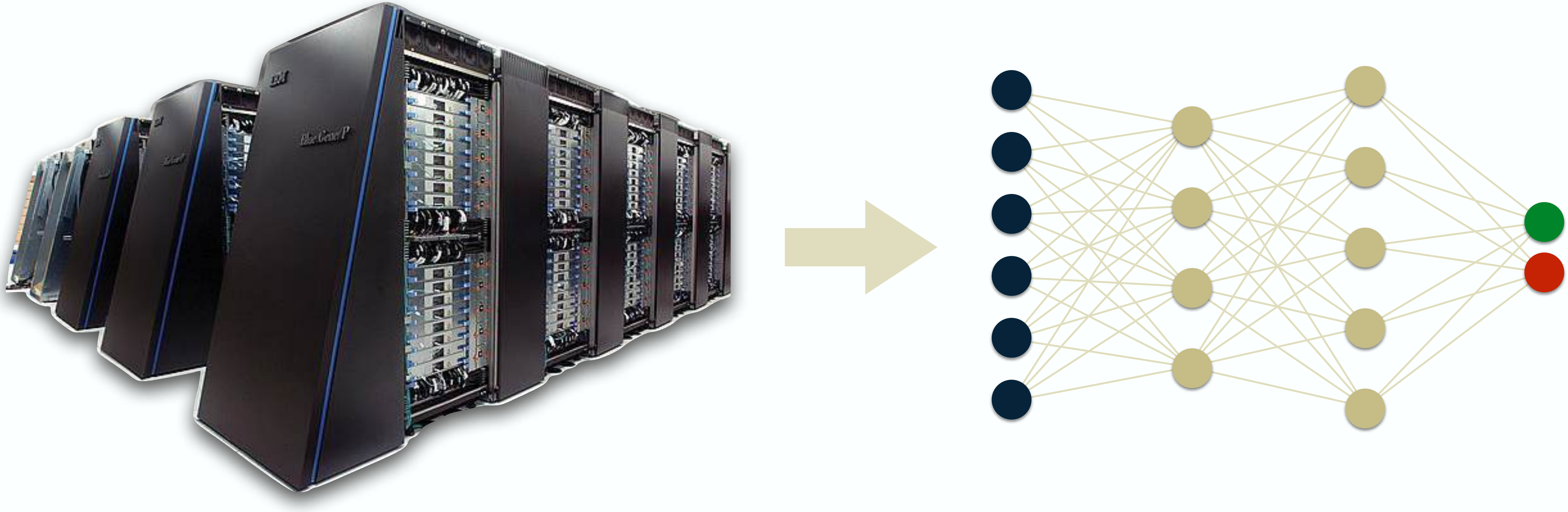
'26.205 Runway overrun awareness and alerting systems

(a) Operators of large aeroplanes used in commercial air transport shall ensure that every aeroplane for which the first individual certificate of airworthiness was issued on or after 1 January 2025, is equipped with a runway overrun awareness and alerting system.

Having regard to Regulation (EU) 2018/1139 of the European Parliament and of the Council of 4 July 2018 on common rules in the field of civil aviation and establishing a European Union Aviation Safety Agency, and amending Regulations (EC) No 2111/2005, (EC) No 1008/2008, (EU) No 996/2010, (EU) No 376/2014 and Directives 2014/30/EU and 2014/53/EU of the European Parliament and of the Council, and repealing Regulations (EC) No 552/2004 and (EC) No 216/2008 of the European Parliament and of the Council and Council Regulation (EEC) No 3922/91 ([1]), and in

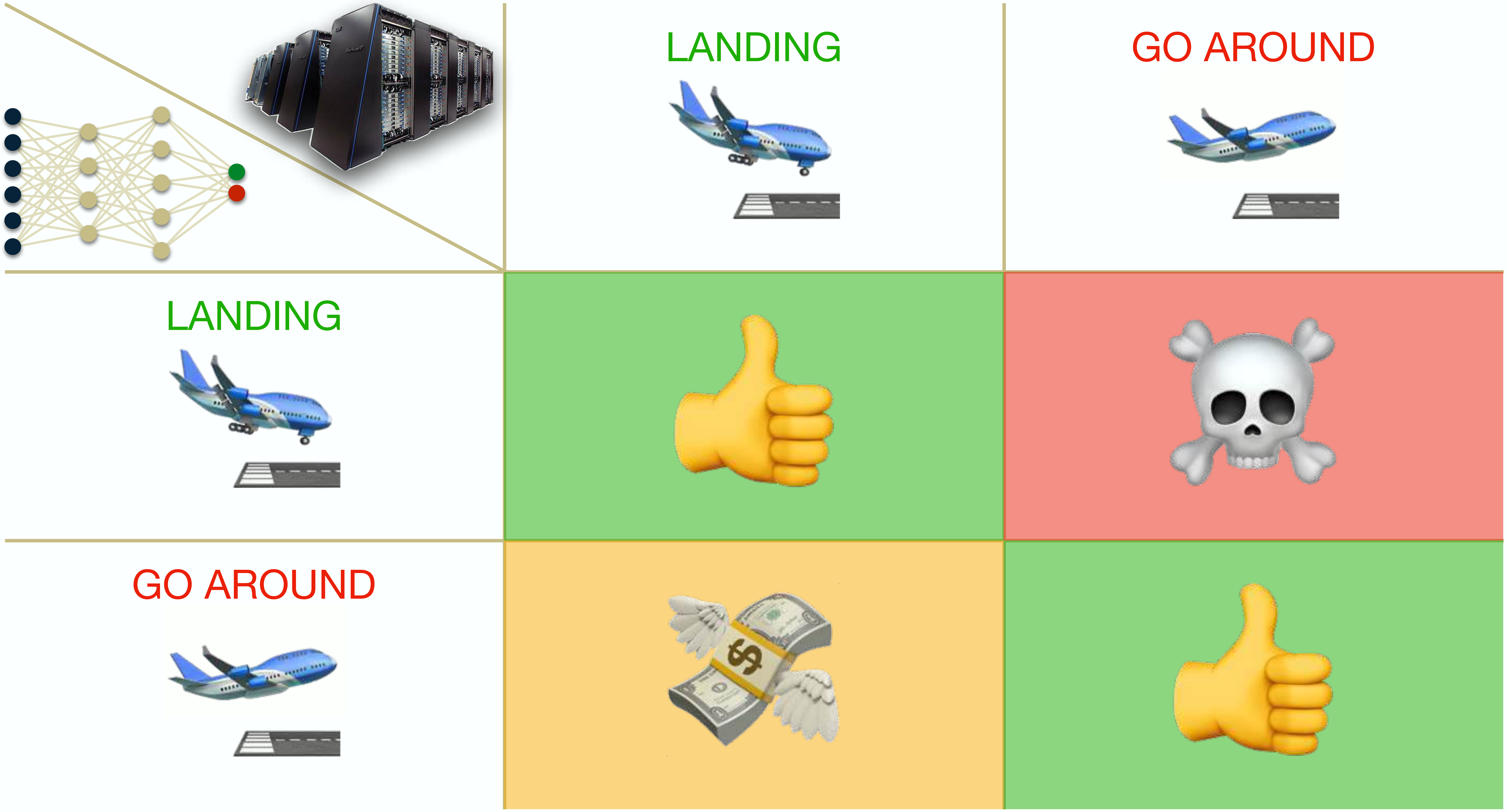
Neural Network Surrogates

Less Computing Power and Less Computing Time



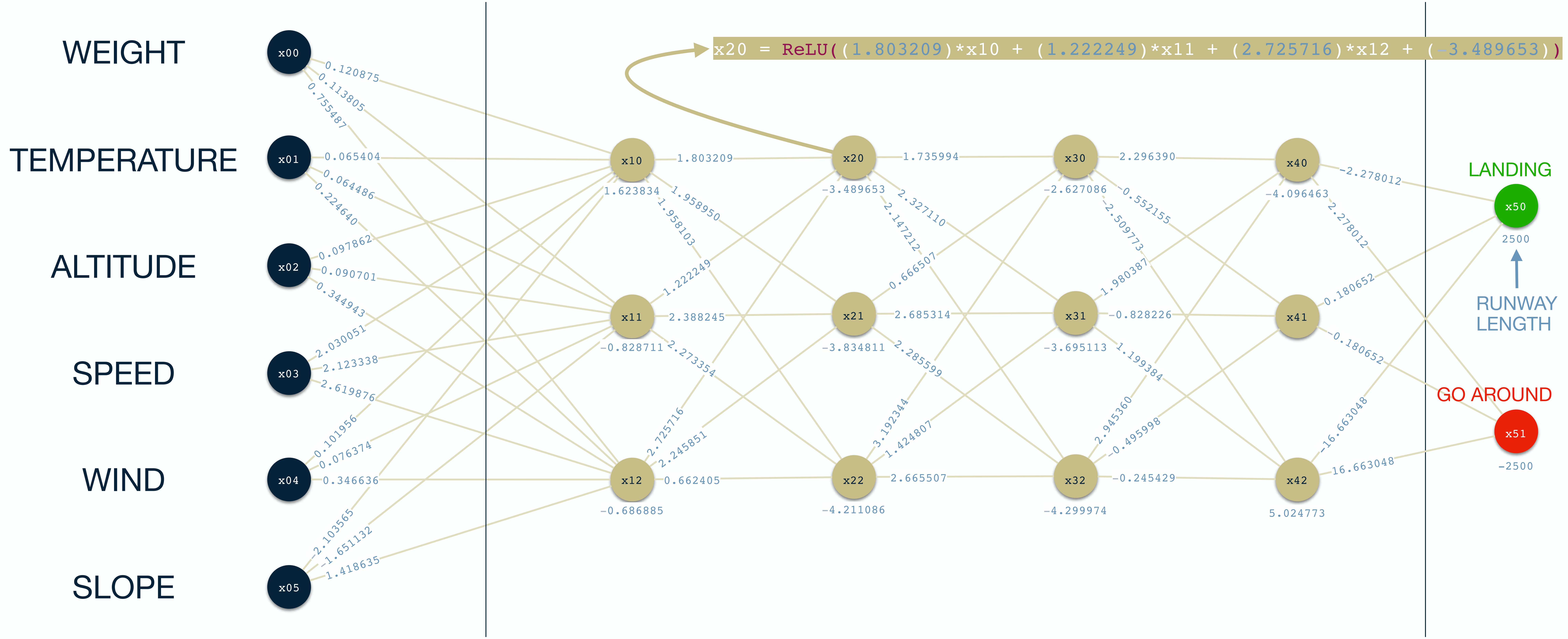
Runway Overrun Warning

Safety of Neural Network Surrogate



Runway Overrun Warning

Toy Example



Runway Overrun Warning

Toy Example

```
x00 = float(input())
x01 = float(input())
x02 = float(input())
x03 = float(input())
x04 = float(input())
x05 = float(input())

x10 = ReLU((0.120875)*x00 + (0.065404)*x01 + (0.097862)*x02 + (2.030051)*x03 + (0.101956)*x04 + (-2.103565)*x05 + (1.623834))
x11 = ReLU((0.113805)*x00 + (0.064486)*x01 + (0.090701)*x02 + (2.123338)*x03 + (0.076374)*x04 + (-1.651132)*x05 + (-0.828711))
x12 = ReLU((0.755487)*x00 + (0.224640)*x01 + (0.344943)*x02 + (2.619876)*x03 + (0.346636)*x04 + (1.418635)*x05 + (-0.686885))

x20 = ReLU((1.803209)*x10 + (1.222249)*x11 + (2.725716)*x12 + (-3.489653))
x21 = ReLU((1.958950)*x10 + (2.388245)*x11 + (2.245851)*x12 + (-3.834811))
x22 = ReLU((1.958103)*x10 + (2.273354)*x11 + (0.662405)*x12 + (-4.211086))

x30 = ReLU((1.735994)*x20 + (0.666507)*x21 + (3.192344)*x22 + (-2.627086))
x31 = ReLU((2.327110)*x20 + (2.685314)*x21 + (1.424807)*x22 + (-3.695113))
x32 = ReLU((2.147212)*x20 + (2.285599)*x21 + (2.665507)*x22 + (-4.299974))

x40 = ReLU((2.296390)*x30 + (1.980387)*x31 + (2.945360)*x32 + (-4.096463))
x41 = ReLU((-0.552155)*x30 + (-0.828226)*x31 + (-0.495998)*x32)
x42 = ReLU((-2.509773)*x30 + (1.199384)*x31 + (-0.245429)*x32 + (5.024773))

x50 = (-2.278012)*x40 + (0.180652)*x41 + (-16.663048)*x42 + (2500)
x51 = (2.278012)*x40 + (-0.180652)*x41 + (16.663048)*x42 + (-2500)
```

Neural Network Verification

Neural Network Explainability

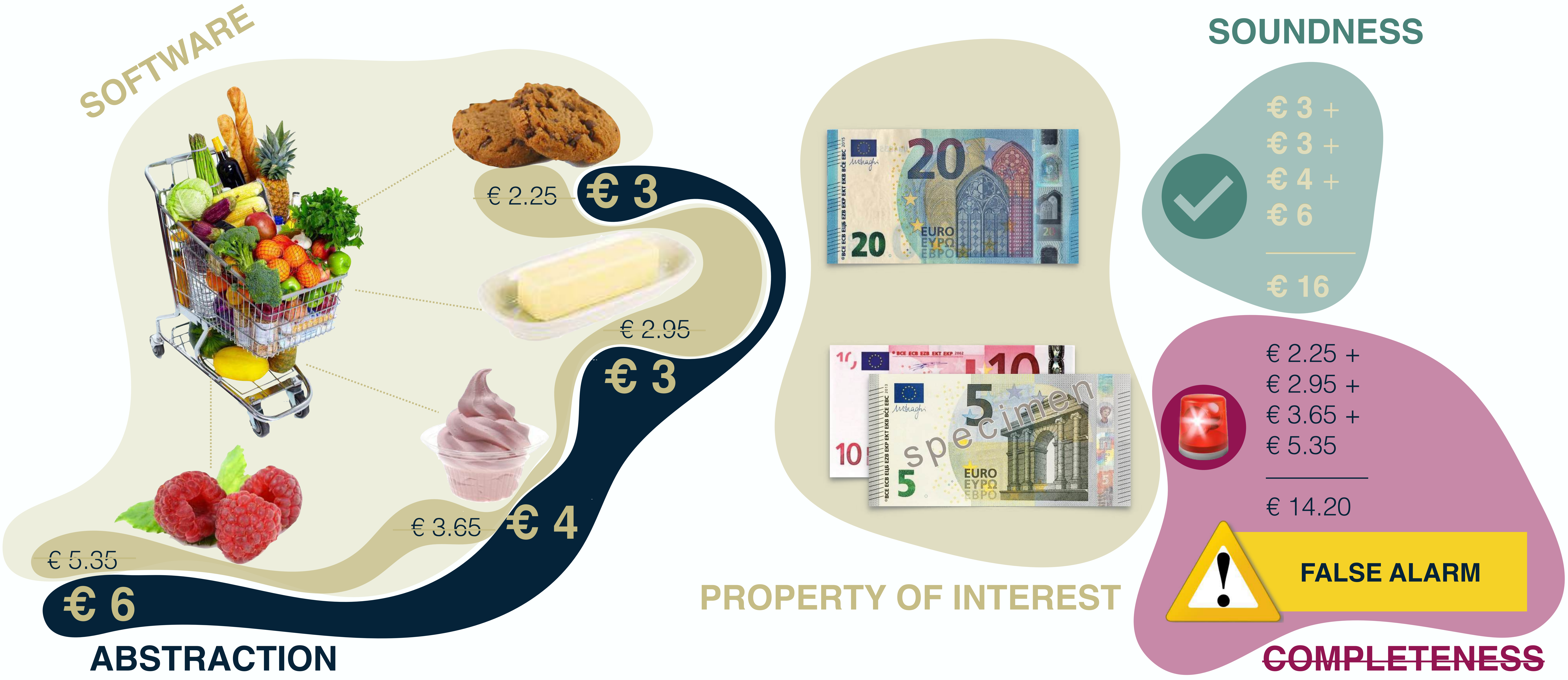
Neural Network Verification

Neural Network Explainability

Static Analysis Methods for Neural Networks

= Abstract Interpretation-Based Static Analysis

Abstract Interpretation

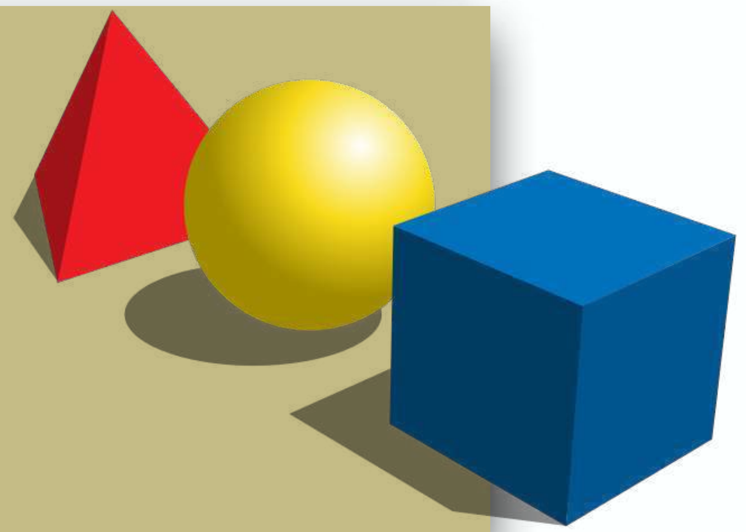


Abstract Interpretation

3-Step Recipe

practical tools
targeting specific programs

abstract semantics, abstract domains
algorithmic approaches to decide program properties



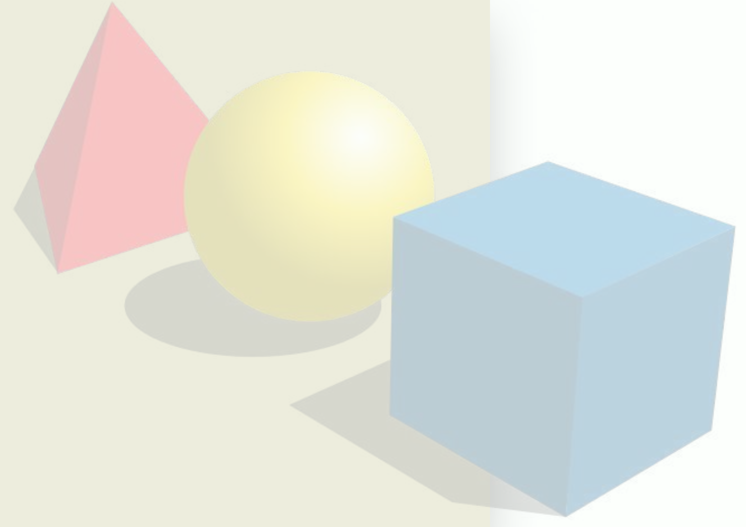
concrete semantics
mathematical models of the program behavior

Abstract Interpretation

3-Step Recipe

practical tools
targeting specific programs

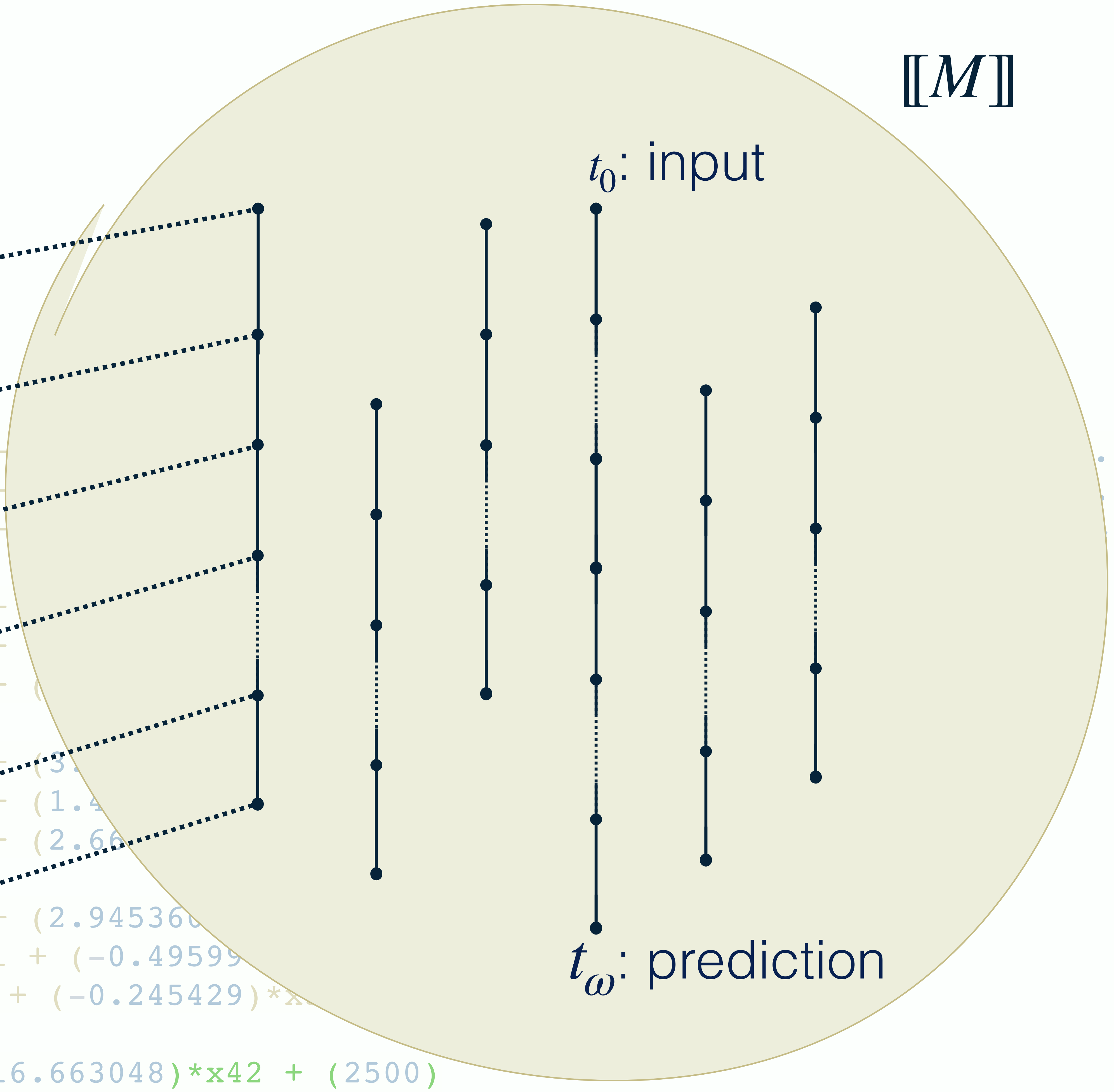
abstract semantics, abstract domains
algorithmic approaches to decide program properties



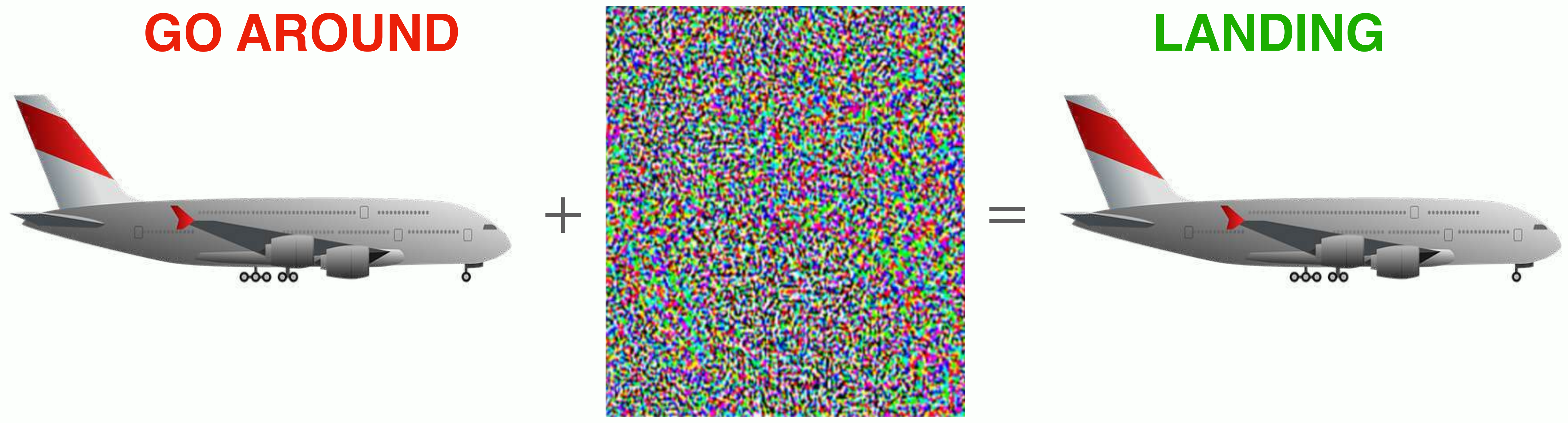
concrete semantics
mathematical models of the program behavior

Trace Semantics

```
x00 = float(input())
x01 = float(input())
x02 = float(input())
x03 = float(input())
x04 = float(input())
x05 = float(input())

x10 = ReLU((0.120875)*x00 + (0.065404)*x01 + … .103565)*x05 + (1.623834))
x11 = ReLU((0.113805)*x00 + (0.064486)*x01 + … .651132)*x05 + (-0.828711))
x12 = ReLU((0.755487)*x00 + (0.224640)*x01 + … 18635)*x05 + (-0.686885))

x20 = ReLU((1.803209)*x10 + (1.222249)*x11 + …
x21 = ReLU((1.958950)*x10 + (2.388245)*x11 + …
x22 = ReLU((1.958103)*x10 + (2.273354)*x11 + (…

x30 = ReLU((1.735994)*x20 + (0.666507)*x21 + (3.…
x31 = ReLU((2.327110)*x20 + (2.685314)*x21 + (1.4…
x32 = ReLU((2.147212)*x20 + (2.205599)*x21 + (2.66…

x40 = ReLU((2.296390)*x30 + (1.980387)*x31 + (2.94536…
x41 = ReLU((-0.552155)*x30 + (-0.828226)*x31 + (-0.49599…
x42 = ReLU((-2.509773)*x30 + (1.199384)*x31 + (-0.245429)*x…

x50 = (-2.278012)*x40 + (0.180652)*x41 + (-16.663048)*x42 + (2500)
x51 = (2.278012)*x40 + (-0.180652)*x41 + (16.663048)*x42 + (-2500)
```

