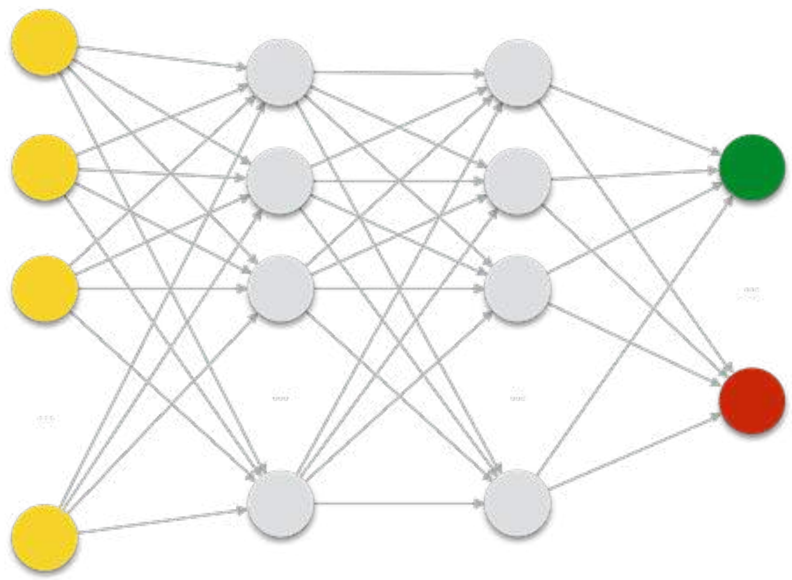
Formal Methods for Machine Learning

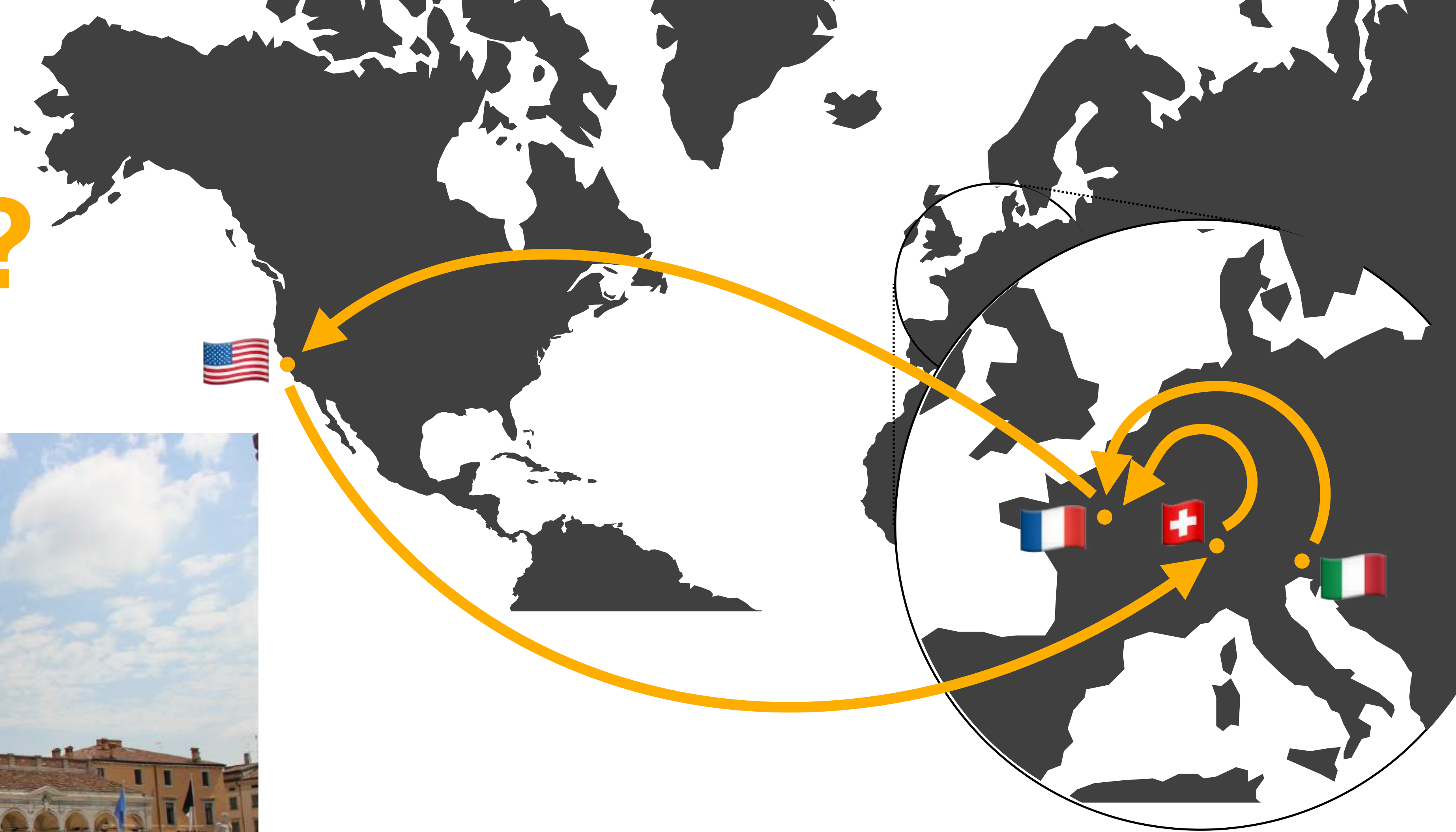
2nd Inria-DFKI European Summer School on Artificial Intelligence
Track A: Trusted AI



Caterina Urban

Tuesday, August 30th, 2022

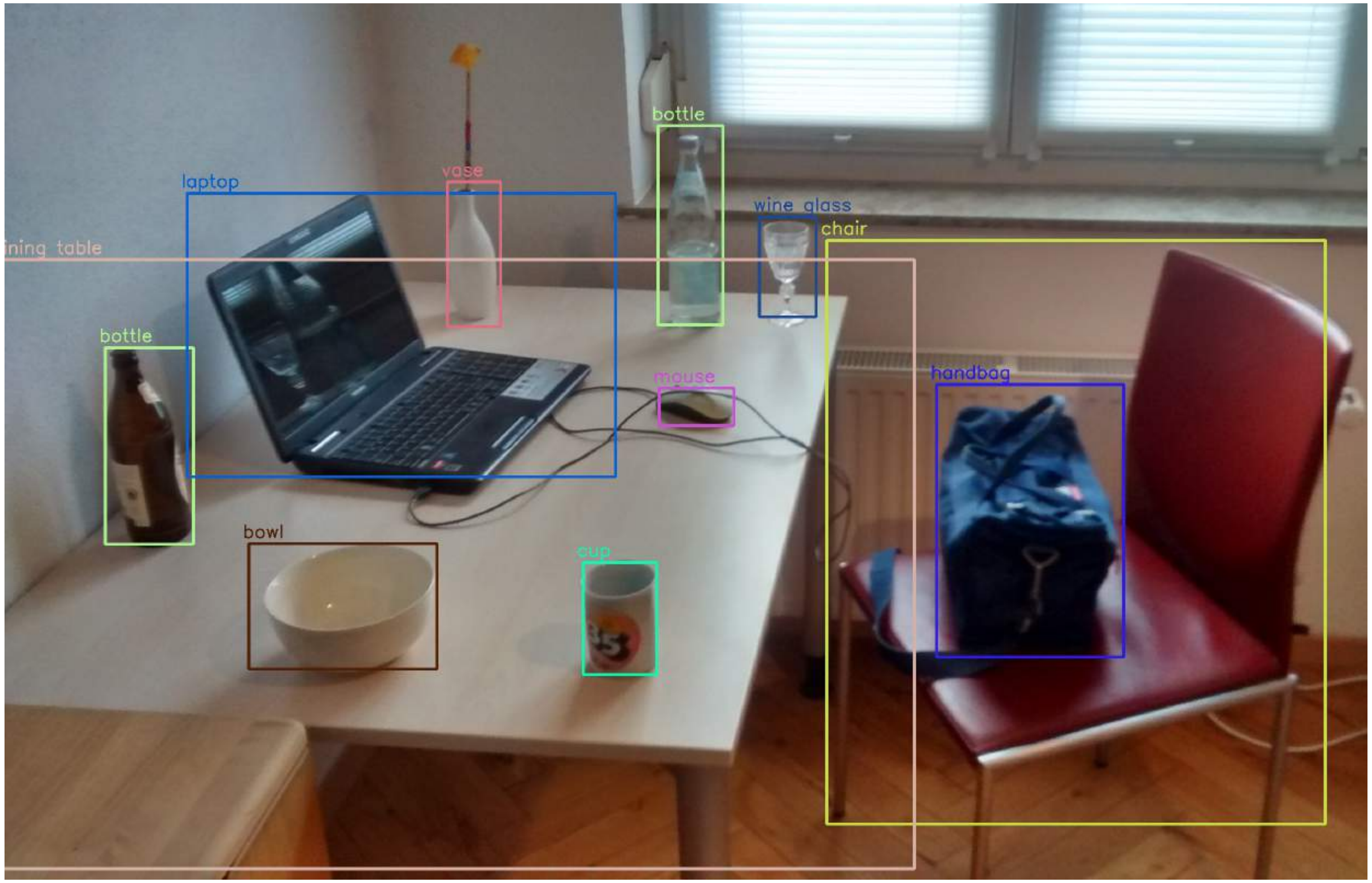
Who am I?



1987	Udine, Italie	
2006 - 2011	Università degli Studi di Udine	BSc, MSc
2011 - 2015	École Normale Supérieure	PhD
2015	NASA & Carnegie Mellon University	Internship
2015 - 2019	ETH Zurich	Postdoc
Since 2019	Inria	

Machine Learning Revolution

Computer software able to efficiently and **autonomously perform tasks** that are difficult or even *impossible* to design using explicit programming



Examples: object recognition, image classification, speech recognition, etc.

ML in Safety-Critical Applications

Enables **new functions that could not be envisioned before**

Self-Driving Cars

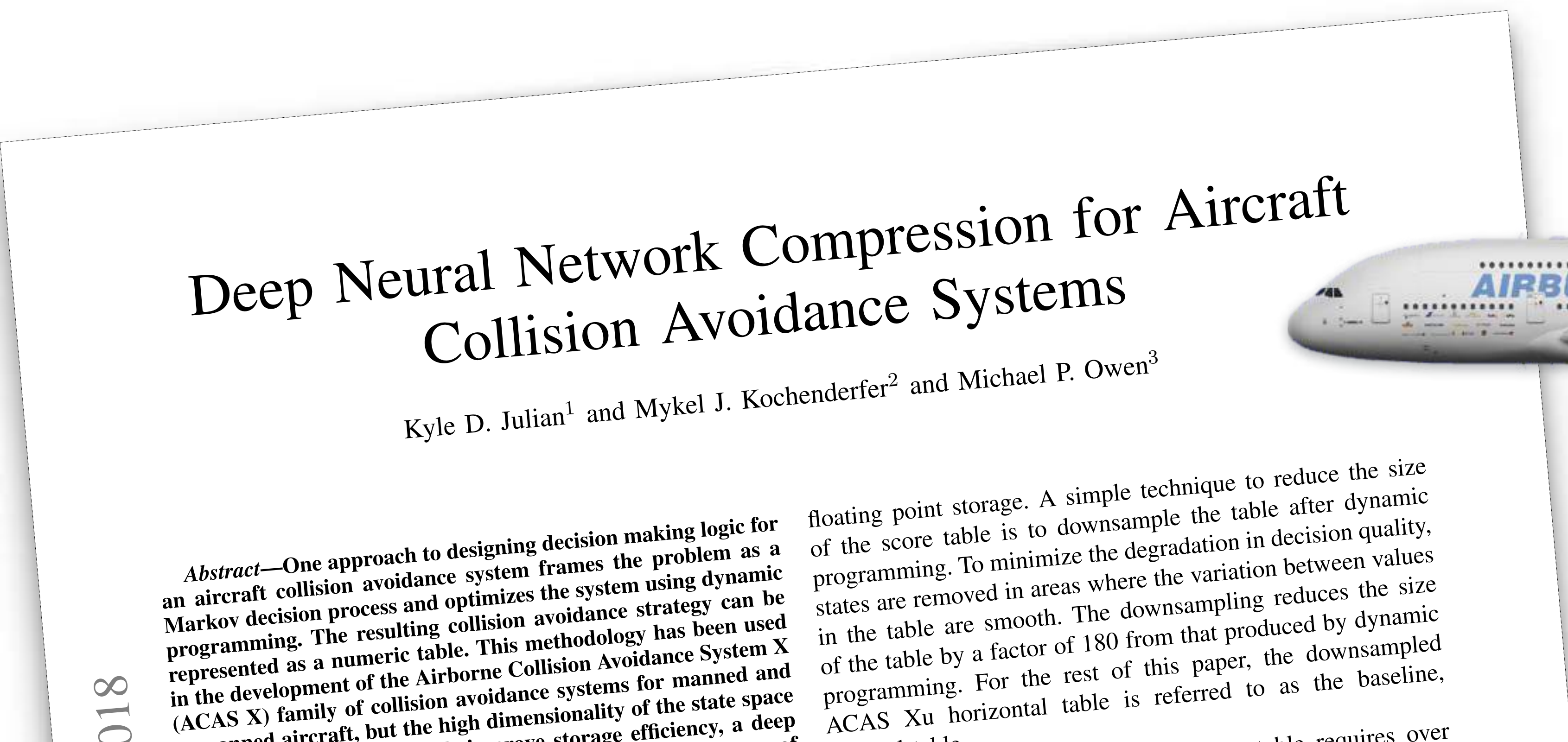
Image-Based Taxiing, Takeoff, Landing

Aircraft Voice Control

ML in Safety-Critical Applications

Approximates complex systems and **automates decision-making**

Diagnosis and Drug Discovery



Deep Neural Network Compression for Aircraft Collision Avoidance Systems

Kyle D. Julian[1] and Mykel J. Kochenderfer[2] and Michael P. Owen[3]

