Abstract Interpretation-Based Certification of Hyperproperties for High-Stakes Machine Learning Software

13th Static Analysis Symposium (SAS 2024)

Caterina Urban
Inria & École Normale Supérieure | Université PSL

Abstract Interpretation-Based Certification of Hyperproperties for High-Stakes Machine Learning Software

= Machine Learning-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

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

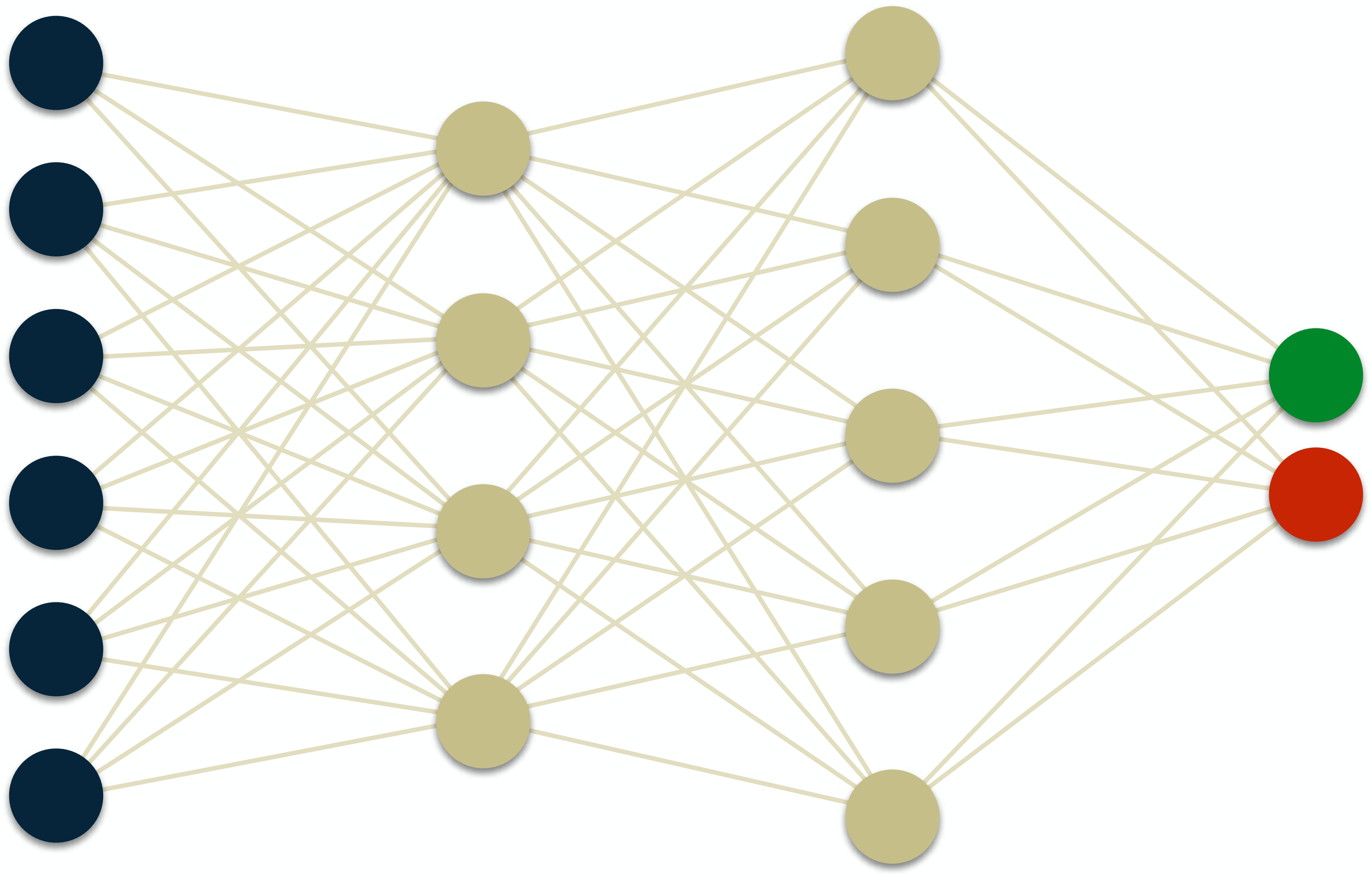
‘26.205 Runway overrun awareness and alerting systems

(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

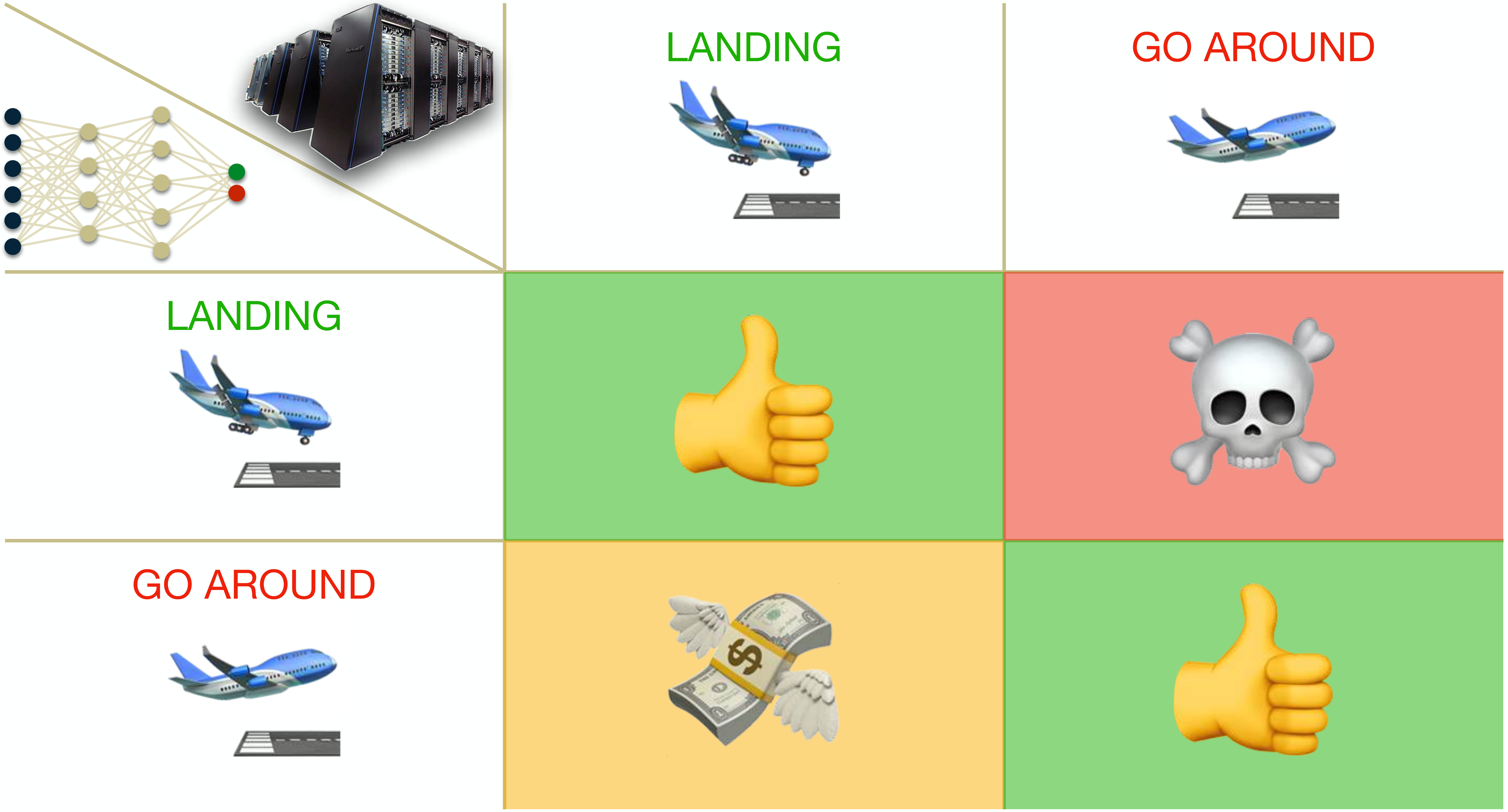
Neural Network Surrogates

Less Computing Power and Less Computing Time



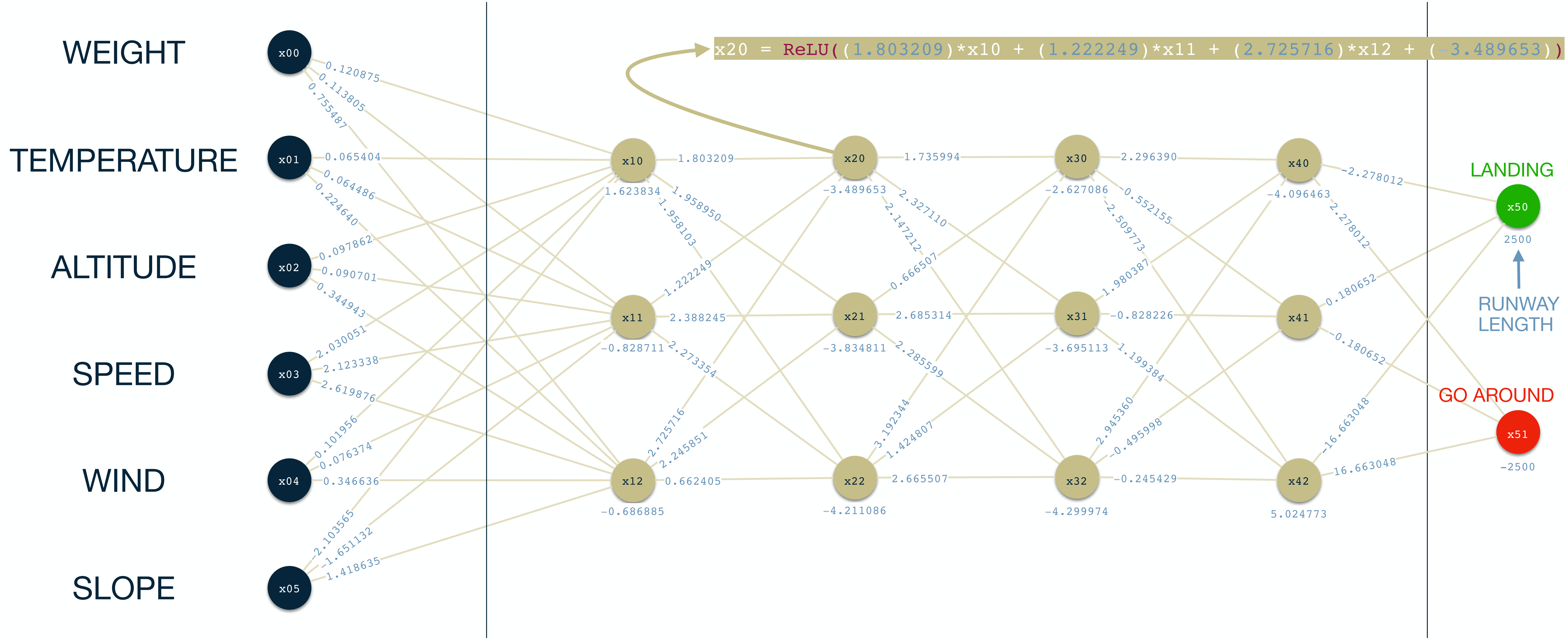
Runway Overrun Warning

Safety of Neural Network Surrogate



Runway Overrun Warning

Toy Example



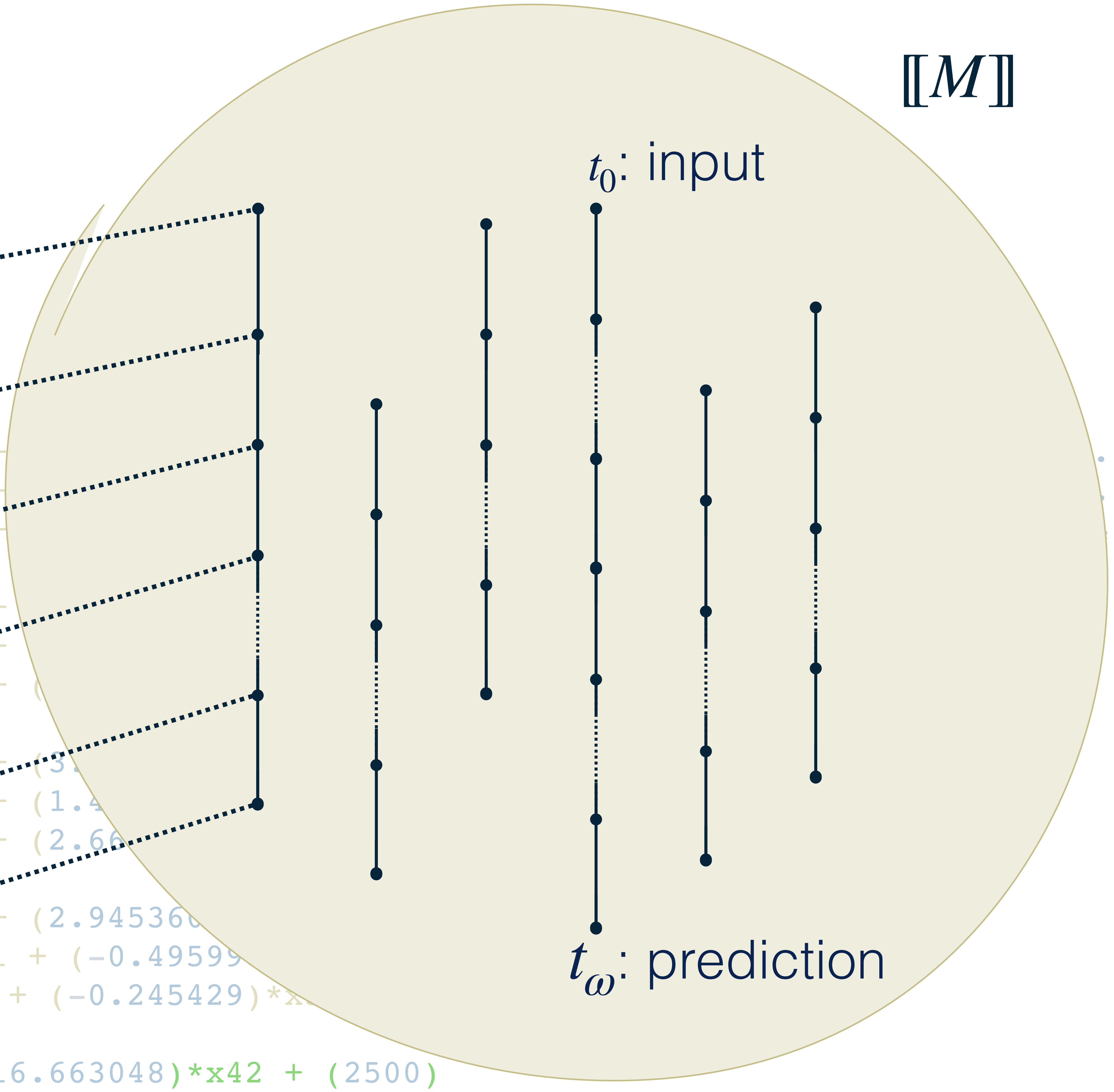
Runway Overrun Warning

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)
```

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)
```