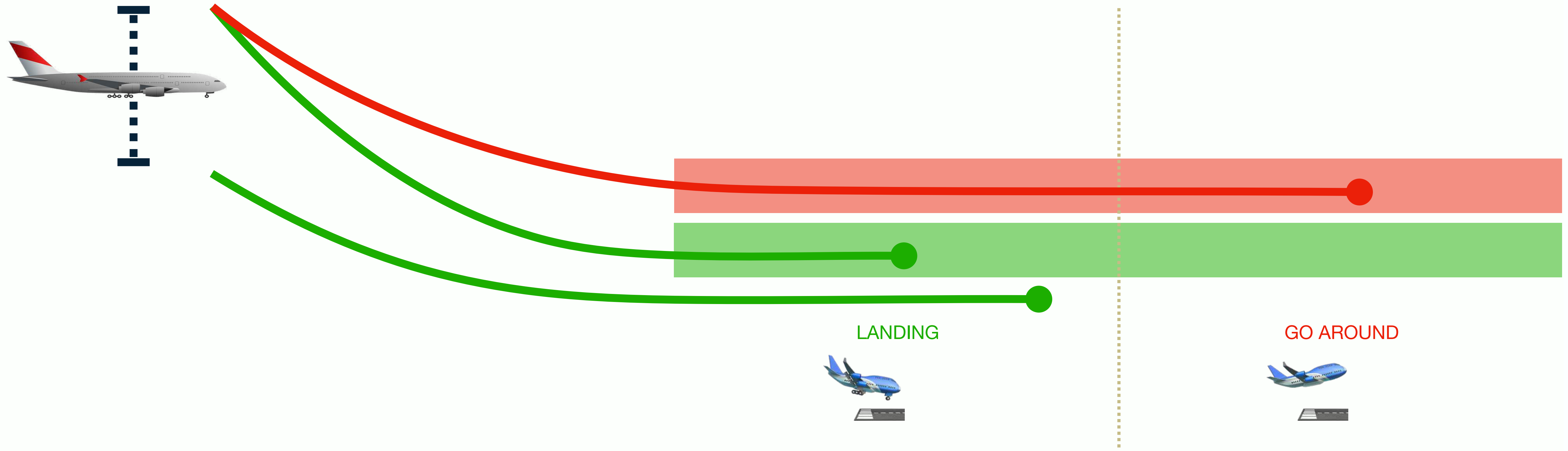


Robustness

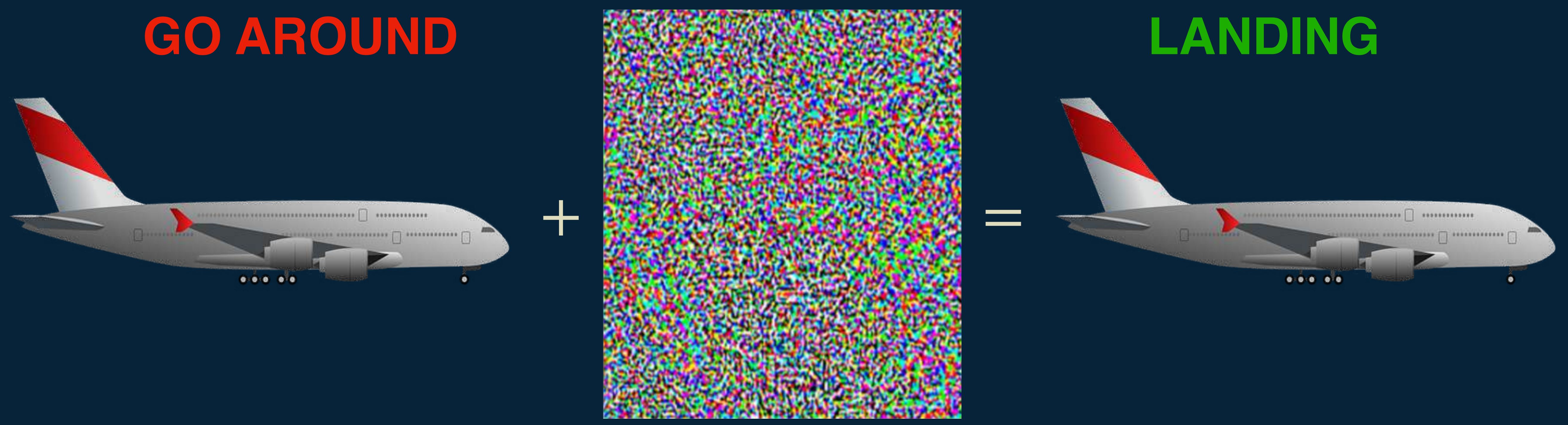


Safety

Hypersafety

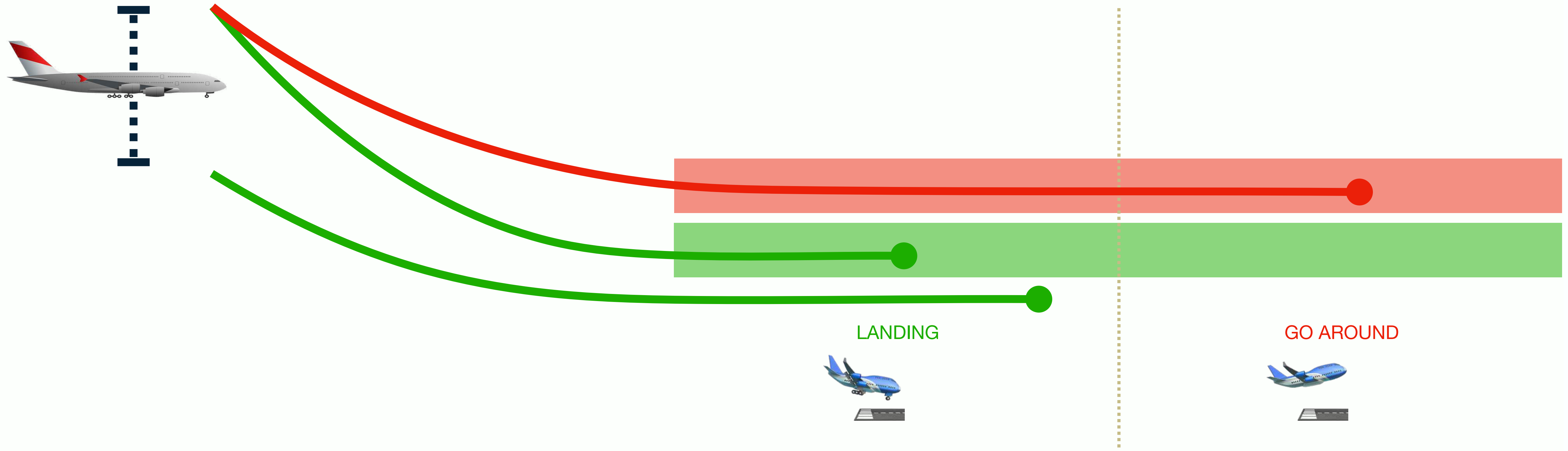


Robustness



Safety

Hypersafety



Local Robustness Verification

Distance-Based Input Perturbations

$P(\mathbf{x}) \overset{\text{def}}{=} \{\mathbf{x}' \mid \delta(\mathbf{x}, \mathbf{x}') \leq \epsilon\}$: perturbation region

$P_\infty(\mathbf{x}) \overset{\text{def}}{=} \{\mathbf{x}' \mid \max_i |\mathbf{x}_i - \mathbf{x}'_i| \leq \epsilon\}$: L_∞ perturbation region

$$\mathcal{R}_\mathbf{x} \overset{\text{def}}{=} \left\{ t \mid t_0 \in P(\mathbf{x}) \Rightarrow t_\omega = M(\mathbf{x}) \right\}$$

prediction of M for $\mathbf{x}$

$\mathcal{R}_\mathbf{x}$ is the set of all executions that are **robust** to perturbations of **x**

Theorem

$$M \models \mathcal{R}_\mathbf{x} \Leftrightarrow [\![M]\!] \subseteq \mathcal{R}_\mathbf{x}$$

Corollary

$$M \models \mathcal{R} \Leftarrow [\![M]\!] \subseteq [\![M]\!]^\natural \subseteq \mathcal{R}$$

Local Robustness Verification

Example

```
x00 = float(input())
x01 = float(input())
x02 = float(input())
x03 = float(input())
x04 = float(input())
x05 = float(input())
```

x:

x00: 0.75
x01: 1
x02: -0.5
x03: 0.75
x04: -0.25
x05: 0.75

$\epsilon = 0.25$ →

$P(\mathbf{x})$:

0.5 ≤ x00 ≤ 1
0.75 ≤ x01 ≤ 1.25
-0.75 ≤ x02 ≤ -0.25
0.5 ≤ x03 ≤ 1
-0.5 ≤ x04 ≤ 0
0.5 ≤ x05 ≤ 1

```
x10 = ReLU((0.120875)*x00 + (0.065404)*x01 + (0.097862)*x02 + (2.030051)*x03 + (0.101956)*x04 + (-2.103565)*x05 + (1.623834))
x11 = ReLU((0.113805)*x00 + (0.064486)*x01 + (0.090701)*x02 + (2.123338)*x03 + (0.076374)*x04 + (-1.651132)*x05 + (-0.828711))
x12 = ReLU((0.755487)*x00 + (0.224640)*x01 + (0.344943)*x02 + (2.619876)*x03 + (0.346636)*x04 + (1.418635)*x05 + (-0.686885))

x20 = ReLU((1.803209)*x10 + (1.222249)*x11 + (2.725716)*x12 + (-3.489653))
x21 = ReLU((1.958950)*x10 + (2.388245)*x11 + (2.245851)*x12 + (-3.834811))
x22 = ReLU((1.958103)*x10 + (2.273354)*x11 + (0.662405)*x12 + (-4.211086))

x30 = ReLU((1.735994)*x20 + (0.666507)*x21 + (3.192344)*x22 + (-2.627086))
x31 = ReLU((2.327110)*x20 + (2.685314)*x21 + (1.424807)*x22 + (-3.695113))
x32 = ReLU((2.147212)*x20 + (2.285599)*x21 + (2.665507)*x22 + (-4.299974))

x40 = ReLU((2.296390)*x30 + (1.980387)*x31 + (2.945360)*x32 + (-4.096463))
x41 = ReLU((-0.552155)*x30 + (-0.828226)*x31 + (-0.495998)*x32)
x42 = ReLU((-2.509773)*x30 + (1.199384)*x31 + (-0.245429)*x32 + (5.024773))

x50 = (-2.278012)*x40 + (0.180652)*x41 + (-16.663048)*x42 + (2500)
x51 = (2.278012)*x40 + (-0.180652)*x41 + (16.663048)*x42 + (-2500)
```

$M(\mathbf{x})$:

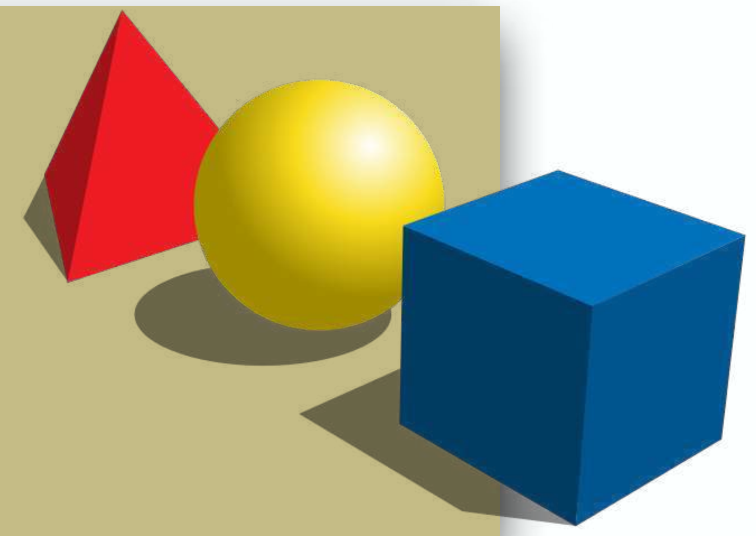
x50 > x51

Abstract Interpretation

3-Step Recipe

practical tools
targeting specific programs

abstract semantics, abstract domains
algorithmic approaches to decide program properties

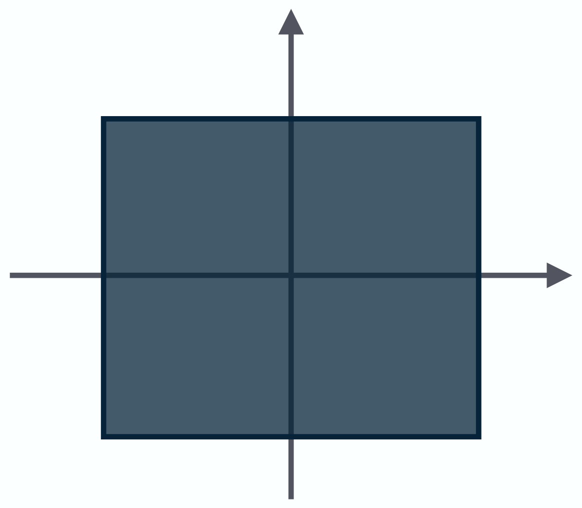


concrete semantics
mathematical models of the program behavior

Local Robustness Verification

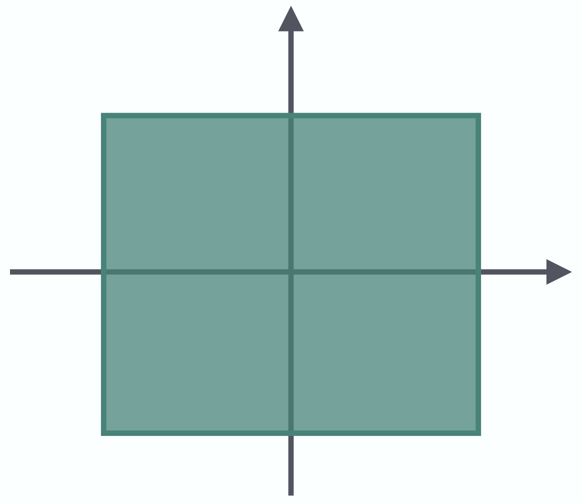
Static Forward Analysis

```
x00 = float(input())
x01 = float(input())
x02 = float(input())
x03 = float(input())
x04 = float(input())
x05 = float(input())

x10 = ReLU((0.120875)*x00 + (0.065404)*x01 + (0.097862)*x02 + (2.030051)*x03 + (0.101956)*x04 + (-2.103565)*x05 + (1.623834))
x11 = ReLU((0.113805)*x00 + (0.064486)*x01 + (0.090701)*x02 + (2.123338)*x03 + (0.076374)*x04 + (-1.651132)*x05 + (-0.828711))
x12 = ReLU((0.755487)*x00 + (0.224640)*x01 + (0.344943)*x02 + (2.619876)*x03 + (0.346636)*x04 + (1.418635)*x05 + (-0.686885))

x20 = ReLU((1.803209)*x10 + (1.222249)*x11 + (2.725716)*x12 + (-3.489653))
x21 = ReLU((1.958950)*x10 + (2.388245)*x11 + (2.245851)*x12 + (-3.834811))
x22 = ReLU((1.958103)*x10 + (2.273354)*x11 + (0.662405)*x12 + (-4.211086))

x30 = ReLU((1.735994)*x20 + (0.666507)*x21 + (3.192344)*x22 + (-2.627086))
x31 = ReLU((2.327110)*x20 + (2.685314)*x21 + (1.424807)*x22 + (-3.695113))
x32 = ReLU((2.147212)*x20 + (2.285599)*x21 + (2.665507)*x22 + (-4.299974))

x40 = ReLU((2.296390)*x30 + (1.980387)*x31 + (2.945360)*x32 + (-4.096463))
x41 = ReLU((-0.552155)*x30 + (-0.828226)*x31 + (-0.495998)*x32)
x42 = ReLU((-2.509773)*x30 + (1.199384)*x31 + (-0.245429)*x32 + (5.024773))

x50 = (-2.278012)*x40 + (0.180652)*x41 + (-16.663048)*x42 + (2500)
x51 = (2.278012)*x40 + (-0.180652)*x41 + (16.663048)*x42 + (-2500)
```



① start from an **abstraction** of all possible inputs

② proceed **forwards** **abstracting** the neural network computations



③ check output for **inclusion** in **expected output**:
included → ✔ **safe**
otherwise → 🚨 **alarm**

Local Robustness Verification

Boxes Abstract Domain

$x_{i,j} \mapsto [a, b]$

$a, b \in \mathscr{R}$

```
x00 = float(input())
x01 = float(input())
x02 = float(input())
x03 = float(input())
x04 = float(input())
x05 = float(input())
```

$P(\mathbf{x})$:
x00: [0.5, 1]
x01: [0.75, 1.25]
x02: [-0.75, -0.25]
x03: [0.5, 1]
x04: [-0.5, 0]
x05: [0.5, 1]

```
x10 = ReLU((0.120875)*x00 + (0.065404)*x01 + (0.097862)*x02 + (2.030051)*x03 + (0.101956)*x04 + (-2.103565)*x05 + (1.623834))
x11 = ReLU((0.113805)*x00 + (0.064486)*x01 + (0.090701)*x02 + (2.123338)*x03 + (0.076374)*x04 + (-1.651132)*x05 + (-0.828711))
x12 = ReLU((0.755487)*x00 + (0.224640)*x01 + (0.344943)*x02 + (2.619876)*x03 + (0.346636)*x04 + (1.418635)*x05 + (-0.686885))

x20 = ReLU((1.803209)*x10 + (1.222249)*x11 + (2.725716)*x12 + (-3.489653))
x21 = ReLU((1.958950)*x10 + (2.388245)*x11 + (2.245851)*x12 + (-3.834811))
x22 = ReLU((1.958103)*x10 + (2.273354)*x11 + (0.662405)*x12 + (-4.211086))

x30 = ReLU((1.735994)*x20 + (0.666507)*x21 + (3.192344)*x22 + (-2.627086))
x31 = ReLU((2.327110)*x20 + (2.685314)*x21 + (1.424807)*x22 + (-3.695113))
x32 = ReLU((2.147212)*x20 + (2.285599)*x21 + (2.665507)*x22 + (-4.299974))

x40 = ReLU((2.296390)*x30 + (1.980387)*x31 + (2.945360)*x32 + (-4.096463))
x41 = ReLU((-0.552155)*x30 + (-0.828226)*x31 + (-0.495998)*x32)
x42 = ReLU((-2.509773)*x30 + (1.199384)*x31 + (-0.245429)*x32 + (5.024773))

x50 = (-2.278012)*x40 + (0.180652)*x41 + (-16.663048)*x42 + (2500)
x51 = (2.278012)*x40 + (-0.180652)*x41 + (16.663048)*x42 + (-2500)
```

$M(\mathbf{x})$: x50 - x51 ⊏ [0, ∞]

Local Robustness Verification

Boxes Abstract Domain

$x_{i,j} \mapsto [a, b]$

$a, b \in \mathscr{R}$

```
x00 = float(input())
x01 = float(input())
x02 = float(input())
x03 = float(input())
x04 = float(input())
x05 = float(input())
```

$P(\mathbf{x})$:
x00: [0.5, 1]
x01: [0.75, 1.25]
x02: [-0.75, -0.25]
x03: [0.5, 1]
x04: [-0.5, 0]
x05: [0.5, 1]

```
x10' = (0.120875)*x00 + (0.065404)*x01 + (0.097862)*x02 + (2.030051)*x03 + (0.101956)*x04 + (-2.103565)*x05 + (1.623834)
```

x10 -> [0.52, 2.78]

```
x10 = ReLU(x10')
```

x10 -> [0.52, 2.78]

```
x11 = ReLU((0.113805)*x00 + (0.064486)*x01 + (0.090701)*x02 + (2.123338)*x03 + (0.076374)*x04 + (-1.651132)*x05 + (-0.828711))
```

x11 -> [0, 0.64]

```
x12 = ReLU((0.755487)*x00 + (0.224640)*x01 + (0.344943)*x02 + (2.619876)*x03 + (0.346636)*x04 + (1.418635)*x05 + (-0.686885))
```

x12 -> [1.45, 4.30]

⋮

```
x50 = (-2.278012)*x40 + (0.180652)*x41 + (-16.663048)*x42 + (2500)
x51 = (2.278012)*x40 + (-0.180652)*x41 + (16.663048)*x42 + (-2500)
```

$M(\mathbf{x})$: x50 - x51 ⊏ [0, ∞]

Local Robustness Verification

Boxes Abstract Domain

$x_{i,j} \mapsto [a, b]$

$a, b \in \mathscr{R}$

```
x00 = float(input())
x01 = float(input())
x02 = float(input())
x03 = float(input())
x04 = float(input())
x05 = float(input())
```

$P(\mathbf{x})$:
x00: [0.5, 1]
x01: [0.75, 1.25]
x02: [-0.75, -0.25]
x03: [0.5, 1]
x04: [-0.5, 0]
x05: [0.5, 1]

```
x10 = ReLU((0.120875)*x00 + (0.065404)*x01 + (0.097862)*x02 + (2.030051)*x03 + (0.101956)*x04 + (-2.103565)*x05 + (1.623834))
x11 = ReLU((0.113805)*x00 + (0.064486)*x01 + (0.090701)*x02 + (2.123338)*x03 + (0.076374)*x04 + (-1.651132)*x05 + (-0.828711))
x12 = ReLU((0.755487)*x00 + (0.224640)*x01 + (0.344943)*x02 + (2.619876)*x03 + (0.346636)*x04 + (1.418635)*x05 + (-0.686885))
```

x10 -> [0.52, 2.78]   x11 -> [0, 0.64]   x12 -> [1.45, 4.30]

```
x20 = ReLU((1.803209)*x10 + (1.222249)*x11 + (2.725716)*x12 + (-3.489653))
x21 = ReLU((1.958950)*x10 + (2.388245)*x11 + (2.245851)*x12 + (-3.834811))
x22 = ReLU((1.958103)*x10 + (2.273354)*x11 + (0.662405)*x12 + (-4.211086))
```

x20 -> [1.39, 14.03]   x21 -> [0.43, 12.80]   x22 -> [0, 5.54]

⋮

x30 -> [0.08, 47.95]   x31 -> [0.71, 71.23]   x32 -> [0, 69.86]

```
x40 = ReLU((2.296390)*x30 + (1.980387)*x31 + (2.945360)*x32 + (-4.096463))
x41 = ReLU((-0.552155)*x30 + (-0.828226)*x31 + (-0.495998)*x32)
x42 = ReLU((-2.509773)*x30 + (1.199384)*x31 + (-0.245429)*x32 + (5.024773))
```

x40 -> [0, 452.83]   x41 -> [0, 0]   x42 -> [0, 90.26]

```
x50 = (-2.278012)*x40 + (0.180652)*x41 + (-16.663048)*x42 + (2500)
x51 = (2.278012)*x40 + (-0.180652)*x41 + (16.663048)*x42 + (-2500)
```

$M(\mathbf{x})$: [-71.23, 5000.0] $\sqsubset$ [0, ∞] ✗

Local Robustness Verification

Symbolic Abstract Domain [Li19]

$$x_{i,j} \mapsto \begin{cases} E_{i,j} \\ [a,b] & a,b \in \mathscr{R} \end{cases}$$

```
x00 = float(input())
x01 = float(input())
x02 = float(input())
x03 = float(input())
x04 = float(input())
x05 = float(input())
```

$P(\mathbf{x})$: x00: $\begin{cases} x00 \\ [0.5,1] \end{cases}$ x01: $\begin{cases} x01 \\ [0.75,1.25] \end{cases}$ x02: $\begin{cases} x02 \\ [-0.75,-0.25] \end{cases}$ x03: $\begin{cases} x03 \\ [0.5,1] \end{cases}$ x04: $\begin{cases} x04 \\ [-0.5,0] \end{cases}$ x05: $\begin{cases} x05 \\ [0.5,1] \end{cases}$

```
x10 = ReLU((0.120875)*x00 + (0.065404)*x01 + (0.097862)*x02 + (2.030051)*x03 + (0.101956)*x04 + (-2.103565)*x05 + (1.623834))
x11 = ReLU((0.113805)*x00 + (0.064486)*x01 + (0.090701)*x02 + (2.123338)*x03 + (0.076374)*x04 + (-1.651132)*x05 + (-0.828711))
x12 = ReLU((0.755487)*x00 + (0.224640)*x01 + (0.344943)*x02 + (2.619876)*x03 + (0.346636)*x04 + (1.418635)*x05 + (-0.686885))

x20 = ReLU((1.803209)*x10 + (1.222249)*x11 + (2.725716)*x12 + (-3.489653))
x21 = ReLU((1.958950)*x10 + (2.388245)*x11 + (2.245851)*x12 + (-3.834811))
x22 = ReLU((1.958103)*x10 + (2.273354)*x11 + (0.662405)*x12 + (-4.211086))

x30 = ReLU((1.735994)*x20 + (0.666507)*x21 + (3.192344)*x22 + (-2.627086))
x31 = ReLU((2.327110)*x20 + (2.685314)*x21 + (1.424807)*x22 + (-3.695113))
x32 = ReLU((2.147212)*x20 + (2.285599)*x21 + (2.665507)*x22 + (-4.299974))

x40 = ReLU((2.296390)*x30 + (1.980387)*x31 + (2.945360)*x32 + (-4.096463))
x41 = ReLU((-0.552155)*x30 + (-0.828226)*x31 + (-0.495998)*x32)
x42 = ReLU((-2.509773)*x30 + (1.199384)*x31 + (-0.245429)*x32 + (5.024773))

x50 = (-2.278012)*x40 + (0.180652)*x41 + (-16.663048)*x42 + (2500)
x51 = (2.278012)*x40 + (-0.180652)*x41 + (16.663048)*x42 + (-2500)
```

$M(\mathbf{x})$: x50 - x51 $\sqsubseteq$ [0, ∞]

Local Robustness Verification

Symbolic Abstract Domain [Li19]

$$x_{i,j} \mapsto \begin{cases} E_{i,j} \\ [a,b] & a,b \in \mathscr{R} \end{cases}$$

```
x00 = float(input())
x01 = float(input())
x02 = float(input())
x03 = float(input())
x04 = float(input())
x05 = float(input())
```

$P(\mathbf{x})$: x00: $\begin{cases} x00 \\ [0.5,1] \end{cases}$ x01: $\begin{cases} x01 \\ [0.75,1.25] \end{cases}$ x02: $\begin{cases} x02 \\ [-0.75,-0.25] \end{cases}$ x03: $\begin{cases} x03 \\ [0.5,1] \end{cases}$ x04: $\begin{cases} x04 \\ [-0.5,0] \end{cases}$ x05: $\begin{cases} x05 \\ [0.5,1] \end{cases}$

```
x10' = (0.120875)*x00 + (0.065404)*x01 + (0.097862)*x02 + (2.030051)*x03 + (0.101956)*x04 + (-2.103565)*x05 + (1.623834)
```

x10': $\begin{cases} (0.120875) * x00 + (0.065404) * x01 + (0.097862) * x02 + (2.030051) * x03 + (0.101956) * x04 + (-2.103565) * x05 + (1.623834) \\ [0.52,\ 2.78] \end{cases}$

```
x10 = ReLU(x10')
```

x10: $\begin{cases} \ldots x10 \\ [0.52,\ 2.78] \end{cases}$

$$x_{i,j} \mapsto \begin{cases} \mathbf{E_{i,j}} \\ \mathbf{[a,b]} \end{cases} \xrightarrow{\text{ReLU}}$$

$$x_{i,j} \mapsto \begin{cases} \mathbf{E_{i,j}} \\ \mathbf{[a,b]} \end{cases} \qquad 0 \leq a$$

$$x_{i,j} \mapsto \begin{cases} \mathbf{x_{i,j}} \\ \mathbf{[0,b]} \end{cases} \qquad a < 0 \wedge 0 < b$$

$$x_{i,j} \mapsto \begin{cases} \mathbf{0} \\ \mathbf{[0,0]} \end{cases} \qquad b \leq 0$$

⋮

```
x50 = (-2.278012)*x40 + (0.180652)*x41 + (-16.663048)*x42 + (2500)
x51 = (2.278012)*x40 + (-0.180652)*x41 + (16.663048)*x42 + (-2500)
```

$M(\mathbf{x})$: x50 - x51 $\sqsubset$ [0, ∞]

Local Robustness Verification

Symbolic Abstract Domain [Li19]

$$x_{i,j} \mapsto \begin{cases} E_{i,j} \\ [a,b] & a,b \in \mathcal{R} \end{cases}$$

```
x00 = float(input())
x01 = float(input())
x02 = float(input())
x03 = float(input())
x04 = float(input())
x05 = float(input())
```

$P(\mathbf{x})$: x00: $\begin{cases} x00 \\ [0.5,1] \end{cases}$ x01: $\begin{cases} x01 \\ [0.75,1.25] \end{cases}$ x02: $\begin{cases} x02 \\ [-0.75,-0.25] \end{cases}$ x03: $\begin{cases} x03 \\ [0.5,1] \end{cases}$ x04: $\begin{cases} x04 \\ [-0.5,0] \end{cases}$ x05: $\begin{cases} x05 \\ [0.5,1] \end{cases}$

```
x10 = ReLU((0.120875)*x00 + (0.065404)*x01 + (0.097862)*x02 + (2.030051)*x03 + (0.101956)*x04 + (-2.103565)*x05 + (1.623834))
x11 = ReLU((0.113805)*x00 + (0.064486)*x01 + (0.090701)*x02 + (2.123338)*x03 + (0.076374)*x04 + (-1.651132)*x05 + (-0.828711))
x12 = ReLU((0.755487)*x00 + (0.224640)*x01 + (0.344943)*x02 + (2.619876)*x03 + (0.346636)*x04 + (1.418635)*x05 + (-0.686885))
```

x10: $\begin{cases} \ldots x10 \\ [0.52,\ 2.78] \end{cases}$ x11: $\begin{cases} x11 \\ [0,\ 0.64] \end{cases}$ x12: $\begin{cases} \ldots x12 \\ [1.45,\ 4.30] \end{cases}$