Abstract—One approach to designing decision making logic for an aircraft collision avoidance system frames the problem as a Markov decision process and optimizes the system using dynamic programming. The resulting collision avoidance strategy can be represented as a numeric table. This methodology has been used in the development of the Airborne Collision Avoidance System X (ACAS X) family of collision avoidance systems for manned and unmanned aircraft, but the high dimensionality of the state space

floating point storage. A simple technique to reduce the size of the score table is to downsample the table after dynamic programming. To minimize the degradation in decision quality, states are removed in areas where the variation between values in the table are smooth. The downsampling reduces the size of the table by a factor of 180 from that produced by dynamic programming. For the rest of this paper, the downsampled ACAS Xu horizontal table is referred to as the baseline, original table.

Aircraft Collision Avoidance

ML in Safety-Critical Applications

[1] STAT+ [2]

IBM's Watson supercomputer recommended 'unsafe and incorrect' cancer treatments, internal documents show

By Casey Ross[3] @caseymross[4] *and* Ike Swetlitz

July 25, 2018

A self-driving Uber ran a red light last December, contrary to company claims

Internal documents reveal that the car was at fault

By Andrew Liptak | @AndrewLiptak | Feb 25, 2017, 11:08am EST

Feds Say Self-Driving Uber SUV Did Not Recognize Jaywalking Pedestrian In Fatal Crash

Richard Gonzales November 7, 201910:57 PM ET

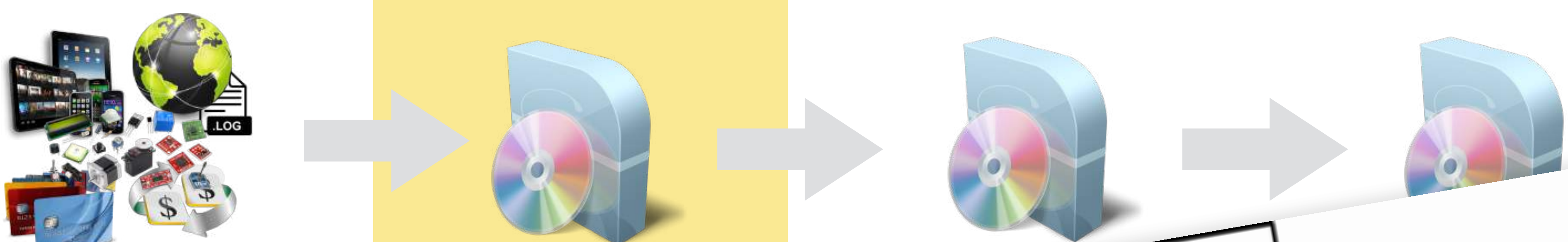
ML in Safety-Critical Applications

Machine Learning Pipeline

Machine Learning Pipeline

Data Preparation is Fragile

insidious silent bugs



data

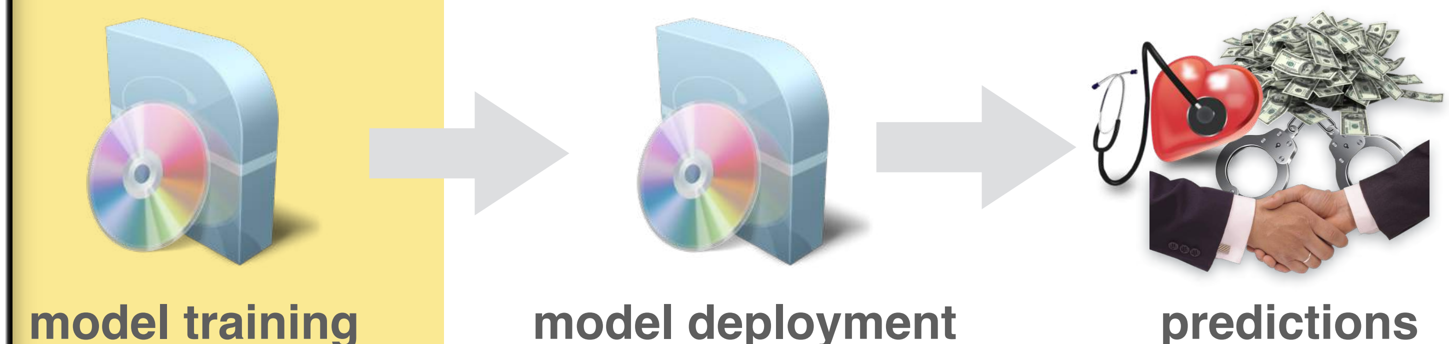
data preparation

https://xkcd.com/2054/

Machine Learning Pipeline

Model Training is Highly Non-Deterministic

https://xkcd.com/1838/



no **predictability** and **traceability**

Machine Learning Pipeline

Models Only Give **Probabilistic Guarantees**

data

https://xkcd.com/2451/

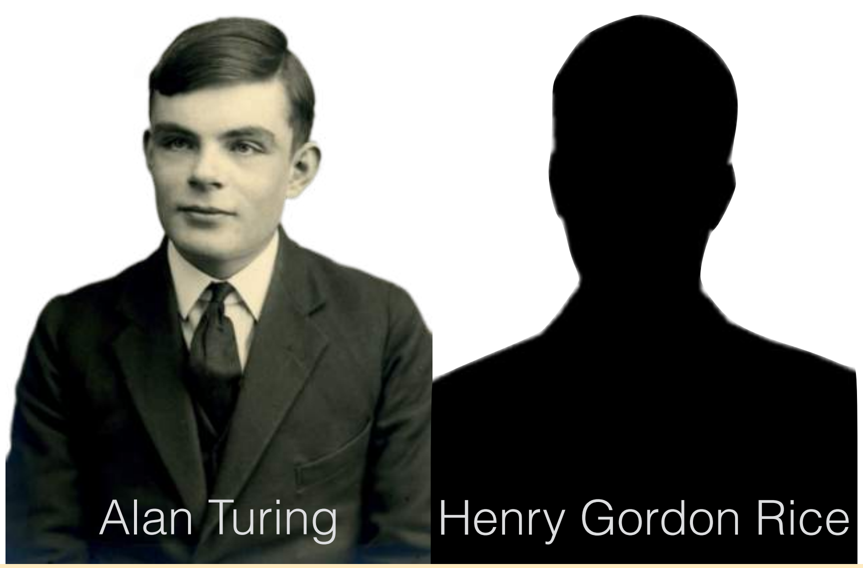
model deployment

predictions

not sufficient for guaranteeing an acceptable failure rate under any circumstance