Safety Verification

Extensional Properties

$\mathbf{I}$: input specification

$\mathbf{O}$: output specification

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

$\mathcal{S}^{\mathbf{I}}_{\mathbf{O}}$ is the set of all executions that **satisfy** the specification

Theorem

$$M \vDash \mathcal{S}^{\mathbf{I}}_{\mathbf{O}} \Leftrightarrow [\![M]\!] \subseteq \mathcal{S}^{\mathbf{I}}_{\mathbf{O}}$$

Corollary

$$M \vDash \mathcal{S}^{\mathbf{I}}_{\mathbf{O}} \Leftarrow [\![M]\!] \subseteq [\![M]\!]^{\natural} \subseteq \mathcal{S}^{\mathbf{I}}_{\mathbf{O}}$$

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

I:

$-1 \leq x00 \leq 1$
$-1 \leq x01 \leq 1$
$-1 \leq x02 \leq 1$
$-1 \leq x03 \leq 1$
$-1 \leq x04 \leq 1$
$-1 \leq x05 \leq 1$

O:

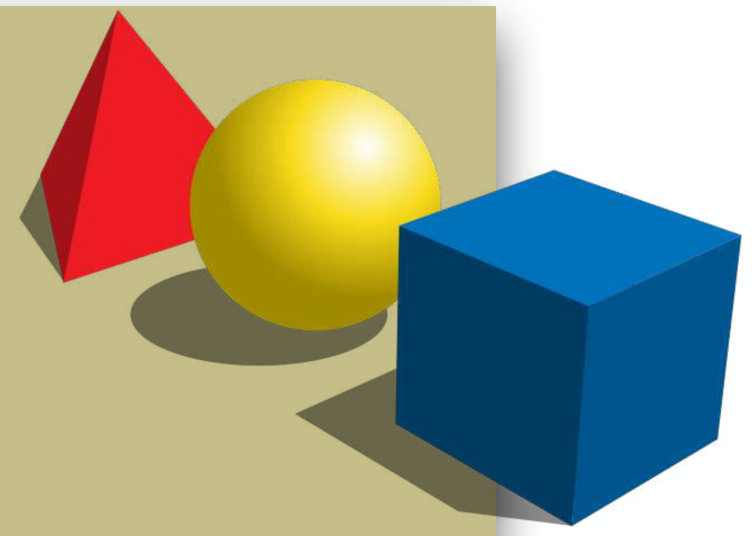
$x50 > x51$

Abstract Interpretation

3-Step Recipe

practical tools
targeting specific programs

abstract semantics, abstract domains
algorithmic approaches to decide program properties

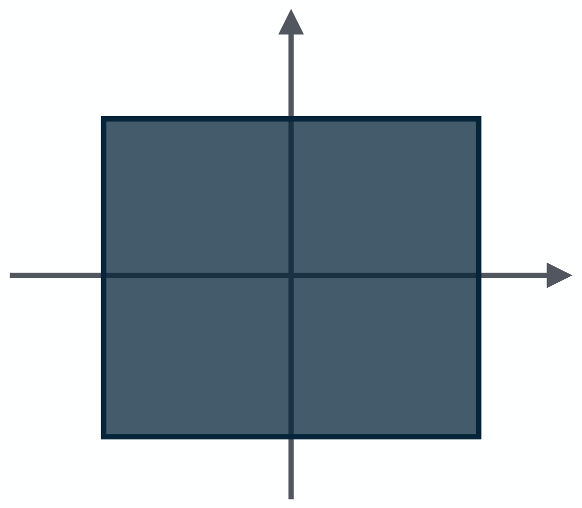


concrete semantics
mathematical models of the program behavior

Safety Verification

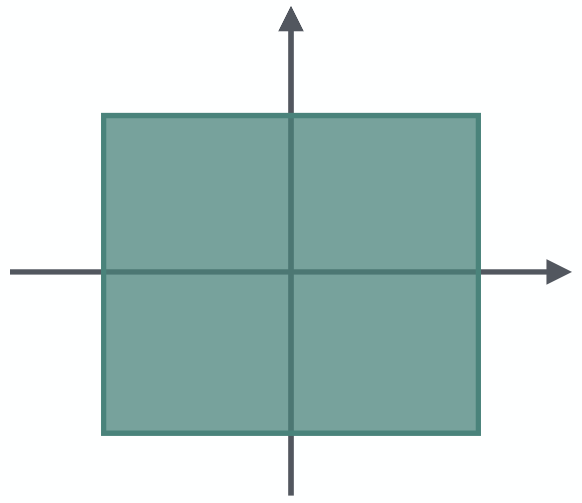
Static Forward Analysis

```
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

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())
```

I:
x00: [-1, 1]
x01: [-1, 1]
x02: [-1, 1]
x03: [-1, 1]
x04: [-1, 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))
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 ⊑ [0, ∞]

Safety Verification

Boxes Abstract Domain

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

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

I:
x00: [-1, 1]
x01: [-1, 1]
x02: [-1, 1]
x03: [-1, 1]
x04: [-1, 1]
x05: [-1, 1]

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

x10 -> [-2.895878, 6.143547]

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

x10 -> [0, 6.143547]

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

x11 -> [0, 3.291125]

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

x12 -> [0, 5.023332]

⋮

```
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

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())
```

I: x00: [-1, 1]
x01: [-1, 1]
x02: [-1, 1]
x03: [-1, 1]
x04: [-1, 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))
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, 6.143547] x11 -> [0, 3.291125] x12 -> [0, 5.023332]

```
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 -> [0, 25.303196] x21 -> [0, 27.341758] x22 -> [0, 18.627984]

⋮

x30 -> [0, 118.989519] x31 -> [0, 155.150698] x32 -> [0, 162.176672]

```
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, 1054.076987] x41 -> [0, 0] x42 -> [0, 191.110038]

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

O: $[-6171.351539, 5000.0] \not\sqsubseteq [0, \infty]$ ✗

Safety Verification

Symbolic Abstract Domain

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

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

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))

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, ∞]

Safety Verification

Symbolic Abstract Domain

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

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

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

x10': $\begin{cases} (0.120875) * x00 + (0.065404) * x01 + (0.097862) * x02 + (2.030051) * x03 + (0.101956) * x04 + (-2.103565) * x05 + (1.623834) \\ [-2.895878, 6.143547] \end{cases}$

$$x_{i-1,0} \mapsto \mathbf{E_{i-1,0}}$$

$\dots$

$$x_{i-1,j} \mapsto \mathbf{E_{i-1,j}}$$

$\dots$

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

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

⋮

```
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 $\sqsubseteq$ [0, ∞]

Safety Verification

Symbolic Abstract Domain

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

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

x10': $\begin{cases} (0.120875) * x00 + (0.065404) * x01 + (0.097862) * x02 + (2.030051) * x03 + (0.101956) * x04 + (-2.103565) * x05 + (1.623834) \\ [-2.895878, 6.143547] \end{cases}$

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

x10: $\begin{cases} x10 \\ [0, 6.143547] \end{cases}$

$$x_{i,j} \mapsto \begin{cases} \mathbf{E_{i,j}} \\ \mathbf{[a,b]} \end{cases} \xrightarrow{\text{ReLU}} \begin{cases} x_{i,j} \mapsto \begin{cases} \mathbf{E_{i,j}} \\ \mathbf{[a,b]} \end{cases} & 0 \leq a \\ x_{i,j} \mapsto \begin{cases} \mathbf{x_{i,j}} \\ \mathbf{[0,b]} \end{cases} & a < 0 \wedge 0 < b \\ x_{i,j} \mapsto \begin{cases} \mathbf{0} \\ \mathbf{[0,0]} \end{cases} & b \leq 0 \end{cases}$$

⋮

```
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 ⊏ [0, ∞]

Safety Verification

Symbolic Abstract Domain

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

```
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: $\begin{cases} x10 \\ [0,6.143547] \end{cases}$ x11: $\begin{cases} x11 \\ [0,3.291125] \end{cases}$ x11: $\begin{cases} x12 \\ [0,5.023332] \end{cases}$

⋮

```
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: $\begin{cases} x40 \\ [0,1054.076987] \end{cases}$ x41: $\begin{cases} (-0.552155) * x30 + (-0.828226) * x31 + (-0.495998) * x32 \\ [0,0] \end{cases}$ x42: $\begin{cases} x42 \\ [0,191.110038] \end{cases}$

```
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: $\begin{cases} (-4.556024) * x40 + (-33.326096) * x42 + 5000 \\ [-6171.351539, 5000.0] \not\sqsubseteq [0, \infty] \end{cases}$

Safety Verification

DeepPoly Abstract Domain

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

```
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

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

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

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': $\begin{cases}[(0.120875) * x00 + (0.065404) * x01 + (0.097862) * x02 + (2.030051) * x03 + (0.101956) * x04 + (-2.103565) * x05 + (1.623834), \\ (0.120875) * x00 + (0.065404) * x01 + (0.097862) * x02 + (2.030051) * x03 + (0.101956) * x04 + (-2.103565) * x05 + (1.623834)] \\ [-2.895878,\ 6.143547]\end{cases}$

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

$\dots$

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

$\dots$

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

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

⋮

```
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 $\sqsubseteq$ [0, ∞]

Safety Verification

DeepPoly Abstract Domain

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

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

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': $\begin{cases} [(0.120875) * x00 + (0.065404) * x01 + (0.097862) * x02 + (2.030051) * x03 + (0.101956) * x04 + (-2.103565) * x05 + (1.623834), \\ \quad (0.120875) * x00 + (0.065404) * x01 + (0.097862) * x02 + (2.030051) * x03 + (0.101956) * x04 + (-2.103565) * x05 + (1.623834)] \\ [-2.895878,\ 6.143547] \end{cases}$

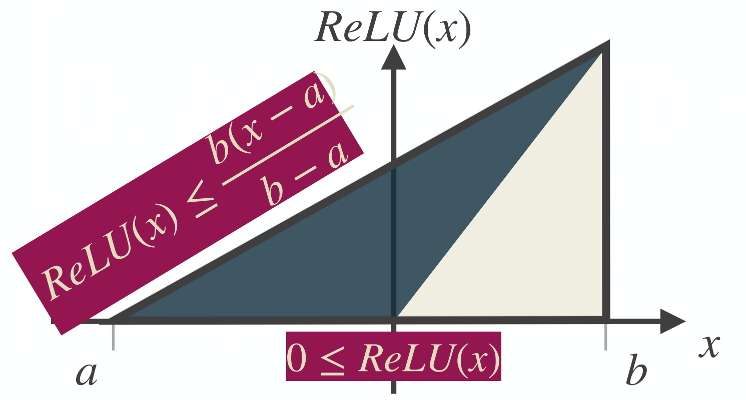
```
x10 = ReLU(x10')
```

x10: $\begin{cases} 0.679639 * x10' \\ \quad +1.968152 \\ [-2.895878,\ 6.143547] \end{cases}$

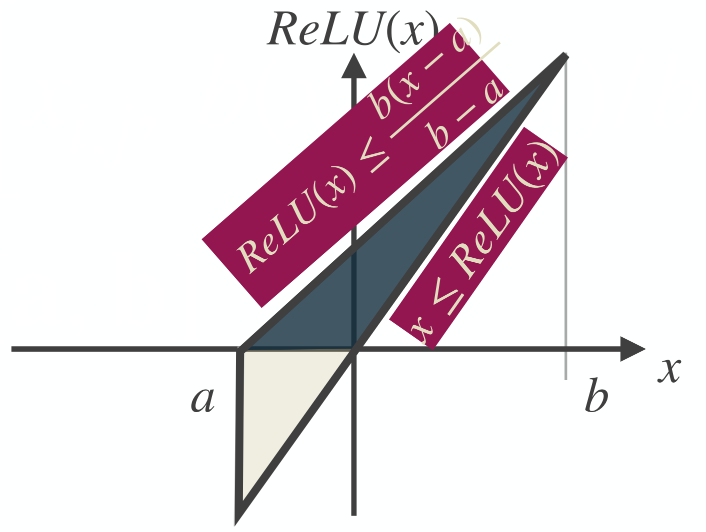
$$x_{i,j} \mapsto \begin{cases} [\mathrm{L_{i,j}}, \mathrm{U_{i,j}}] \\ [\mathrm{a}, \mathrm{b}] \end{cases}$$

ReLU

$a < 0 \land 0 < b \land -b \leq a$: $x_{i,j} \mapsto$



$a < 0 \land 0 < b \land -a < b$: $x_{i,j} \mapsto$



⋮