⋮

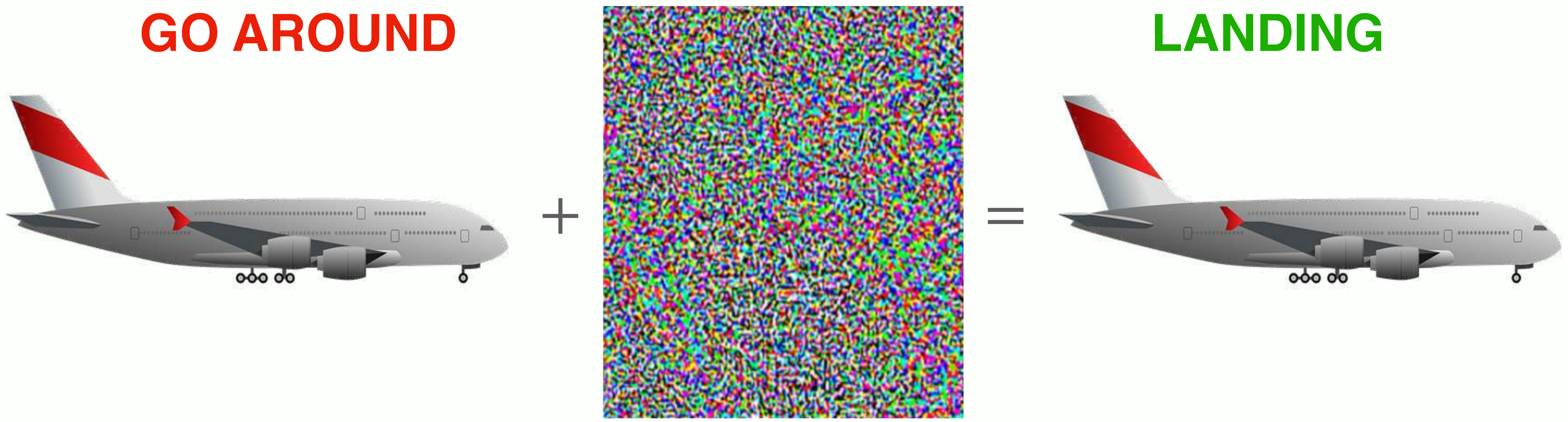
```
x40 = ReLU((2.296390)*x30 + (1.980387)*x31 + (2.945360)*x32 + (-4.096463))
x41 = ReLU((-0.552155)*x30 + (-0.828226)*x31 + (-0.495998)*x32)
x42 = ReLU((-2.509773)*x30 + (1.199384)*x31 + (-0.245429)*x32 + (5.024773))
```

x40: $\begin{cases} 60.23 * x00 + \ldots - 11.6 * x05 + 50.67 * x11 + 18 * x22 - 96.25 \\ [47.02,\ 398.89] \end{cases}$ x41: $\begin{cases} \ldots x40 \\ [0,\ 0] \end{cases}$ x42: $\begin{cases} \ldots x42 \\ [0,\ 3.82] \end{cases}$

```
x50 = (-2.278012)*x40 + (0.180652)*x41 + (-16.663048)*x42 + (2500)
x51 = (2.278012)*x40 + (-0.180652)*x41 + (16.663048)*x42 + (-2500)
```

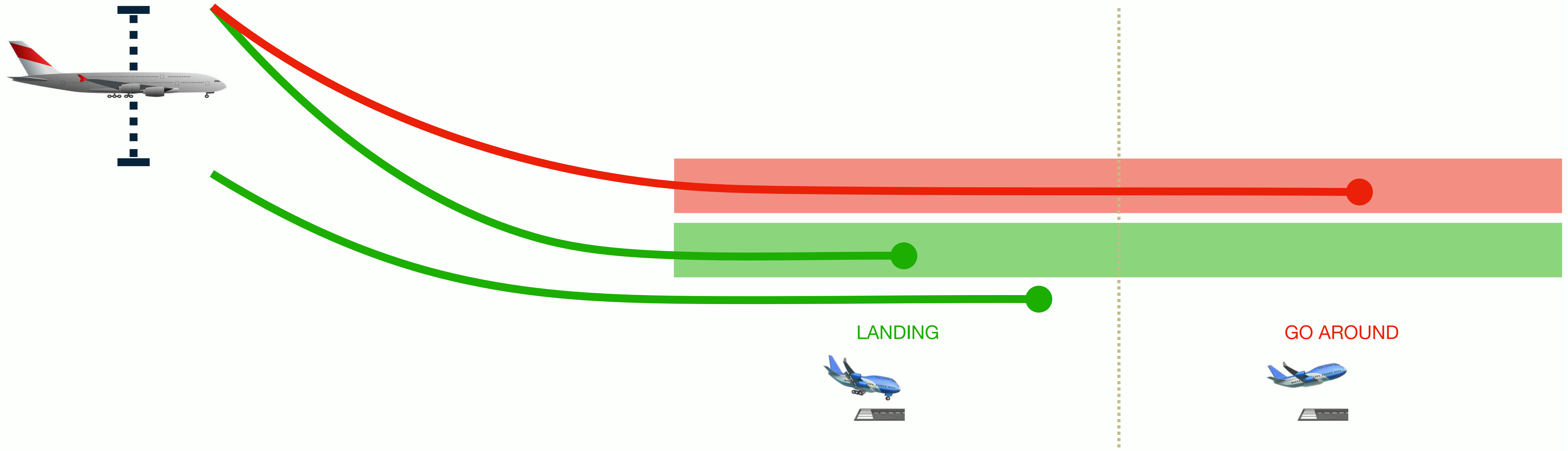
$M(\mathbf{x})$: x50 - x51: $\begin{cases} \ldots - 33.32 * x42 + 5438.52 \\ [3078.07,\ 4785.79] \sqsubseteq [0,\infty] \end{cases}$ ✓

Robustness



Safety

Hypersafety



Safety Verification

Extensional Properties

$\mathbf{I}$: input specification

$\mathbf{O}$: output specification

$$\mathcal{S} \stackrel{\text{def}}{=} \left\{ t \mid t_0 \models \mathbf{I} \Rightarrow t_\omega \models \mathbf{O} \right\}$$

$\mathcal{S}$ is the set of all executions that **satisfy** the specification

Theorem

$$M \models \mathcal{S} \Leftrightarrow [\![M]\!] \subseteq \mathcal{S}$$

Corollary

$$M \models \mathcal{S} \Leftarrow [\![M]\!] \subseteq [\![M]\!]^\natural \subseteq \mathcal{S}$$

Safety Verification

Example

```
x00 = float(input())
x01 = float(input())
x02 = float(input())
x03 = float(input())
x04 = float(input())
x05 = float(input())
```

I:

-1 ≤ x00 ≤ 1
-1 ≤ x01 ≤ 1
-1 ≤ x02 ≤ 1
-1 ≤ x03 ≤ 1
-1 ≤ x04 ≤ 1
-1 ≤ x05 ≤ 1

```
x10 = ReLU((0.120875)*x00 + (0.065404)*x01 + (0.097862)*x02 + (2.030051)*x03 + (0.101956)*x04 + (-2.103565)*x05 + (1.623834))
x11 = ReLU((0.113805)*x00 + (0.064486)*x01 + (0.090701)*x02 + (2.123338)*x03 + (0.076374)*x04 + (-1.651132)*x05 + (-0.828711))
x12 = ReLU((0.755487)*x00 + (0.224640)*x01 + (0.344943)*x02 + (2.619876)*x03 + (0.346636)*x04 + (1.418635)*x05 + (-0.686885))

x20 = ReLU((1.803209)*x10 + (1.222249)*x11 + (2.725716)*x12 + (-3.489653))
x21 = ReLU((1.958950)*x10 + (2.388245)*x11 + (2.245851)*x12 + (-3.834811))
x22 = ReLU((1.958103)*x10 + (2.273354)*x11 + (0.662405)*x12 + (-4.211086))

x30 = ReLU((1.735994)*x20 + (0.666507)*x21 + (3.192344)*x22 + (-2.627086))
x31 = ReLU((2.327110)*x20 + (2.685314)*x21 + (1.424807)*x22 + (-3.695113))
x32 = ReLU((2.147212)*x20 + (2.285599)*x21 + (2.665507)*x22 + (-4.299974))

x40 = ReLU((2.296390)*x30 + (1.980387)*x31 + (2.945360)*x32 + (-4.096463))
x41 = ReLU((-0.552155)*x30 + (-0.828226)*x31 + (-0.495998)*x32)
x42 = ReLU((-2.509773)*x30 + (1.199384)*x31 + (-0.245429)*x32 + (5.024773))

x50 = (-2.278012)*x40 + (0.180652)*x41 + (-16.663048)*x42 + (2500)
x51 = (2.278012)*x40 + (-0.180652)*x41 + (16.663048)*x42 + (-2500)
```

O:

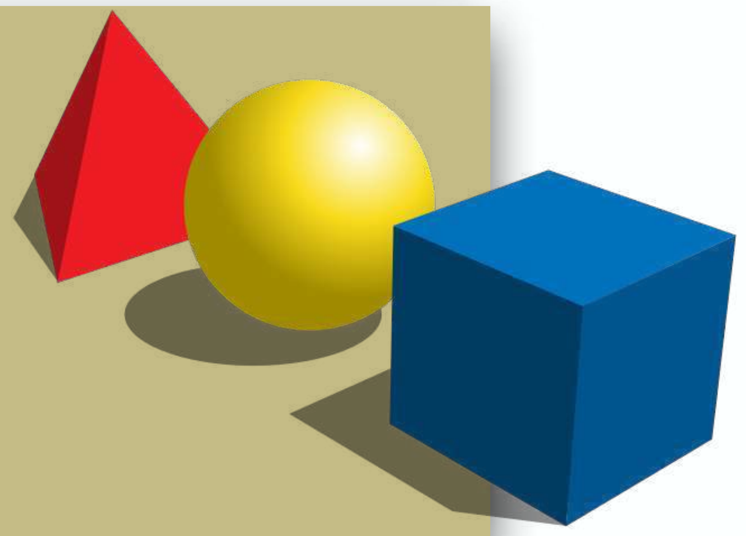
x50 > x51

Abstract Interpretation

3-Step Recipe

practical tools
targeting specific programs

abstract semantics, abstract domains
algorithmic approaches to decide program properties

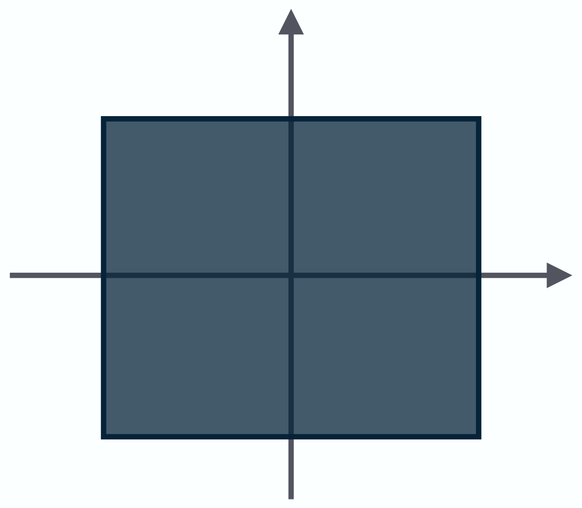


concrete semantics
mathematical models of the program behavior

Safety Verification

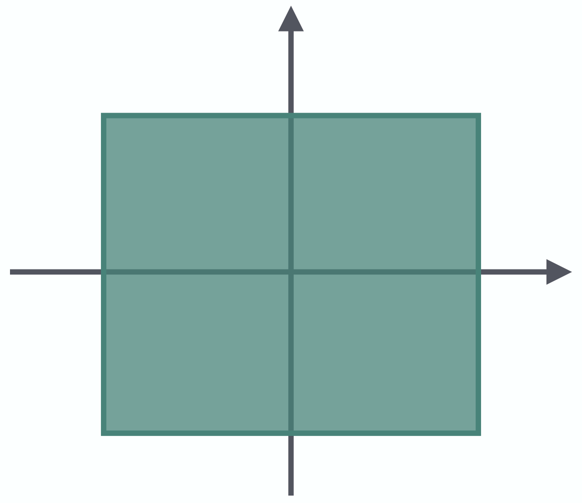
Static Forward Analysis

```
x00 = float(input())
x01 = float(input())
x02 = float(input())
x03 = float(input())
x04 = float(input())
x05 = float(input())

x10 = ReLU((0.120875)*x00 + (0.065404)*x01 + (0.097862)*x02 + (2.030051)*x03 + (0.101956)*x04 + (-2.103565)*x05 + (1.623834))
x11 = ReLU((0.113805)*x00 + (0.064486)*x01 + (0.090701)*x02 + (2.123338)*x03 + (0.076374)*x04 + (-1.651132)*x05 + (-0.828711))
x12 = ReLU((0.755487)*x00 + (0.224640)*x01 + (0.344943)*x02 + (2.619876)*x03 + (0.346636)*x04 + (1.418635)*x05 + (-0.686885))

x20 = ReLU((1.803209)*x10 + (1.222249)*x11 + (2.725716)*x12 + (-3.489653))
x21 = ReLU((1.958950)*x10 + (2.388245)*x11 + (2.245851)*x12 + (-3.834811))
x22 = ReLU((1.958103)*x10 + (2.273354)*x11 + (0.662405)*x12 + (-4.211086))

x30 = ReLU((1.735994)*x20 + (0.666507)*x21 + (3.192344)*x22 + (-2.627086))
x31 = ReLU((2.327110)*x20 + (2.685314)*x21 + (1.424807)*x22 + (-3.695113))
x32 = ReLU((2.147212)*x20 + (2.285599)*x21 + (2.665507)*x22 + (-4.299974))

x40 = ReLU((2.296390)*x30 + (1.980387)*x31 + (2.945360)*x32 + (-4.096463))
x41 = ReLU((-0.552155)*x30 + (-0.828226)*x31 + (-0.495998)*x32)
x42 = ReLU((-2.509773)*x30 + (1.199384)*x31 + (-0.245429)*x32 + (5.024773))

x50 = (-2.278012)*x40 + (0.180652)*x41 + (-16.663048)*x42 + (2500)
x51 = (2.278012)*x40 + (-0.180652)*x41 + (16.663048)*x42 + (-2500)
```



① start from an **abstraction** of all possible inputs

② proceed **forwards** **abstracting** the neural network computations



③ check output for **inclusion** in **expected output**:
included → ✔ **safe**
otherwise → 🚨 **alarm**

Safety Verification

Symbolic Abstract Domain [Li19]

$$x_{i,j} \mapsto \begin{cases} E_{i,j} \\ [a,b] \quad a,b \in \mathscr{R} \end{cases}$$

```
x00 = float(input())
x01 = float(input())
x02 = float(input())
x03 = float(input())
x04 = float(input())
x05 = float(input())
```

I: x00: $\begin{cases} x00 \\ [-1,1] \end{cases}$ x01: $\begin{cases} x01 \\ [-1,1] \end{cases}$ x02: $\begin{cases} x02 \\ [-1,1] \end{cases}$ x03: $\begin{cases} x03 \\ [-1,1] \end{cases}$ x04: $\begin{cases} x04 \\ [-1,1] \end{cases}$ x05: $\begin{cases} x05 \\ [-1,1] \end{cases}$

```
x10 = ReLU((0.120875)*x00 + (0.065404)*x01 + (0.097862)*x02 + (2.030051)*x03 + (0.101956)*x04 + (-2.103565)*x05 + (1.623834))
x11 = ReLU((0.113805)*x00 + (0.064486)*x01 + (0.090701)*x02 + (2.123338)*x03 + (0.076374)*x04 + (-1.651132)*x05 + (-0.828711))
x12 = ReLU((0.755487)*x00 + (0.224640)*x01 + (0.344943)*x02 + (2.619876)*x03 + (0.346636)*x04 + (1.418635)*x05 + (-0.686885))
```

x10: $\begin{cases} x10 \\ [0,\ 6.14] \end{cases}$ x11: $\begin{cases} x11 \\ [0,\ 3.29] \end{cases}$ x11: $\begin{cases} x12 \\ [0,\ 5.02] \end{cases}$

⋮

```
x40 = ReLU((2.296390)*x30 + (1.980387)*x31 + (2.945360)*x32 + (-4.096463))
x41 = ReLU((-0.552155)*x30 + (-0.828226)*x31 + (-0.495998)*x32)
x42 = ReLU((-2.509773)*x30 + (1.199384)*x31 + (-0.245429)*x32 + (5.024773))
```

x40: $\begin{cases} x40 \\ [0,\ 1054.08] \end{cases}$ x41: $\begin{cases} (-0.552155) * x30 + (-0.828226) * x31 + (-0.495998) * x32 \\ [0,0] \end{cases}$ x42: $\begin{cases} x42 \\ [0,\ 191.11] \end{cases}$

```
x50 = (-2.278012)*x40 + (0.180652)*x41 + (-16.663048)*x42 + (2500)
x51 = (2.278012)*x40 + (-0.180652)*x41 + (16.663048)*x42 + (-2500)
```

O: x50 - x51: $\begin{cases} (-4.56) * x40 + (-33.33) * x42 + 5000 \\ [-6171.35,\ 5000.0] \not\sqsubseteq [0,\infty] \end{cases}$ ✗

Safety Verification

DeepPoly Abstract Domain [Singh19]

$$x_{i,j} \mapsto \begin{cases} [L_{i,j}, U_{i,j}] \\ [a, b] \qquad a, b \in \mathscr{R} \end{cases}$$

```
x00 = float(input())
x01 = float(input())
x02 = float(input())
x03 = float(input())
x04 = float(input())
x05 = float(input())
```

I: x00: $\begin{cases}[x00,x00]\\ [-1,1]\end{cases}$ x01: $\begin{cases}[x01,x01]\\ [-1,1]\end{cases}$ x02: $\begin{cases}[x02,x02]\\ [-1,1]\end{cases}$ x03: $\begin{cases}[x03,x03]\\ [-1,1]\end{cases}$ x04: $\begin{cases}[x04,x04]\\ [-1,1]\end{cases}$ x05: $\begin{cases}[x05,x05]\\ [-1,1]\end{cases}$

```
x10 = ReLU((0.120875)*x00 + (0.065404)*x01 + (0.097862)*x02 + (2.030051)*x03 + (0.101956)*x04 + (-2.103565)*x05 + (1.623834))
x11 = ReLU((0.113805)*x00 + (0.064486)*x01 + (0.090701)*x02 + (2.123338)*x03 + (0.076374)*x04 + (-1.651132)*x05 + (-0.828711))
x12 = ReLU((0.755487)*x00 + (0.224640)*x01 + (0.344943)*x02 + (2.619876)*x03 + (0.346636)*x04 + (1.418635)*x05 + (-0.686885))

x20 = ReLU((1.803209)*x10 + (1.222249)*x11 + (2.725716)*x12 + (-3.489653))
x21 = ReLU((1.958950)*x10 + (2.388245)*x11 + (2.245851)*x12 + (-3.834811))
x22 = ReLU((1.958103)*x10 + (2.273354)*x11 + (0.662405)*x12 + (-4.211086))

x30 = ReLU((1.735994)*x20 + (0.666507)*x21 + (3.192344)*x22 + (-2.627086))
x31 = ReLU((2.327110)*x20 + (2.685314)*x21 + (1.424807)*x22 + (-3.695113))
x32 = ReLU((2.147212)*x20 + (2.285599)*x21 + (2.665507)*x22 + (-4.299974))

x40 = ReLU((2.296390)*x30 + (1.980387)*x31 + (2.945360)*x32 + (-4.096463))
x41 = ReLU((-0.552155)*x30 + (-0.828226)*x31 + (-0.495998)*x32)
x42 = ReLU((-2.509773)*x30 + (1.199384)*x31 + (-0.245429)*x32 + (5.024773))

x50 = (-2.278012)*x40 + (0.180652)*x41 + (-16.663048)*x42 + (2500)
x51 = (2.278012)*x40 + (-0.180652)*x41 + (16.663048)*x42 + (-2500)
```

O: x50 - x51 $\sqsubseteq [0, \infty]$

Safety Verification

DeepPoly Abstract Domain [Singh19]

$$x_{i,j} \mapsto \begin{cases} [L_{i,j}, U_{i,j}] \\ [a, b] \qquad a, b \in \mathscr{R} \end{cases}$$

```
x00 = float(input())
x01 = float(input())
x02 = float(input())
x03 = float(input())
x04 = float(input())
x05 = float(input())
```

I: x00: $\begin{cases} [x00, x00] \\ [-1,1] \end{cases}$ x01: $\begin{cases} [x01, x01] \\ [-1,1] \end{cases}$ x02: $\begin{cases} [x02, x02] \\ [-1,1] \end{cases}$ x03: $\begin{cases} [x03, x03] \\ [-1,1] \end{cases}$ x04: $\begin{cases} [x04, x04] \\ [-1,1] \end{cases}$ x05: $\begin{cases} [x05, x05] \\ [-1,1] \end{cases}$

```
x10' = (0.120875)*x00 + (0.065404)*x01 + (0.097862)*x02 + (2.030051)*x03 + (0.101956)*x04 + (-2.103565)*x05 + (1.623834)
```

x10': $\begin{cases} [(0.120875) * x00 + (0.065404) * x01 + (0.097862) * x02 + (2.030051) * x03 + (0.101956) * x04 + (-2.103565) * x05 + (1.623834), \\ (0.120875) * x00 + (0.065404) * x01 + (0.097862) * x02 + (2.030051) * x03 + (0.101956) * x04 + (-2.103565) * x05 + (1.623834)] \\ [-2.90,\ 6.14] \end{cases}$

$x_{i-1,0} \mapsto [\mathbf{L_{i-1,0}}, \mathbf{U_{i-1,0}}]$

$\cdots$

$x_{i-1,j} \mapsto [\mathbf{L_{i-1,j}}, \mathbf{U_{i-1,j}}]$

$\cdots$

$$x_{i,j} = \sum_k w_{j,k}^{i-1} \cdot x_{i-1,k} + b_{i,j}$$

$$x_{i,j} \mapsto \sum_k w_{j,k}^{i-1} \cdot \mathbf{x_{i-1,k}} + b_{i,j}$$

⋮

```
x50 = (-2.278012)*x40 + (0.180652)*x41 + (-16.663048)*x42 + (2500)
x51 = (2.278012)*x40 + (-0.180652)*x41 + (16.663048)*x42 + (-2500)
```

O: x50 - x51 $\sqsubset$ [0, ∞]

Safety Verification

DeepPoly Abstract Domain [Singh19]

$$x_{i,j} \mapsto \begin{cases} [L_{i,j}, U_{i,j}] \\ [a, b] \qquad a, b \in \mathscr{R} \end{cases}$$

```
x00 = float(input())
x01 = float(input())
x02 = float(input())
x03 = float(input())
x04 = float(input())
x05 = float(input())
```

I: x00: $\begin{cases}[x00,x00]\\ [-1,1]\end{cases}$ x01: $\begin{cases}[x01,x01]\\ [-1,1]\end{cases}$ x02: $\begin{cases}[x02,x02]\\ [-1,1]\end{cases}$ x03: $\begin{cases}[x03,x03]\\ [-1,1]\end{cases}$ x04: $\begin{cases}[x04,x04]\\ [-1,1]\end{cases}$ x05: $\begin{cases}[x05,x05]\\ [-1,1]\end{cases}$

```
x10' = (0.120875)*x00 + (0.065404)*x01 + (0.097862)*x02 + (2.030051)*x03 + (0.101956)*x04 + (-2.103565)*x05 + (1.623834)
```

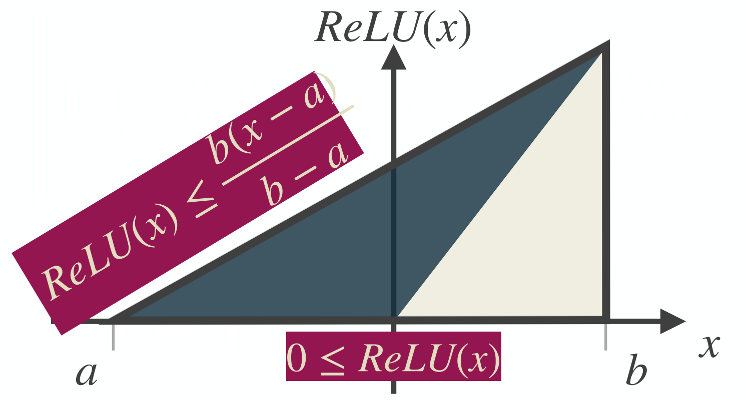
x10': $\begin{cases}[(0.120875)*x00 + (0.065404)*x01 + (0.097862)*x02 + (2.030051)*x03 + (0.101956)*x04 + (-2.103565)*x05 + (1.623834), \\ \ (0.120875)*x00 + (0.065404)*x01 + (0.097862)*x02 + (2.030051)*x03 + (0.101956)*x04 + (-2.103565)*x05 + (1.623834)] \\ [-2.90,\ 6.14]\end{cases}$

```
x10 = ReLU(x10')
```

x10: $\begin{cases}[x10', 0.68 * x10' + 1.97] \\ [-2.90,\ 6.14]\end{cases}$

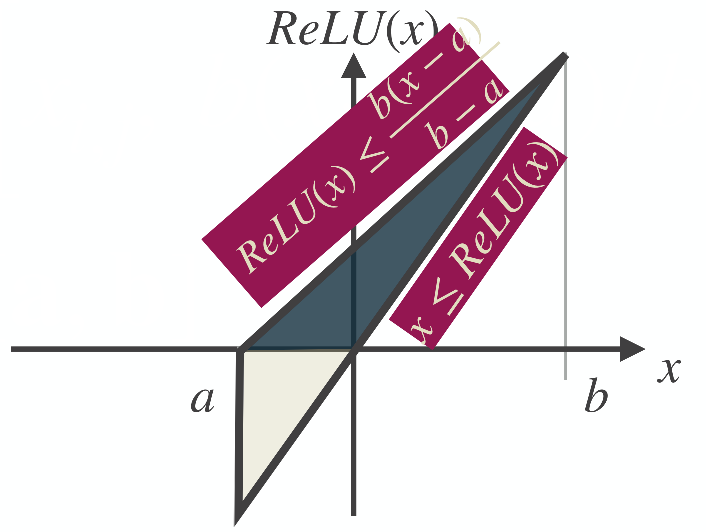
$$x_{i,j} \mapsto \begin{cases} [\mathbf{L_{i,j}}, \mathbf{U_{i,j}}] \\ [\mathbf{a}, \mathbf{b}] \end{cases}$$

$a < 0 \wedge 0 < b \wedge -b \leq a$ → $x_{i,j} \mapsto$



ReLU

$a < 0 \wedge 0 < b \wedge -a < b$ → $x_{i,j} \mapsto$



⋮

```
x50 = (-2.278012)*x40 + (0.180652)*x41 + (-16.663048)*x42 + (2500)
x51 = (2.278012)*x40 + (-0.180652)*x41 + (16.663048)*x42 + (-2500)
```

O: x50 - x51 ⊏ [0, ∞]

Safety Verification

DeepPoly Abstract Domain [Singh19]

$$x_{i,j} \mapsto \begin{cases} [L_{i,j}, U_{i,j}] \\ [a, b] & a, b \in \mathcal{R} \end{cases}$$

```
x00 = float(input())
x01 = float(input())
x02 = float(input())
x03 = float(input())
x04 = float(input())
x05 = float(input())
```

x04: $\begin{cases}[x04, x04] \\ [-1,1]\end{cases}$ x05: $\begin{cases}[x05, x05] \\ [-1,1]\end{cases}$