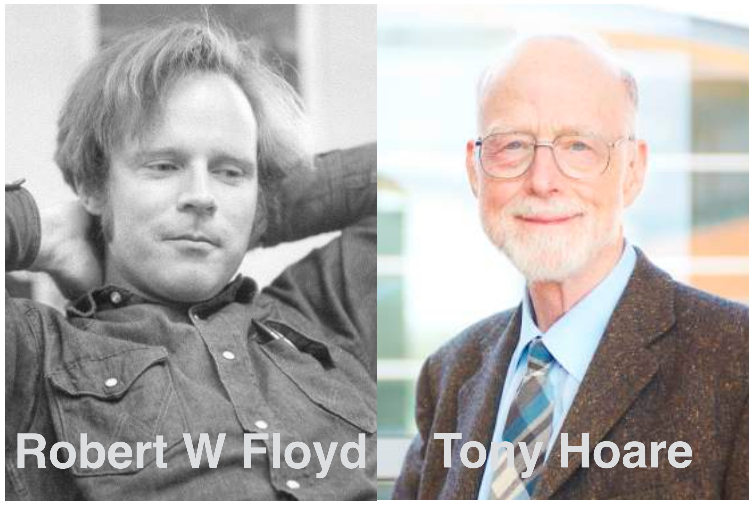
Correctness Guarantees

A Mathematically Proven Hard Problem

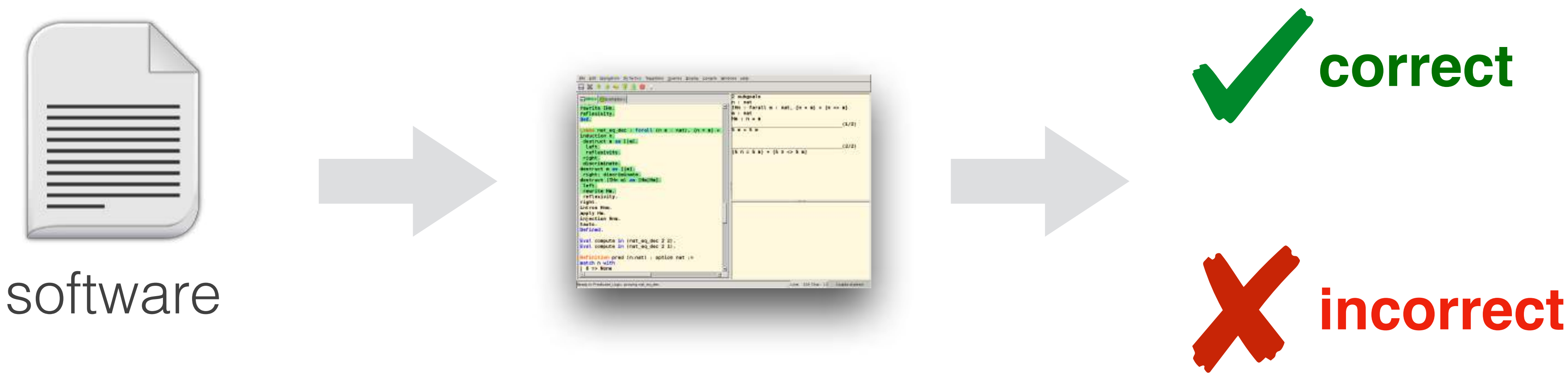


Formal Methods

Deductive Verification

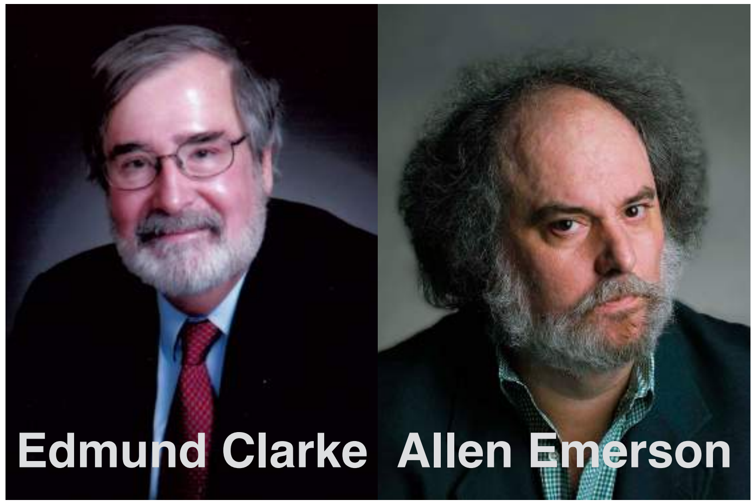


- extremely **expressive**
- **relies on the user** to guide the proof

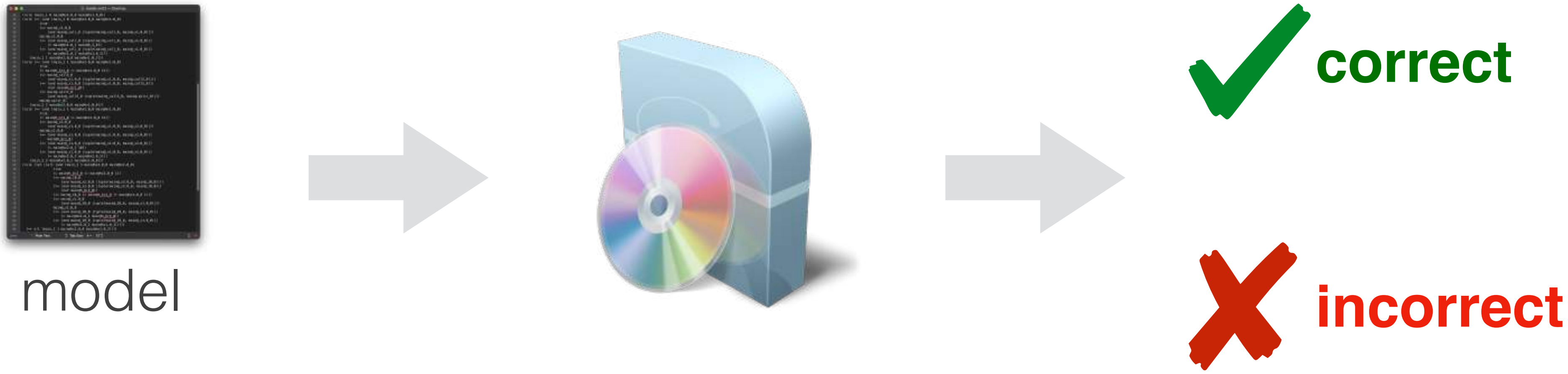


Formal Methods

Model Checking



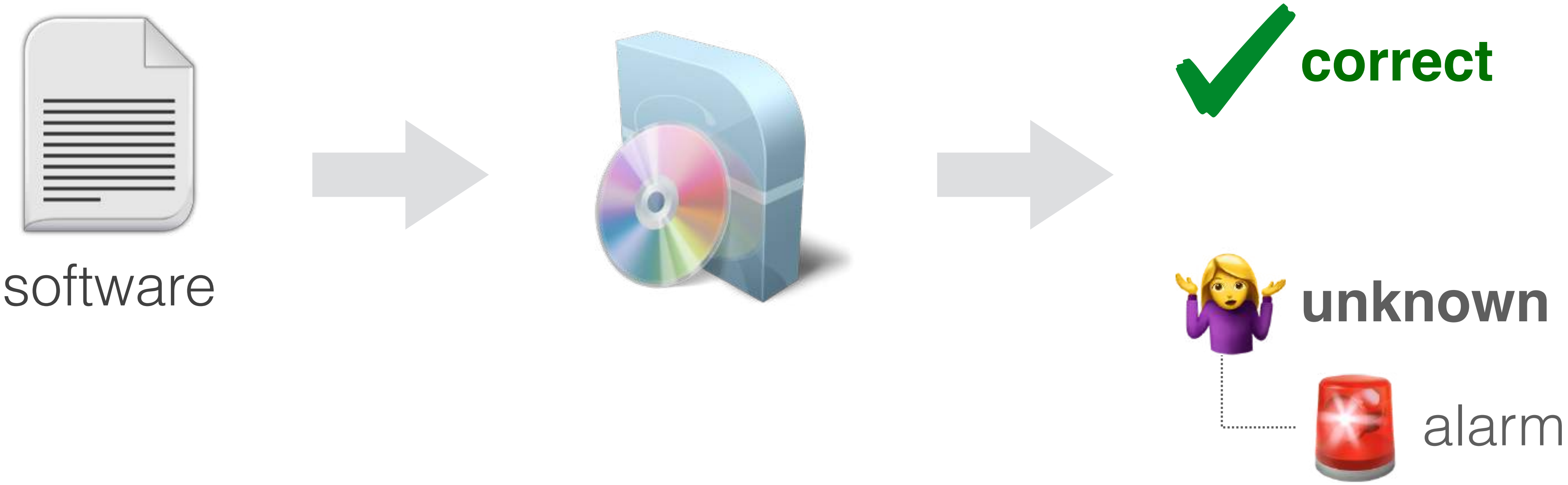
- analysis of a **model** of the software
- **sound and complete with respect to the model**



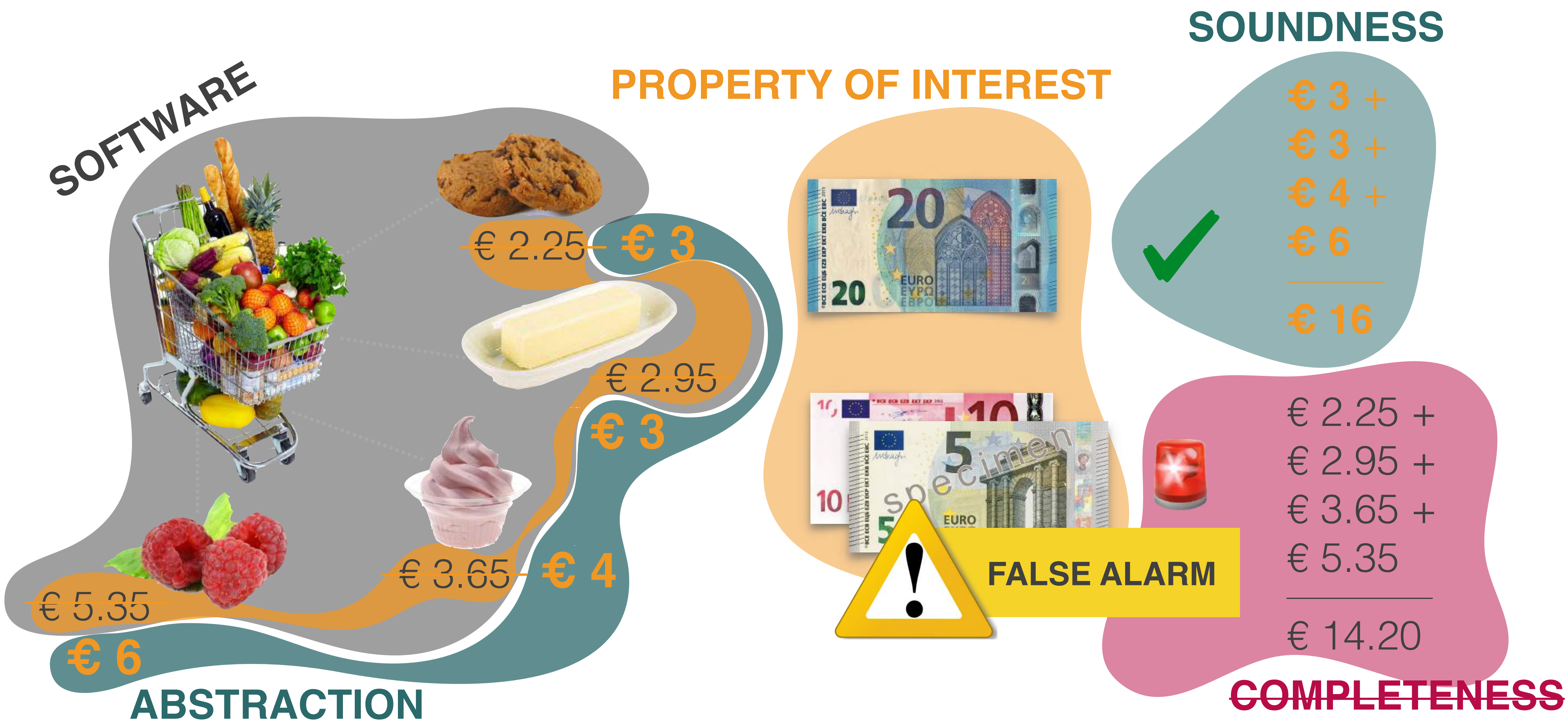
Formal Methods

Static Analysis by Abstract Interpretation

- analysis of the **source or object code**
- fully **automatic** and **sound** by construction
- generally **not complete**



Abstract Interpretation



Abstract Interpretation Today

integral part of the development of safety-critical software

aviation software

space software

automotive software

nuclear software

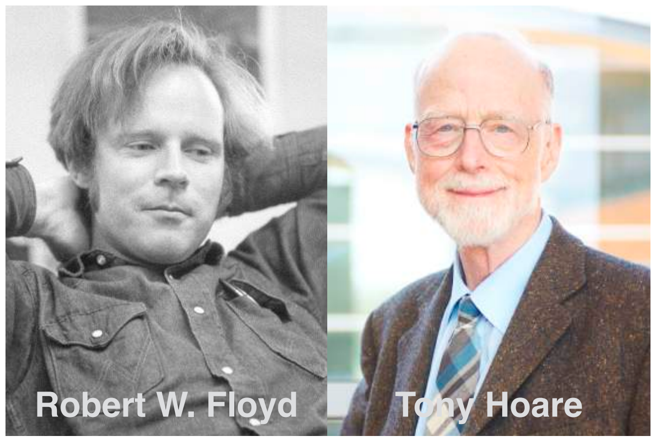
successfully employed by software companies

Google

Facebook

Microsoft

Formal Methods for ML



Deductive Verification

- extremely **expressive**
- **relies on the user** to guide the proof

Model Checking

- analysis of a **model** of the software
- **sound and complete**
  with respect to the model

Static Analysis

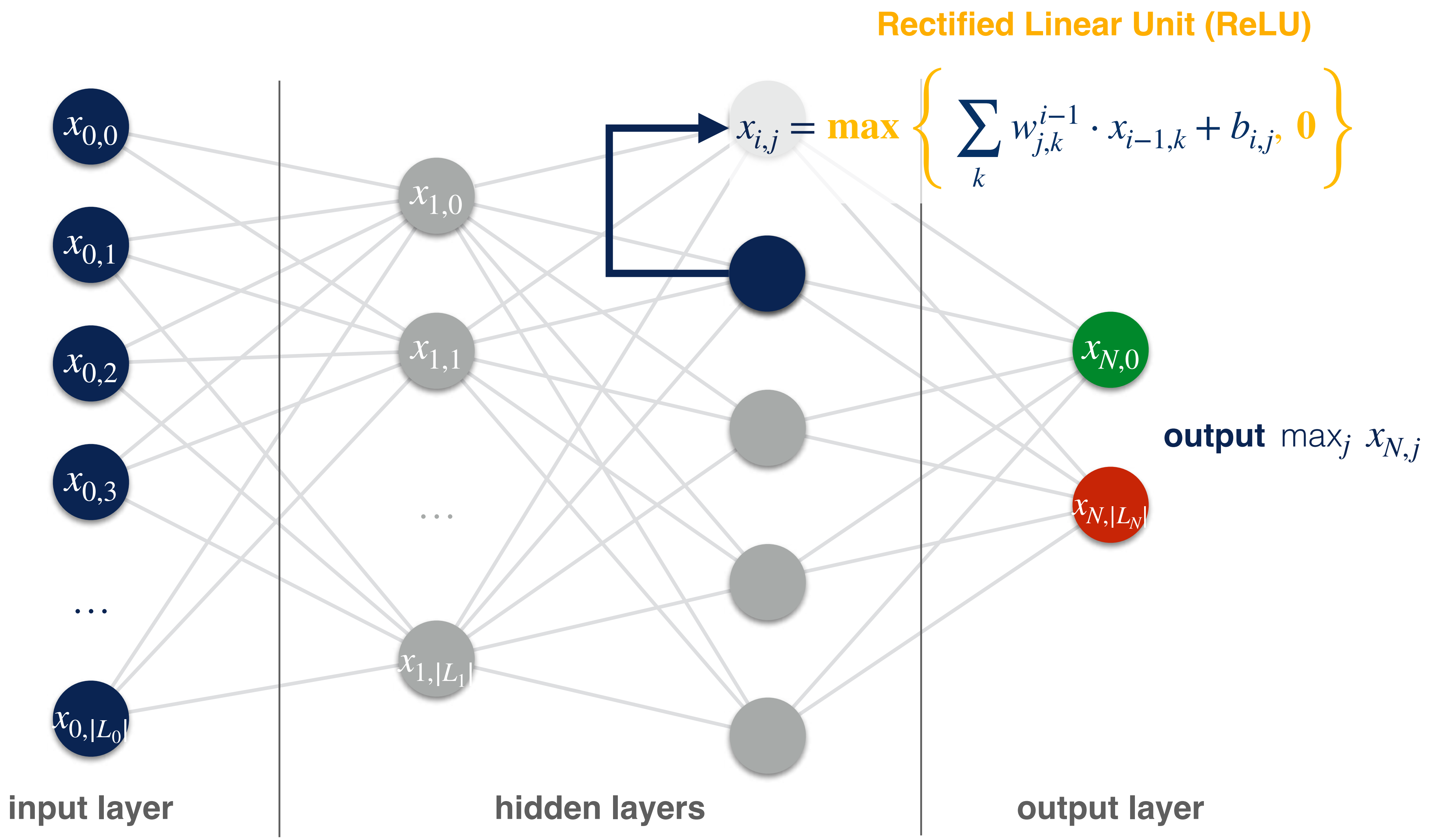
- analysis of the **source or object code**
- fully **automatic** and **sound** by construction
- generally **not complete**