```
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

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

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

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

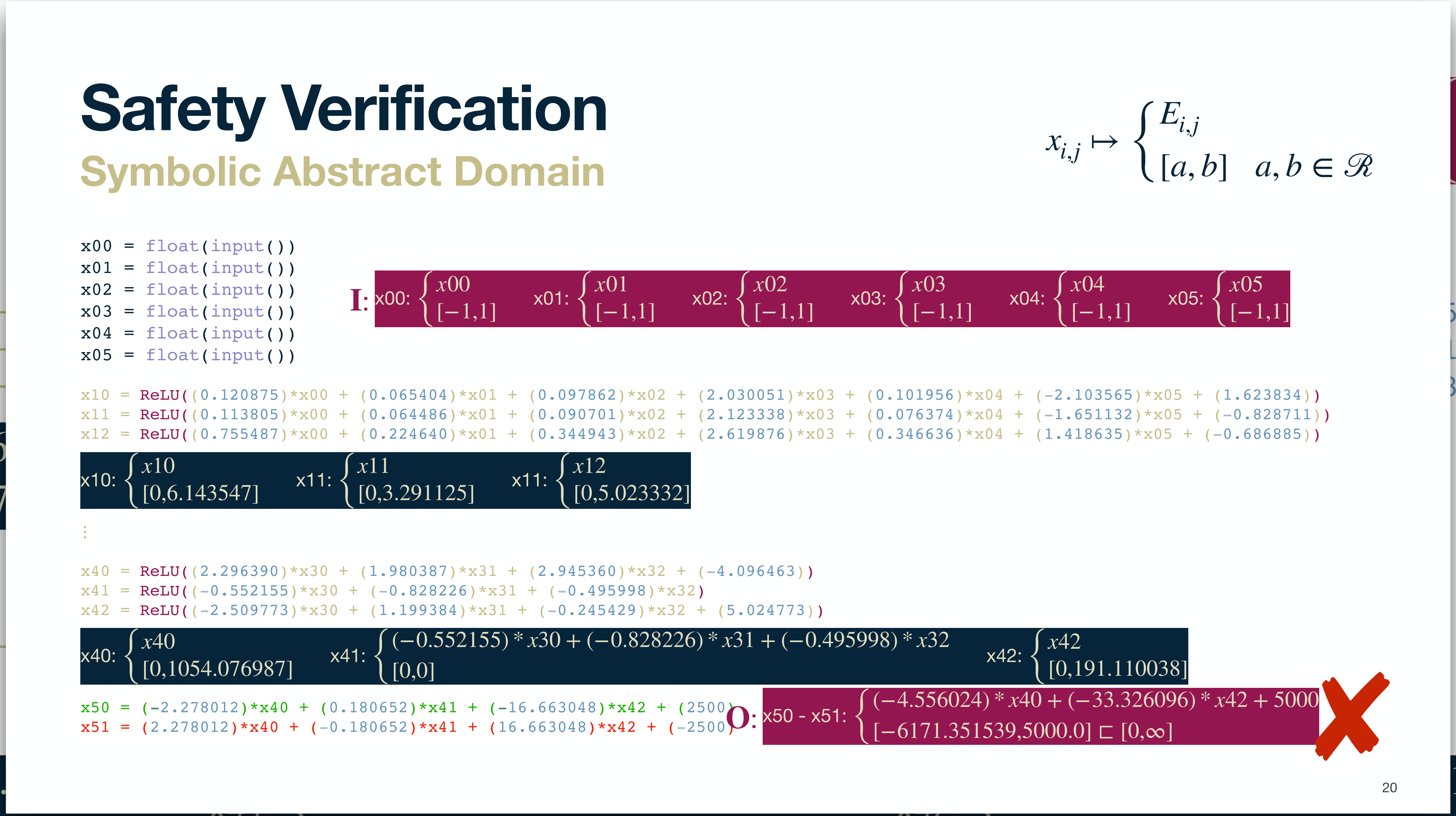
```
x10 = ReLU((0.120875)*x00 + ...65)*x05 + (1.623834))
x11 = ReLU((0.113805)*x00 + ...32)*x05 + (-0.828711))
x12 = ReLU((0.755487)*x00 + ...35)*x05 + (-0.686885))
```

x10: $\begin{cases} 0.679639 * x10' + 1.96\ldots \\ [-2.895878,\ 6.143547\ldots \end{cases}$

⋮

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

x40: $\begin{cases} 0.670470 * x40' + 313\ldots \\ [-467.102459,\ 950.380211] \end{cases}$ x41: $[0,\ 118.628114]$ x42: $\begin{cases} \ldots 152 \\ [-142.203290,\ 162.093422] \end{cases}$



Safety Verification

Symbolic Abstract Domain

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

```
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: $\begin{cases} x10 \\ [0,6.143547] \end{cases}$ x11: $\begin{cases} x11 \\ [0,3.291125] \end{cases}$ x11: $\begin{cases} x12 \\ [0,5.023332] \end{cases}$

⋮

```
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: $\begin{cases} x40 \\ [0,1054.076987] \end{cases}$ x41: $\begin{cases} (-0.552155) * x30 + (-0.828226) * x31 + (-0.495998) * x32 \\ [0,0] \end{cases}$ x42: $\begin{cases} x42 \\ [0,191.110038] \end{cases}$

```
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: $\begin{cases} (-4.556024) * x40 + (-33.326096) * x42 + 5000 \\ [-6171.351539,5000.0] \sqsubset [0,\infty] \end{cases}$ ✗

20

```
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: $\begin{cases} \cdots \\ [-1424.797461,\ 9072.124338] \sqsubset [0,\infty] \end{cases}$ ✗

Reduced Product Domain

Symbolic Abstract Domain & DeepPoly Abstract Domain

$$\begin{tikzcd} \text{Symbolic} \arrow[rr, bend right, "{[a_s, b_s]}"'] & & \text{DeepPoly} \arrow[ll, bend right, "{[a_d, b_d]}"'] \end{tikzcd}$$