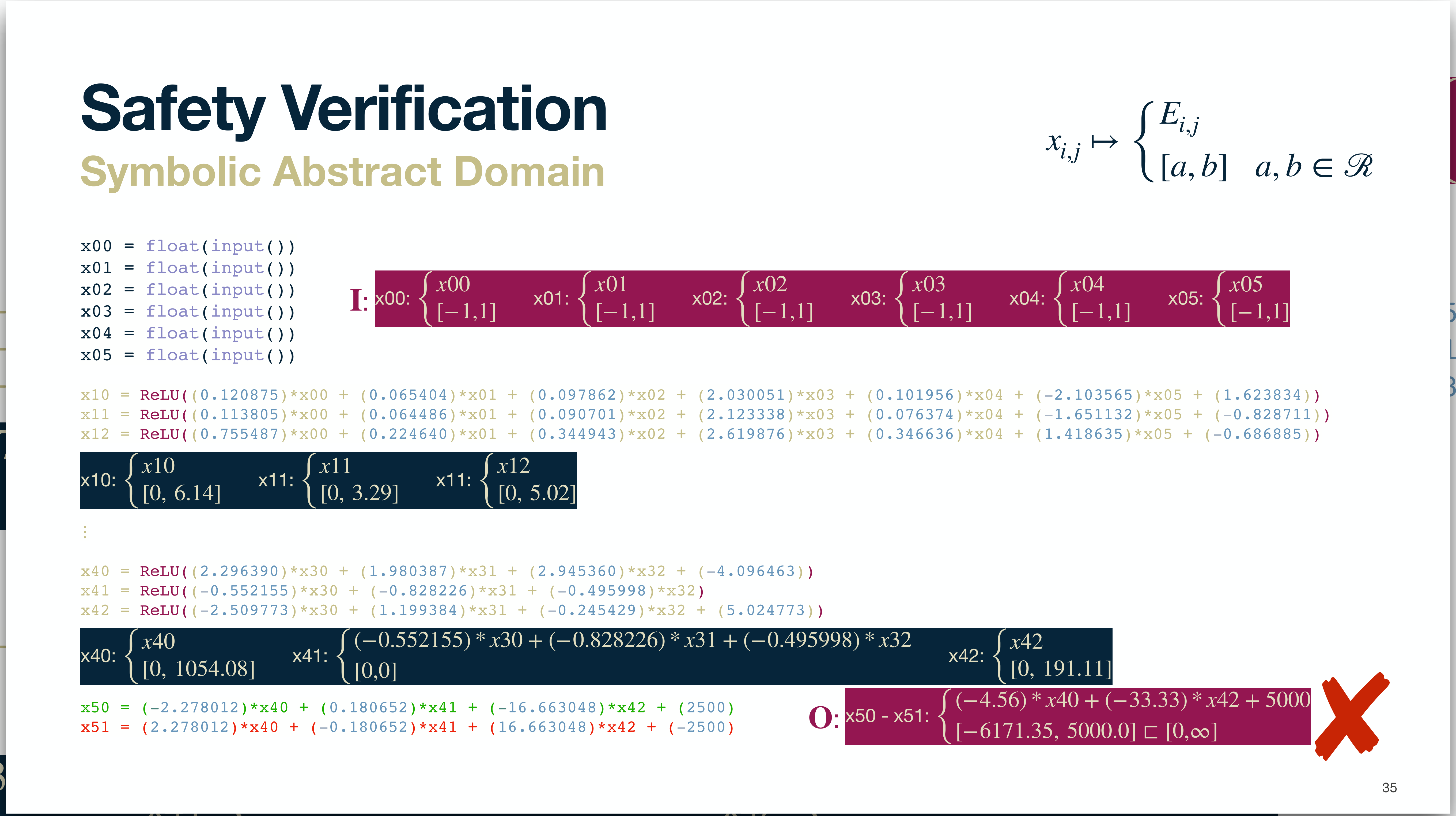
```
x10 = ReLU((0.120875)*x00 +          65)*x05 + (1.623834))
x11 = ReLU((0.113805)*x00 +          32)*x05 + (-0.828711))
x12 = ReLU((0.755487)*x00 +          35)*x05 + (-0.686885))
```

x10: $\begin{cases}[x10', 0.68 * x10' + 1.97 \\ [-2.90,\ 6.14]\end{cases}$

⋮

```
x40 = ReLU((2.296390)*x30 +
x41 = ReLU((-0.552155)*x30
x42 = ReLU((-2.509773)*x30
```

x40: $\begin{cases}[x40', 0.67 * x40' + 313 \\ [-467.10,\ 950.38]\end{cases}$ x41: $[0,\ 118.63]$ x42: $[-142.20,\ 162.09]$

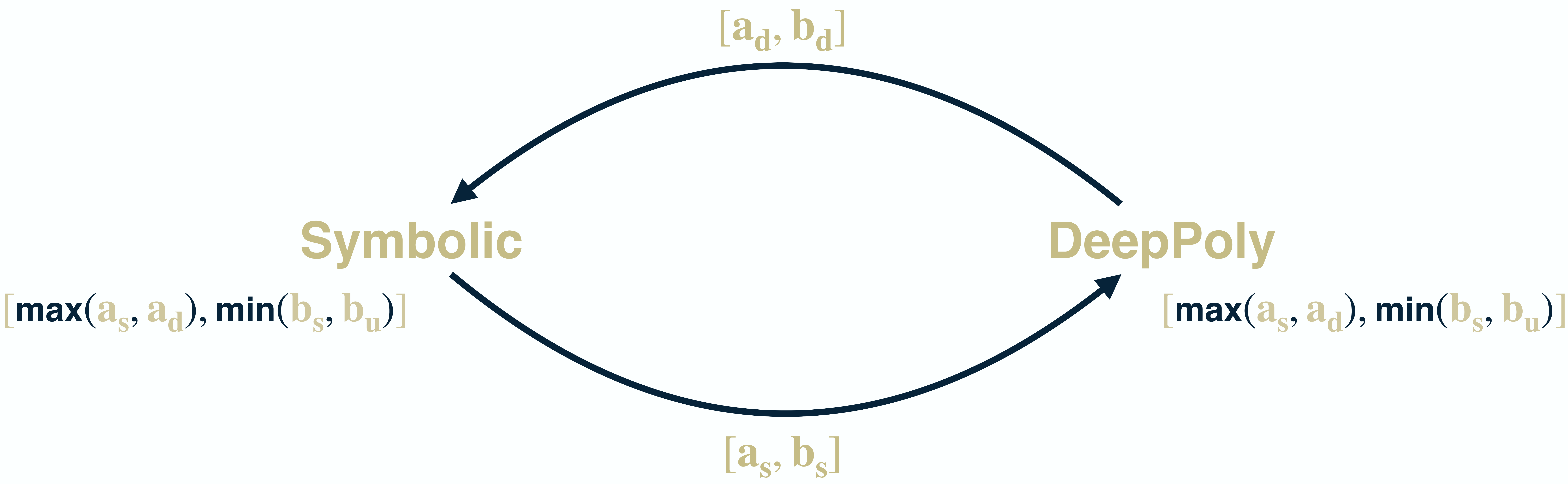



```
x50 = (-2.278012)*x40 + (0.180652)*x41 + (-16.663048)*x42 + (2500)
x51 = (2.278012)*x40 + (-0.180652)*x41 + (16.663048)*x42 + (-2500)
```

O: x50 - x51: $\begin{cases} \dots \\ [-1424.80,\ 9072.12] \sqsubset [0, \infty] \end{cases}$ ✗

Reduced Product Domain

Symbolic Abstract Domain & DeepPoly Abstract Domain



D. Mazzucato and CU. Reduced Products of Abstract Domains for Fairness Certification of Neural Networks. In SAS, 2021

Safety Verification

Symbolic & DeepPoly Product Abstract Domain

```
x00 = float(input())
x01 = float(input())
x02 = float(input())
x03 = float(input())
x04 = float(input())
x05 = float(input())
```

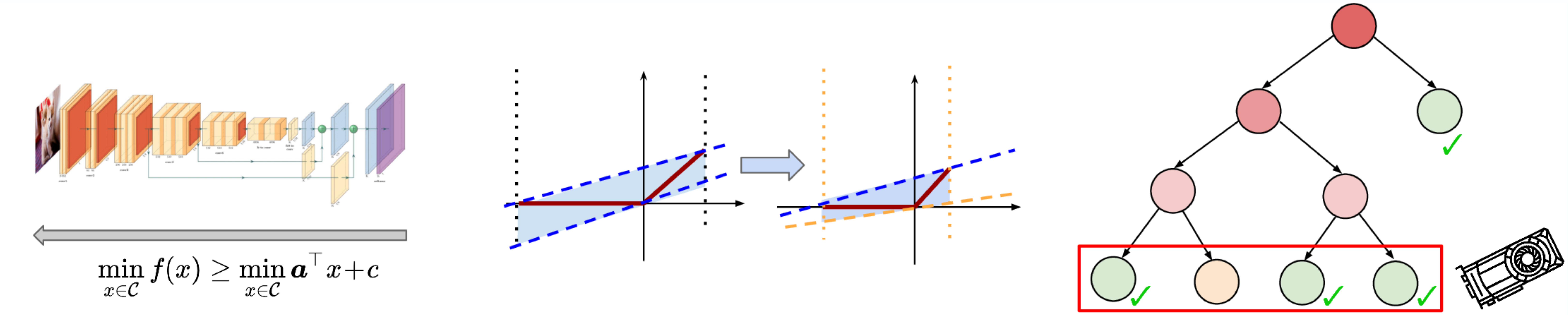
I: x00: $\begin{cases} x00 \\ [x00, x00] \\ [-1, 1] \end{cases}$ x01: $\begin{cases} x01 \\ [x01, x01] \\ [-1, 1] \end{cases}$ x02: $\begin{cases} x02 \\ [x02, x02] \\ [-1, 1] \end{cases}$ x03: $\begin{cases} x03 \\ [x03, x03] \\ [-1, 1] \end{cases}$ x04: $\begin{cases} x04 \\ [x04, x04] \\ [-1, 1] \end{cases}$ x05: $\begin{cases} x05 \\ [x05, x05] \\ [-1, 1] \end{cases}$

```
x10 = ReLU((0.120875)*x00 + (0.065404)*x01 + (0.097862)*x02 + (2.030051)*x03 + (0.101956)*x04 + (-2.103565)*x05 + (1.623834))
x11 = ReLU((0.113805)*x00 + (0.064486)*x01 + (0.090701)*x02 + (2.123338)*x03 + (0.076374)*x04 + (-1.651132)*x05 + (-0.828711))
x12 = ReLU((0.755487)*x00 + (0.224640)*x01 + (0.344943)*x02 + (2.619876)*x03 + (0.346636)*x04 + (1.418635)*x05 + (-0.686885))

x20 = ReLU((1.803209)*x10 + (1.222249)*x11 + (2.725716)*x12 + (-3.489653))
x21 = ReLU((1.958950)*x10 + (2.388245)*x11 + (2.245851)*x12 + (-3.834811))
x22 = ReLU((1.958103)*x10 + (2.273354)*x11 + (0.662405)*x12 + (-4.211086))

x30 = ReLU((1.735994)*x20 + (0.666507)*x21 + (3.192344)*x22 + (-2.627086))
x31 = ReLU((2.327110)*x20 + (2.685314)*x21 + (1.424807)*x22 + (-3.695113))
x32 = ReLU((2.147212)*x20 + (2.285599)*x21 + (2.665507)*x22 + (-4.299974))

x40 = ReLU((2.296390)*x30 + (1.980387)*x31 + (2.945360)*x32 + (-4.096463))
x41 = ReLU((-0.552155)*x30 + (-0.828226)*x31 + (-0.495998)*x32)
x42 = ReLU((-2.509773)*x30 + (1.199384)*x31 + (-0.245429)*x32 + (5.024773))

x50 = (-2.278012)*x40 + (0.180652)*x41 + (-16.663048)*x42 + (2500)
x51 = (2.278012)*x40 + (-0.180652)*x41 + (16.663048)*x42 + (-2500)
```

O: x50 - x51: $\begin{cases} \vdots \\ [670.04, 5000.0] \sqsubseteq [0, \infty] \end{cases}$ ✓

Safety Verification

Going Farther: $\alpha\beta$-CROWN



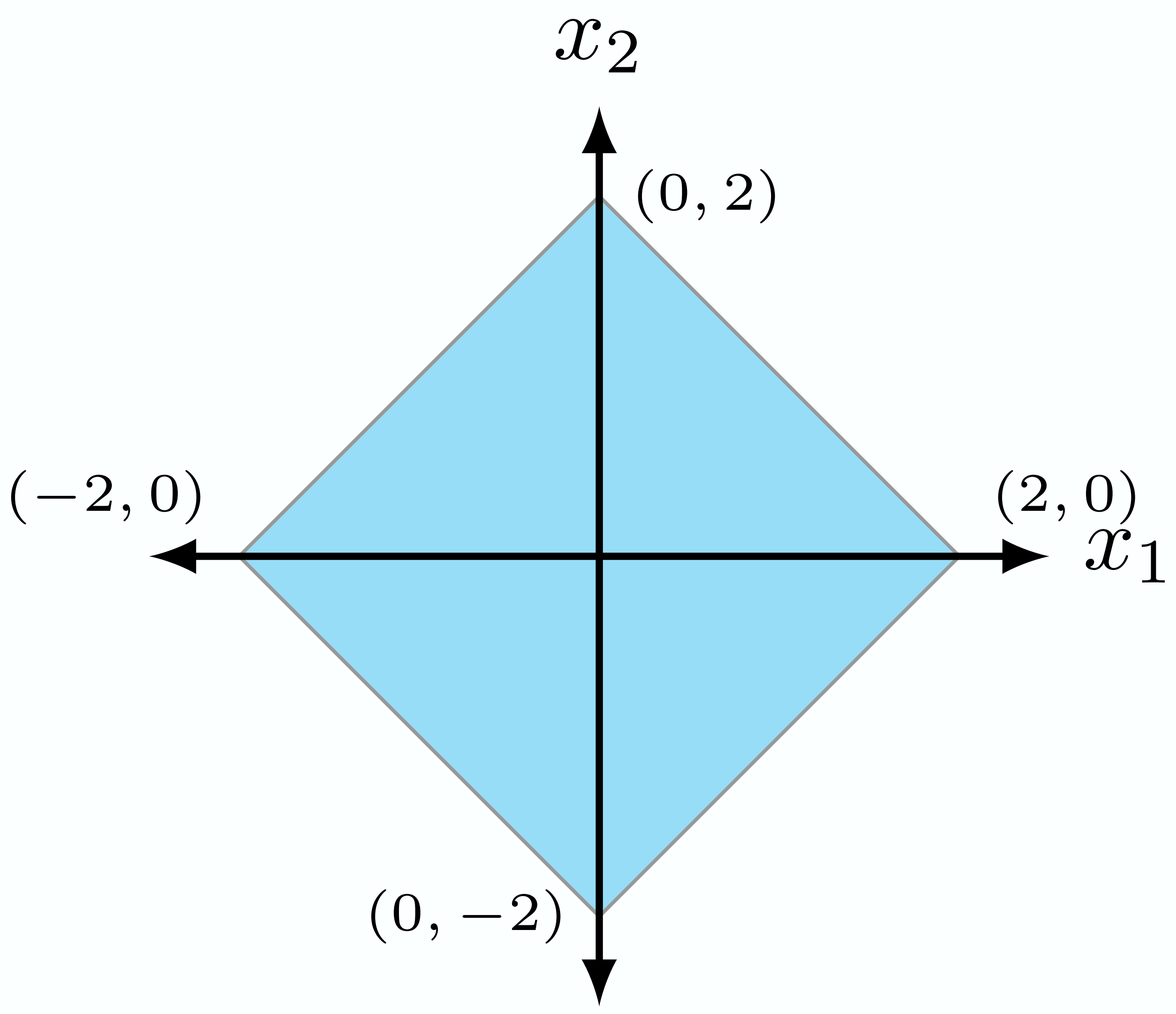
Efficient bound propagation (**CROWN**) GPU optimized relaxation (***α*-CROWN**) Parallel branch and bound (***β*-CROWN**)

Winner of the International Verification of Neural Networks Competition since 2021

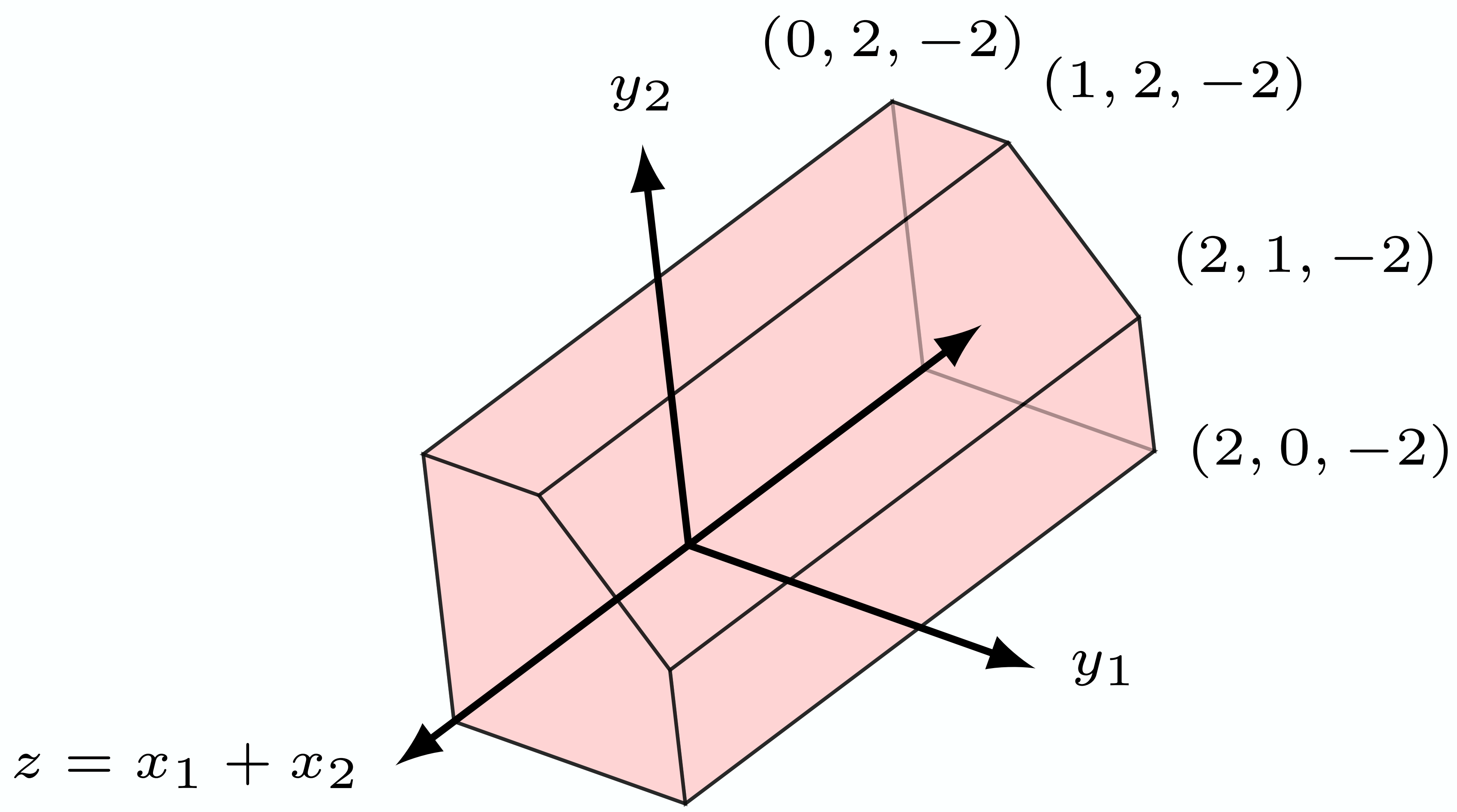
https://github.com/Verified-Intelligence/alpha-beta-CROWN

Safety Verification

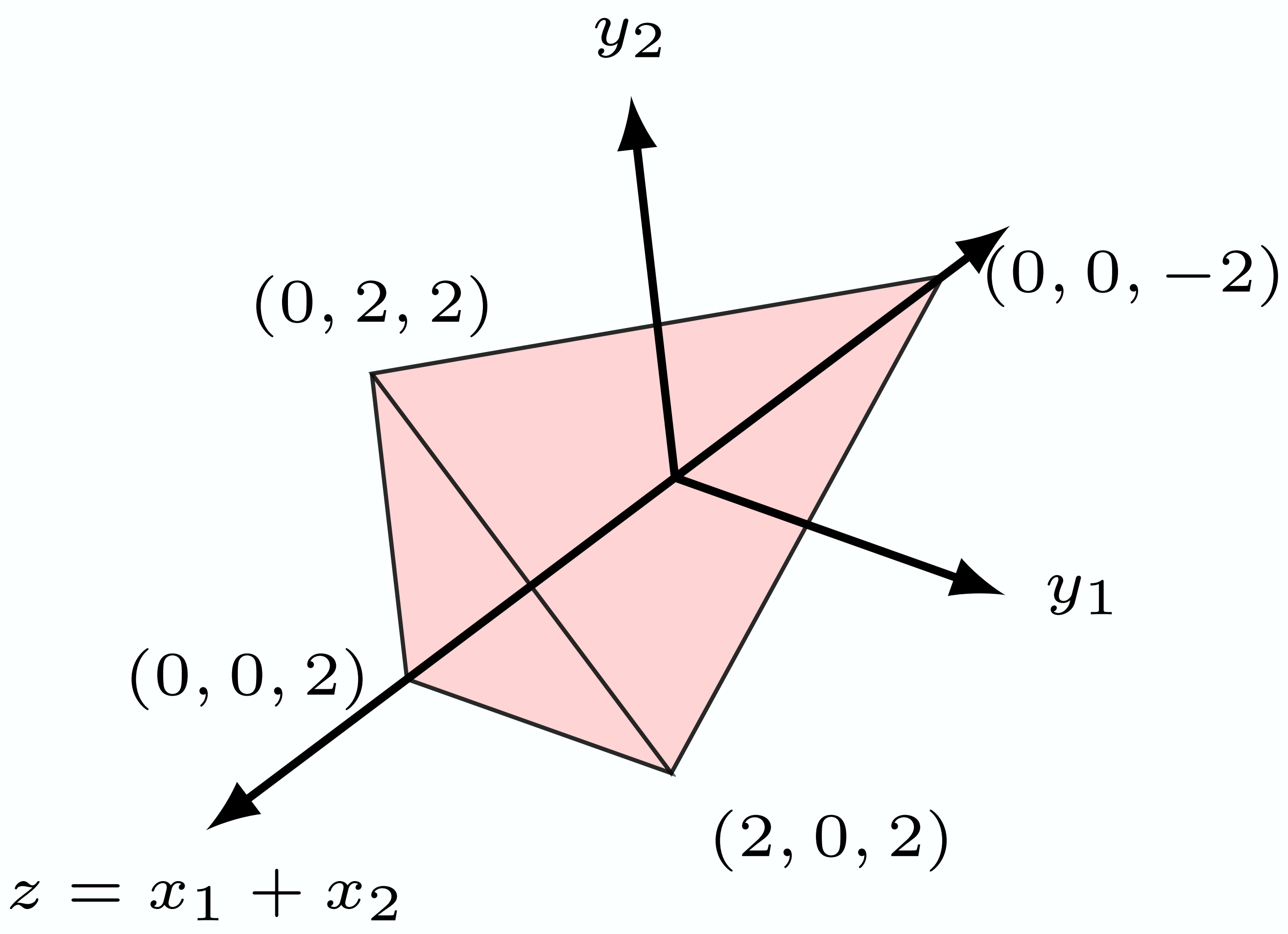
Going Farther: Multi-Neuron Abstractions



(a) Input shape



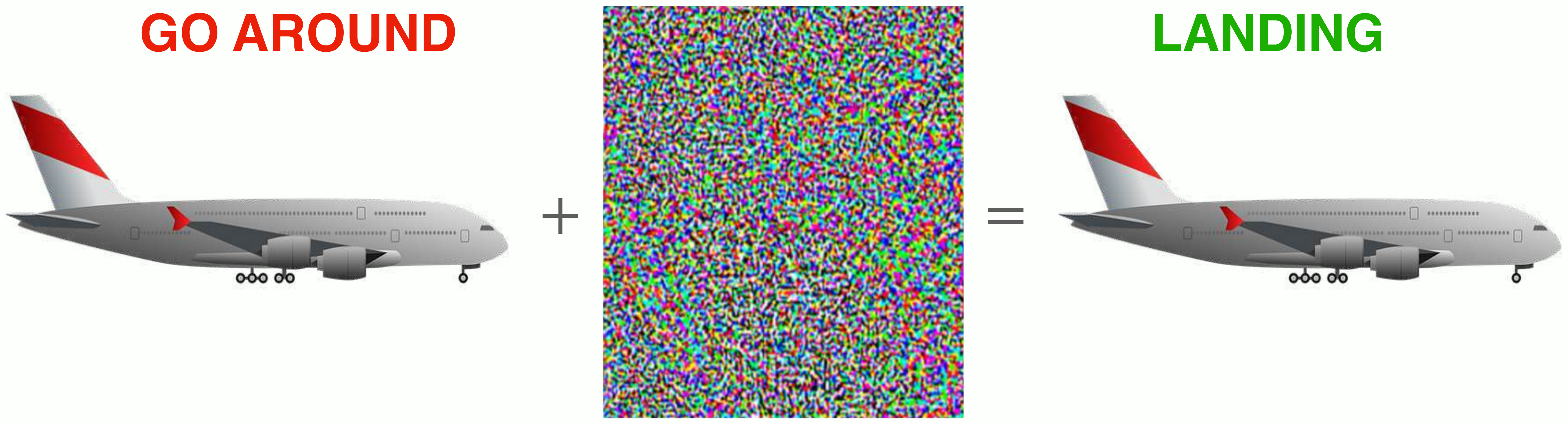
(b) 1-ReLU



(c) 2-ReLU

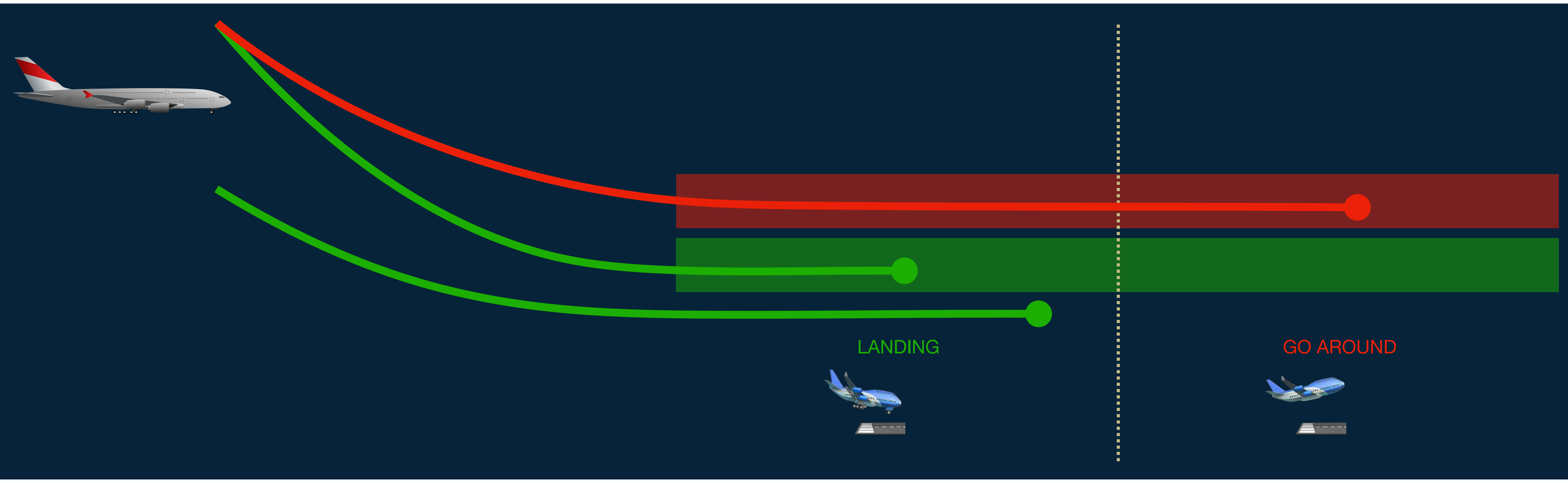
Singh, Ganvir, Püschel and Vechev. Beyond the Single Neuron Convex Barrier for Neural Network Certification. In NeurIPS, 2019

Robustness



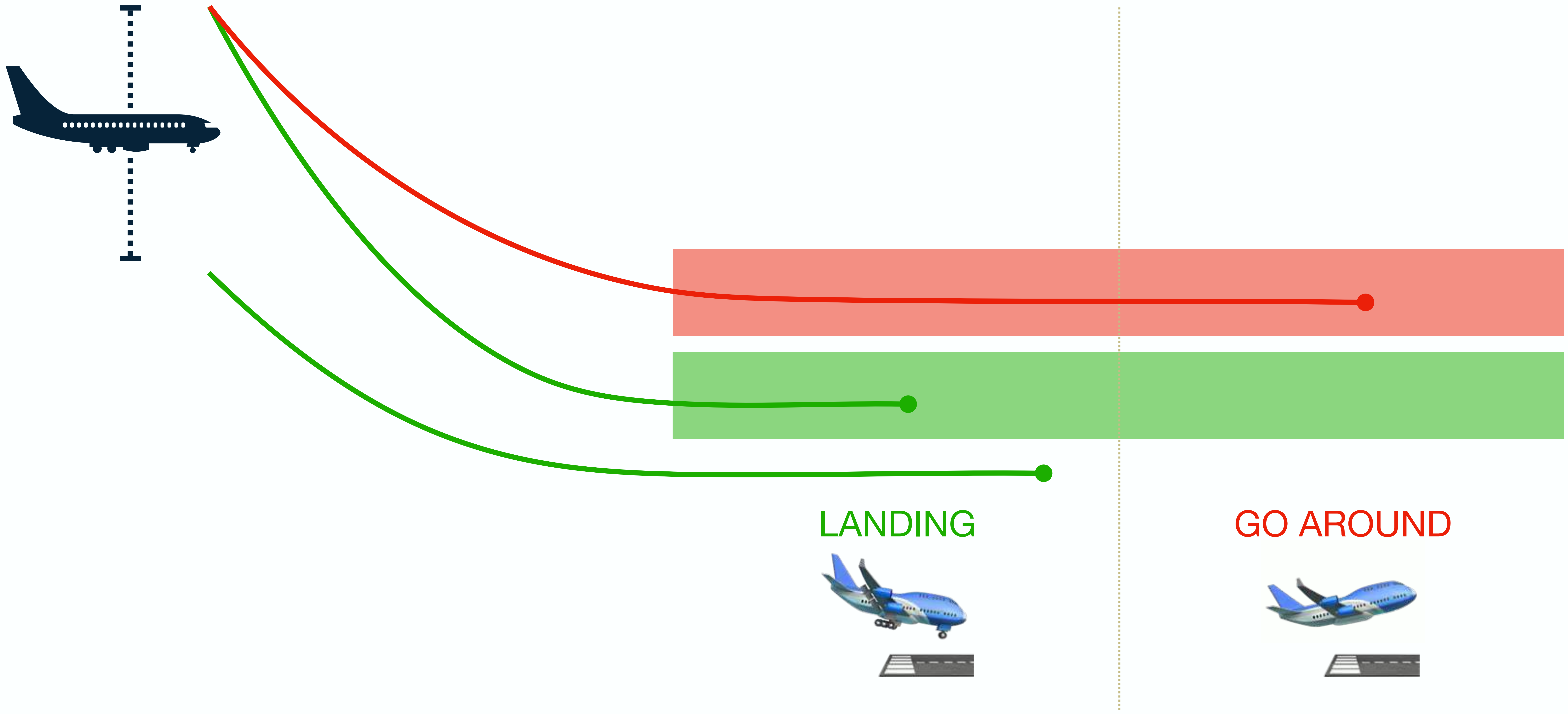
Safety

Hypersafety



Runway Overrun Warning

HyperSafety of Neural Network Surrogate



Hyperproperty Verification

Abstract Non-Interference Properties

η: input abstraction

ρ: output abstraction

$$\mathcal{H} \overset{\text{def}}{=} \left\{ T \mid \forall t, t' \in T \colon \eta(t_0) = \eta(t'_0) \Rightarrow \rho(t_\omega) = \rho(t'_\omega) \right\}$$

$\mathcal{H}$ is the set of all executions that **satisfy** abstract non-interference with respect to η and ρ

Theorem

$$M \models \mathcal{H} \Leftrightarrow [\![M]\!] \in \mathcal{H} \Leftrightarrow \{[\![M]\!]\} \subseteq \mathcal{H}$$

Corollary

$$M \models \mathcal{H} \Leftarrow \{[\![M]\!]\} \subseteq [\![M]\!]^\natural \subseteq \mathcal{H}$$

Giacobazzi and Mastroeni. Abstract Non-Interference: A Unifying Framework for Weakening Information-Flow. In TOPS, 2018.