Formal Methods for Trained Models

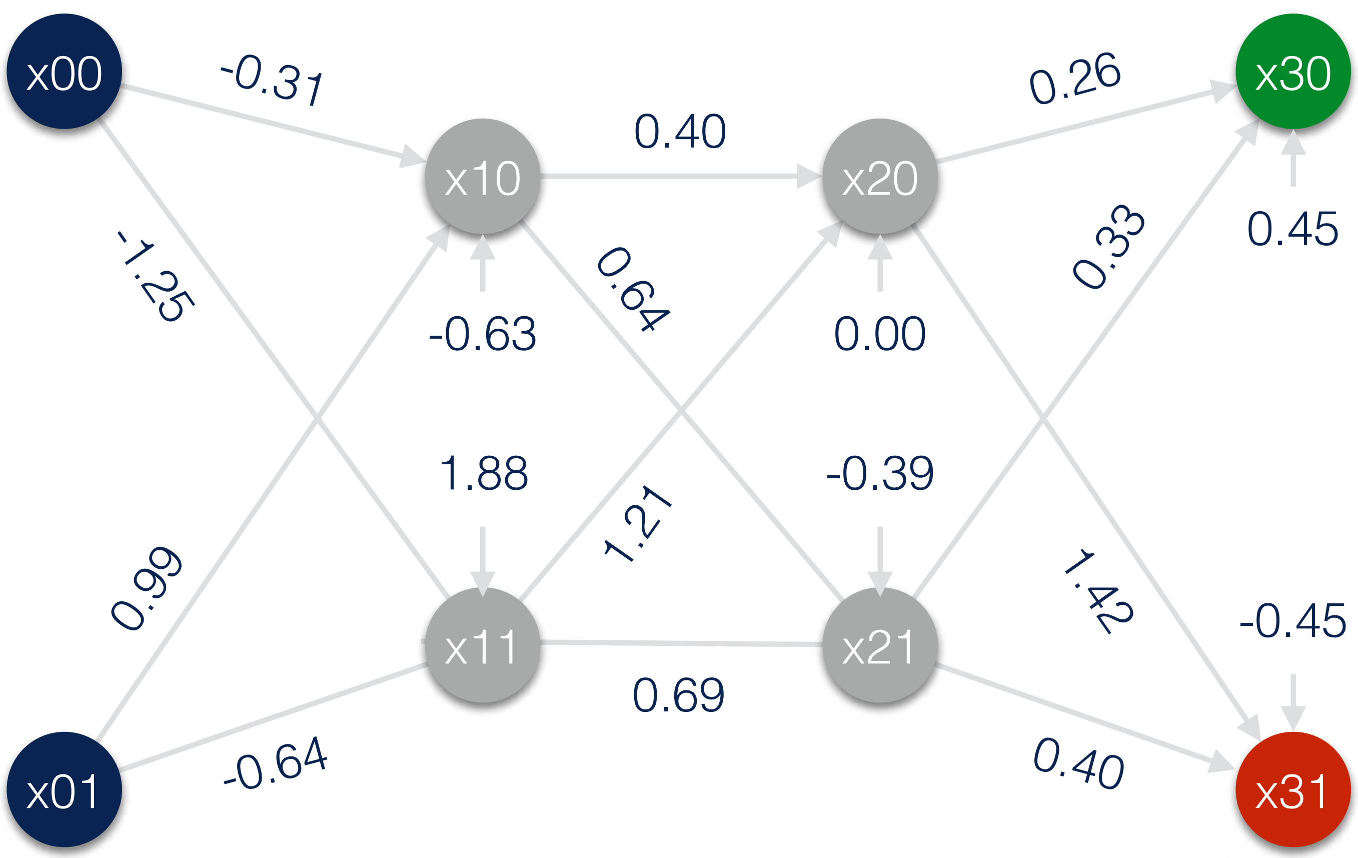
Neural Networks

Neural Networks

Feed-Forward Fully-Connected Neural Networks with ReLU Activation Functions

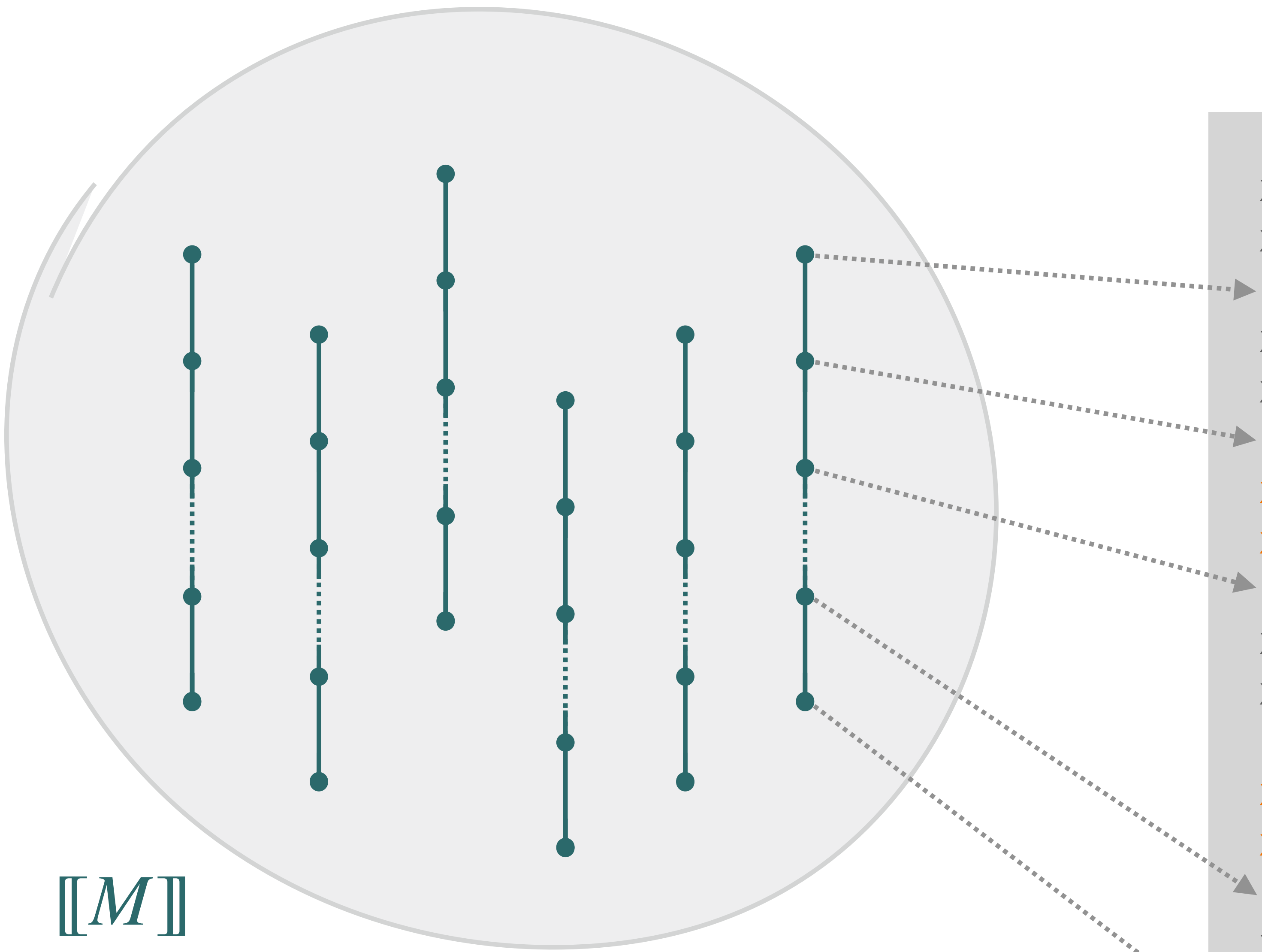


Feed-Forward Fully-Connected ReLU Networks as Programs




```
x00 = input()
x01 = input()

x10 = -0.31 * x00 + 0.99 * x01 + (-0.63)
x11 = -1.25 * x00 + (-0.64) * x01 + 1.88

x10 = 0 if x10 < 0 else x10
x11 = 0 if x11 < 0 else x11

x20 = 0.40 * x10 + 1.21 * x11 + 0.00
x21 = 0.64 * x10 + 0.69 * x11 + (-0.39)

x20 = 0 if x20 < 0 else x20
x21 = 0 if x21 < 0 else x21

x30 = 0.26 * x20 + 0.33 * x21 + 0.45
X31 = 1.42 * x20 + 0.40 * x21 + (-0.45)

return ‘ ’ if x31 < 30 else ‘ ’
```

Maximal Trace Semantics

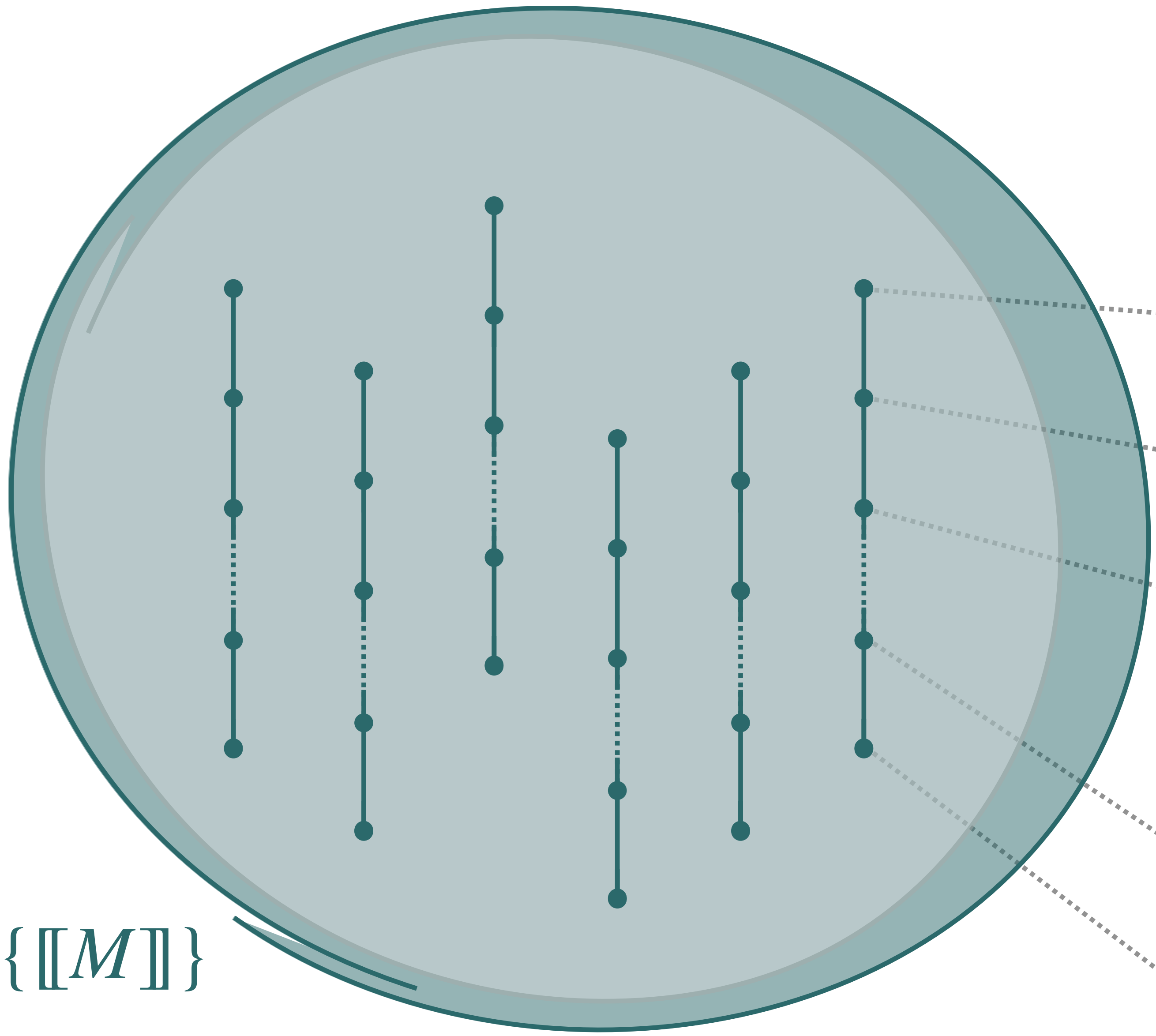


M

```
x00 = input()
x01 = input()

x10 = -0.31 * x00 + 0.99 * x01 + (-0.63)
x11 = -1.25 * x00 + (-0.64) * x01 + 1.88

x10 = 0 if x10 < 0 else x10
x11 = 0 if x11 < 0 else x11

x20 = 0.40 * x10 + 1.21 * x11 + 0.00
x21 = 0.64 * x10 + 0.69 * x11 + (-0.39)

x20 = 0 if x20 < 0 else x20
x21 = 0 if x21 < 0 else x21

x30 = 0.26 * x20 + 0.33 * x21 + 0.45
X31 = 1.42 * x20 + 0.40 * x21 + (-0.45)

return ' ' if x31 < 30 else ' '
```