Symbolic: $[\max(a_s, a_d), \min(b_s, b_u)]$

DeepPoly: $[\max(a_s, a_d), \min(b_s, b_u)]$

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())
```

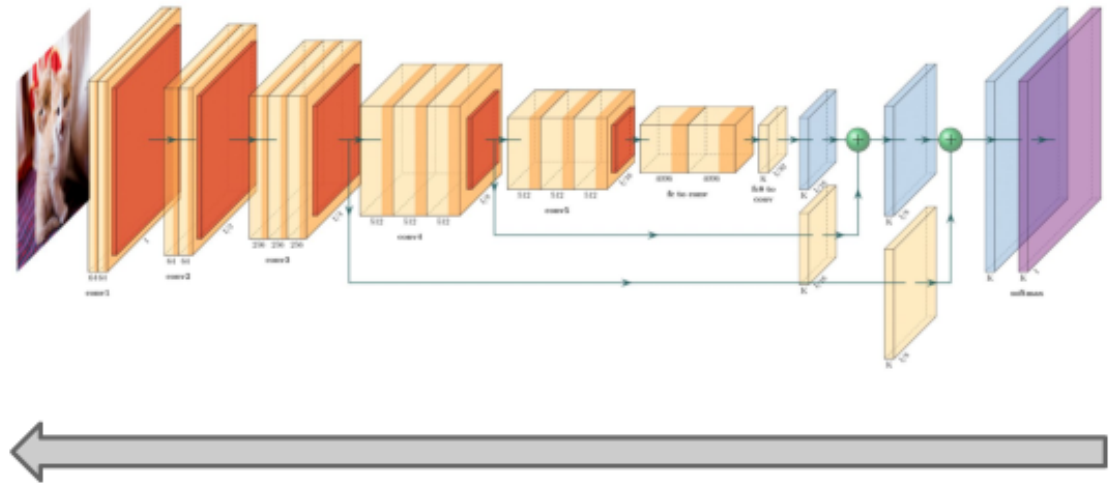
I: x00: $\begin{cases} x00 \\ [x00, x00] \\ [-1,1] \end{cases}$ x01: $\begin{cases} x01 \\ [x01, x01] \\ [-1,1] \end{cases}$ x02: $\begin{cases} x02 \\ [x02, x02] \\ [-1,1] \end{cases}$ x03: $\begin{cases} x03 \\ [x03, x03] \\ [-1,1] \end{cases}$ x04: $\begin{cases} x04 \\ [x04, x04] \\ [-1,1] \end{cases}$ 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: $\begin{cases} \vdots \\ [670.044947961025,\ 5000.0] \sqsubseteq [0, \infty] \end{cases}$

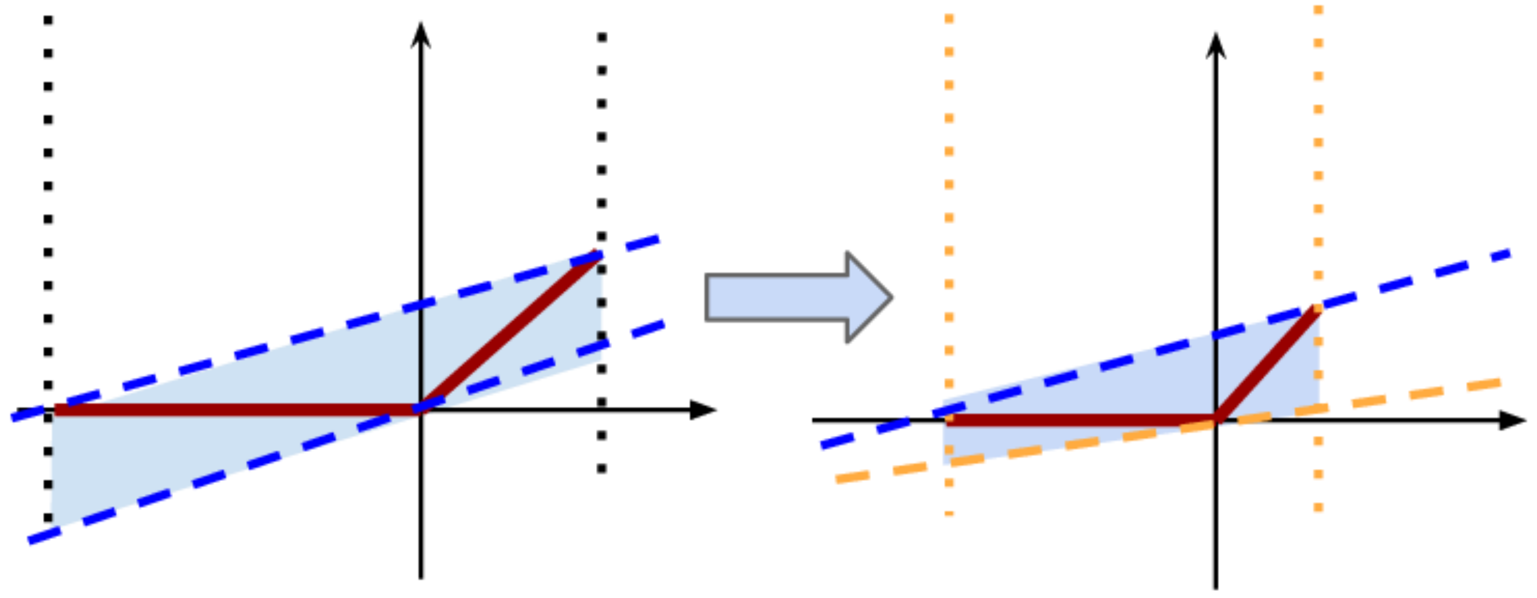
Safety Verification

Going Farther: $\alpha\beta$-CROWN

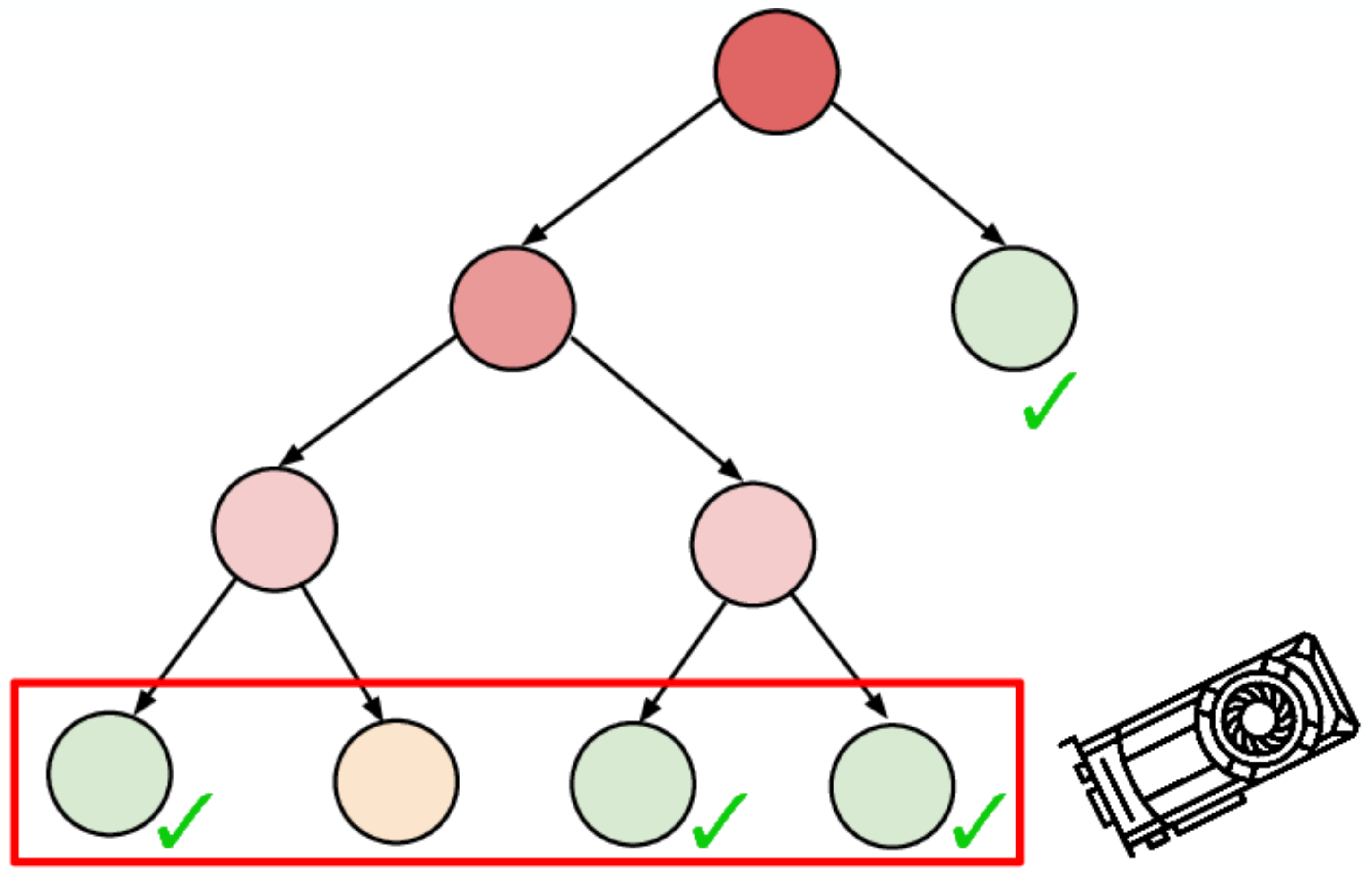


$$\min_{x \in \mathcal{C}} f(x) \geq \min_{x \in \mathcal{C}} \boldsymbol{a}^\top x + c$$

Efficient bound propagation (**CROWN**)



GPU optimized relaxation (***α*-CROWN**)



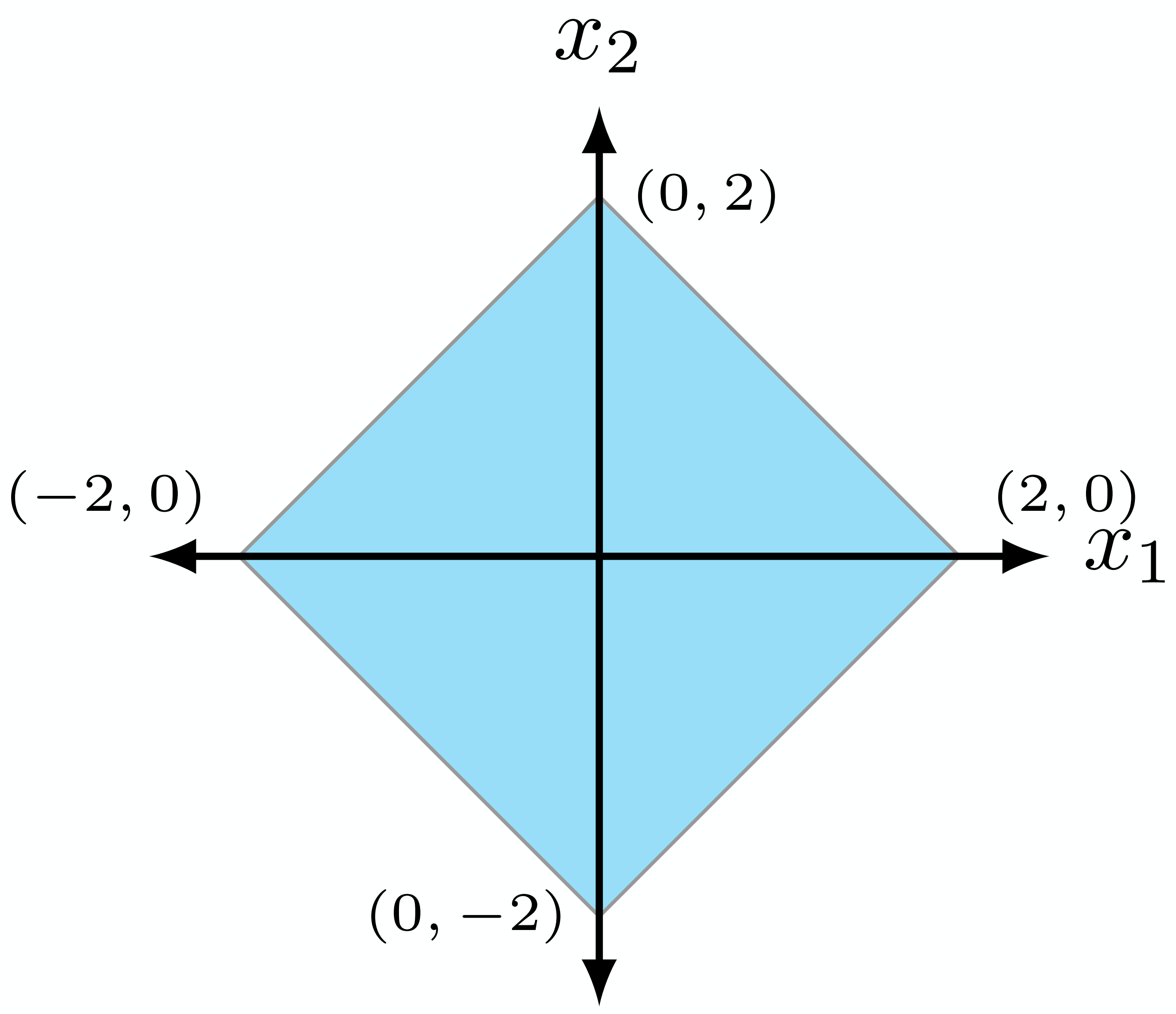
Parallel branch and bound (***β*-CROWN**)

Winner of the International Verification of Neural Networks Competition since 2021

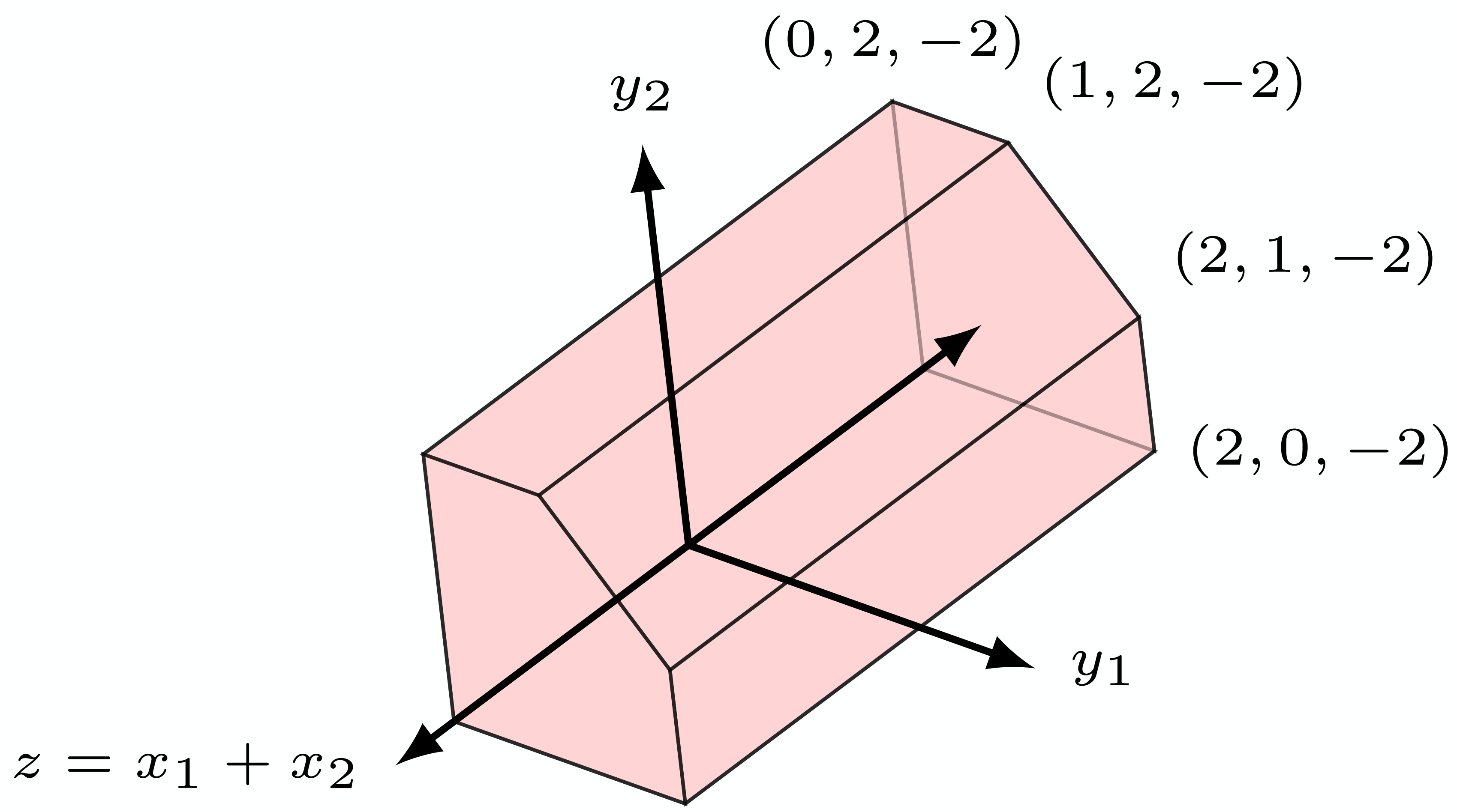
https://github.com/Verified-Intelligence/alpha-beta-CROWN

Safety Verification

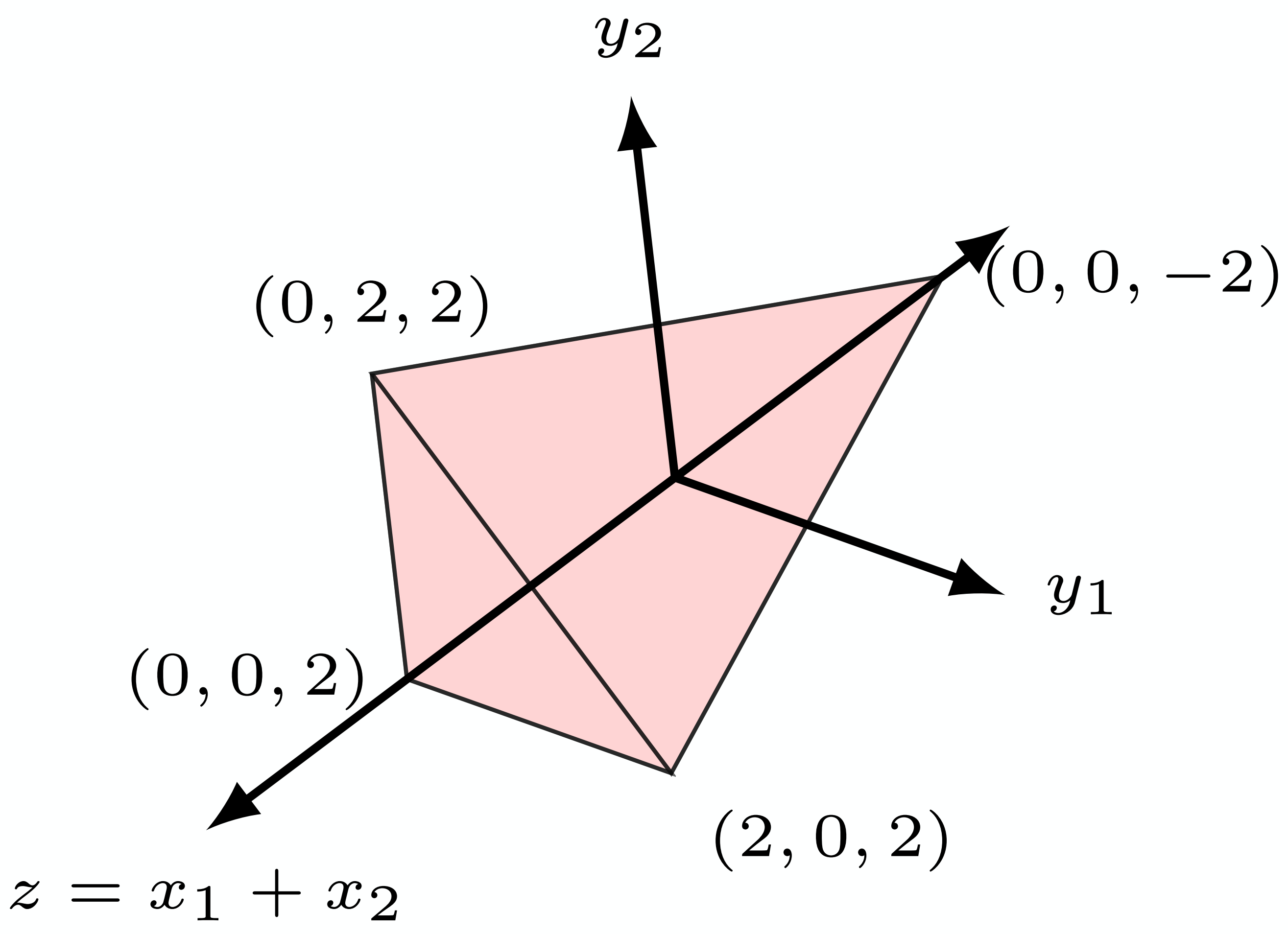
Going Farther: Multi-Neuron Abstractions



(a) Input shape



(b) 1-ReLU

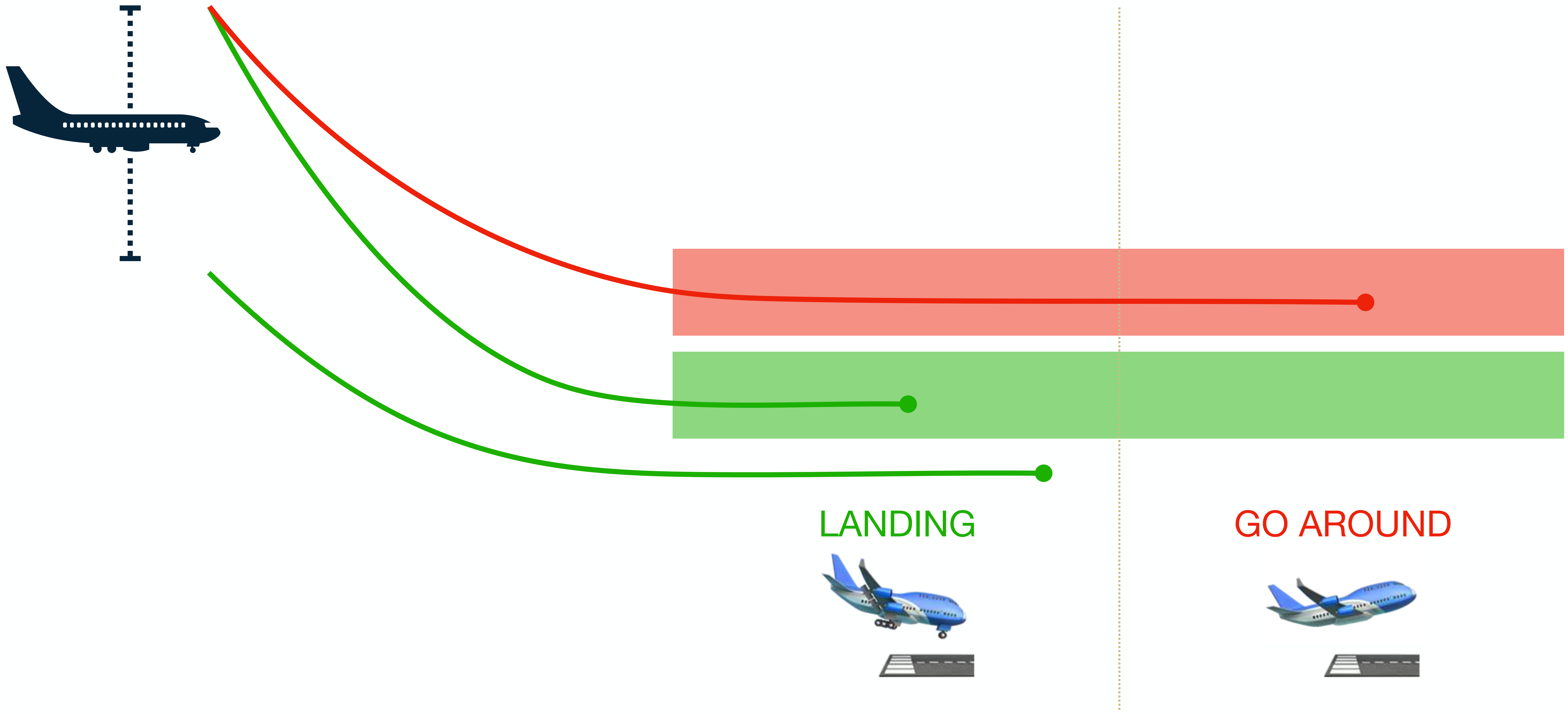


(c) 2-ReLU

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

Runway Overrun Warning

HyperSafety of Neural Network Surrogate



Hyperproperty Verification

Abstract Non-Interference Properties

η: input abstraction

ρ: output abstraction

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

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

Theorem

$$M \models \mathcal{H}_\rho^\eta \Leftrightarrow [\![M]\!] \in \mathcal{H}_\rho^\eta \Leftrightarrow \{[\![M]\!]\} \subseteq \mathcal{H}_\rho^\eta$$

Corollary

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

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:

$\eta(x00) = x00$

$\eta(x01) = x01$

ALTITUDE $\eta(x02) = \top$

$\eta(x03) = x03$

$\eta(x04) = x04$

$\eta(x05) = x05$

“**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

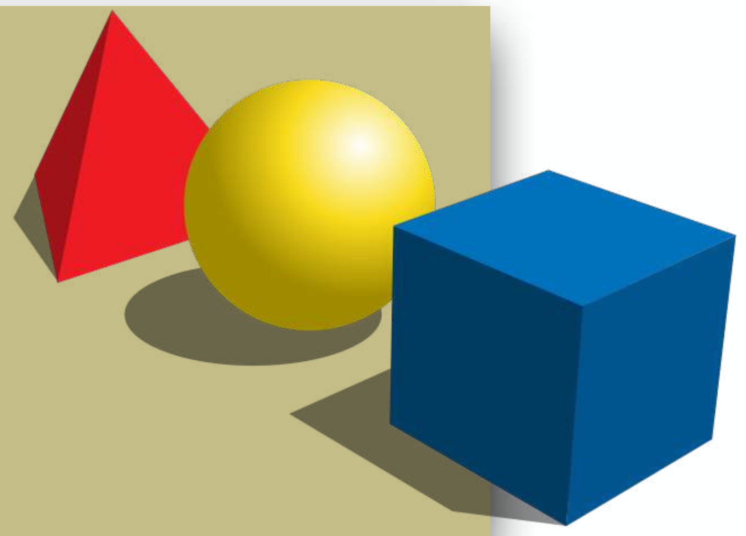
$\rho(x51) = 1$ if $x51 > x50$ else 0

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

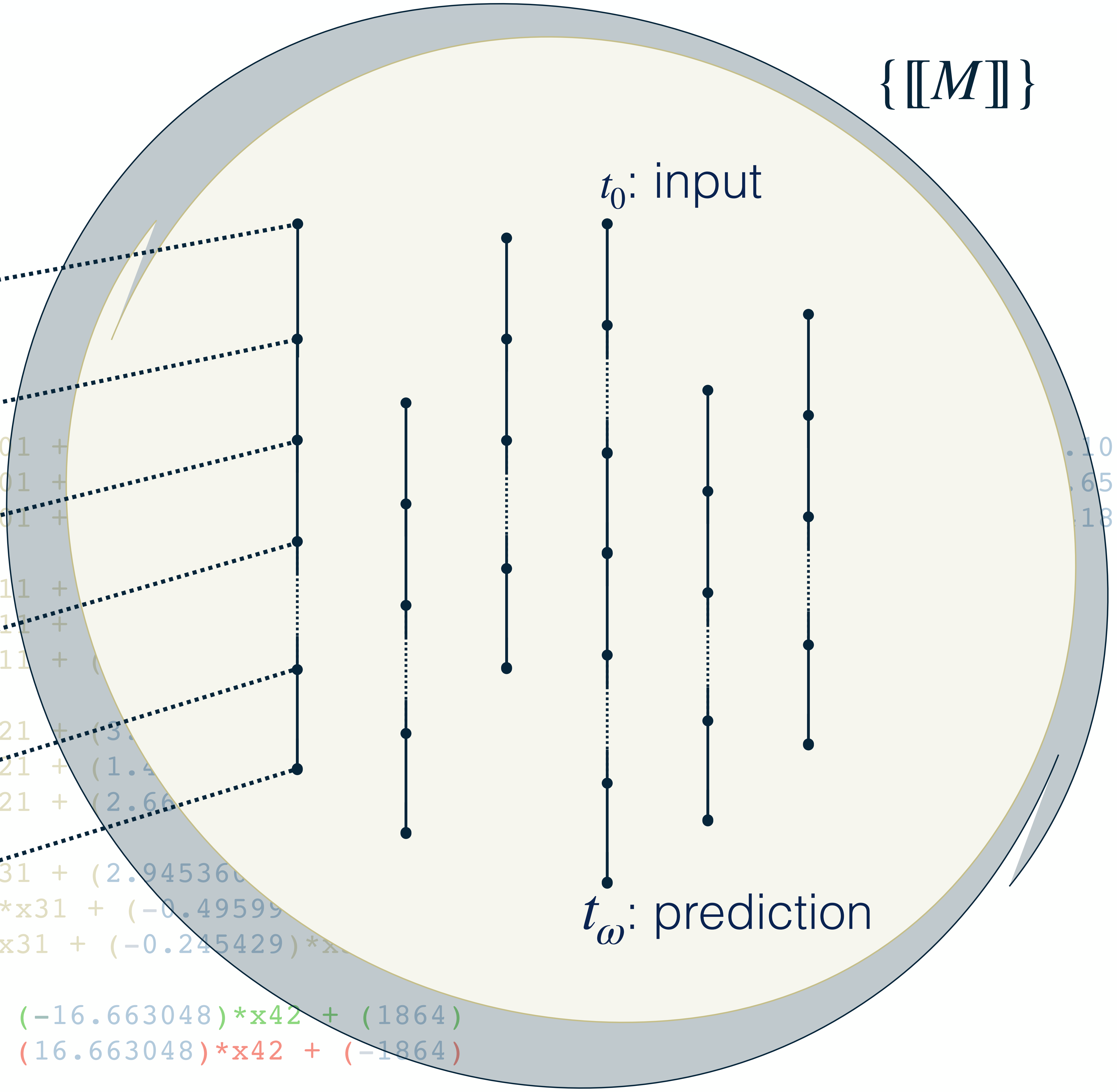
practical tools
targeting specific programs

abstract semantics, abstract domains
algorithmic approaches to decide program properties

concrete semantics
mathematical models of the program behavior

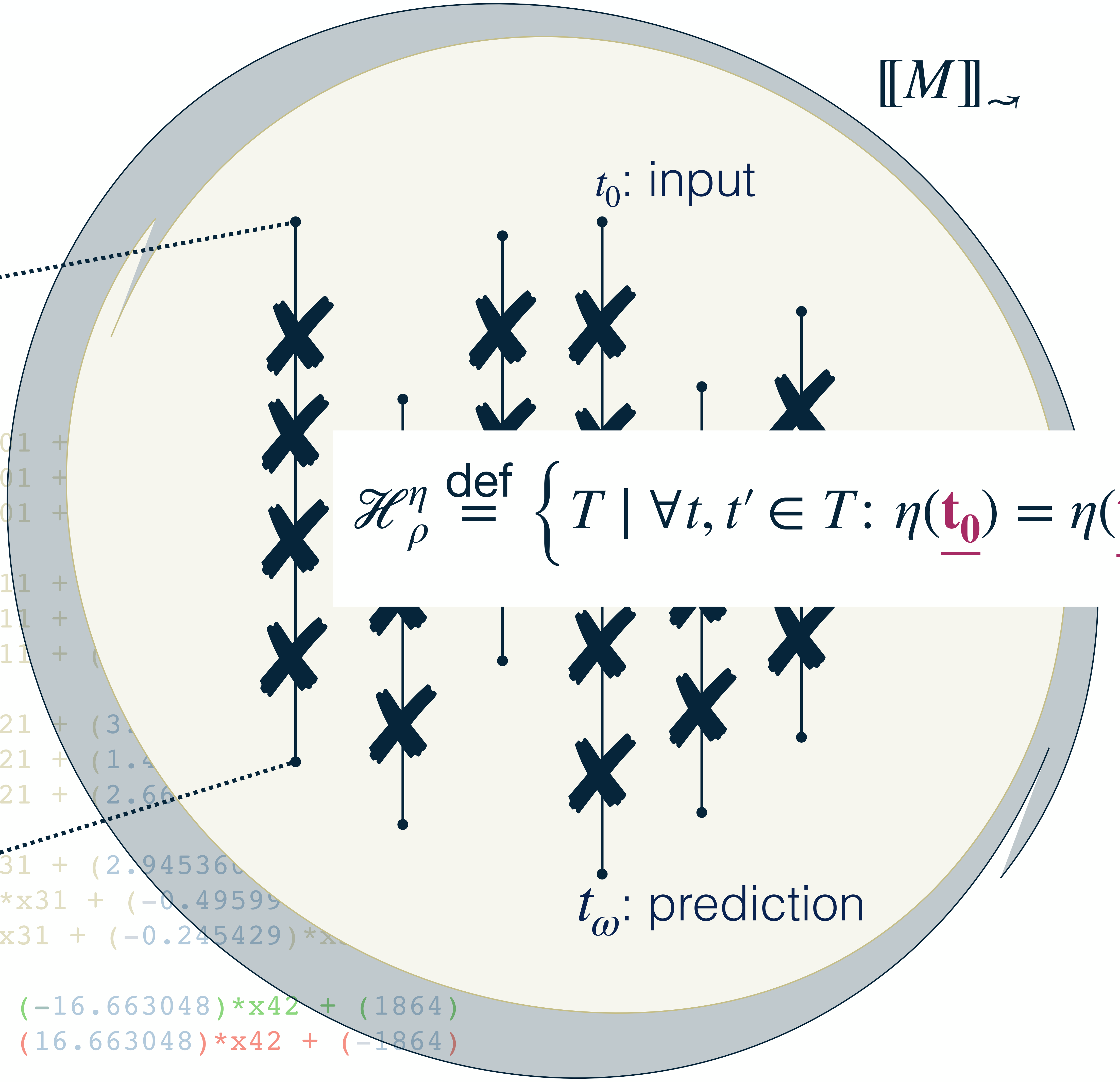
Collecting Semantics