Abstract Non-Interference Verification

Example

```
x00 = float(input())
x01 = float(input())
x02 = float(input())
x03 = float(input())
x04 = float(input())
x05 = float(input())

x10 = ReLU((0.120875)*x00 + (0.065404)*x01 + (0.097862)*x02 + (2.030051)*x03 + (0.101956)*x04 + (-2.103565)*x05 + (1.623834))
x11 = ReLU((0.113805)*x00 + (0.064486)*x01 + (0.090701)*x02 + (2.123338)*x03 + (0.076374)*x04 + (-1.651132)*x05 + (-0.828711))
x12 = ReLU((0.755487)*x00 + (0.224640)*x01 + (0.344943)*x02 + (2.619876)*x03 + (0.346636)*x04 + (1.418635)*x05 + (-0.686885))

x20 = ReLU((1.803209)*x10 + (1.222249)*x11 + (2.725716)*x12 + (-3.489653))
x21 = ReLU((1.958950)*x10 + (2.388245)*x11 + (2.245851)*x12 + (-3.834811))
x22 = ReLU((1.958103)*x10 + (2.273354)*x11 + (0.662405)*x12 + (-4.211086))

x30 = ReLU((1.735994)*x20 + (0.666507)*x21 + (3.192344)*x22 + (-2.627086))
x31 = ReLU((2.327110)*x20 + (2.685314)*x21 + (1.424807)*x22 + (-3.695113))
x32 = ReLU((2.147212)*x20 + (2.285599)*x21 + (2.665507)*x22 + (-4.299974))

x40 = ReLU((2.296390)*x30 + (1.980387)*x31 + (2.945360)*x32 + (-4.096463))
x41 = ReLU((-0.552155)*x30 + (-0.828226)*x31 + (-0.495998)*x32)
x42 = ReLU((-2.509773)*x30 + (1.199384)*x31 + (-0.245429)*x32 + (5.024773))

x50 = (-2.278012)*x40 + (0.180652)*x41 + (-16.663048)*x42 + (1864)
x51 = (2.278012)*x40 + (-0.180652)*x41 + (16.663048)*x42 + (-1864)
```

η:

$\eta(x00) = x00$

ALTITUDE

$\eta(x01) = x01$

$\eta(x02) = \top$

$\eta(x03) = x03$

$\eta(x04) = x04$

$\eta(x05) = x05$

"**the risk of a runway overrun does not change when only varying the altitude** at which it is measured (in the expected range) and nothing else"

ρ:

$\rho(x50) = 1$ if $x50 > x51$ else 0

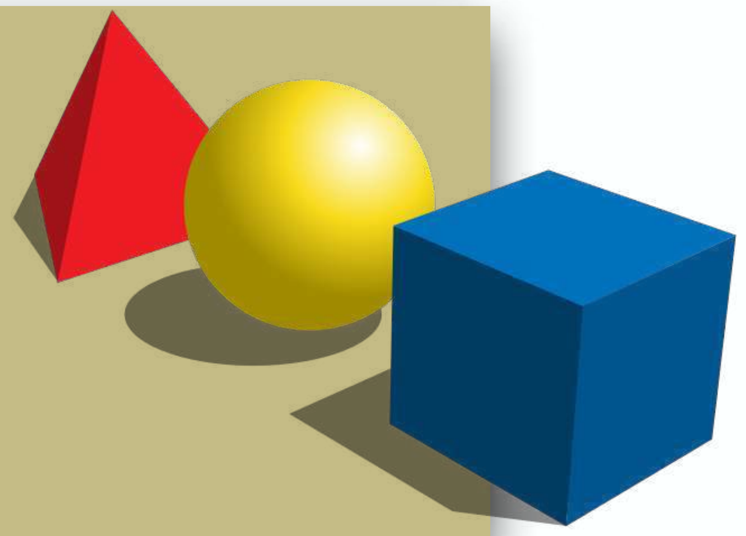
$\rho(x51) = 1$ if $x51 > x50$ else 0

Abstract Interpretation

3-Step Recipe

practical tools
targeting specific programs

abstract semantics, abstract domains
algorithmic approaches to decide program properties



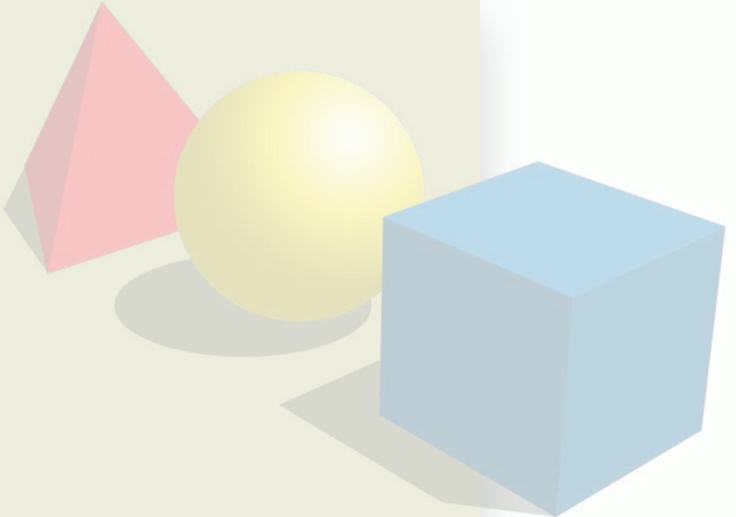
concrete semantics
mathematical models of the program behavior

Abstract Interpretation

3-Step Recipe

practical tools
targeting specific programs

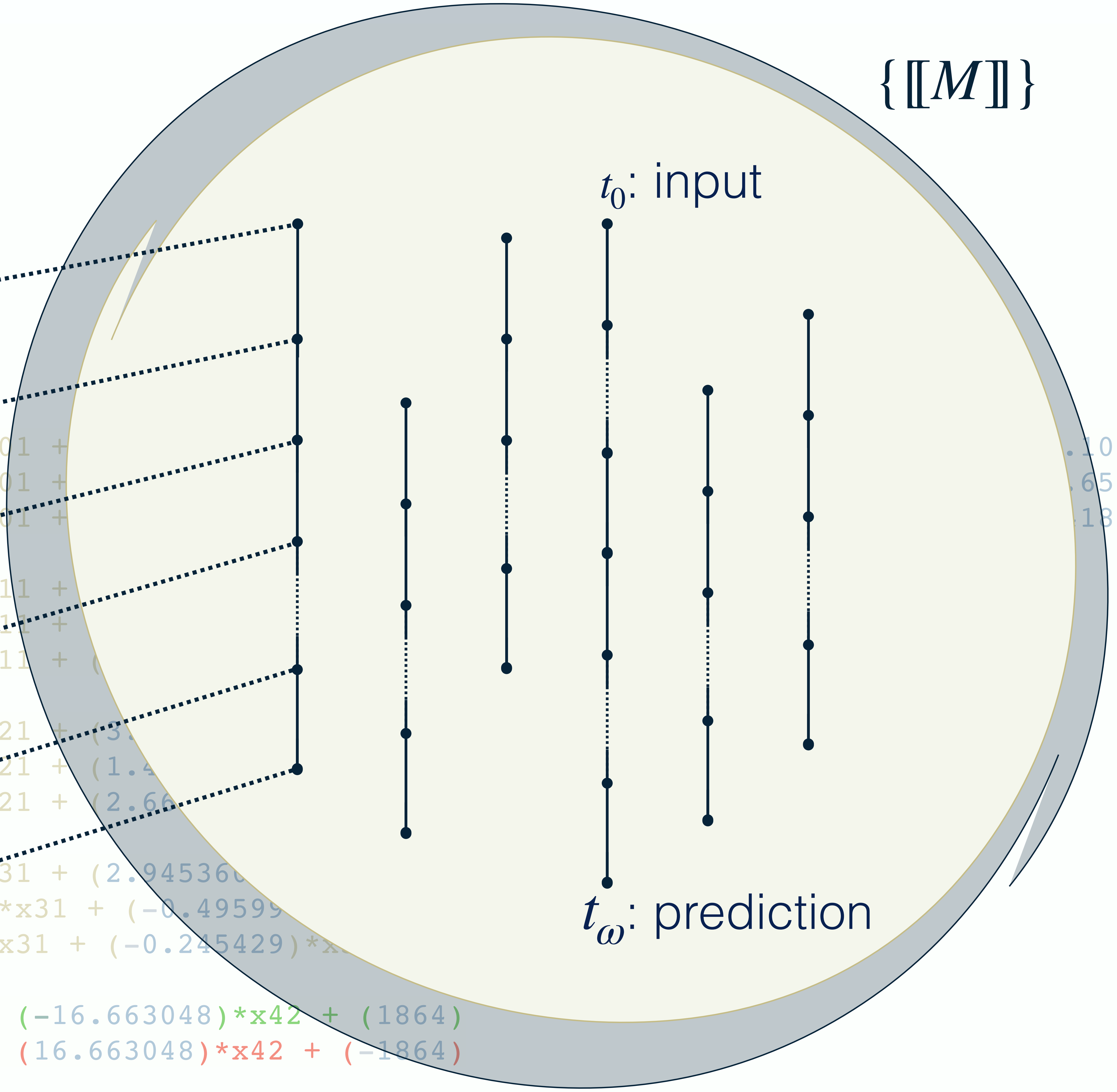
abstract semantics, abstract domains
algorithmic approaches to decide program properties



concrete semantics
mathematical models of the program behavior

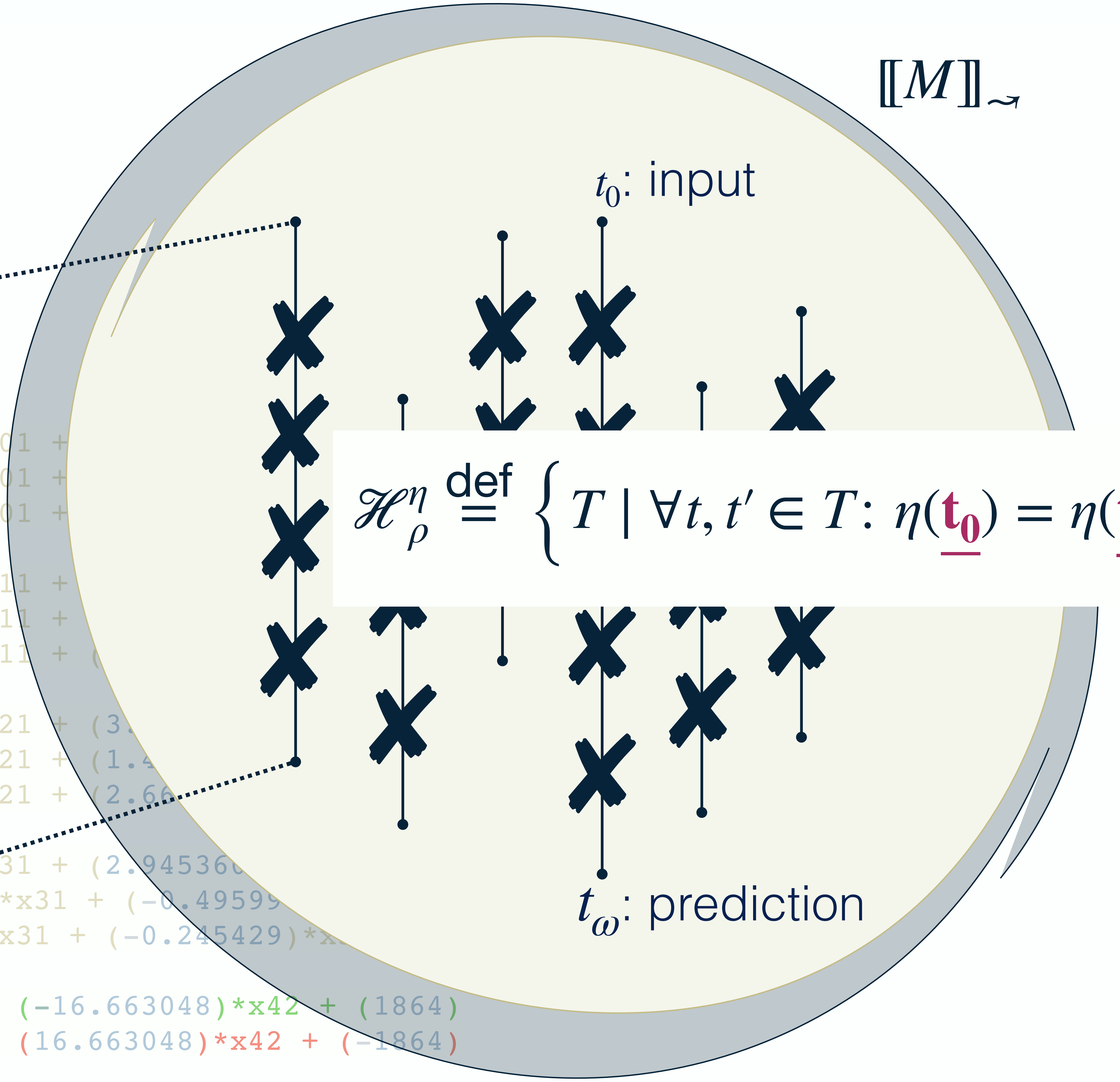
Collecting Semantics

```
x00 = float(input())
x01 = float(input())
x02 = float(input())
x03 = float(input())
x04 = float(input())
x05 = float(input())

x10 = ReLU((0.120875)*x00 + (0.065404)*x01 + ... .103565)*x05 + (1.623834))
x11 = ReLU((0.113805)*x00 + (0.064486)*x01 + ... .651132)*x05 + (-0.828711))
x12 = ReLU((0.755487)*x00 + (0.224640)*x01 + ... .18635)*x05 + (-0.686885))

x20 = ReLU((1.803209)*x10 + (1.222249)*x11 + ...
x21 = ReLU((1.958950)*x10 + (2.388245)*x11 + ...
x22 = ReLU((1.958103)*x10 + (2.273354)*x11 + (...

x30 = ReLU((1.735994)*x20 + (0.666507)*x21 + (3....
x31 = ReLU((2.327110)*x20 + (2.685314)*x21 + (1.4...
x32 = ReLU((2.147212)*x20 + (2.205599)*x21 + (2.66...

x40 = ReLU((2.296390)*x30 + (1.980387)*x31 + (2.945360...
x41 = ReLU((-0.552155)*x30 + (-0.828226)*x31 + (-0.49599...
x42 = ReLU((-2.509773)*x30 + (1.199384)*x31 + (-0.245429)*x...

x50 = (-2.278012)*x40 + (0.180652)*x41 + (-16.663048)*x42 + (1864)
x51 = (2.278012)*x40 + (-0.180652)*x41 + (16.663048)*x42 + (-1864)
```



Dependency Semantics

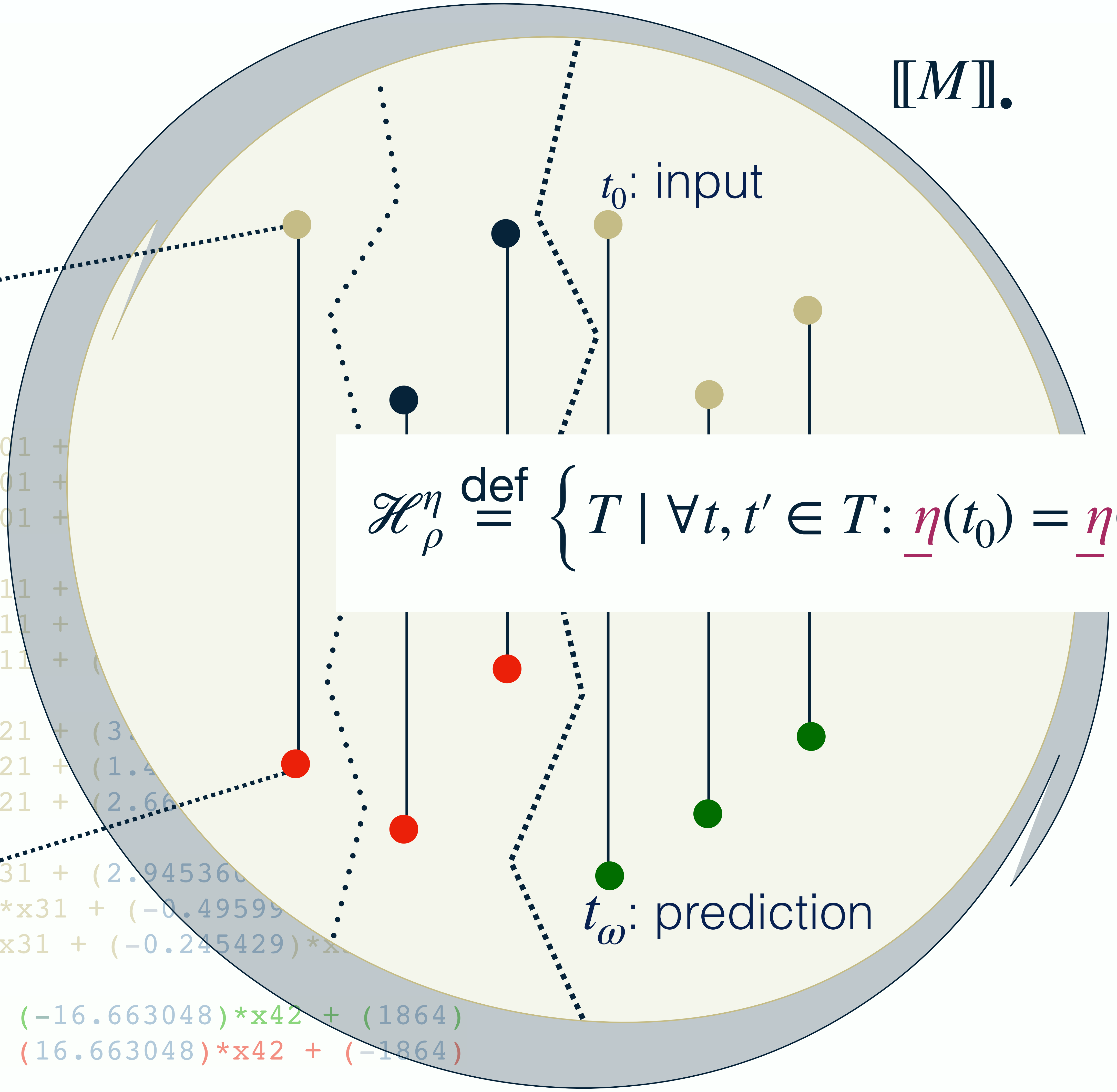
```
x00 = float(input())
x01 = float(input())
x02 = float(input())
x03 = float(input())
x04 = float(input())
x05 = float(input())

x10 = ReLU((0.120875)*x00 + (0.065404)*x01 +
x11 = ReLU((0.113805)*x00 + (0.064486)*x01 +
x12 = ReLU((0.755487)*x00 + (0.224640)*x01 +

x20 = ReLU((1.803209)*x10 + (1.222249)*x11 +
x21 = ReLU((1.958950)*x10 + (2.388245)*x11 +
x22 = ReLU((1.958103)*x10 + (2.273354)*x11 + (

x30 = ReLU((1.735994)*x20 + (0.666507)*x21 + (3.
x31 = ReLU((2.327110)*x20 + (2.685314)*x21 + (1.4
x32 = ReLU((2.147212)*x20 + (2.285599)*x21 + (2.66

x40 = ReLU((2.296390)*x30 + (1.980387)*x31 + (2.945360
x41 = ReLU((-0.552155)*x30 + (-0.828226)*x31 + (-0.49599
x42 = ReLU((-2.509773)*x30 + (1.199384)*x31 + (-0.245429)*x

x50 = (-2.278012)*x40 + (0.180652)*x41 + (-16.663048)*x42 + (1864)
x51 = (2.278012)*x40 + (-0.180652)*x41 + (16.663048)*x42 + (-1864)
```



$$\mathscr{H}^{\eta}_{\rho} \overset{\text{def}}{=} \left\{ T \mid \forall t, t' \in T\colon \eta(\underline{\mathbf{t_0}}) = \eta(\underline{\mathbf{t'_0}}) \Rightarrow \rho(\underline{\mathbf{t_\omega}}) = \rho(\underline{\mathbf{t'_\omega}}) \right\}$$

Parallel Semantics

```
x00 = float(input())
x01 = float(input())
x02 = float(input())
x03 = float(input())
x04 = float(input())
x05 = float(input())

x10 = ReLU((0.120875)*x00 + (0.065404)*x01 +
x11 = ReLU((0.113805)*x00 + (0.064486)*x01 +
x12 = ReLU((0.755487)*x00 + (0.224640)*x01 +

x20 = ReLU((1.803209)*x10 + (1.222249)*x11 +
x21 = ReLU((1.958950)*x10 + (2.388245)*x11 +
x22 = ReLU((1.958103)*x10 + (2.273354)*x11 + (

x30 = ReLU((1.735994)*x20 + (0.666507)*x21 + (3.
x31 = ReLU((2.327110)*x20 + (2.685314)*x21 + (1.4
x32 = ReLU((2.147212)*x20 + (2.285599)*x21 + (2.66

x40 = ReLU((2.296390)*x30 + (1.980387)*x31 + (2.945360
x41 = ReLU((-0.552155)*x30 + (-0.828226)*x31 + (-0.49599
x42 = ReLU((-2.509773)*x30 + (1.199384)*x31 + (-0.245429)*x

x50 = (-2.278012)*x40 + (0.180652)*x41 + (-16.663048)*x42 + (1864)
x51 = (2.278012)*x40 + (-0.180652)*x41 + (16.663048)*x42 + (-1864)
```



$$\mathcal{H}^{\eta}_{\rho} \overset{\text{def}}{=} \left\{ T \mid \forall t, t' \in T \colon \underline{\eta}(t_0) = \underline{\eta}(t'_0) \Rightarrow \underline{\rho}(t_\omega) = \underline{\rho}(t'_\omega) \right\}$$

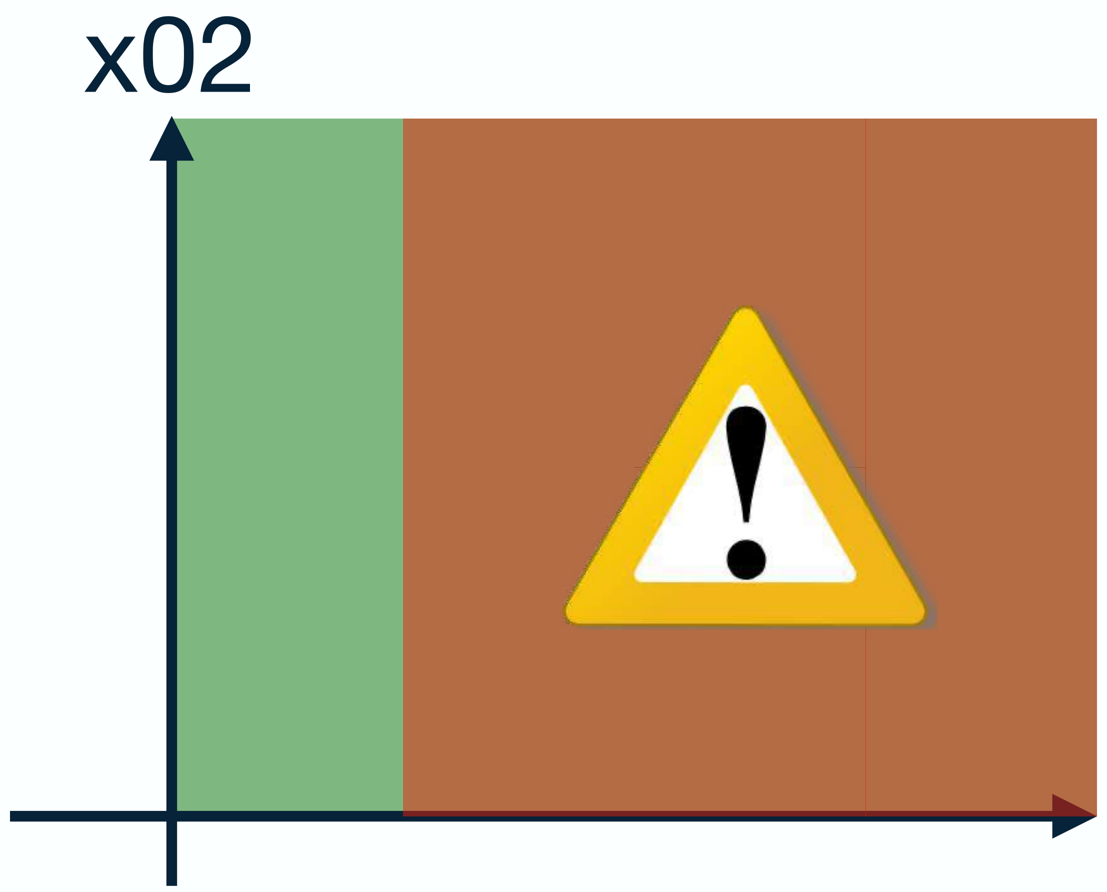
Hyperproperty Verification

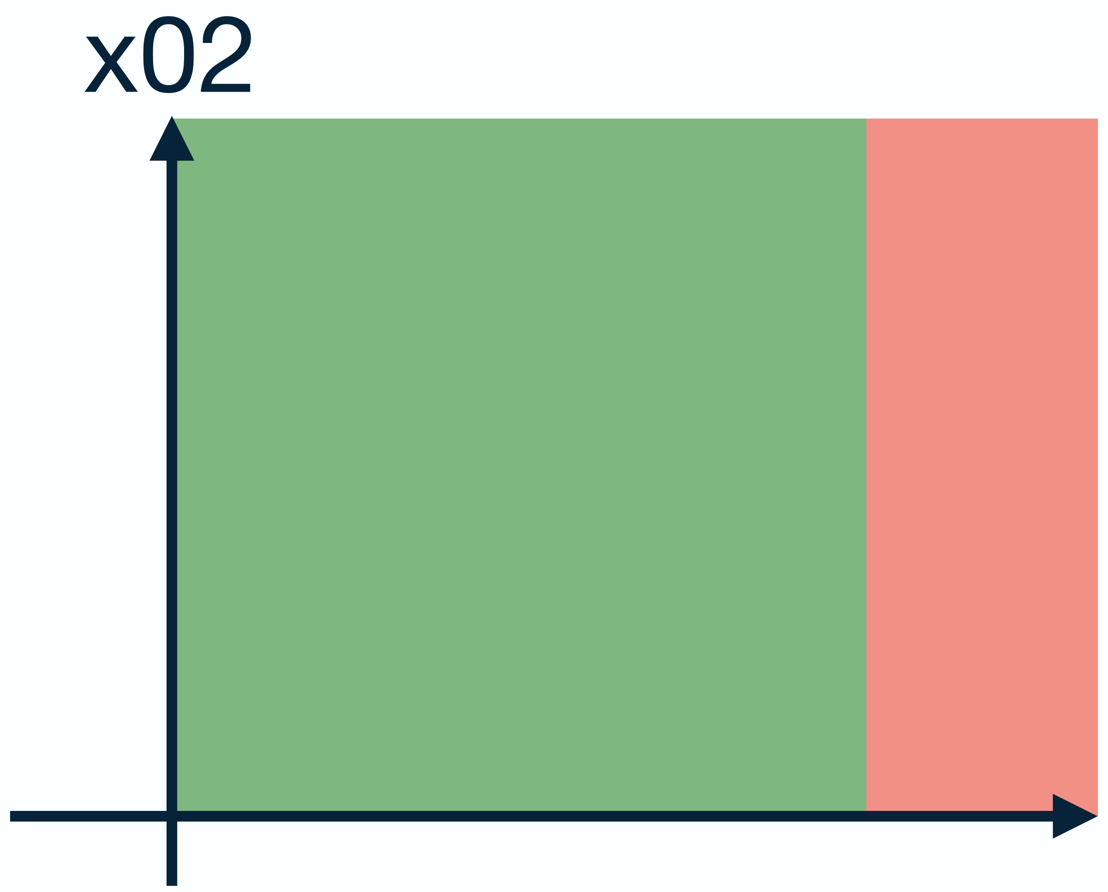
Abstract Non-Interference Properties

$$\mathscr{H} \overset{\text{def}}{=} \left\{ T \mid \forall t, t' \in T \colon \eta(t_0) = \eta(t'_0) \Rightarrow \rho(t_\omega) = \rho(t'_\omega) \right\}$$