Neural Network Verification

Collecting Semantics



$\{[\![M]\!]\}$

M

```
x00 = input()
x01 = input()

x10 = -0.31 * x00 + 0.99 * x01 + (-0.63)
x11 = -1.25 * x00 + (-0.64) * x01 + 1.88

x10 = 0 if x10 < 0 else x10
x11 = 0 if x11 < 0 else x11

x20 = 0.40 * x10 + 1.21 * x11 + 0.00
x21 = 0.64 * x10 + 0.69 * x11 + (-0.39)

x20 = 0 if x20 < 0 else x20
x21 = 0 if x21 < 0 else x21

x30 = 0.26 * x20 + 0.33 * x21 + 0.45
X31 = 1.42 * x20 + 0.40 * x21 + (-0.45)

return ‘ ’ if x31 < 30 else ‘ ’
```

Collecting Semantics

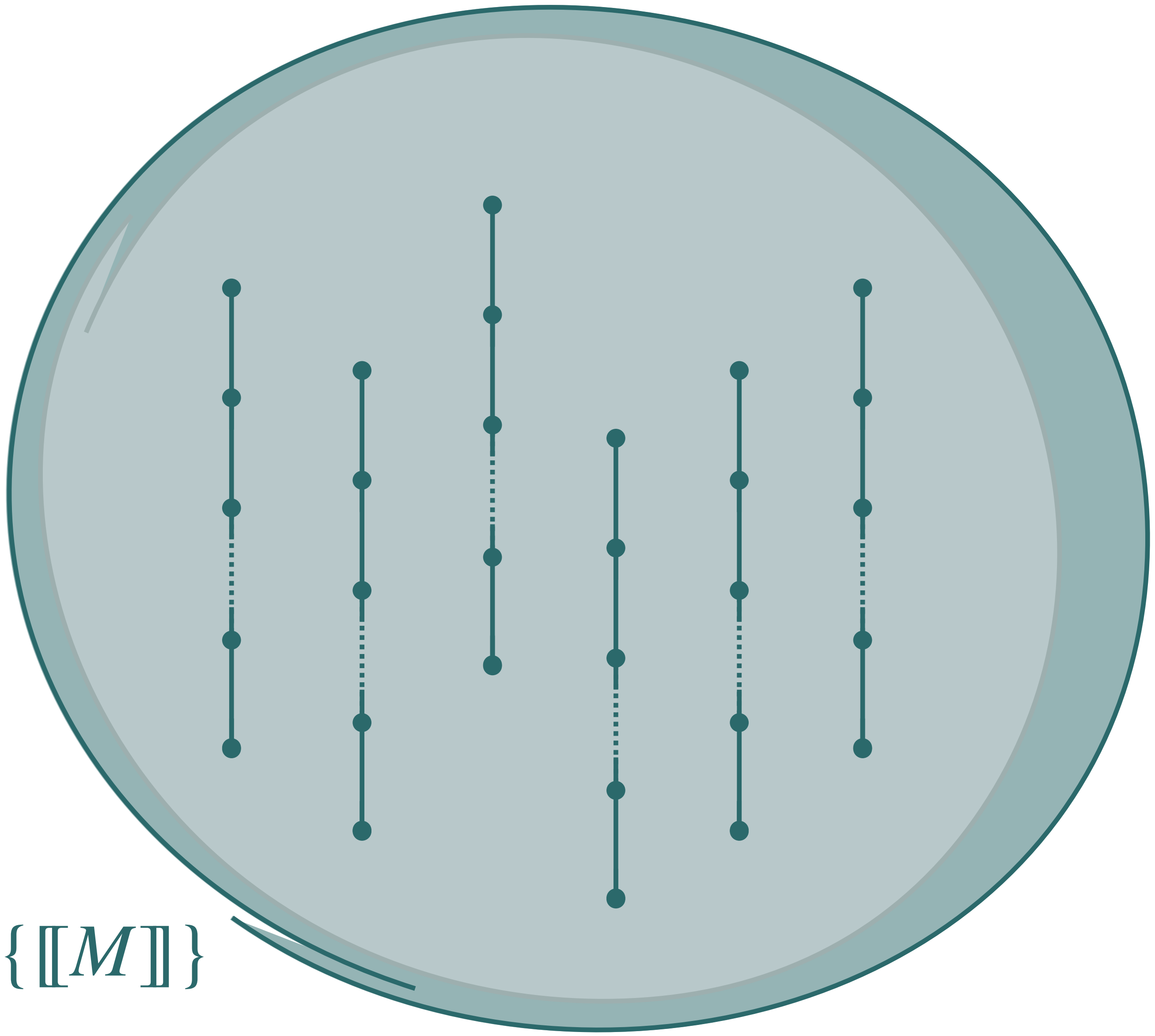
Intuition

Property (*by extension*): set of elements that have that property

Property “being Patrick Cousot”

Property “being neural network M”

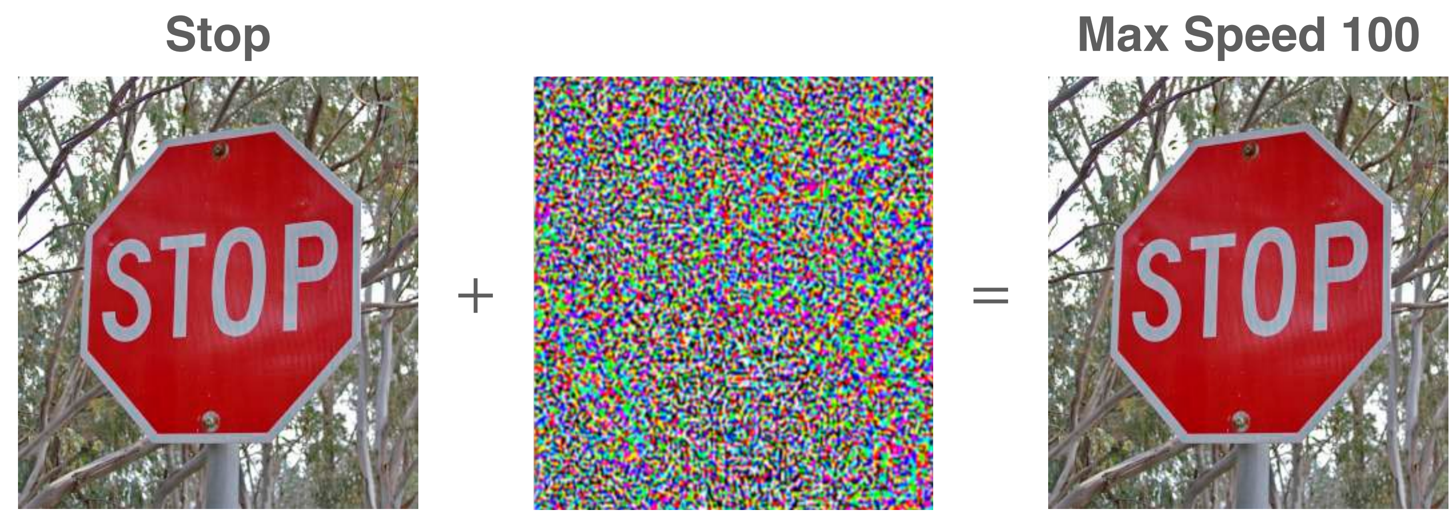
$\{ [\![M]\!] \}$



$\{ [\![M]\!] \}$

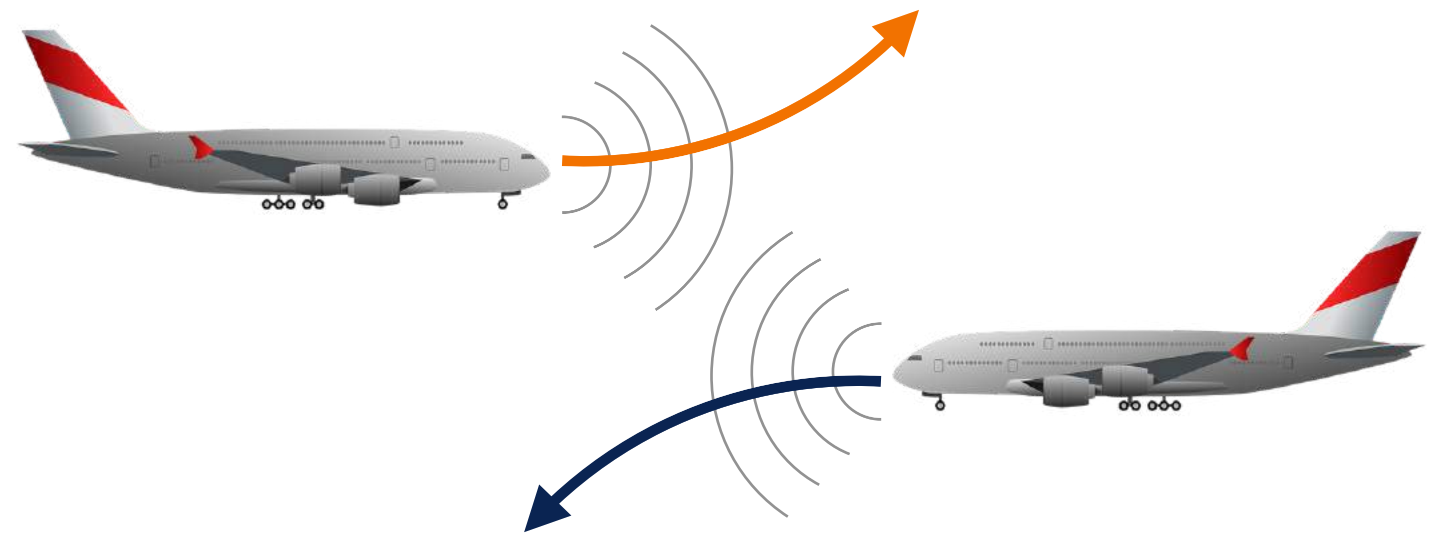
Stability

Goal G3 in [Kurd03]

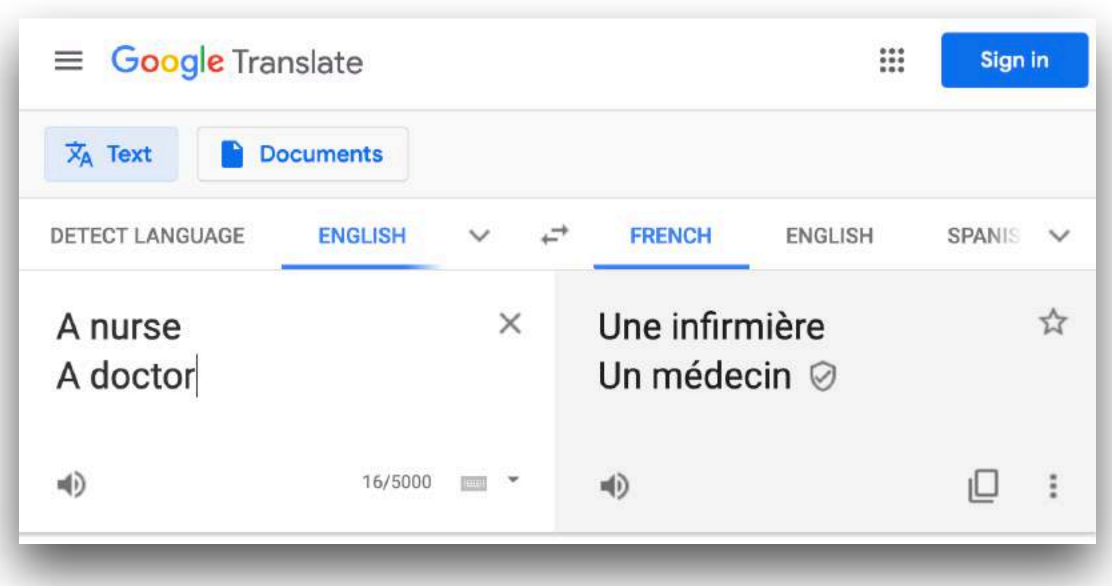


Safety

Goal G4 in [Kurd03]