```
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

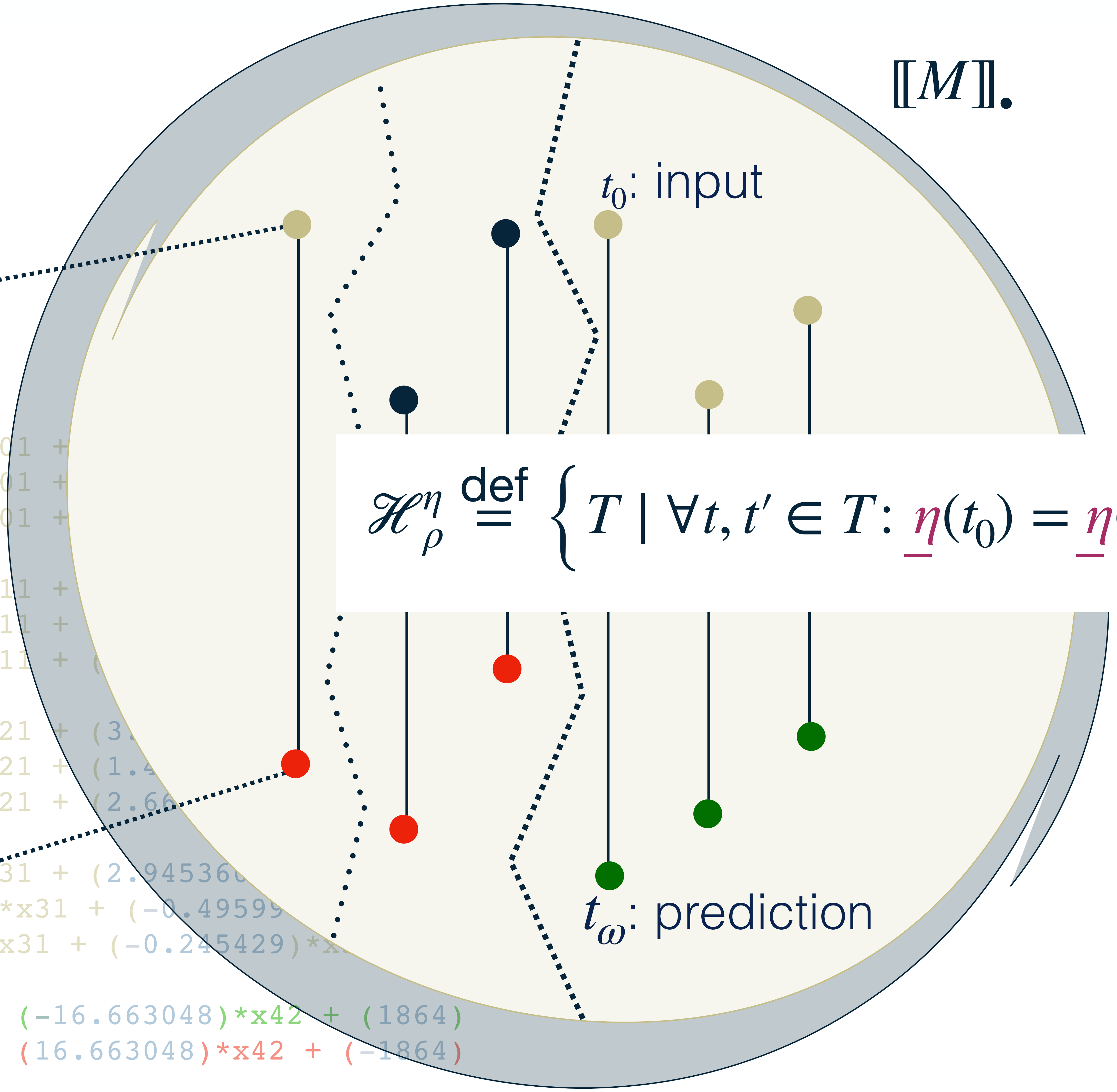
```
x00 = float(input())
x01 = float(input())
x02 = float(input())
x03 = float(input())
x04 = float(input())
x05 = float(input())

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)
```