Lemma

$$M \models \mathscr{H} \Leftrightarrow \forall I \in \mathbb{I} \colon \forall A, B \in [\![M]\!]^{\mathbb{I}}_{\bullet} \colon \rho(A^I_\omega) \sqcap \rho(B^I_\omega) = \bot \Rightarrow \eta(A^I_0) \sqcap \eta(B^I_0) = \bot$$





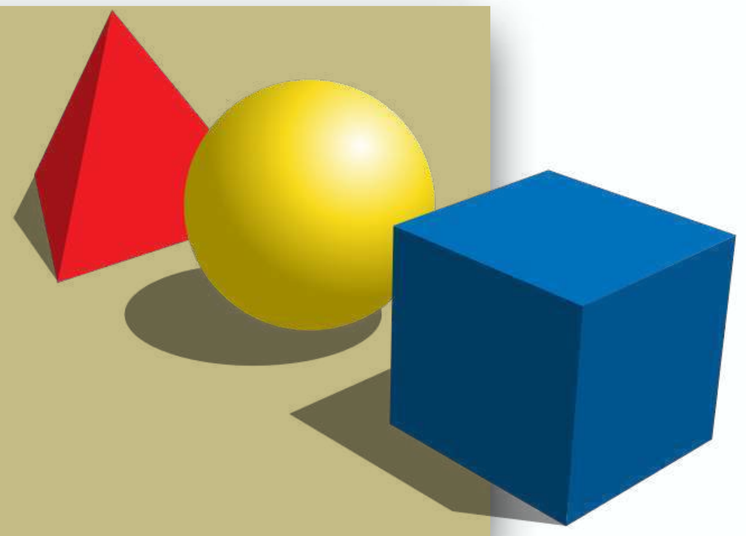
Giacobazzi and Mastroeni. Abstract Non-Interference: A Unifying Framework for Weakening Information-Flow. In TOPS, 2018.

Abstract Interpretation

3-Step Recipe

practical tools
targeting specific programs

abstract semantics, abstract domains
algorithmic approaches to decide program properties

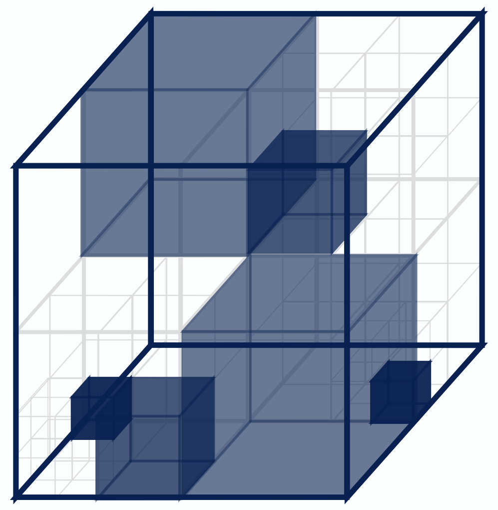


concrete semantics
mathematical models of the program behavior

Hyperproperty Verification [Urban20]

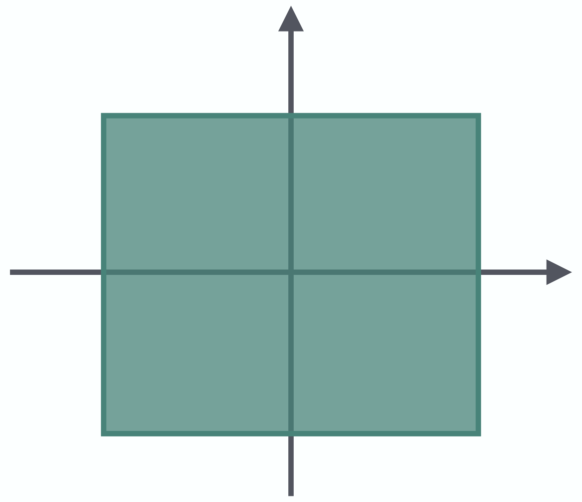
Static Forward Analysis

```
x00 = float(input())
x01 = float(input())
x02 = float(input())
x03 = float(input())
x04 = float(input())
x05 = float(input())

x10 = ReLU((0.120875)*x00 + (0.065404)*x01 + (0.097862)*x02 + (2.030051)*x03 + (0.101956)*x04 + (-2.103565)*x05 + (1.623834))
x11 = ReLU((0.113805)*x00 + (0.064486)*x01 + (0.090701)*x02 + (2.123338)*x03 + (0.076374)*x04 + (-1.651132)*x05 + (-0.828711))
x12 = ReLU((0.755487)*x00 + (0.224640)*x01 + (0.344943)*x02 + (2.619876)*x03 + (0.346636)*x04 + (1.418635)*x05 + (-0.686885))

x20 = ReLU((1.803209)*x10 + (1.222249)*x11 + (2.725716)*x12 + (-3.489653))
x21 = ReLU((1.958950)*x10 + (2.388245)*x11 + (2.245851)*x12 + (-3.834811))
x22 = ReLU((1.958103)*x10 + (2.273354)*x11 + (0.662405)*x12 + (-4.211086))

x30 = ReLU((1.735994)*x20 + (0.666507)*x21 + (3.192344)*x22 + (-2.627086))
x31 = ReLU((2.327110)*x20 + (2.685314)*x21 + (1.424807)*x22 + (-3.695113))
x32 = ReLU((2.147212)*x20 + (2.285599)*x21 + (2.665507)*x22 + (-4.299974))

x40 = ReLU((2.296390)*x30 + (1.980387)*x31 + (2.945360)*x32 + (-4.096463))
x41 = ReLU((-0.552155)*x30 + (-0.828226)*x31 + (-0.495998)*x32)
x42 = ReLU((-2.509773)*x30 + (1.199384)*x31 + (-0.245429)*x32 + (5.024773))

x50 = (-2.278012)*x40 + (0.180652)*x41 + (-16.663048)*x42 + (1864)
x51 = (2.278012)*x40 + (-0.180652)*x41 + (16.663048)*x42 + (-1864)
```



① start from a **partition** of the input space

② proceed **forwards** **in parallel** from all partitions



③ check output for:
- **unique classification outcome** → ✓ **safe**
- **abstract activation pattern**

Static Forward Analysis

Symbolic & DeepPoly Product Abstract Domain

```
x00 = float(input())
x01 = float(input())
x02 = float(input())
x03 = float(input())
x04 = float(input())
x05 = float(input())

x10 = ReLU((0.120875)*x00 + (0.065404)*x01 + (0.09…62)*x02 + (2.03…51)*x03 + (0.101956)*x04 + (-2.103565)*x05 + (1.623834))
x11 = ReLU((0.113805)*x00 + (0.064486)*x01 + (0.09…01)*x02 + (2.12…38)*x03 + (0.076374)*x04 + (-1.651132)*x05 + (-0.828711))
x12 = ReLU((0.755487)*x00 + (0.224640)*x01 + (0.34…43)*x02 + (2.61…76)*x03 + (0.346636)*x04 + (1.418635)*x05 + (-0.686885))

x20 = ReLU((1.803209)*x10 + (1.222249)*x11 + (2.72…16)*x12 + (-3.4…653))
x21 = ReLU((1.958950)*x10 + (2.388245)*x11 + (2.24…51)*x12 + (-3.8…811))
x22 = ReLU((1.958103)*x10 + (2.273354)*x11 + (0.66…05)*x12 + (-4.2…086))

x30 = ReLU((1.735994)*x20 + (0.666507)*x21 + (3.19…44)*x22 + (-2.6…086))
x31 = ReLU((2.327110)*x20 + (2.685314)*x21 + (1.42…07)*x22 + (-3.6…113))
x32 = ReLU((2.147212)*x20 + (2.285599)*x21 + (2.66…07)*x22 + (-4.2…974))

x40 = ReLU((2.296390)*x30 + (1.980387)*x31 + (2.94…60)*x32 + (-4.0…463))
x41 = ReLU((-0.552155)*x30 + (-0.828226)*x31 + (-0…95998)*x32)
x42 = ReLU((-2.509773)*x30 + (1.199384)*x31 + (-0.…5429)*x32 + (5.…4773))

x50 = (-2.278012)*x40 + (0.180652)*x41 + (-16.663048)*x42 + (1864)
x51 = (2.278012)*x40 + (-0.180652)*x41 + (16.663048)*x42 + (-1864)
```

η:
- x00: [-1, 1]
- x01: [-1, 1]
- x02: T
- x03: [-1, 0]
- x04: [-1, 1]
- x05: [-1, 1]

Activation pattern: 1, 1, ?, 1, 1, 1, 1, 1, 1, 1, 0, ?

ρ:
- x50: [1, 1]
- x51: [0, 0]

η:
- x00: [0, 1]
- x01: [-1, 0]
- x02: T
- x03: [0.5, 1]
- x04: [0, 1]
- x05: [-1, 0]

ρ:
- x50: [0, 1]
- x51: [0, 1]

η:
- x00: [0, 1]
- x01: [0, 1]
- x02: T
- x03: [0.5, 1]
- x04: [0, 1]
- x05: [-1, 0]

Activation pattern: 1, 1, ?, 1, 1, 1, 1, 1, 1, 1, 0, ?

ρ:
- x50: [0, 1]
- x51: [0, 1]

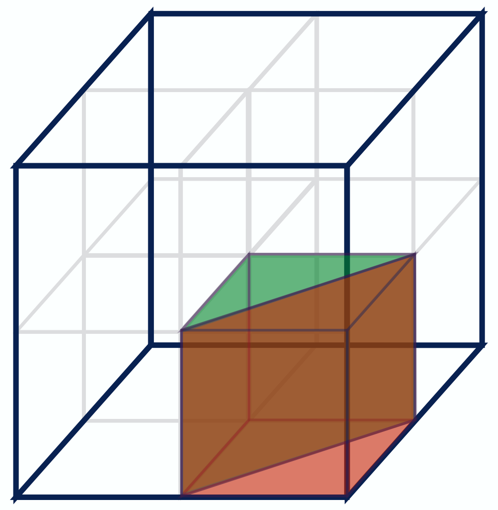
...

several partitions share the same abstract activation pattern

Hyperproperty Verification [Urban20]

Static Backward Analysis

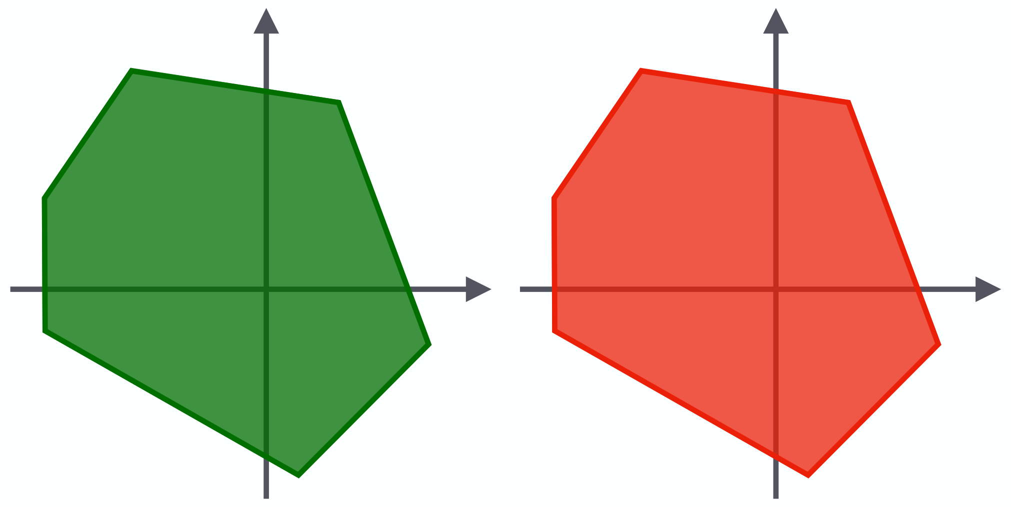
```
x00 = float(input())
x01 = float(input())
x02 = float(input())
x03 = float(input())
x04 = float(input())
x05 = float(input())

x10 = ReLU((0.120875)*x00 + (0.065404)*x01 + (0.097862)*x02 + (2.030051)*x03 + (0.101956)*x04 + (-2.103565)*x05 + (1.623834))
x11 = ReLU((0.113805)*x00 + (0.064486)*x01 + (0.090701)*x02 + (2.123338)*x03 + (0.076374)*x04 + (-1.651132)*x05 + (-0.828711))
x12 = ReLU((0.755487)*x00 + (0.224640)*x01 + (0.344943)*x02 + (2.619876)*x03 + (0.346636)*x04 + (1.418635)*x05 + (-0.686885))

x20 = ReLU((1.803209)*x10 + (1.222249)*x11 + (2.725716)*x12 + (-3.489653))
x21 = ReLU((1.958950)*x10 + (2.388245)*x11 + (2.245851)*x12 + (-3.834811))
x22 = ReLU((1.958103)*x10 + (2.273354)*x11 + (0.662405)*x12 + (-4.211086))

x30 = ReLU((1.735994)*x20 + (0.666507)*x21 + (3.192344)*x22 + (-2.627086))
x31 = ReLU((2.327110)*x20 + (2.685314)*x21 + (1.424807)*x22 + (-3.695113))
x32 = ReLU((2.147212)*x20 + (2.285599)*x21 + (2.665507)*x22 + (-4.299974))

x40 = ReLU((2.296390)*x30 + (1.980387)*x31 + (2.945360)*x32 + (-4.096463))
x41 = ReLU((-0.552155)*x30 + (-0.828226)*x31 + (-0.495998)*x32)
x42 = ReLU((-2.509773)*x30 + (1.199384)*x31 + (-0.245429)*x32 + (5.024773))

x50 = (-2.278012)*x40 + (0.180652)*x41 + (-16.663048)*x42 + (1864)
x51 = (2.278012)*x40 + (-0.180652)*x41 + (16.663048)*x42 + (-1864)
```



① check for **disjunction** in corresponding **input partitions**:
disjoint → ✔ **safe**
otherwise → 🚨 **alarm**

② proceed **backwards** in parallel **for each abstract activation pattern**

① start from an **abstraction** for each possible classification outcome



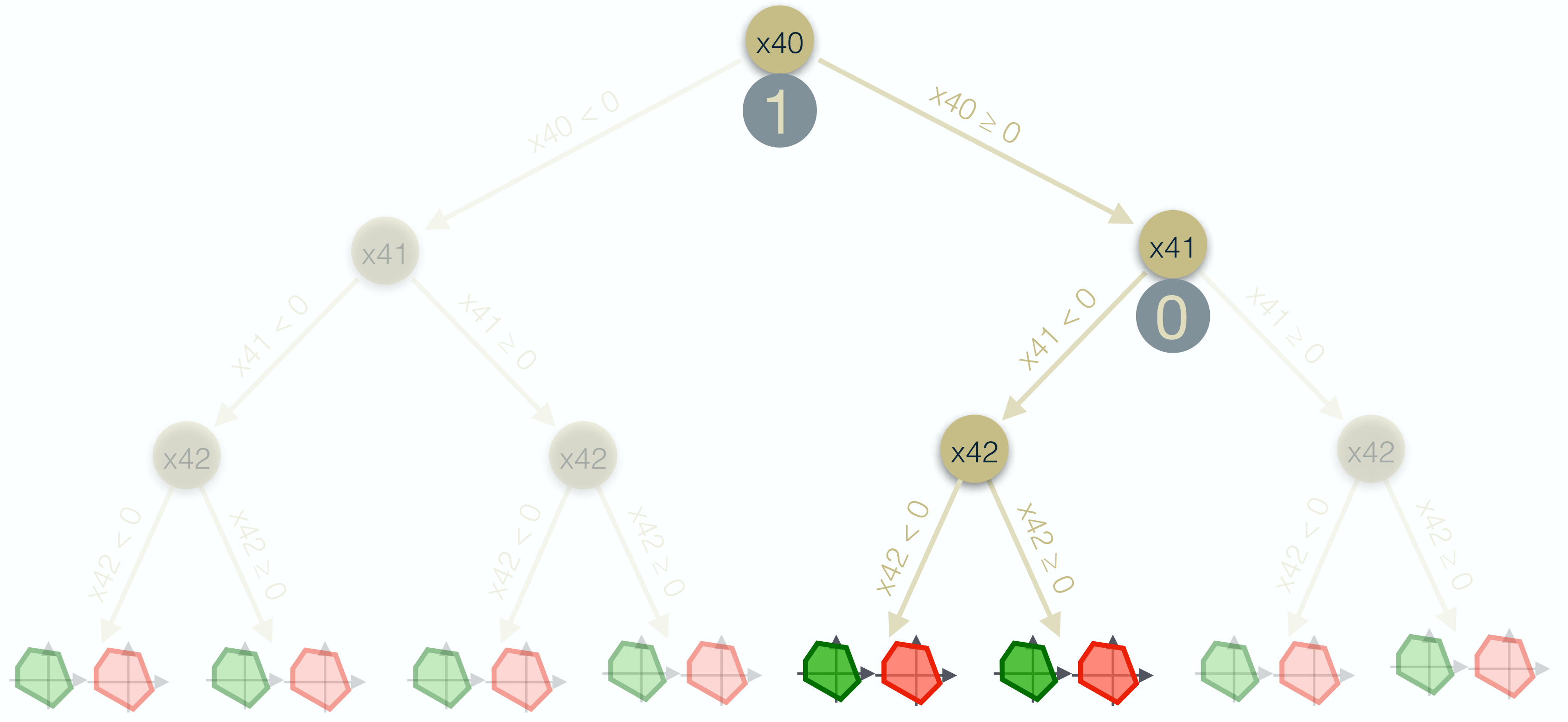
Static Backward Analysis

Symbolic & DeepPoly Product Abstract Domain

```
x00 = float(input())
x01 = float(input())
x02 = float(input())
x03 = float(input())
x04 = float(input())
x05 = float(input())
```

η:	η:
x00: [0, 1]	x00: [0, 1]
x01: [-1, 0]	x01: [0, 1]
x02: ⊤	x02: ⊤
x03: [0.5, 1]	x03: [0.5, 1]
x04: [0, 1]	x04: [0, 1]
x05: [-1, 0]	x05: [-1, 0]

```
1 x10 = ReLU((0.120875)*x00 + (0.065404)*x01 + (0.097862)*x02 + (2.030051)*x03 + (0.101956)*x04 + (-2.103565)*x05 + (1.623834))
1 x11 = ReLU((0.113805)*x00 + (0.064486)*x01 + (0.090701)*x02 + (2.123338)*x03 + (0.076374)*x04 + (-1.651132)*x05 + (-0.828711))
? x12 = ReLU((0.755487)*x00 + (0.224640)*x01 + (0.344943)*x02 + (2.619876)*x03 + (0.346636)*x04 + (1.418635)*x05 + (-0.686885))

1 x20 = ReLU((1.803209)*x10 + (1.222249)*x11 + (2.725716)*x12 + (-3.489653))
1 x21 = ReLU((1.958950)*x10 + (2.388245)*x11 + (2.245851)*x12 + (-3.834811))
1 x22 = ReLU((1.958103)*x10 + (2.273354)*x11 + (0.662405)*x12 + (-4.211086))

1 x30 = ReLU((1.735994)*x20 + (0.666507)*x21 + (3.192344)*x22 + (-2.627086))
1 x31 = ReLU((2.327110)*x20 + (2.685314)*x21 + (1.424807)*x22 + (-3.695113))
1 x32 = ReLU((2.147212)*x20 + (2.285599)*x21 + (2.665507)*x22 + (-4.299974))

1 x40 = ReLU((2.296390)*x30 + (1.980387)*x31 + (2.945360)*x32 + (-4.096463))
0 x41 = ReLU((-0.552155)*x30 + (-0.828226)*x31 + (-0.495998)*x32)
? x42 = ReLU((-2.509773)*x30 + (1.199384)*x31 + (-0.245429)*x32 + (5.024773))

x50 = (-2.278012)*x40 + (0.180652)*x41 + (-16.663048)*x42 + (1864)
x51 = (2.278012)*x40 + (-0.180652)*x41 + (16.663048)*x42 + (-1864)
```



$(-4.556024) * x40 + (0.361304) * x41 + (-33.326096) * x42 + (3728) > 0$

$(4.556024) * x40 + (33.326096) * x42 - 3728 > 0$

Static Backward Analysis

Symbolic & DeepPoly Product Abstract Domain

```
x00 = float(input())
x01 = float(input())
x02 = float(input())
x03 = float(input())
x04 = float(input())
x05 = float(input())
```

η:
x00: [0, 1]
x01: [-1, 0]
x02: T
x03: [0.5, 1]
x04: [0, 1]
x05: [-1, 0]
✓

η:
x00: [0, 1]
x01: [0, 1]
x02: T
x03: [0.5, 1]
x04: [0, 1]
x05: [-1, 0]
✗

counterexample

x00: 1	x00: 1
x01: 1	x01: 1
x02: -1	x02: 1
x03: 1	x03: 1
x04: 1	x04: 1
x05: -1	x05: -1

```
1 x10 = ReLU((0.120875)*x00 + (0.065404)*x01 + (0.097862)*x02 + (2.030051)*x03 + (0.101956)*x04 + (-2.103565)*x05 + (1.623834))
1 x11 = ReLU((0.113805)*x00 + (0.064486)*x01 + (0.090701)*x02 + (2.123338)*x03 + (0.076374)*x04 + (-1.651132)*x05 + (-0.828711))
? x12 = ReLU((0.755487)*x00 + (0.224640)*x01 + (0.344943)*x02 + (2.619876)*x03 + (0.346636)*x04 + (1.418635)*x05 + (-0.686885))

1 x20 = ReLU((1.803209)*x10 + (1.222249)*x11 + (2.725716)*x12 + (-3.489653))
1 x21 = ReLU((1.958950)*x10 + (2.388245)*x11 + (2.245851)*x12 + (-3.834811))
1 x22 = ReLU((1.958103)*x10 + (2.273354)*x11 + (0.662405)*x12 + (-4.211086))

⋮

1 x40 = ReLU((2.296390)*x30 + (1.980387)*x31 + (2.945360)*x32 + (-4.096463))
0 x41 = ReLU((-0.552155)*x30 + (-0.828226)*x31 + (-0.495998)*x32)
? x42 = ReLU((-2.509773)*x30 + (1.199384)*x31 + (-0.245429)*x32 + (5.024773))

x50 = (-2.278012)*x40 + (0.180652)*x41 + (-16.663048)*x42 + (1864)
x51 = (2.278012)*x40 + (-0.180652)*x41 + (16.663048)*x42 + (-1864)
```

$(-4.556024) * x40 + (0.361304) * x41 + (-33.326096) * x42 + (3728) > 0$

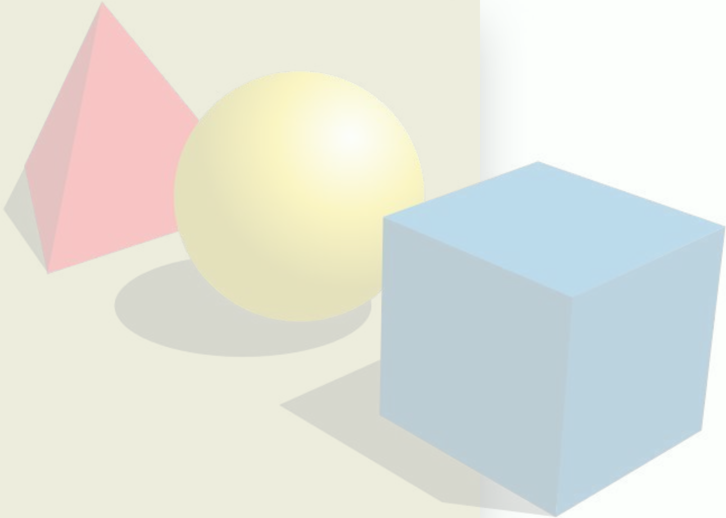
$(4.556024) * x40 + (33.326096) * x42 - 3728 > 0$

Abstract Interpretation

3-Step Recipe

practical tools
targeting specific programs

abstract semantics, abstract domains
algorithmic approaches to decide program properties

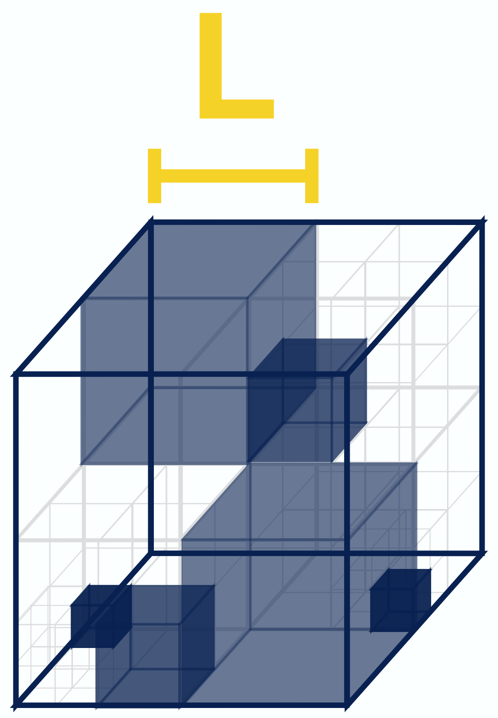


concrete semantics
mathematical models of the program behavior

Hyperproperty Verification [Urban20]