Fairness



Stability

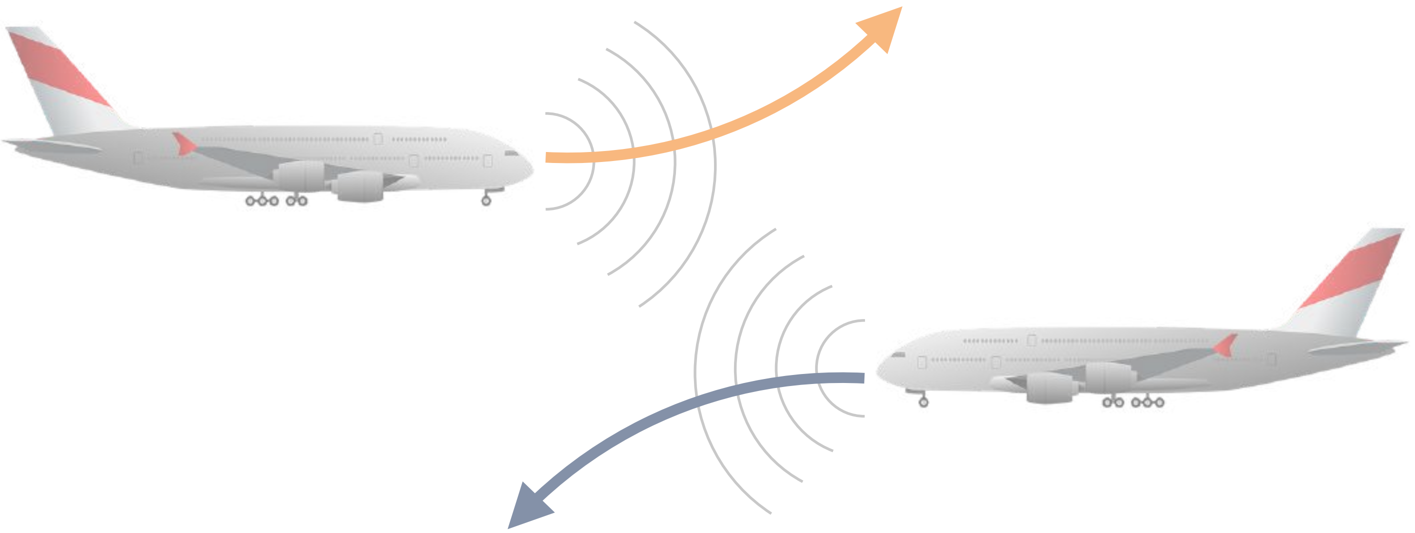
Goal G3 in [Kurd03]

Stop

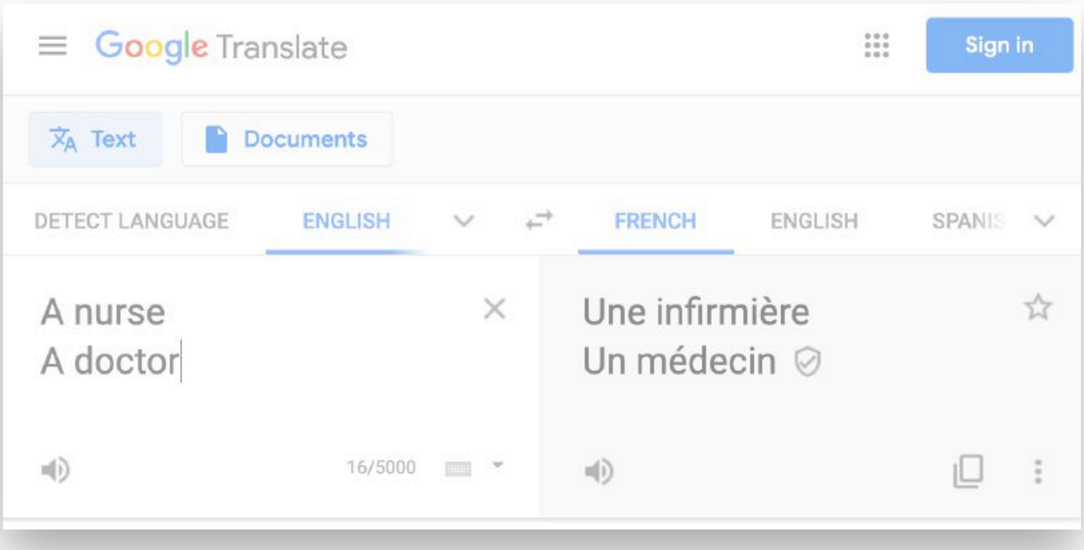
Max Speed 100

Safety

Goal G4 in [Kurd03]

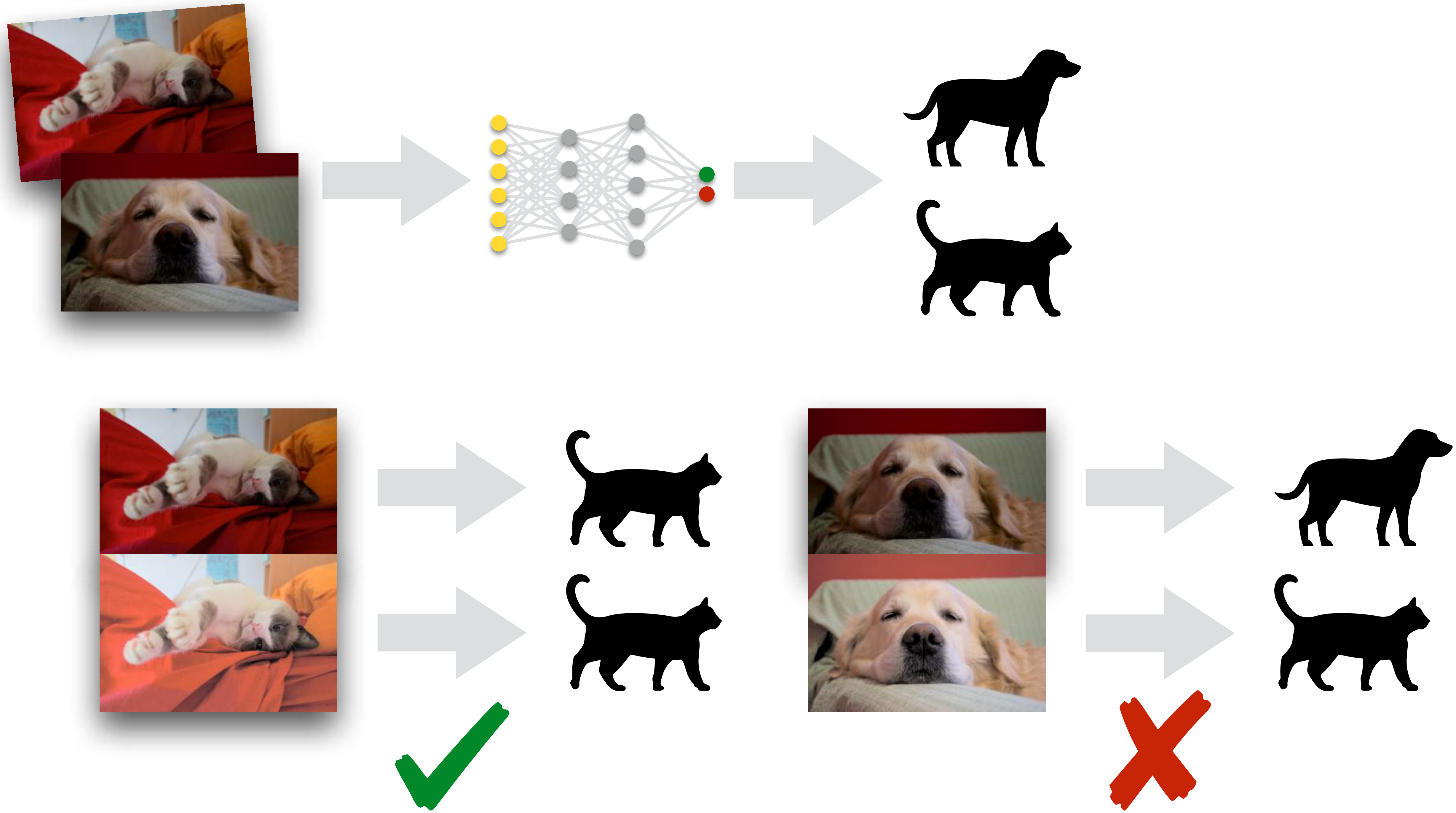


Fairness



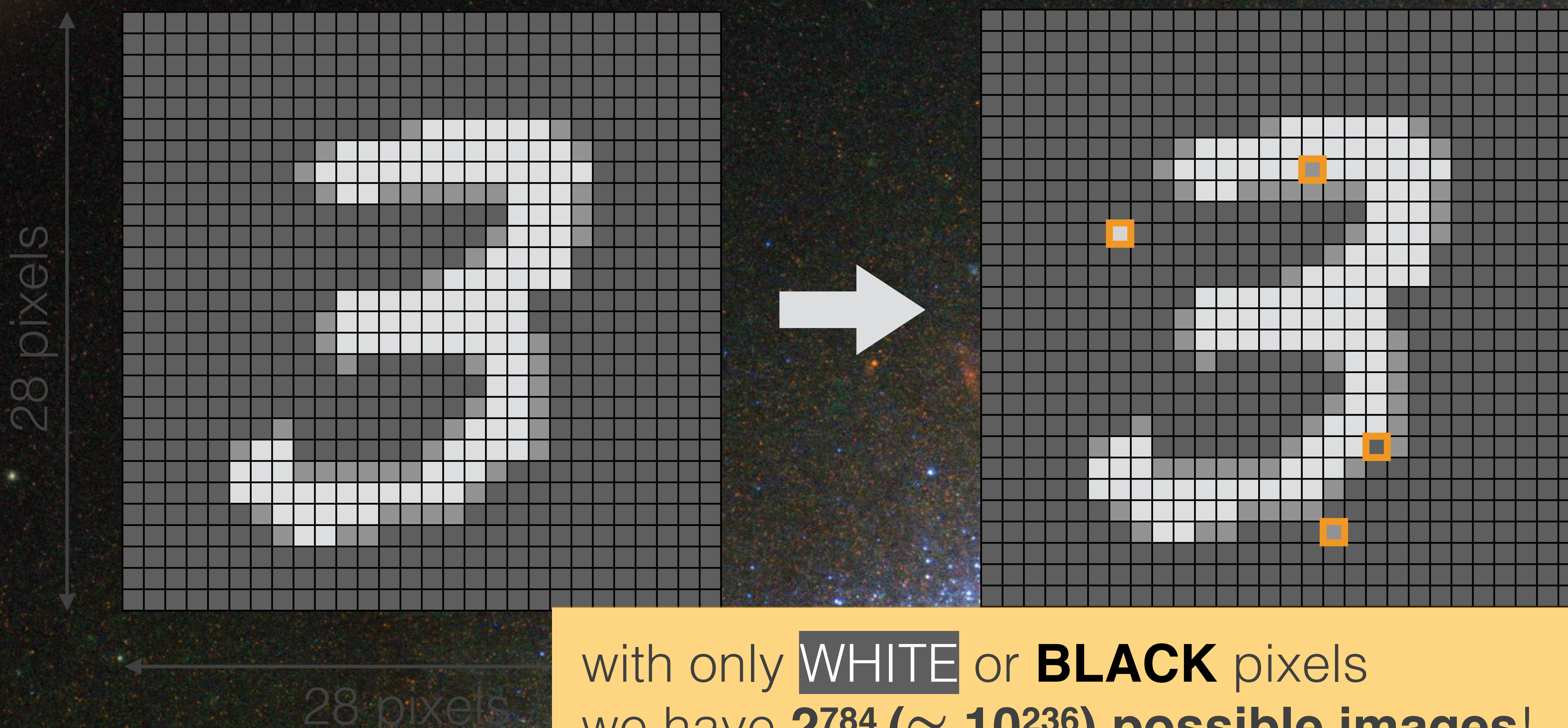
Local Stability

The **classification** is **unaffected by small input perturbations**



Local Stability

Random Noise



with only WHITE or **BLACK** pixels
we have **2^{784} ($\simeq$ 10^{236}) possible images**!

more than the estimated number of atoms in the visible universe ($\simeq$ 10^{80})!