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

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)
```



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

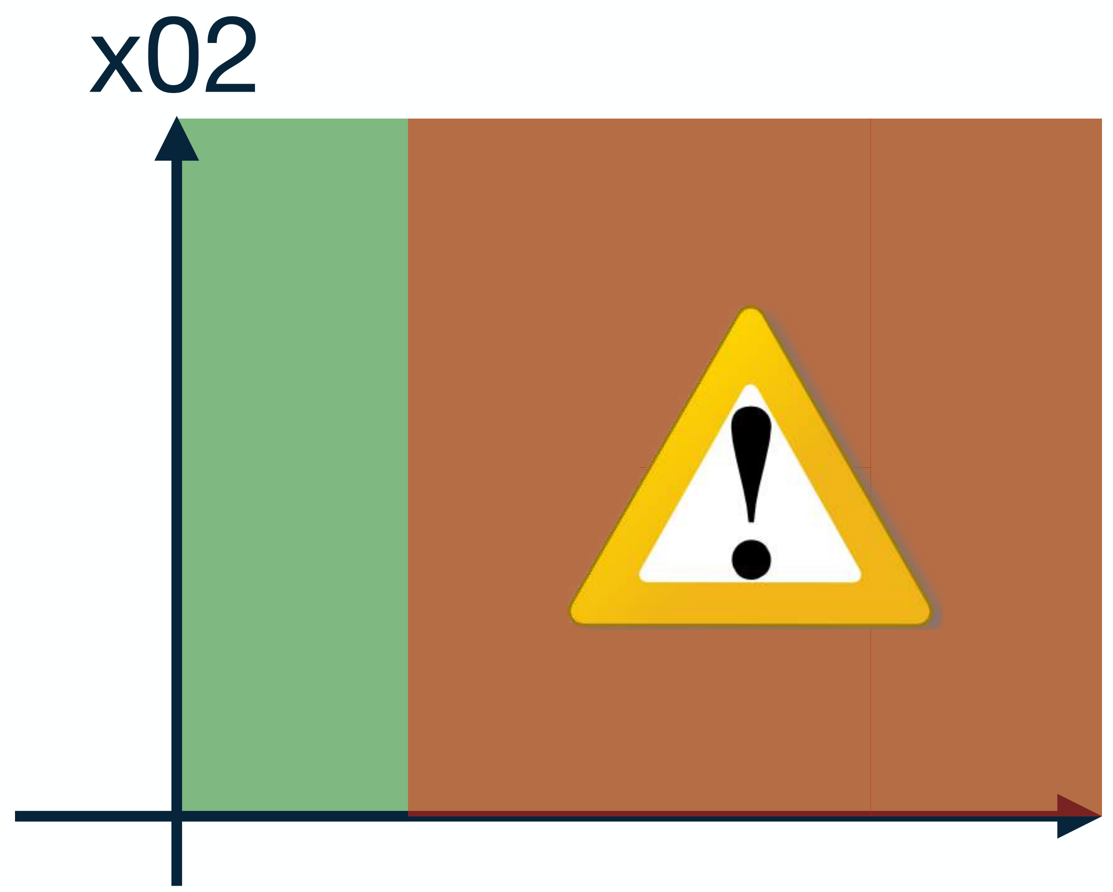
Hyperproperty Verification

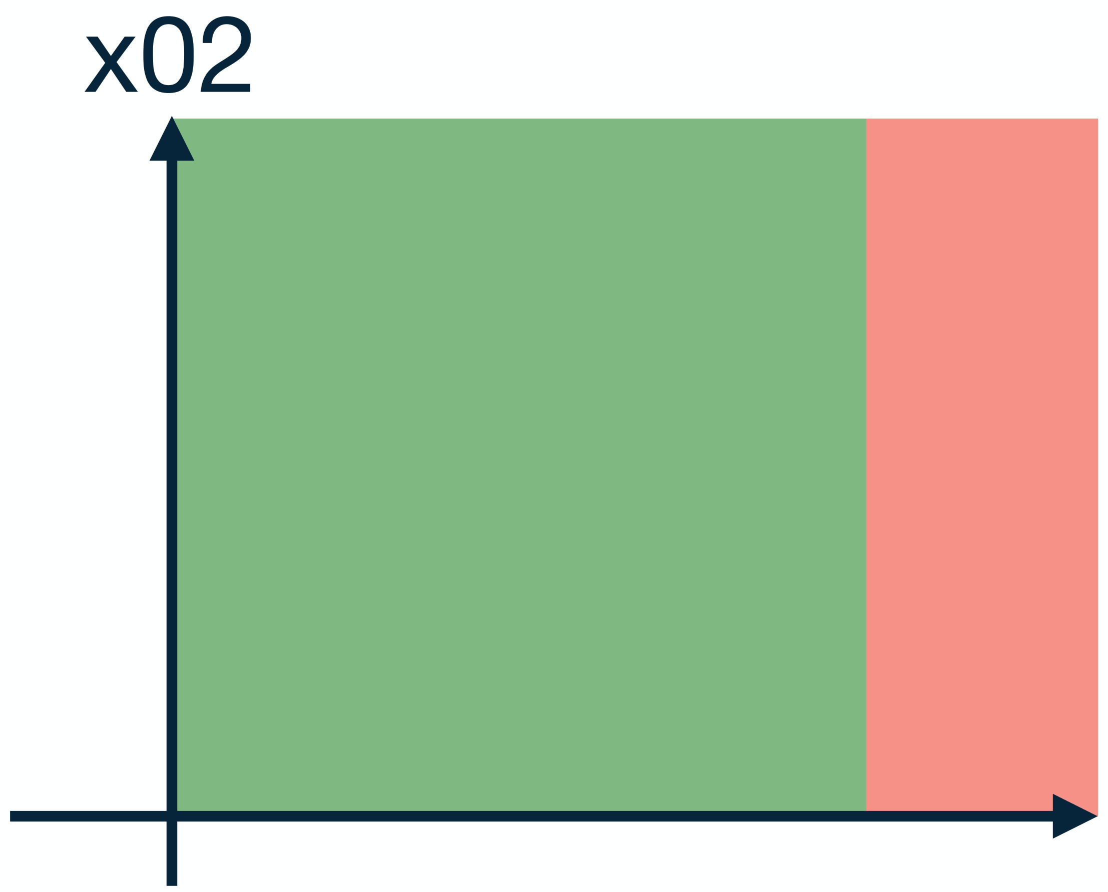
Abstract Non-Interference Properties

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

Lemma

$$M \models \mathcal{H}^{\eta}_{\rho} \Leftrightarrow \forall I \in \mathbb{I} \colon \forall A, B \in [\![M]\!]^{\mathbb{I}}_{\bullet} \colon \rho(A^{I}_{\omega}) \sqcap \rho(B^{I}_{\omega}) = \bot \Rightarrow \eta(A^{I}_{0}) \sqcap \eta(B^{I}_{0}) = \bot$$





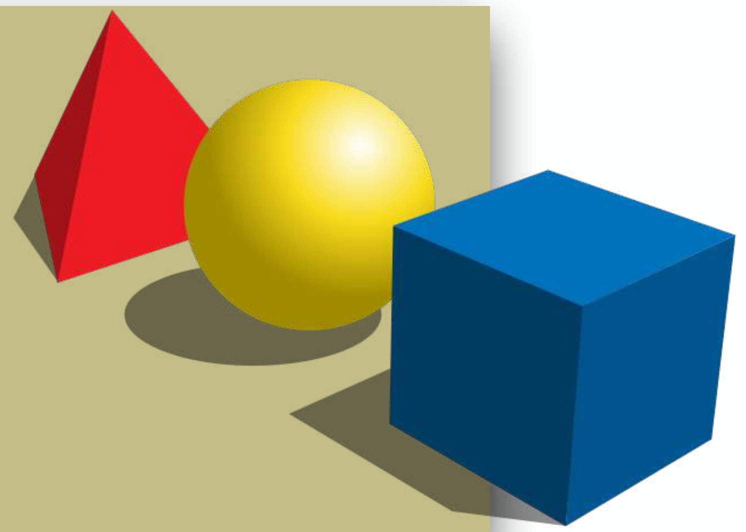
Giacobazzi and Mastroeni. Abstract Non-Interference: A Unifying Framework for Weakening Information-Flow. In TOPS, 2018.

Abstract Interpretation

3-Step Recipe

practical tools
targeting specific programs

abstract semantics, abstract domains
algorithmic approaches to decide program properties

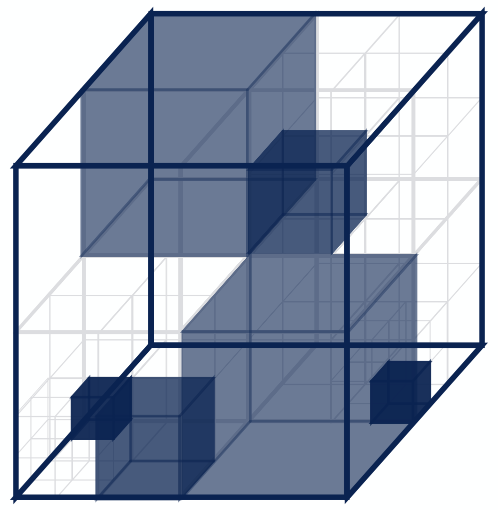


concrete semantics
mathematical models of the program behavior

Hyperproperty Verification

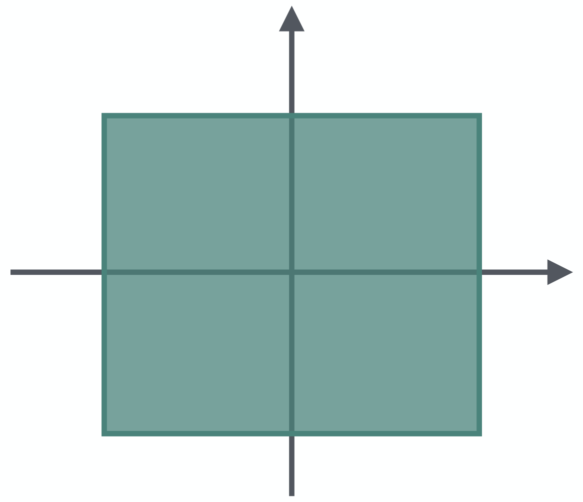
Static Forward Analysis

```
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())
```

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

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

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

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

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

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

Activation patterns (columns 1 / 2): x10: 1 / 1; x11: 1 / 1; x12: ? / ?; x20: 1 / 1; x21: 1 / 1; x22: 1 / 1; x30: 1 / 1; x31: 1 / 1; x32: 1 / 1; x40: 1 / 1; x41: 0 / 0; x42: ? / ?

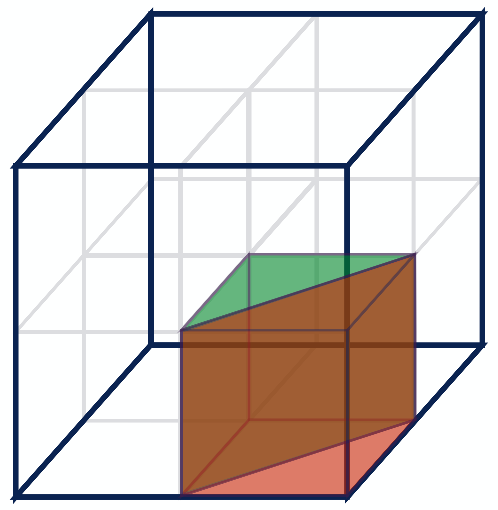
ρ:	ρ:	ρ:
x50: [1, 1]	x50: [0, 1]	x50: [0, 1]
x51: [0, 0]	x51: [0, 1]	x51: [0, 1]

several partitions share the same abstract activation pattern

Hyperproperty Verification

Static Backward Analysis

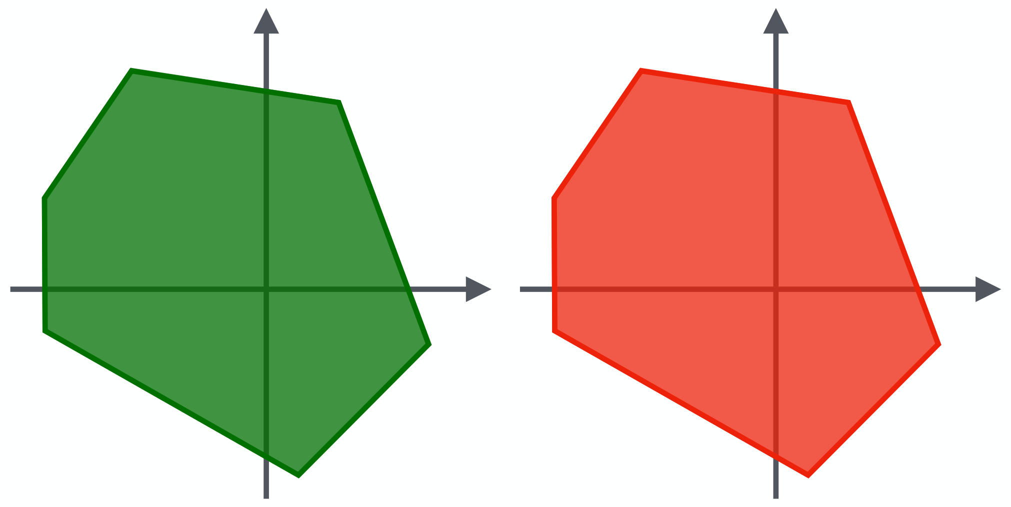
```
x00 = float(input())
x01 = float(input())
x02 = float(input())
x03 = float(input())
x04 = float(input())
x05 = float(input())

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())

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

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

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

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

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

	η:	η:
	x00: [0, 1]	x00: [0, 1]
	x01: [-1, 0]	x01: [0, 1]
	x02: T	x02: T
	x03: [0.5, 1]	x03: [0.5, 1]
	x04: [0, 1]	x04: [0, 1]
	x05: [-1, 0]	x05: [-1, 0]
x10	1	1
x11	1	1
x12	?	?
x20	1	1
x21	1	1
x22	1	1
x30	1	1
x31	1	1
x32	1	1
x40	1	1
x41	0	0
x42	?	?