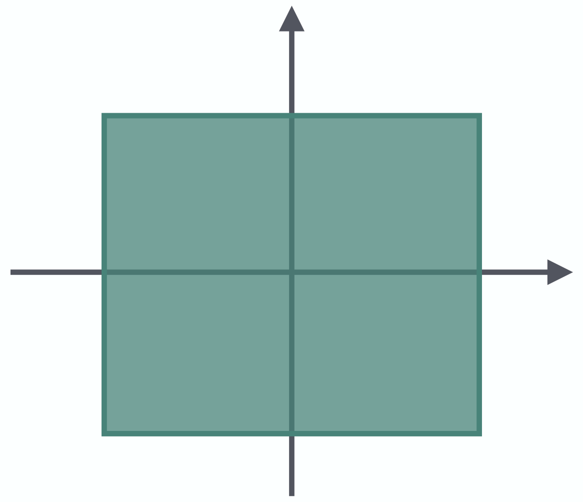
Static Forward Analysis

```
x00 = float(input())
x01 = float(input())
x02 = float(input())
x03 = float(input())
x04 = float(input())
x05 = float(input())

1 x10 = ReLU((0.120875)*x00 + (0.065404)*x01 + (0.097862)*x02 + (2.030051)*x03 + (0.101956)*x04 + (-2.103565)*x05 + (1.623834))
1 x11 = ReLU((0.113805)*x00 + (0.064486)*x01 + (0.090701)*x02 + (2.123338)*x03 + (0.076374)*x04 + (-1.651132)*x05 + (-0.828711))
? x12 = ReLU((0.755487)*x00 + (0.224640)*x01 + (0.344943)*x02 + (2.619876)*x03 + (0.346636)*x04 + (1.418635)*x05 + (-0.686885))

? x20 = ReLU((1.803209)*x10 + (1.222249)*x11 + (2.725716)*x12 + (-3.489653))
? x21 = ReLU((1.958950)*x10 + (2.388245)*x11 + (2.245851)*x12 + (-3.834811))
? x22 = ReLU((1.958103)*x10 + (2.273354)*x11 + (0.662405)*x12 + (-4.211086))

? x30 = ReLU((1.735994)*x20 + (0.666507)*x21 + (3.192344)*x22 + (-2.627086))
1 x31 = ReLU((2.327110)*x20 + (2.685314)*x21 + (1.424807)*x22 + (-3.695113))
0 x32 = ReLU((2.147212)*x20 + (2.285599)*x21 + (2.665507)*x22 + (-4.299974))

1 x40 = ReLU((2.296390)*x30 + (1.980387)*x31 + (2.945360)*x32 + (-4.096463))
0 x41 = ReLU((-0.552155)*x30 + (-0.828226)*x31 + (-0.495998)*x32)
0 x42 = ReLU((-2.509773)*x30 + (1.199384)*x31 + (-0.245429)*x32 + (5.024773))

x50 = (-2.278012)*x40 + (0.180652)*x41 + (-16.663048)*x42 + (1864)
x51 = (2.278012)*x40 + (-0.180652)*x41 + (16.663048)*x42 + (-1864)
```



① **iteratively** partition the input space

② proceed **forwards** **in parallel** from all partitions



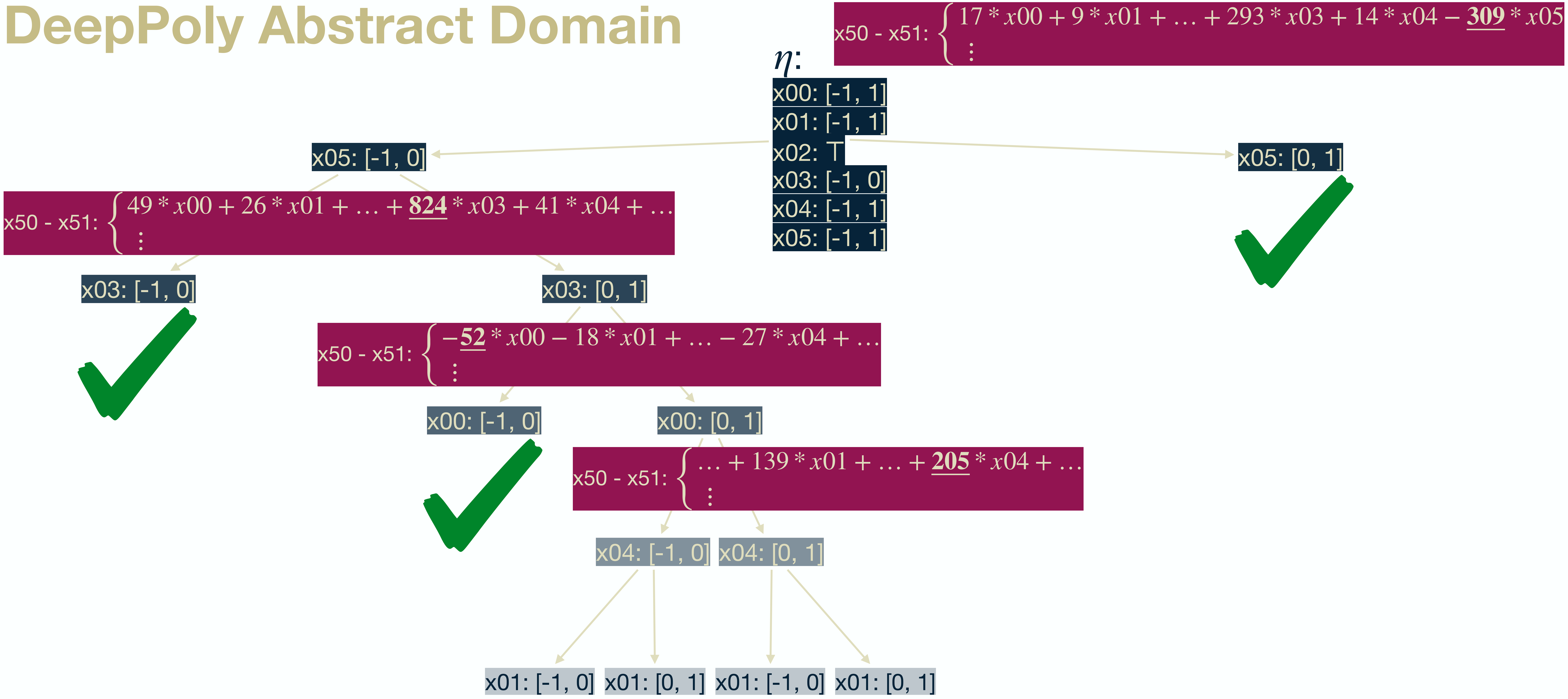
③ check output for:
- **unique classification outcome** → **safe**
- **abstract activation pattern** ∪

Partitioning Strategies: Interval Range

DeepPoly Abstract Domain

Partitioning Strategies: ReCIPH

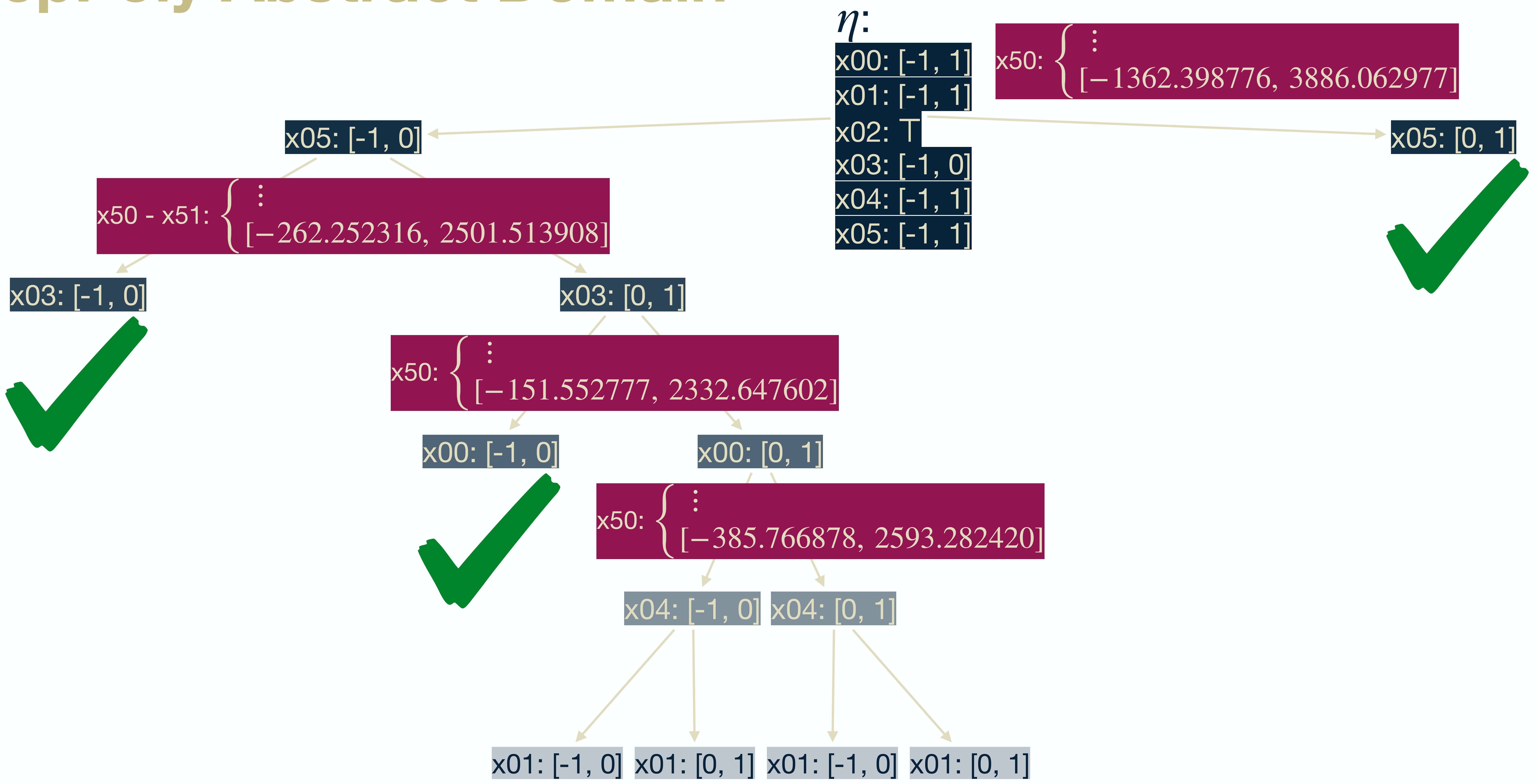
DeepPoly Abstract Domain



Durand, Lemesle, Chihani, CU, and Terrier. ReCIPH: Relational Coefficients for Input Partitioning Heuristic. In WFVML, 2022

Input Refinement $\not\Rightarrow$ Output Refinement

DeepPoly Abstract Domain



Input Refinement ⇏ Output Refinement

DeepPoly with Input Range Partitioning

Scalability-vs-Precision Tradeoff

Analyzed Input Space Percentage

L	U	Boxes	Symbolic	DeepPoly		Product	
				Input Range Partitioning	ReCIPH	Input Range Partitioning	ReCIPH
1	2	46,9 %	46,9 %	68,8 %	87,5 %	90,6 %	90,6 %
	6	46,9 %	46,9 %	68,8 %	87,5 %	90,6 %	90,6 %
0.5	2	76,9 %	89,2 %	100,0 %	100,0 %	100,0 %	100,0 %
	6	84,4 %	89,9 %	100,0 %	100,0 %	100,0 %	100,0 %

Execution Time

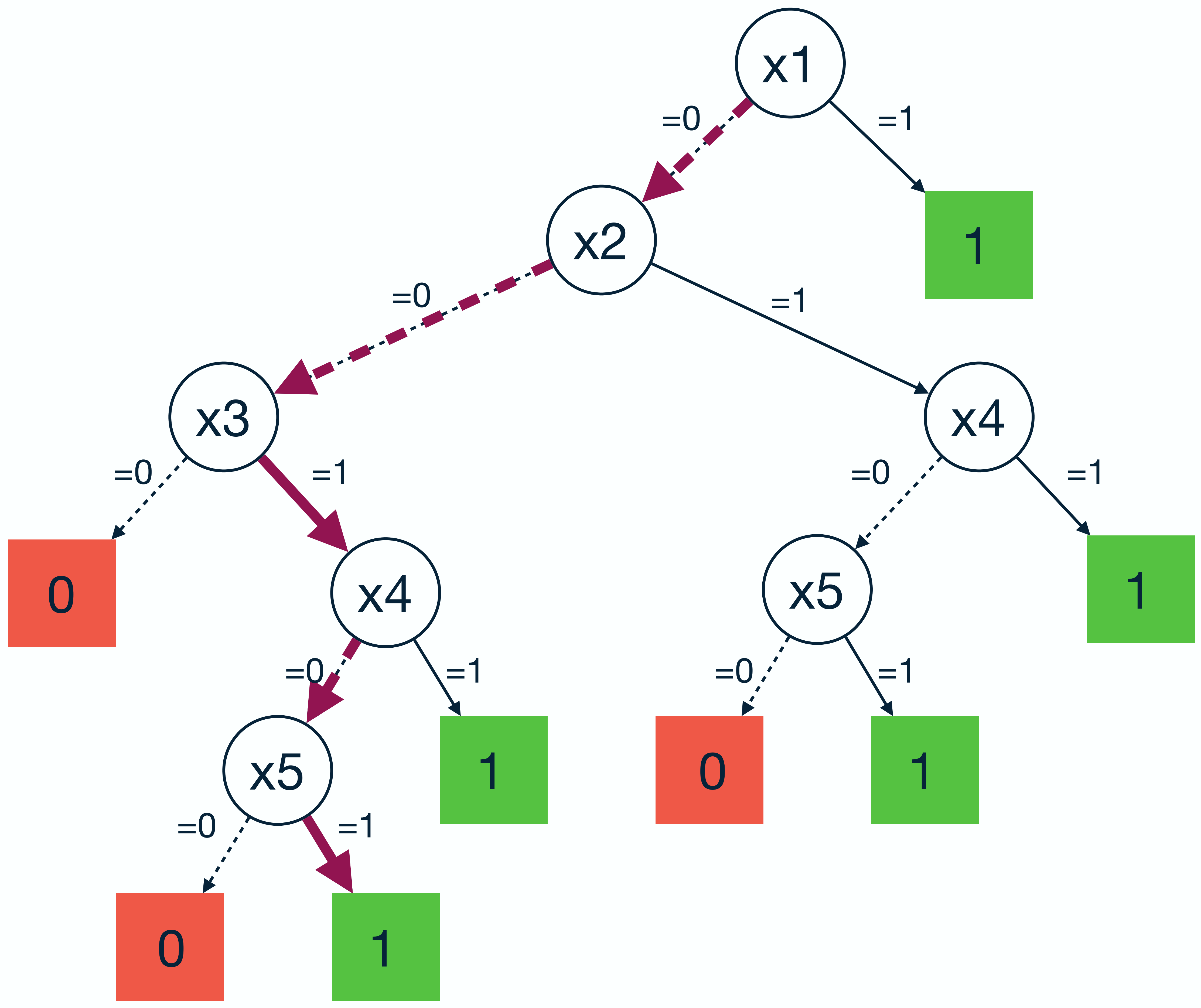
L	U	Boxes	Symbolic	DeepPoly		Product	
				Input Range Partitioning	ReCIPH	Input Range Partitioning	ReCIPH
1	2	0,08s	0,14s	0,26s	0,11s	0,26s	0,12s
	6	0,16s	0,31s	0,51s	0,20s	0,35s	0,20s
0.5	2	8,88s	5,76s	2,60s	1,61s	2,10s	1,61s
	6	64,67s	40,90s	2,65s	1,63s	2,10s	1,62s

Neural Network Verification

Neural Network Explainability

Abductive Explanations (AXp) [Marques-Silva21]

Subset-Minimal Set of Input Features Sufficient for Ensuring Prediction



AXp = { 3, 5 }

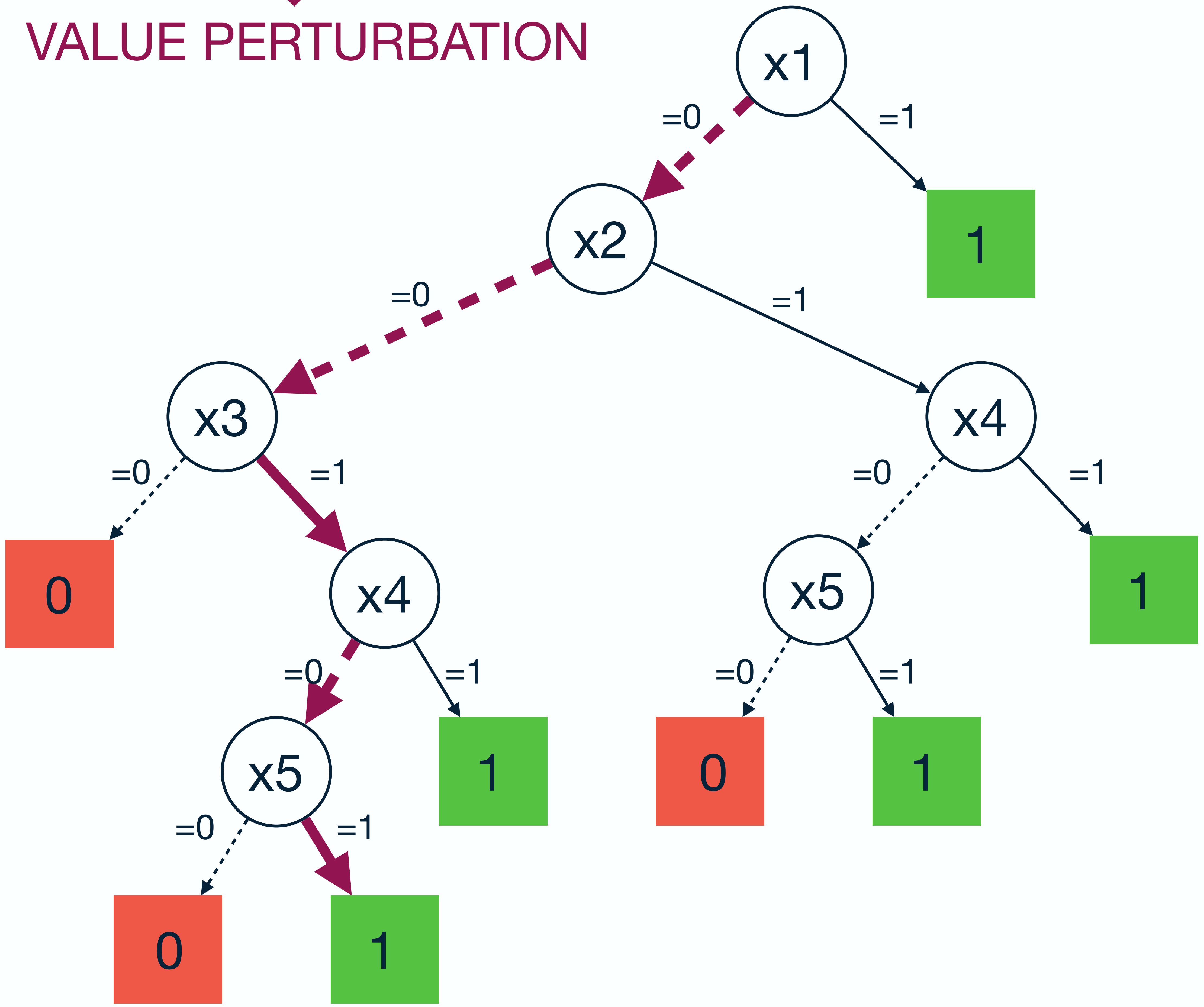
x3	x5	x1	x2	x4		
1	1	0	0	0	→	1
1	1	0	0	1	→	1
1	1	0	1	0	→	1
1	1	0	1	1	→	1
1	1	1	0	0	→	1
1	1	1	0	1	→	1
1	1	1	1	0	→	1
1	1	1	1	1	→	1

Computing One AXp [Marques-Silva21]

Drop (i.e., Free) Input Features While AXp Condition Holds

VALUE PERTURBATION

LOCAL ROBUSTNESS



{ 1, 2, 3, 4, 5 } → 1

Free 1: { 2, 3, 4, 5 } → 1

Free 2: { 3, 4, 5 } → 1

Free 3: { 4, 5 } → ✗

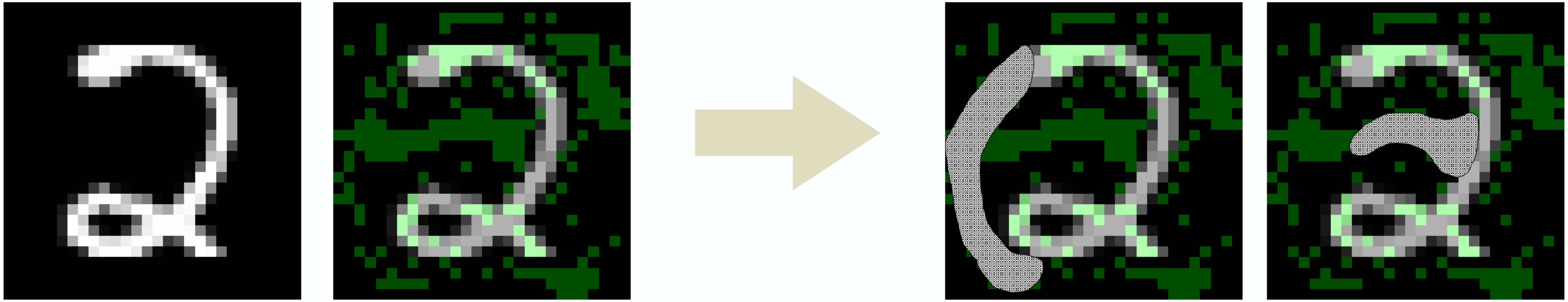
Free 4: { 3, 5 } → 1

Free 5: { 3 } → ✗

AXp = { 3, 5 }

VeriX [Wu23]

Distance-Restricted AXps



(a) Original "2" (c) VeriX (e) "2" into "0" (f) "2" into "3"

Abstract AXps

Example

```
x00 = float(input())
x01 = float(input())
x02 = float(input())
x03 = float(input())
x04 = float(input())
x05 = float(input())
```

x:

x00: 0.75
x01: 1
x02: -0.5
x03: 0.75
x04: -0.25
x05: 0.75

```
x10 = ReLU((0.120875)*x00 + (0.065404)*x01 + (0.097862)*x02 + (2.030051)*x03 + (0.101956)*x04 + (-2.103565)*x05 + (1.623834))
x11 = ReLU((0.113805)*x00 + (0.064486)*x01 + (0.090701)*x02 + (2.123338)*x03 + (0.076374)*x04 + (-1.651132)*x05 + (-0.828711))
x12 = ReLU((0.755487)*x00 + (0.224640)*x01 + (0.344943)*x02 + (2.619876)*x03 + (0.346636)*x04 + (1.418635)*x05 + (-0.686885))

x20 = ReLU((1.803209)*x10 + (1.222249)*x11 + (2.725716)*x12 + (-3.489653))
x21 = ReLU((1.958950)*x10 + (2.388245)*x11 + (2.245851)*x12 + (-3.834811))
x22 = ReLU((1.958103)*x10 + (2.273354)*x11 + (0.662405)*x12 + (-4.211086))

x30 = ReLU((1.735994)*x20 + (0.666507)*x21 + (3.192344)*x22 + (-2.627086))
x31 = ReLU((2.327110)*x20 + (2.685314)*x21 + (1.424807)*x22 + (-3.695113))
x32 = ReLU((2.147212)*x20 + (2.285599)*x21 + (2.665507)*x22 + (-4.299974))

x40 = ReLU((2.296390)*x30 + (1.980387)*x31 + (2.945360)*x32 + (-4.096463))
x41 = ReLU((-0.552155)*x30 + (-0.828226)*x31 + (-0.495998)*x32)
x42 = ReLU((-2.509773)*x30 + (1.199384)*x31 + (-0.245429)*x32 + (5.024773))

x50 = (-2.278012)*x40 + (0.180652)*x41 + (-16.663048)*x42 + (1864)
x51 = (2.278012)*x40 + (-0.180652)*x41 + (16.663048)*x42 + (-1864)
```

Abstract AXps

BOXES

{ x03, x05 }
{ x02, x04, x05 }

DEEPPOLY

{ x03 }
{ x05 }

SYMBOLIC

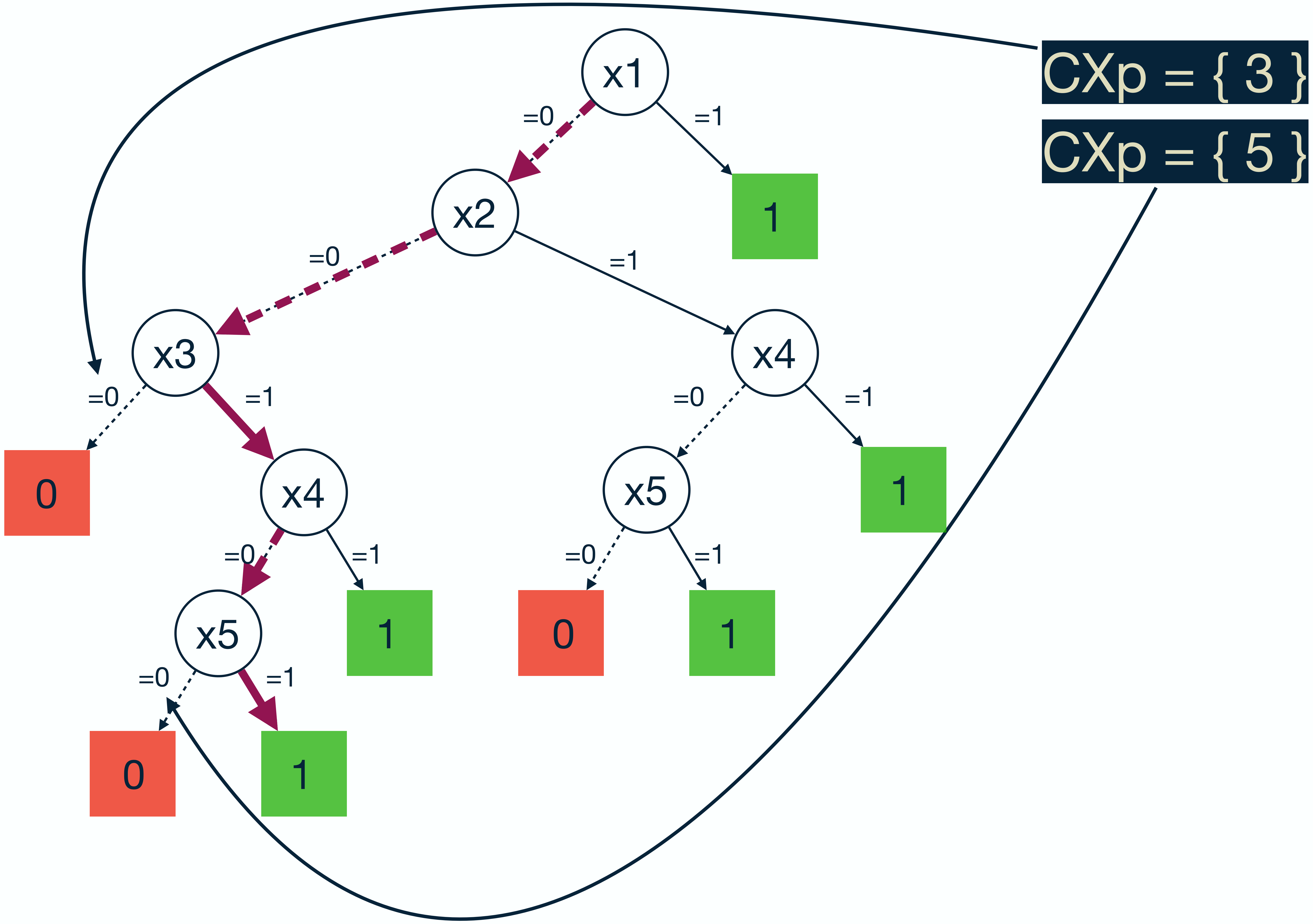
{ x00, x01, x02, x03 }
{ x03, x05 }
{ x02, x04, x05 }
{ x00, x01, x03, x04 }

PRODUCT

{ x00, x02, x04 }
{ x03 }
{ x05 }

Contrastive Explanations (CXp) [Marques-Silva21]