Local Stability

Distance-Based Perturbations

$$P_{\delta,\epsilon}(\mathbf{x}) \stackrel{\text{def}}{=} \{\mathbf{x}' \in \mathscr{R}^{|L_0|} \mid \delta(\mathbf{x}, \mathbf{x}') \leq \epsilon\}$$

Example (L_∞ distance): $P_{\infty,\epsilon}(\mathbf{x}) \stackrel{\text{def}}{=} \{\mathbf{x}' \in \mathscr{R}^{|L_0|} \mid \text{max}_i |\mathbf{x}_i - \mathbf{x}'_i| \leq \epsilon\}$

$$\mathscr{R}_{\mathbf{x}}^{\delta,\epsilon} \stackrel{\text{def}}{=} \{[\![M]\!] \in \mathscr{P}(\Sigma^*) \mid \text{STABLE}_{\mathbf{x}}^{\delta,\epsilon}([\![M]\!])\}$$

$\mathscr{R}_{\mathbf{x}}^{\delta,\epsilon}$ is the set of all neural networks M (or, rather, their semantics $[\![M]\!]$) that are **stable** in the neighborhood $P_{\delta,\epsilon}(\mathbf{x})$ of a given input $\mathbf{x}$

$$\begin{aligned}\text{STABLE}_{\mathbf{x}}^{\delta,\epsilon}([\![M]\!]) \stackrel{\text{def}}{=}\ & \forall t \in [\![M]\!] \colon (\exists t' \in [\![M]\!] \colon \forall 0 \leq i \leq |L_0| \colon t'_0(x_{0,i}) = \mathbf{x}_i) \\ & \wedge (\exists \mathbf{x}' \in P_{\delta,\epsilon}(\mathbf{x}) \colon \forall 0 \leq i \leq |L_0| \colon t_0(x_{0,i}) = \mathbf{x}'_i) \\ & \Rightarrow \text{max}_j\, t_\omega(x_{N,j}) = \text{max}_j\, t'_\omega(x_{N,j})\end{aligned}$$

Theorem

$$M \models \mathscr{R}_{\mathbf{x}}^{\delta,\epsilon} \Leftrightarrow \{[\![M]\!]\} \subseteq \mathscr{R}_{\mathbf{x}}^{\delta,\epsilon}$$

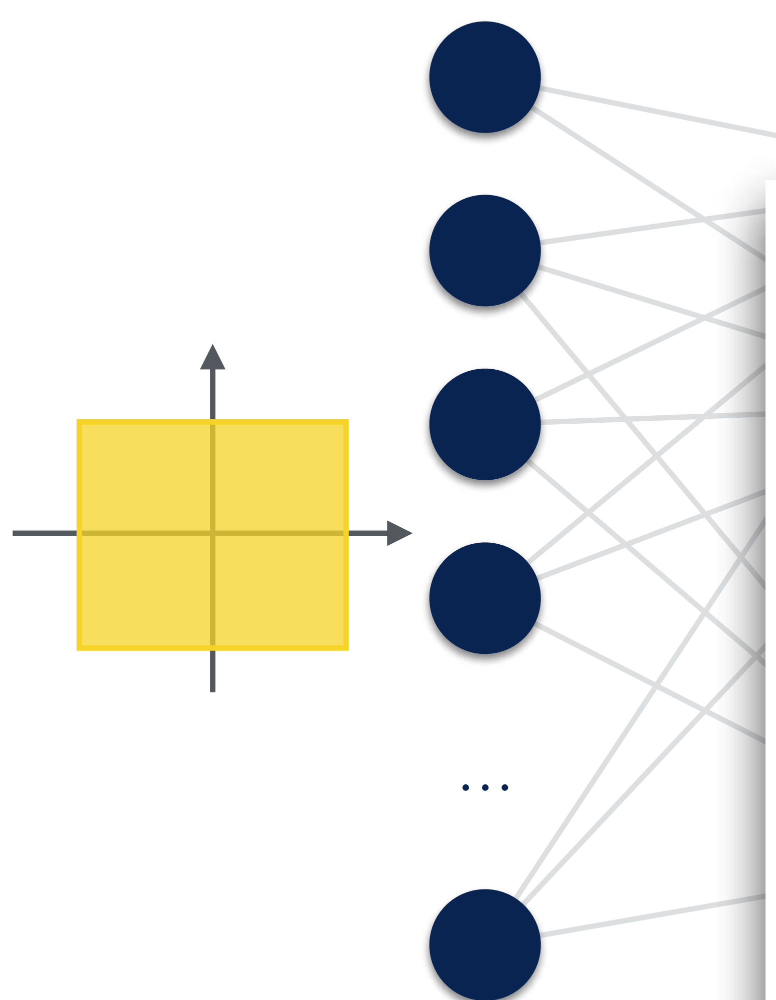
Corollary

$$M \models \mathscr{R}_{\mathbf{x}}^{\delta,\epsilon} \Leftrightarrow [\![M]\!] \subseteq \bigcup \mathscr{R}_{\mathbf{x}}^{\delta,\epsilon}$$

Static Analysis Methods

Forward Analysis

② check output for **inclusion** in **expected output**:
included → ✓ **stable**
otherwise → 🚨 **alarm**



Local Stability

Distance-Based Perturbations

$P_{\delta,\epsilon}(\mathbf{x}) \stackrel{\text{def}}{=} \{\mathbf{x}' \in \mathscr{R}^{|L_0|} \mid \delta(\mathbf{x}, \mathbf{x}') \leq \epsilon\}$

Example (L_∞ distance): $P_{\infty,\epsilon}(\mathbf{x}) \stackrel{\text{def}}{=} \{\mathbf{x}' \in \mathscr{R}^{|L_0|} \mid \max_i |\mathbf{x}_i - \mathbf{x}'_i| \leq \epsilon\}$

$$\mathscr{R}_{\mathbf{x}}^{\delta,\epsilon} \stackrel{\text{def}}{=} \{[\![M]\!] \in \mathscr{P}(\Sigma^*) \mid \text{STABLE}_{\mathbf{x}}^{\delta,\epsilon}([\![M]\!])\}$$

$\mathscr{R}_{\mathbf{x}}^{\delta,\epsilon}$ is the set of all neural networks M (or, rather, their semantics $[\![M]\!]$) that are **stable** in the neighborhood $P_{\delta,\epsilon}(\mathbf{x})$ of a given input $\mathbf{x}$

$$\begin{aligned}\text{STABLE}_{\mathbf{x}}^{\delta,\epsilon}([\![M]\!]) \stackrel{\text{def}}{=} \; & \forall t \in [\![M]\!] \colon (\exists t' \in [\![M]\!] \colon \forall 0 \leq i \leq |L_0| \colon t'_0(x_{0,i}) = \mathbf{x}_i) \\ & \wedge (\exists \mathbf{x}' \in P_{\delta,\epsilon}(\mathbf{x}) \colon \forall 0 \leq i \leq |L_0| \colon t_0(x_{0,i}) = \mathbf{x}'_i) \\ & \Rightarrow \max_j t_\omega(x_{N,j}) = \max_j t'_\omega(x_{N,j})\end{aligned}$$

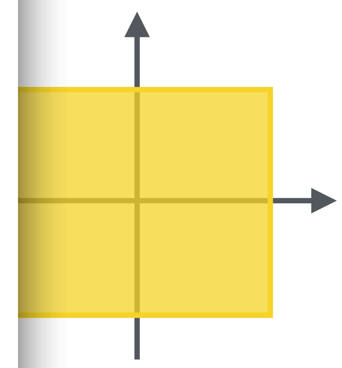
Theorem

$M \models \mathscr{R}_{\mathbf{x}}^{\delta,\epsilon} \Leftrightarrow \{[\![M]\!]\} \subseteq \mathscr{R}_{\mathbf{x}}^{\delta,\epsilon}$

Corollary

$M \models \mathscr{R}_{\mathbf{x}}^{\delta,\epsilon} \Leftrightarrow [\![M]\!] \subseteq \bigcup \mathscr{R}_{\mathbf{x}}^{\delta,\epsilon}$

31

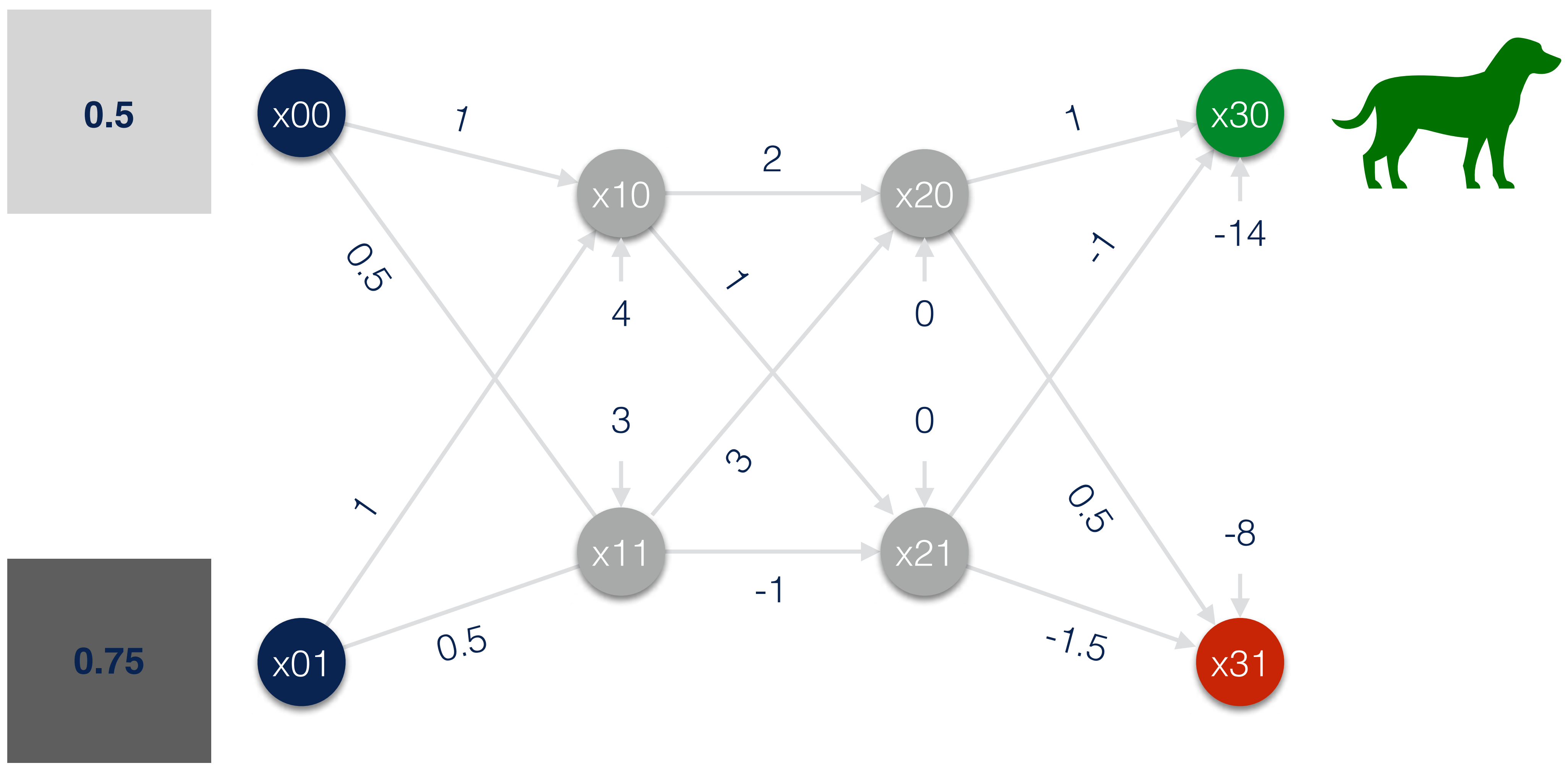


① proceed **forwards from an abstraction** of all possible perturbations

Theorem

$$\{[\![M]\!]\} \subseteq \{[\![M]\!]\}^\natural \subseteq \mathscr{R}_{\mathbf{x}}^{\delta,\epsilon} \Rightarrow M \models \mathscr{R}_{\mathbf{x}}^{\delta,\epsilon}$$