ρ: x50: [0, 1], x51: [0, 1]

ρ: x50: [0, 1], x51: [0, 1]

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: T	x02: T
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)
```

$x50 > x51$ $x51 > x50$

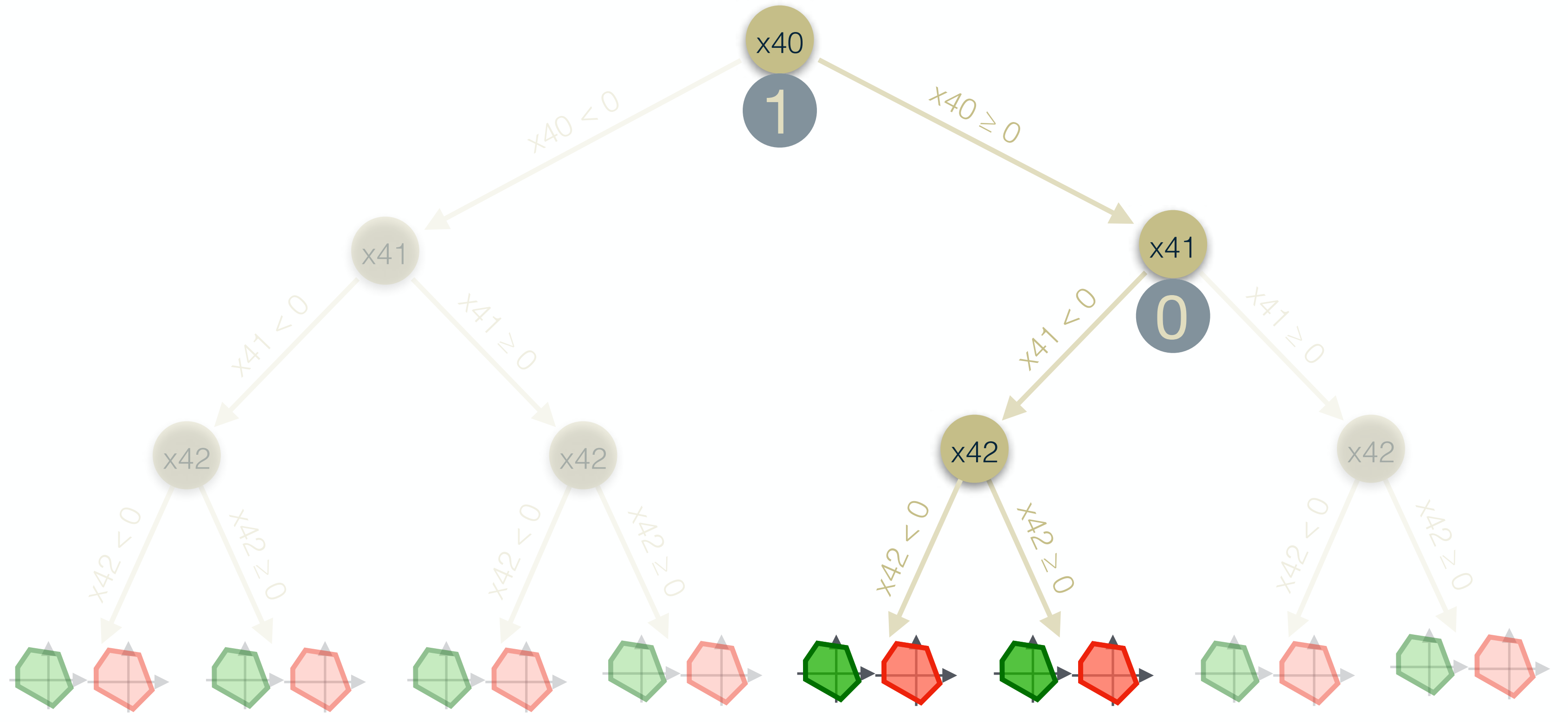
Static Backward Analysis

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)
```



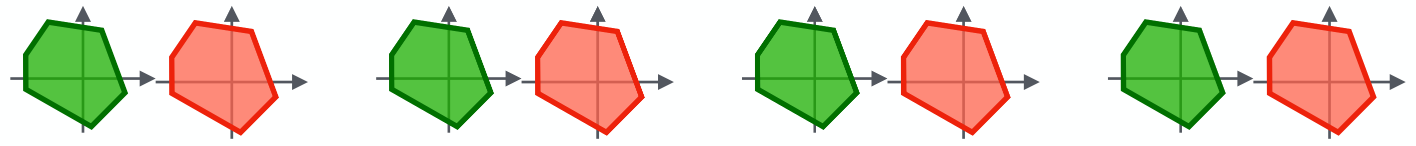
$(-4.556024) * x40 + (0.361304) * x41 + (-33.326096) * x42 + (3728) > 0$

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

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]

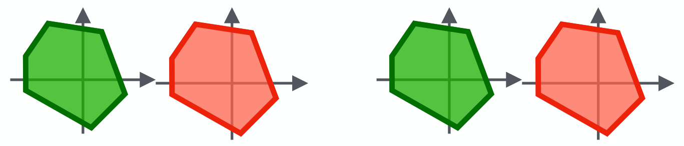
η:

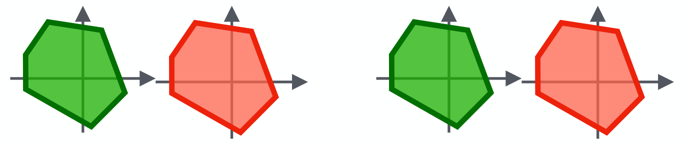
x00: [0, 1]
x01: [0, 1]
x02: T
x03: [0.5, 1]
x04: [0, 1]
x05: [-1, 0]

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

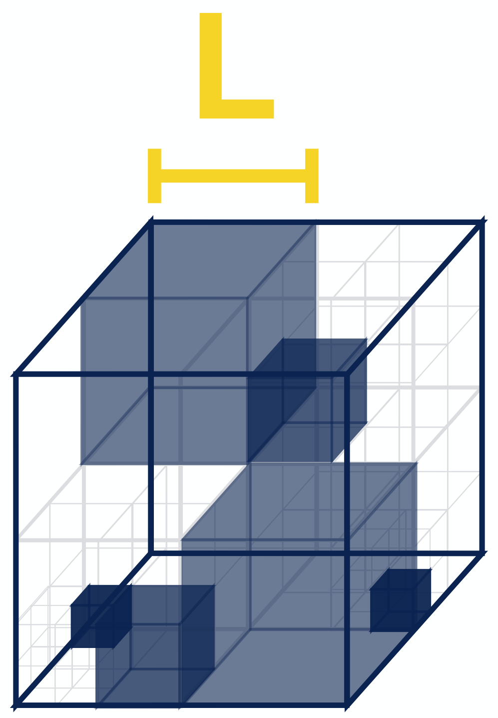
abstract semantics, abstract domains
algorithmic approaches to decide program properties