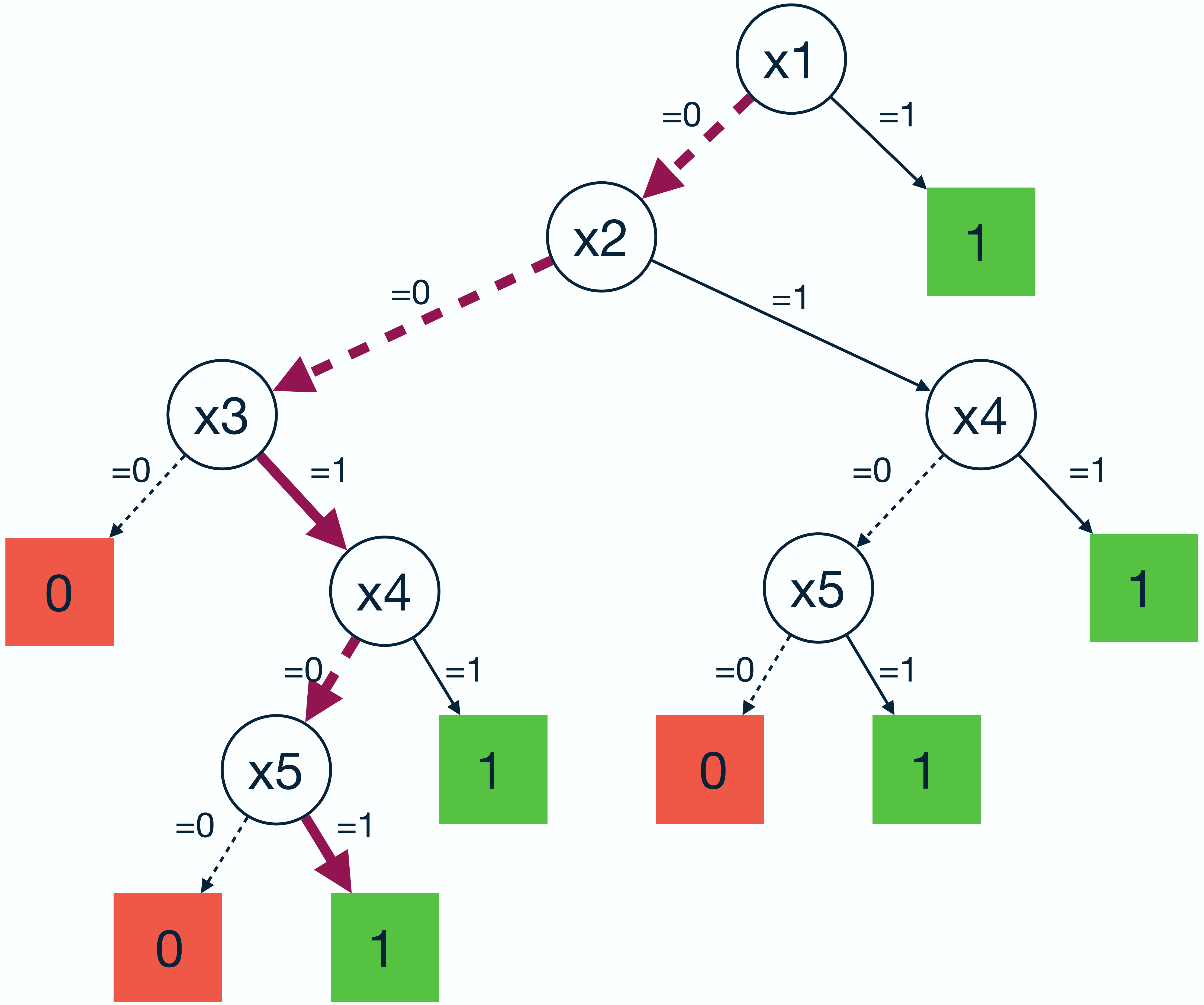
Subset-Minimal Set of Input Features Sufficient for Changing Prediction



Computing One CXp [Marques-Silva21]

Drop (i.e., Fix) Input Features While CXp Condition Holds

¬ (LOCAL ROBUSTNESS)



{ 1, 2, 3, 4, 5 } → 1 0

Fix 1: { 2, 3, 4, 5 } → 1 0

Fix 2: { 3, 4, 5 } → 1 0

Fix 3: { 4, 5 } → 1 0

Fix 4: { 5 } → 1 0

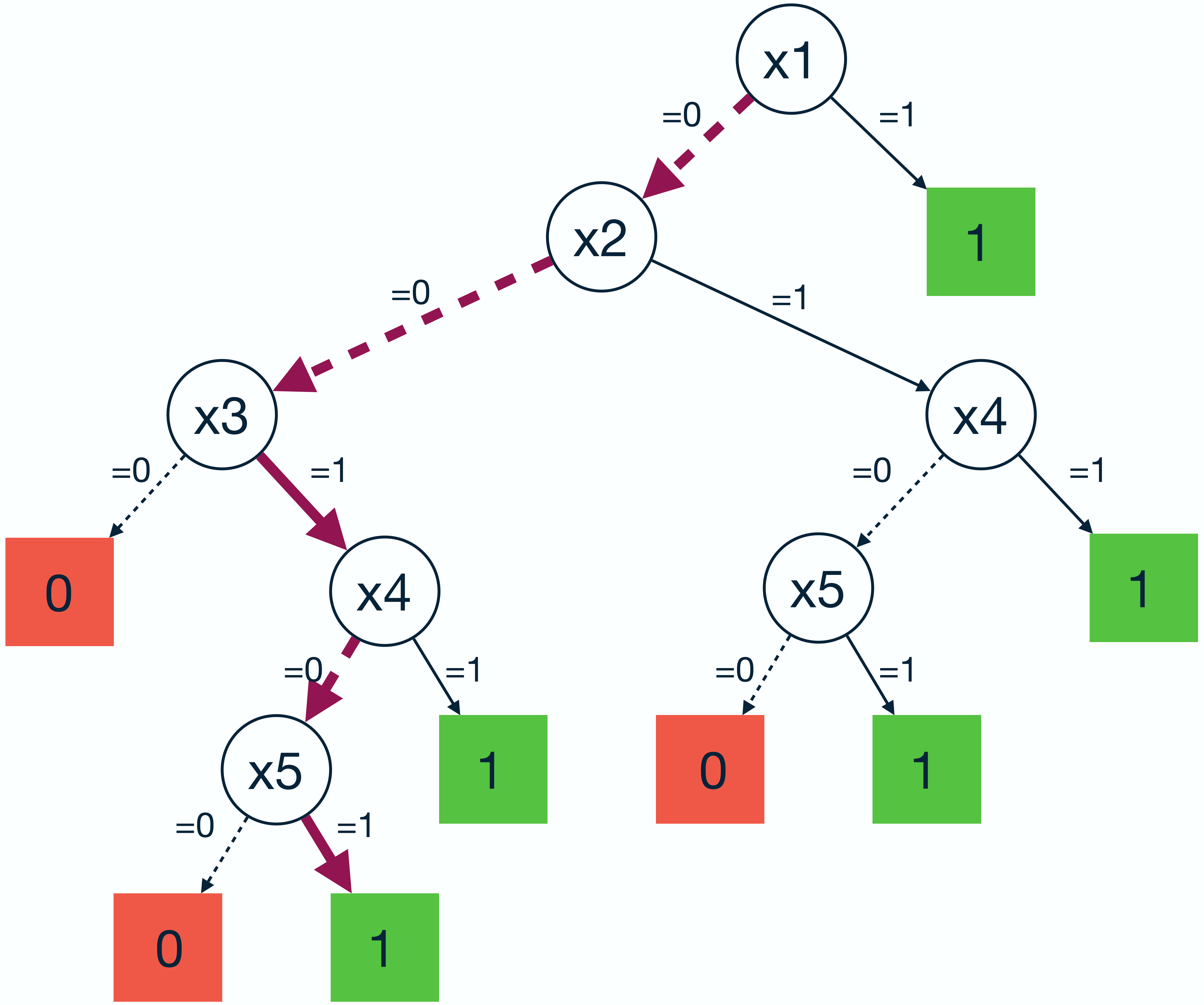
Fix 5: ∅ → ✗

CXp = { 5 }

Computing One CXp [Marques-Silva21]

Drop (i.e., Fix) Input Features While CXp Condition Holds

¬ (LOCAL ROBUSTNESS)



{ 1, 2, 3, 4, 5 } → 1 0

Fix 5: { 1, 2, 3, 4 } → 1 0

Fix 4: { 1, 2, 3 } → 1 0

Fix 3: { 1, 2 } → ✗

Fix 2: { 1, 3 } → 1 0

Fix 1: { 3 } → 1 0

CXp = { 3 }

Abstract CXps

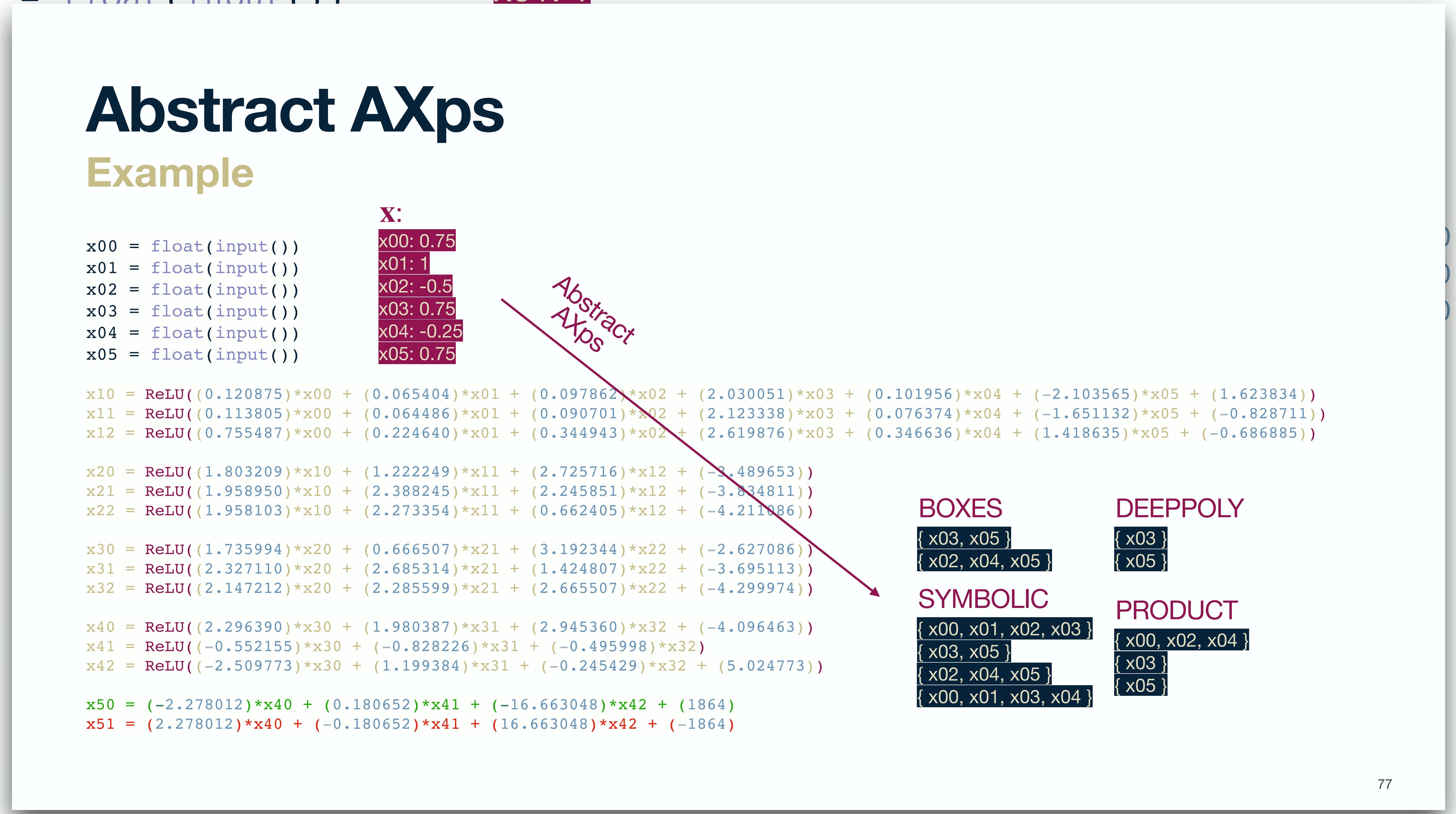
Example

X:

x00: 0.75

x01: 1

```
x00 = float(input())
x01 = float(input())
x02
x03
x04
x05

x10                                                  ).101956)*x04 + (-2.103565)*x05 + (1.623834))
x11                                                  ).076374)*x04 + (-1.651132)*x05 + (-0.828711))
x12                                                  ).346636)*x04 + (1.418635)*x05 + (-0.686885))

x20
x21
x22

x30
x31
x32

x40
x41
x42

x50 = (-2.278012)*x40 + (0.180652)*x41 + (-16.663048)*x42 + (1864)
x51 = (2.278012)*x40 + (-0.180652)*x41 + (16.663048)*x42 + (-1864)
```



BOXES

{ x05 }

{ x03, x04 }

{ x02, x03 }

DEEPPOLY

{ x03, x05 }

SYMBOLIC

{ x03, x04 }

{ x02, x03 }

{ x02, x04, x05 }

{ x00, x05 }

{ x01, x05 }

{ x03, x05 }

PRODUCT

{ x02, x03, x05 }

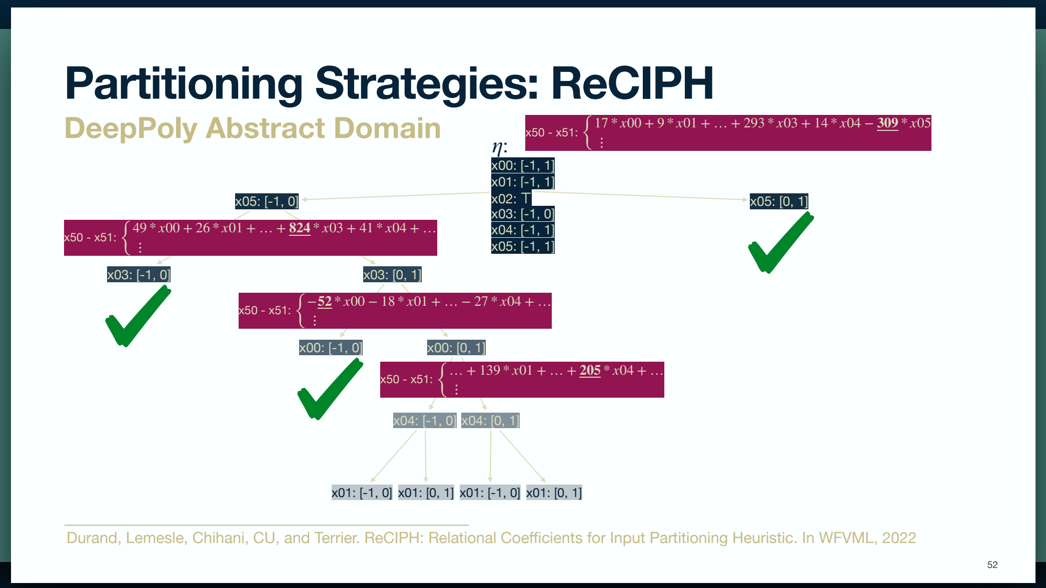
{ x00, x03, x05 }

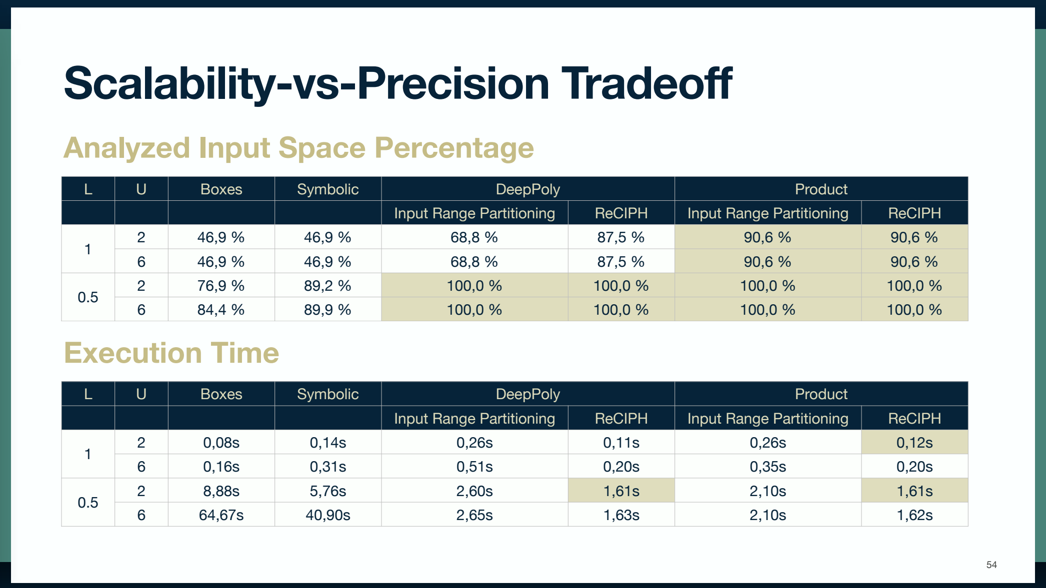
{ x03, x04, x05 }

Verification and Explainability

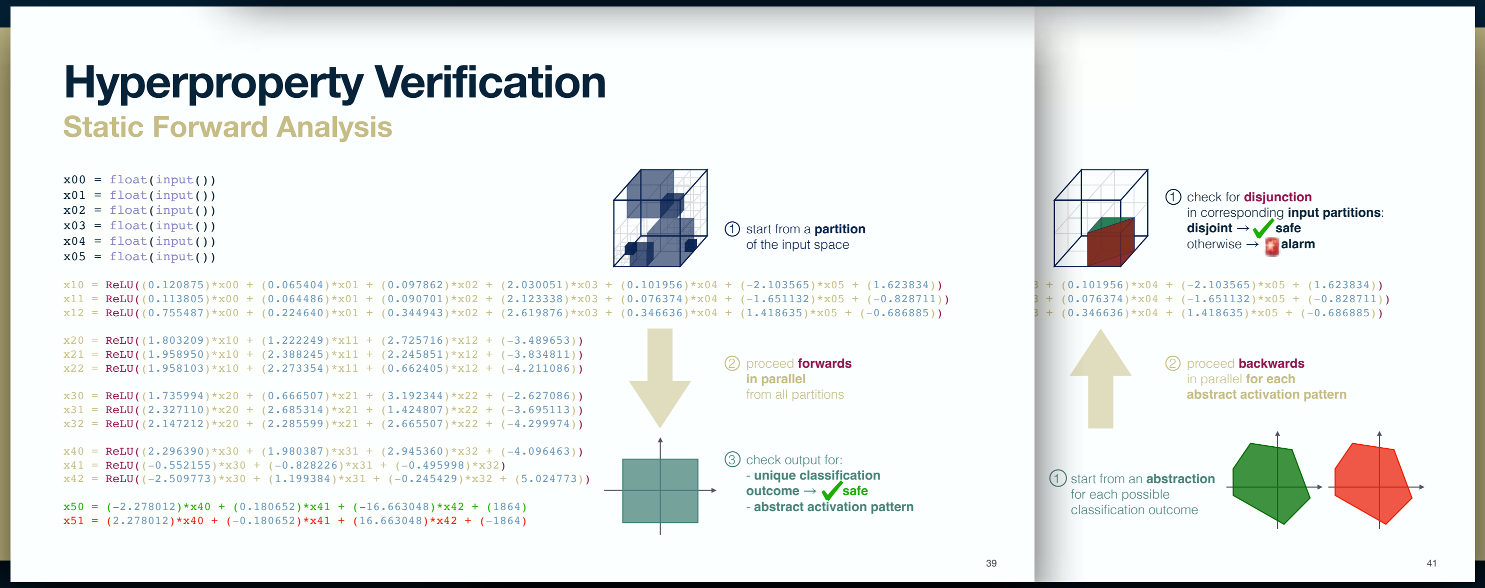
Safety-Critical Neural Networks

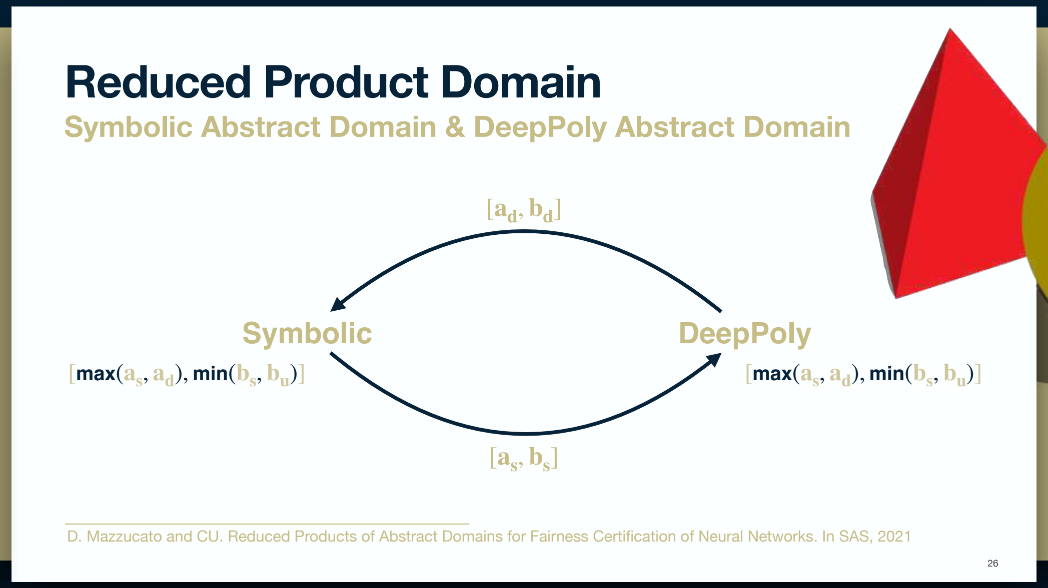
practical tools
targeting specific programs

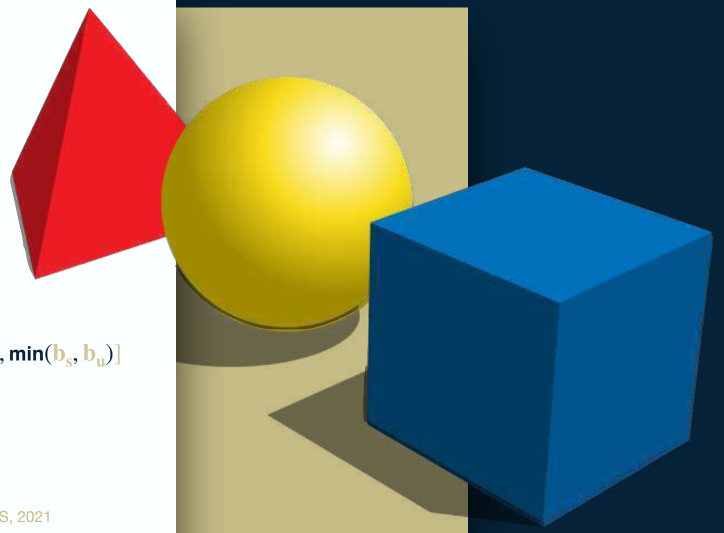




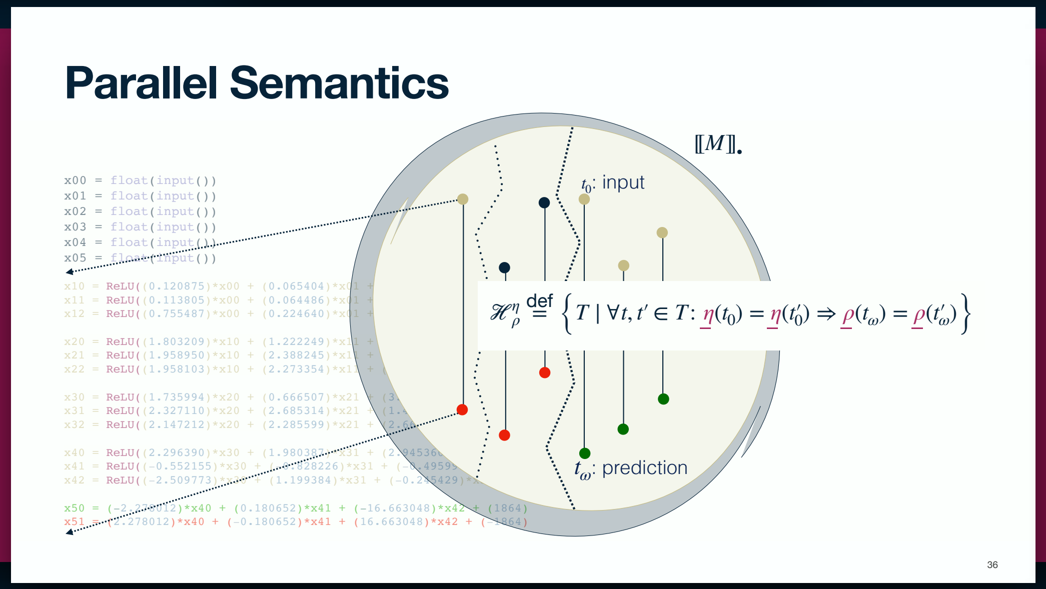
algorithmic approaches
to decide program properties

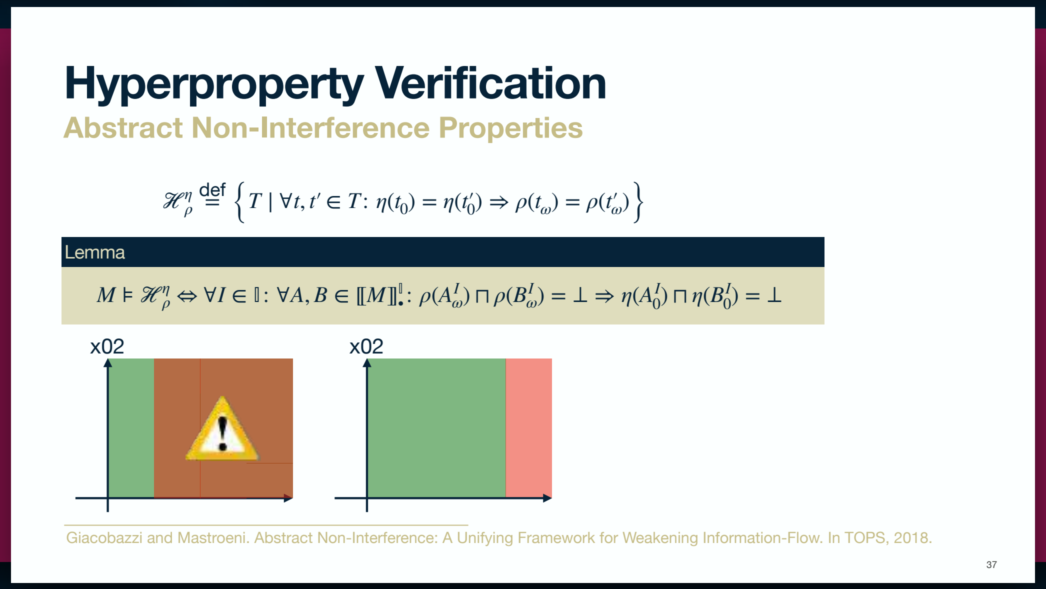






mathematical models
of the program behavior





THANKS!

References

[Li19] **Jianlin Li, Jiangchao Liu, Pengfei Yang, Liqian Chen, Xiaowei Huang, and Lijun Zhang**. Analyzing Deep Neural Networks with Symbolic Propagation: Towards Higher Precision and Faster Verification. In SAS, page 296–319, 2019.
symbolic abstraction

[Singh19] **Gagandeep Singh, Timon Gehr, Markus Püschel, and Martin T. Vechev**. An Abstract Domain for Certifying Neural Networks. In POPL, pages 41:1 - 41:30, 2019.
deeppoly abstraction

[Urban20] **Caterina Urban, Maria Christakis, Valentin Wüstholz, and Fuyuan Zhang**. Perfectly Parallel Fairness Certification of Neural Networks. In OOPSLA, pages 185:1–185:30, 2020.
hypersafety verification

[Marques-Silva21] **João Marques-Silva, Thomas Gerspacher, Martin C. Cooper, Alexey Ignatiev, and Nina Narodytska**. Explanations for Monotonic Classifiers. In ICML, pages 7469-7479, 2021.
[Wu23] **Min Wu, Haoze Wu, Clark W. Barrett**. VeriX: Towards Verified Explainability of Deep Neural Networks. In NeurIPS, 2023.
logic-based explanations