concrete semantics
mathematical models of the program behavior

Hyperproperty Verification

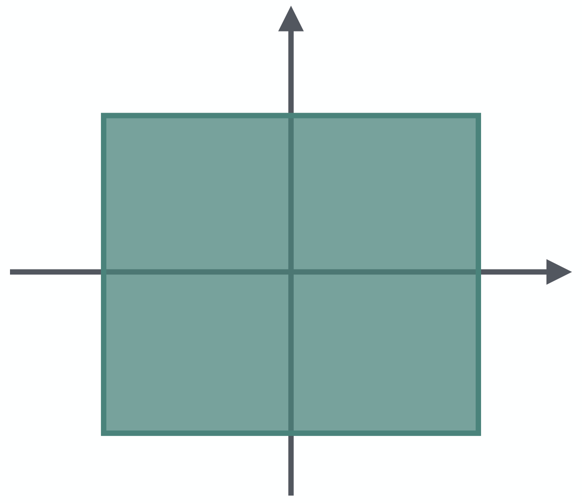
Static Forward Analysis

```
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



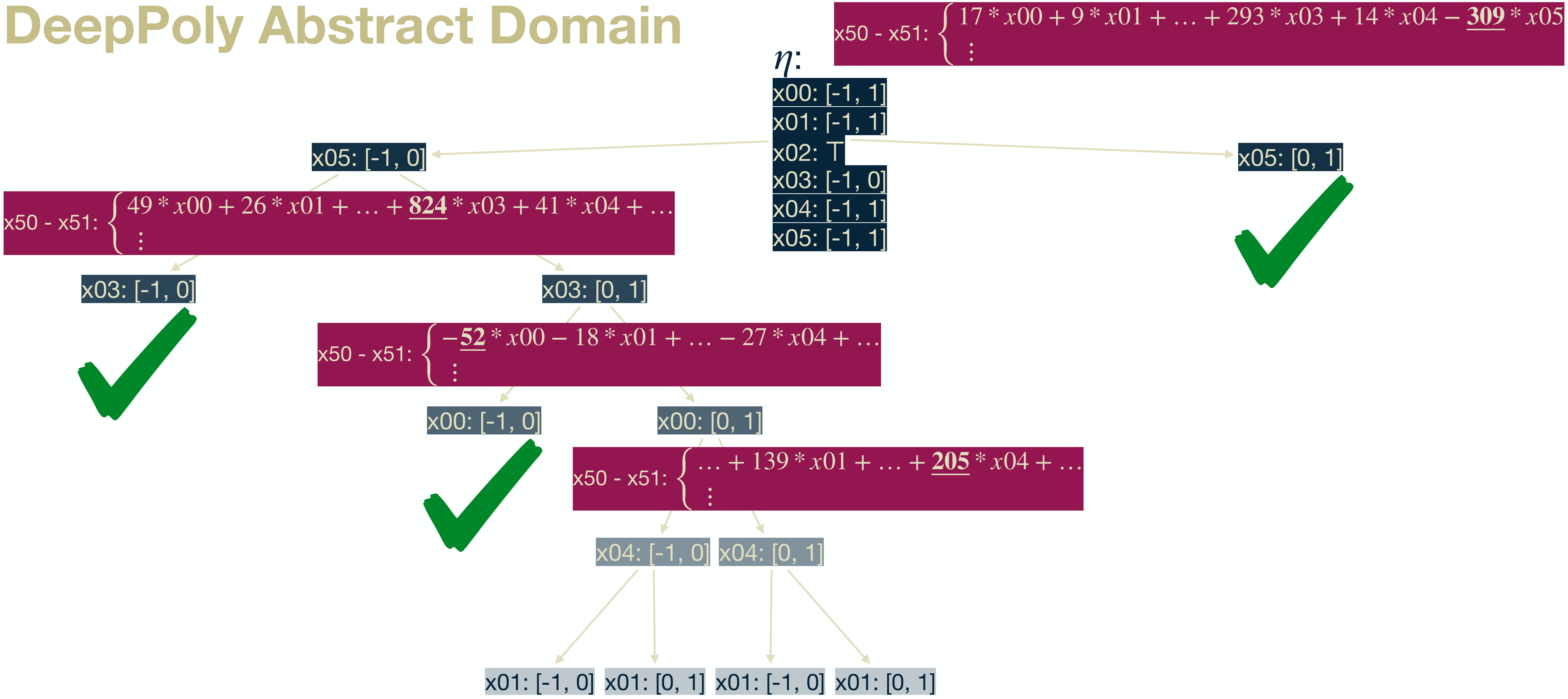
3 check output for:
- **unique classification outcome** → ✓ **safe**
- **abstract activation pattern** ∪

Partitioning Strategies: Interval Range

DeepPoly Abstract Domain

Partitioning Strategies: ReCIPH

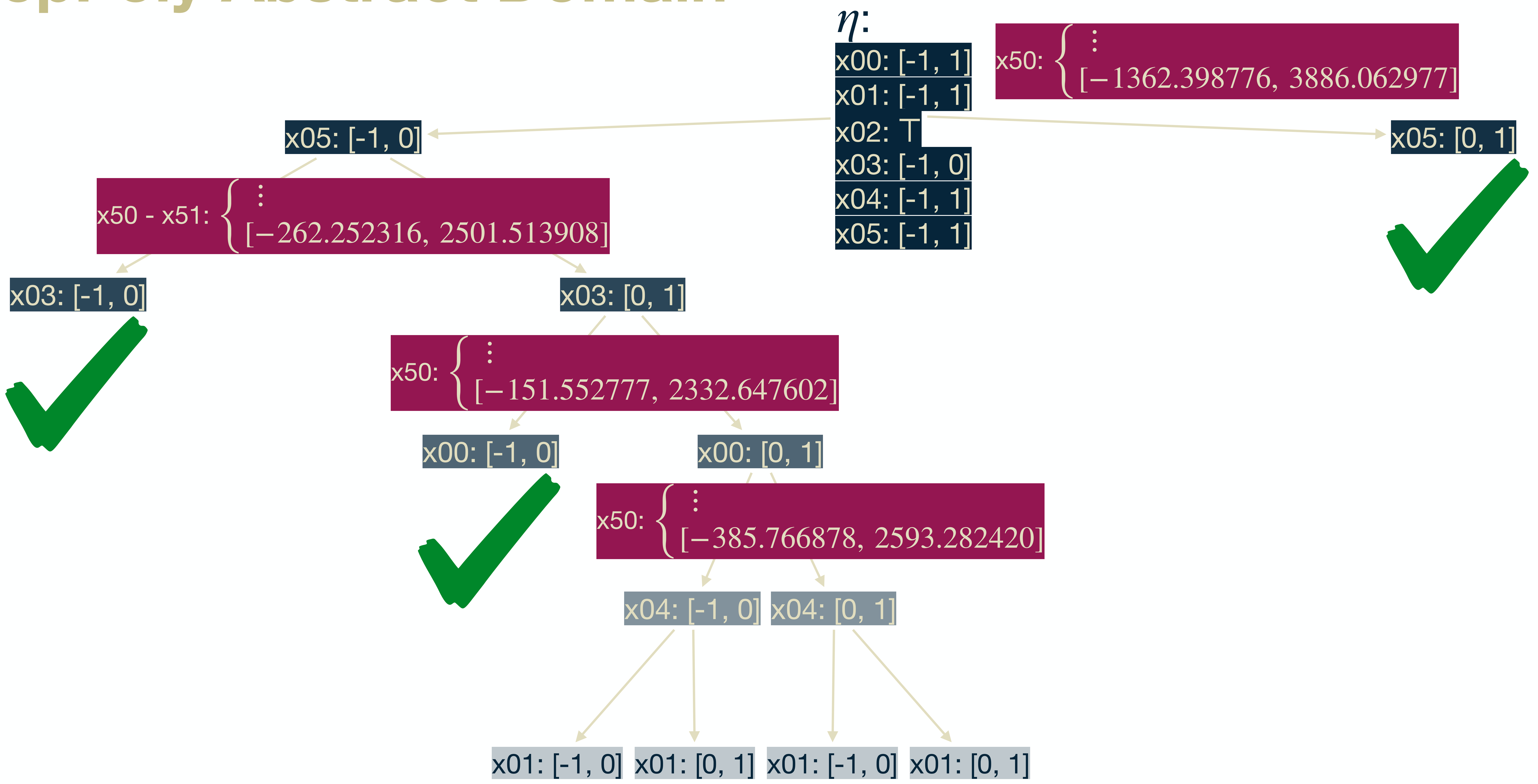
DeepPoly Abstract Domain



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

Input Refinement $\nRightarrow$ Output Refinement

DeepPoly Abstract Domain



Input Refinement $\not\Rightarrow$ 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

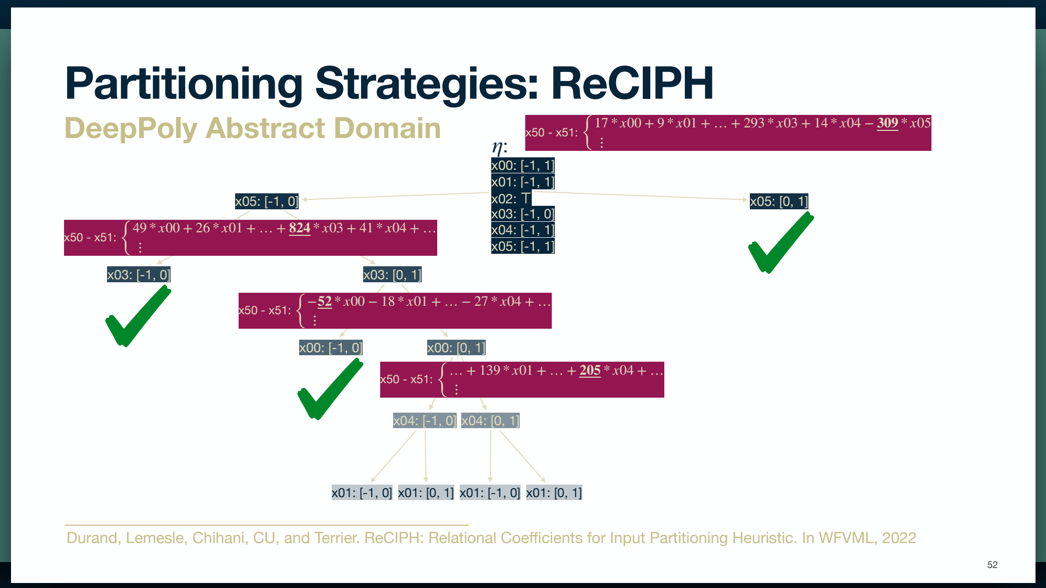
L	U	Boxes	Symbolic	DeepPoly		Product	
				Input Range Partitioning	ReCIPH	Input Range Partitioning	ReCIPH
1	2	0,08s	0,14s	0,26s	0,11s	0,26s	0,12s
	6	0,16s	0,31s	0,51s	0,20s	0,35s	0,20s
0.5	2	8,88s	5,76s	2,60s	1,61s	2,10s	1,61s
	6	64,67s	40,90s	2,65s	1,63s	2,10s	1,62s

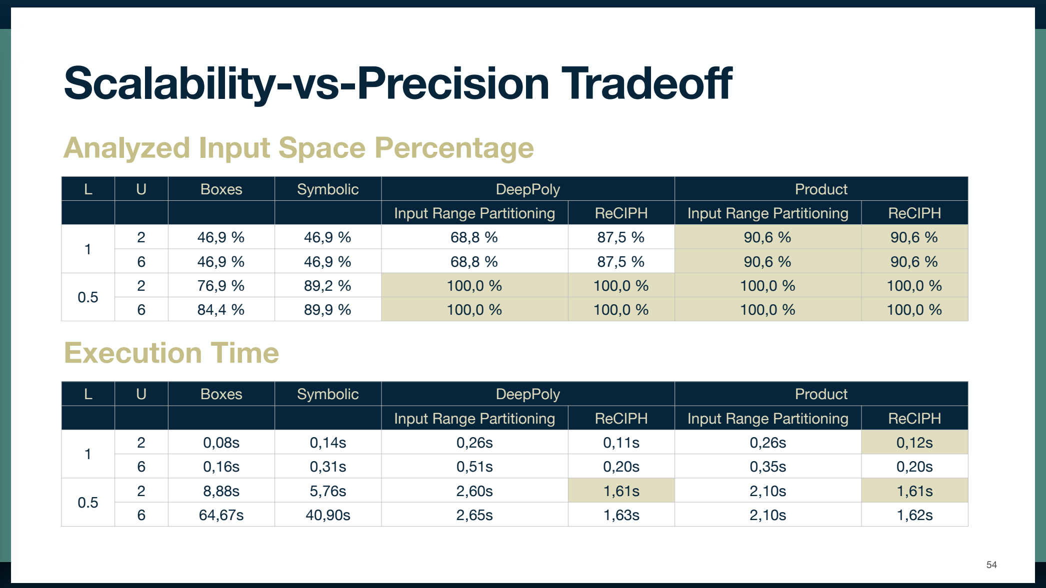
HyperProperty Verification

High-Stakes Machine Learning Software

practical tools

targeting specific programs



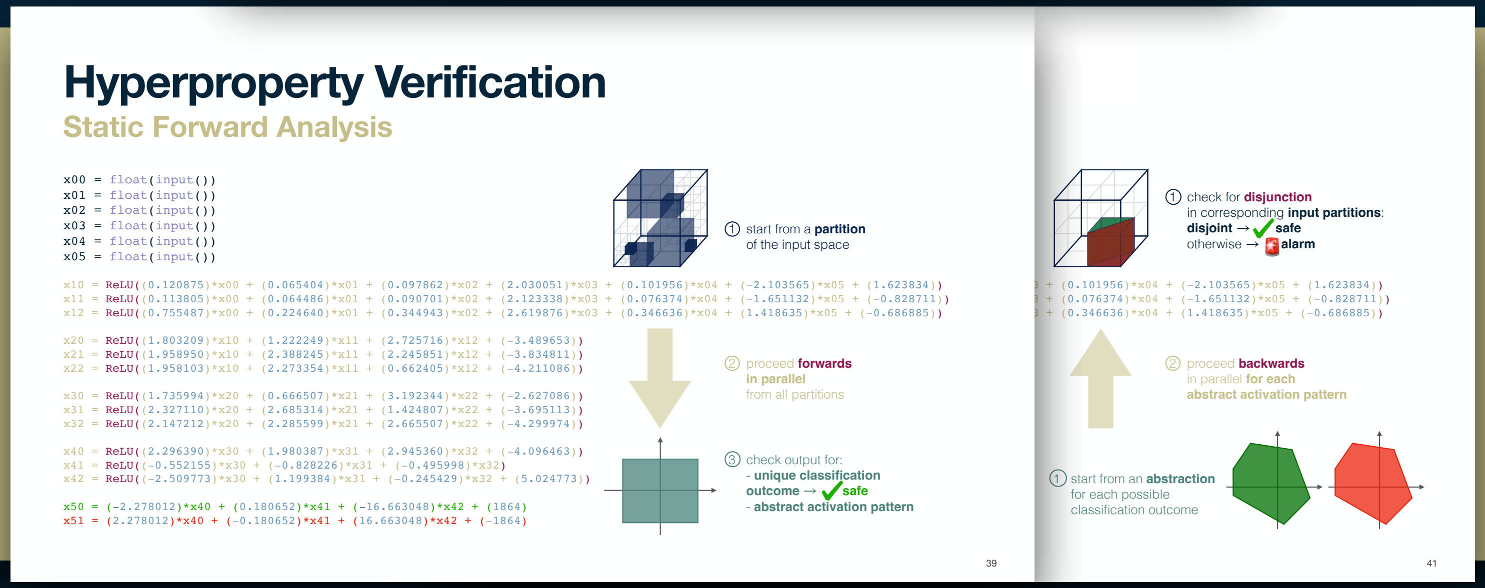


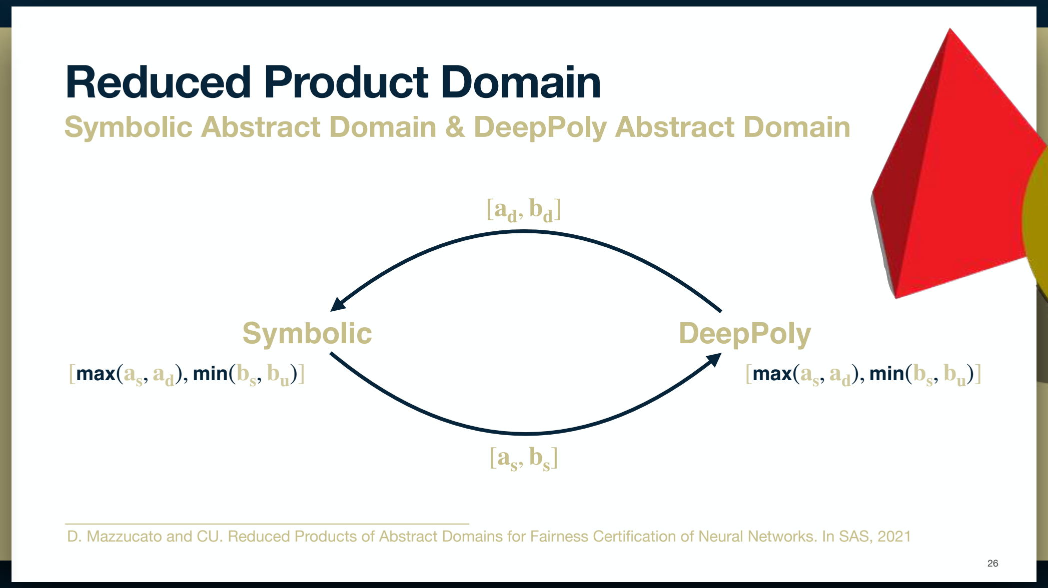
L	U	Boxes	Symbolic	DeepPoly		Product	
				Input Range Partitioning	ReCIPH	Input Range Partitioning	ReCIPH
1	2	46,9 %	46,9 %	68,8 %	87,5 %	90,6 %	90,6 %
	6	46,9 %	46,9 %	68,8 %	87,5 %	90,6 %	90,6 %
0.5	2	76,9 %	89,2 %	100,0 %	100,0 %	100,0 %	100,0 %
	6	84,4 %	89,9 %	100,0 %	100,0 %	100,0 %	100,0 %

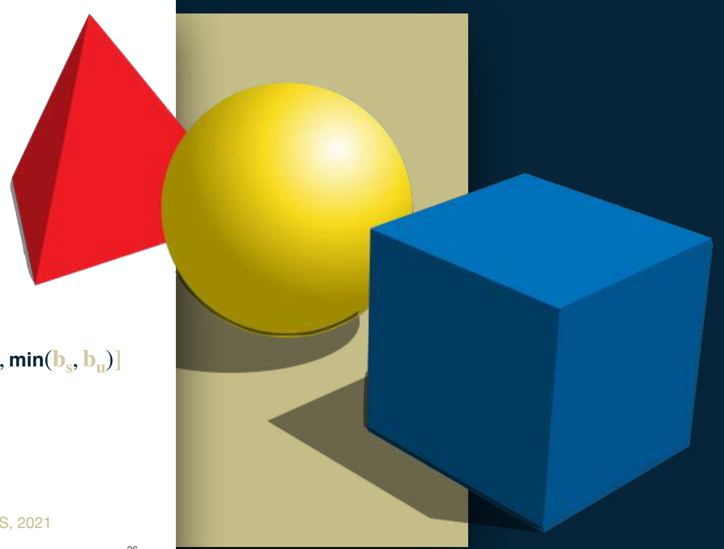
L	U	Boxes	Symbolic	DeepPoly		Product	
				Input Range Partitioning	ReCIPH	Input Range Partitioning	ReCIPH
1	2	0,08s	0,14s	0,26s	0,11s	0,26s	0,12s
	6	0,16s	0,31s	0,51s	0,20s	0,35s	0,20s
0.5	2	8,88s	5,76s	2,60s	1,61s	2,10s	1,61s
	6	64,67s	40,90s	2,65s	1,63s	2,10s	1,62s

algorithmic approaches

to decide program properties

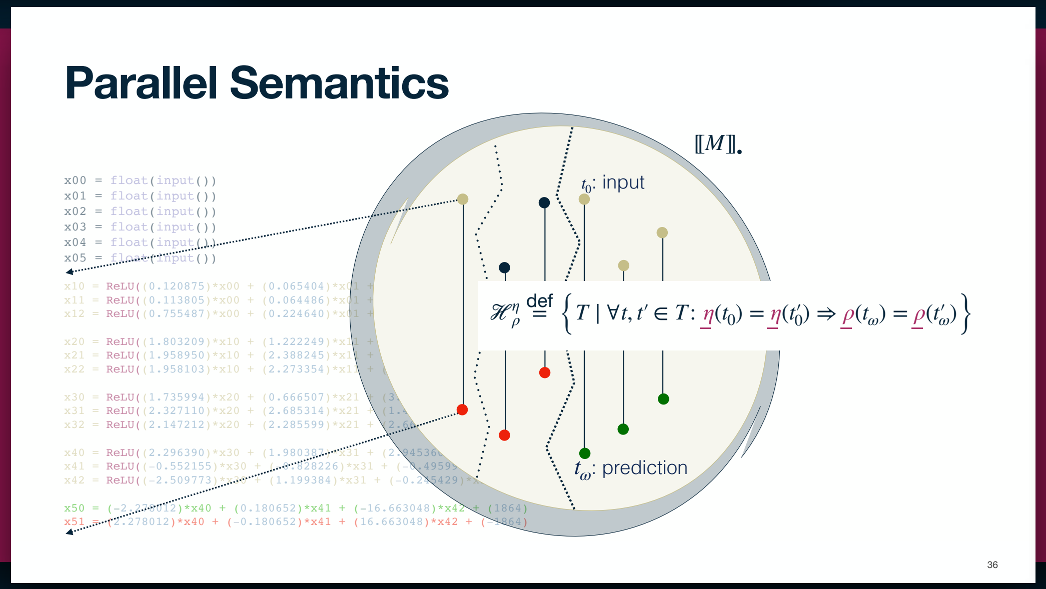


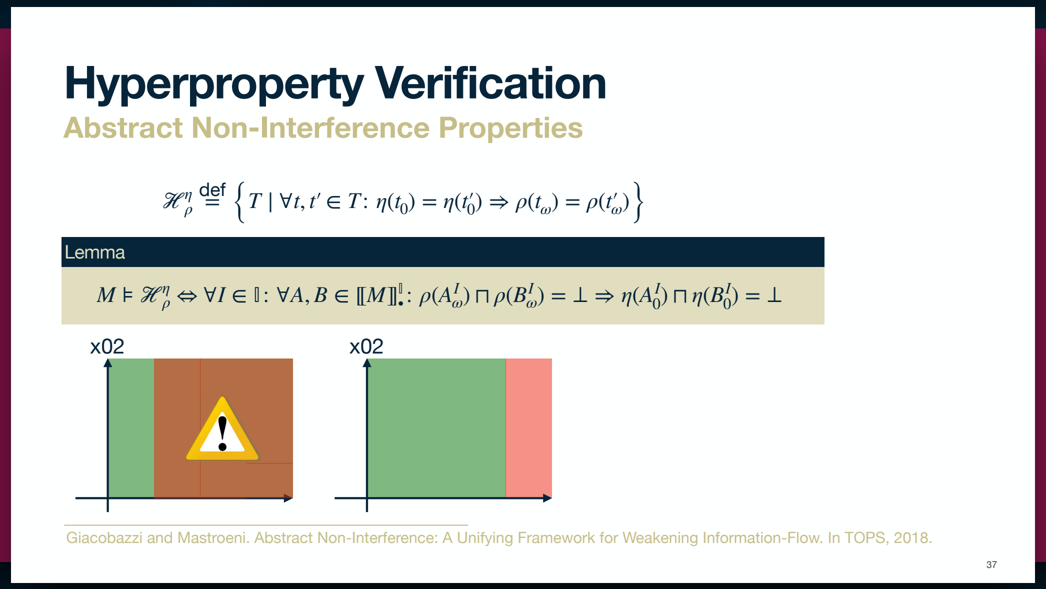




mathematical models

of the program behavior





THANKS!