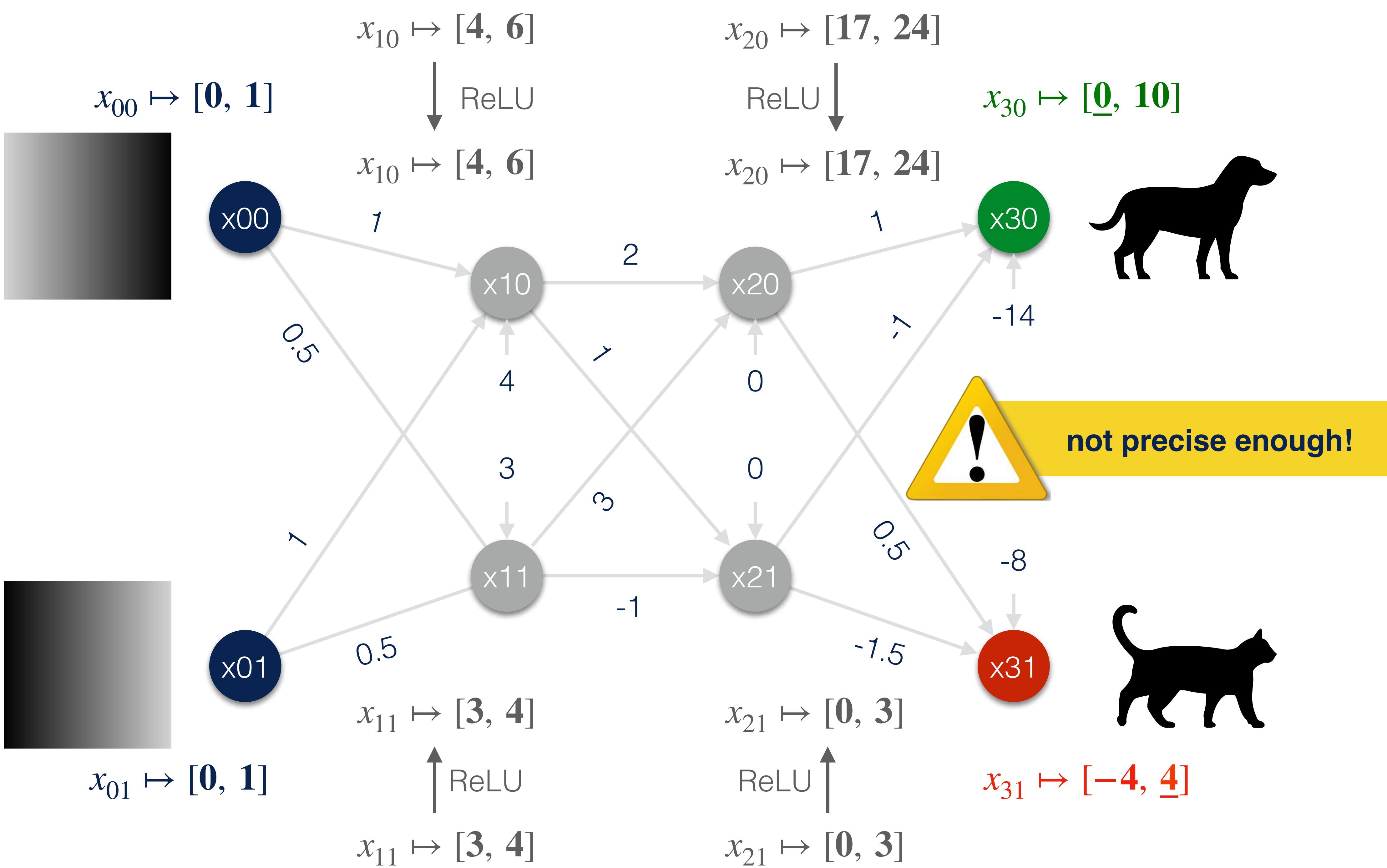
Example



$$P(\langle 0.5, 0.75 \rangle) \stackrel{\text{def}}{=} \{\mathbf{x} \in \mathscr{R} \times \mathscr{R} \mid 0 \leq \mathbf{x}_0 \leq 1 \wedge 0 \leq \mathbf{x}_1 \leq 1\}$$

Interval Abstraction

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



Abstract Interpretation

Improving Precision

~~€ 2.25~~ **€ 2,5**

~~€ 2.95~~

€ 3

~~€ 3.65~~ **€ 4**

~~€ 5.35~~

€ 5,5

€ 2,5 +
€ 3 +
€ 4 +
€ 5,5

€ 15

€ 2.25 +
€ 2.95 +
€ 3.65 +
€ 5.35

€ 14.20

Interval Abstraction

with Symbolic Constant Propagation [Li19]

each neuron as a linear combination of the inputs and the previous ReLUs

$$x_{i,j} \mapsto \begin{cases} \sum_{k=0}^{i-1} \mathbf{c}_k \cdot \mathbf{x}_k + \mathbf{c} & \mathbf{c}_k, \mathbf{c} \in \mathscr{R}^{|\mathbf{X}_k|} \\ [a, b] & a, b \in \mathscr{R} \end{cases}$$

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

$\ldots$

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

$\ldots$

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

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

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

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

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

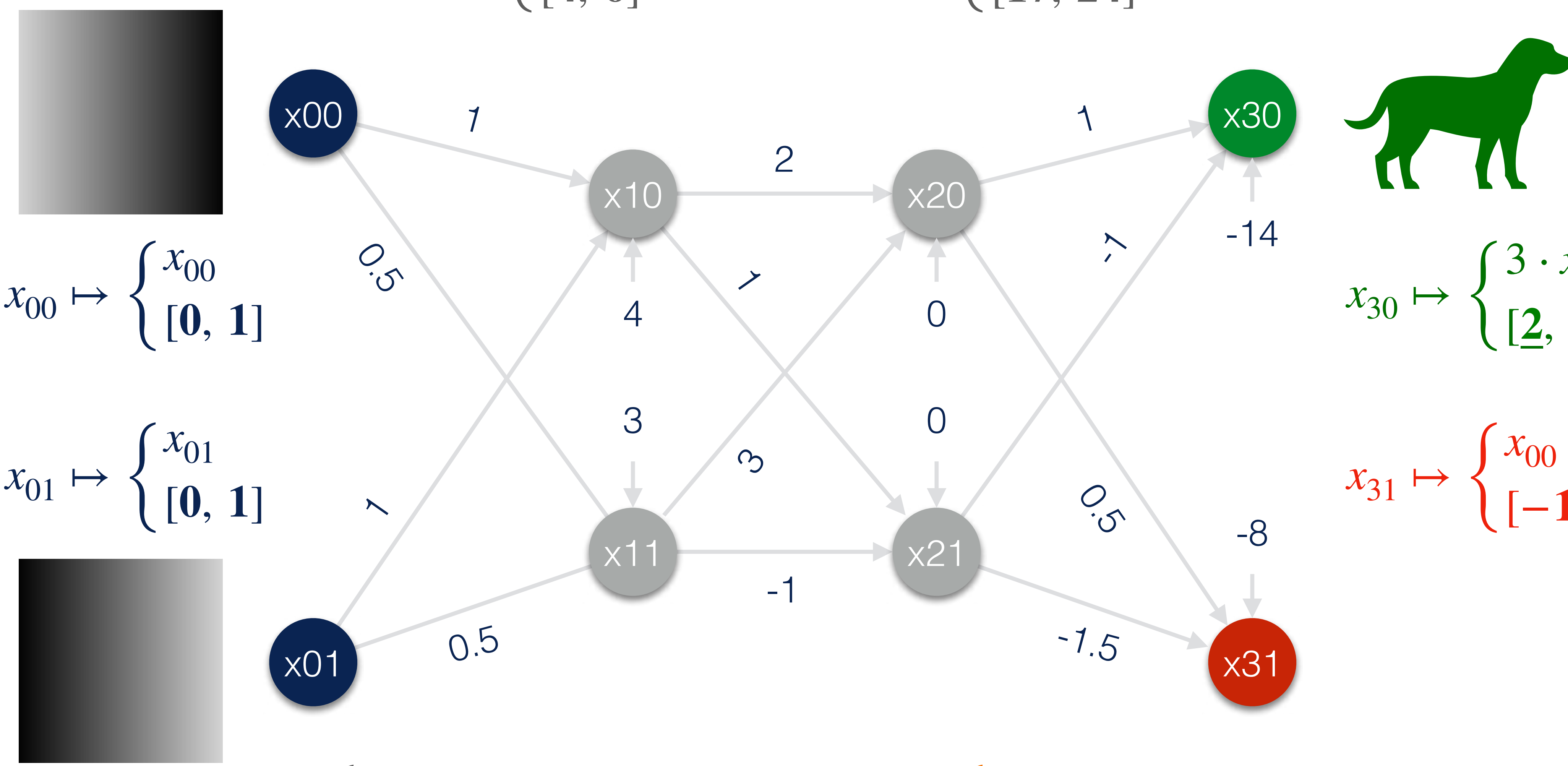
$$x_{i,j} \mapsto \begin{cases} \mathbf{0} \\ [\mathbf{0}, \mathbf{0}] \end{cases} \qquad b \leq 0$$

J. Li et al. - Analyzing Deep Neural Networks with Symbolic Propagation (SAS 2019)

Interval Abstraction

with Symbolic Constant Propagation [Li19]

$x_{10} \mapsto \begin{cases} x_{00} + x_{01} + 4 \\ [\mathbf{4}, \mathbf{6}] \end{cases}$ $\quad x_{20} \mapsto \begin{cases} 2 \cdot (x_{00} + x_{01} + 4) + 3 \cdot (0.5 \cdot x_{00} + 0.5 \cdot x_{01} + 3) \\ [\mathbf{17}, \mathbf{24}] \end{cases}$



$x_{00} \mapsto \begin{cases} x_{00} \\ [\mathbf{0}, \mathbf{1}] \end{cases}$

$x_{01} \mapsto \begin{cases} x_{01} \\ [\mathbf{0}, \mathbf{1}] \end{cases}$

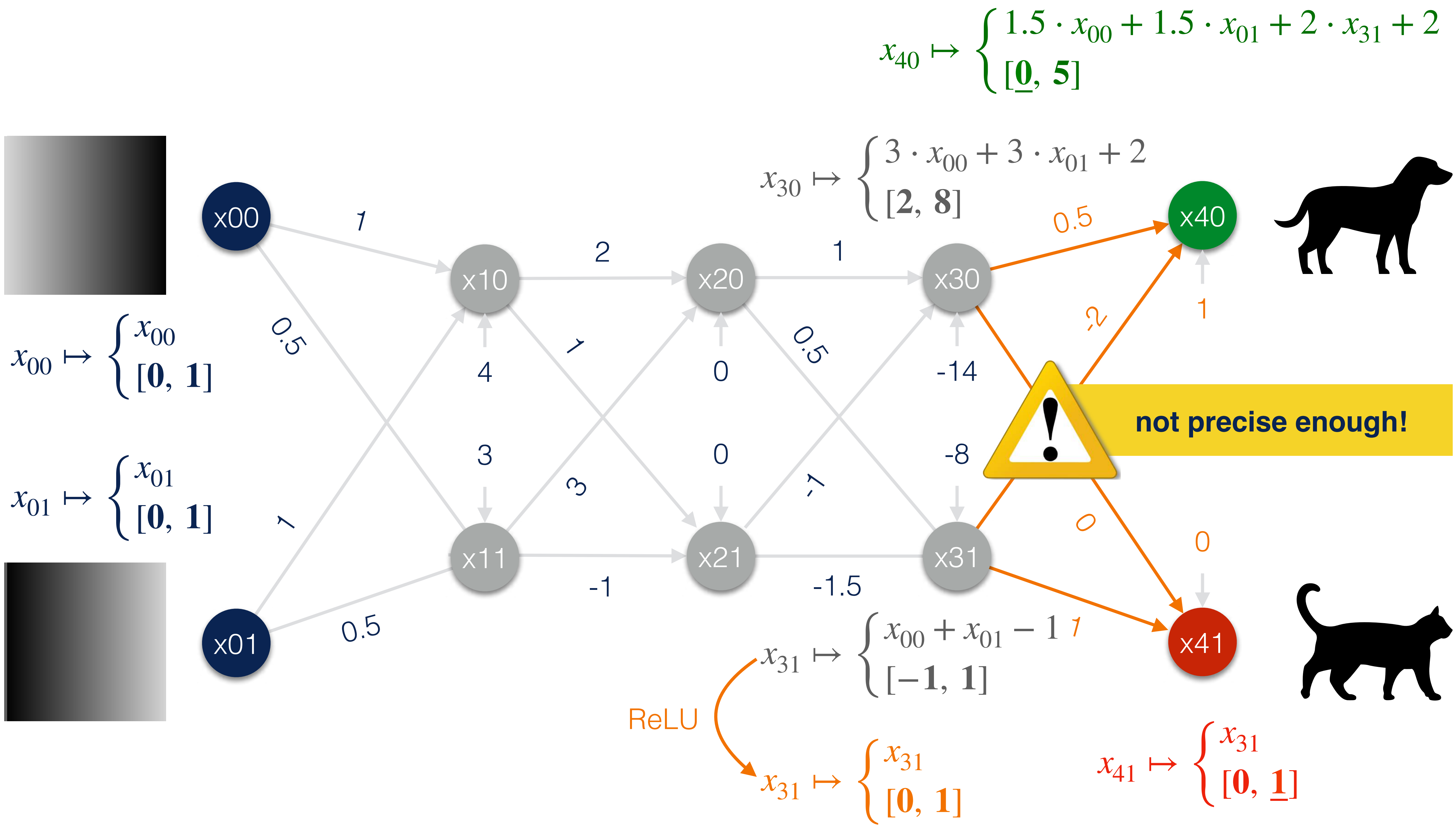
$x_{30} \mapsto \begin{cases} 3 \cdot x_{00} + 3 \cdot x_{01} + 2 \\ [\underline{\mathbf{2}}, \mathbf{8}] \end{cases}$

$x_{31} \mapsto \begin{cases} x_{00} + x_{01} - 1 \\ [\mathbf{-1}, \underline{\mathbf{1}}] \end{cases}$

$x_{11} \mapsto \begin{cases} 0.5 \cdot x_{00} + 0.5 \cdot x_{01} + 3 \\ [\mathbf{3}, \mathbf{4}] \end{cases}$ $\quad x_{21} \mapsto \begin{cases} (x_{00} + x_{01} + 4) - 1 \cdot (0.5 \cdot x_{00} + 0.5 \cdot x_{01} + 3) \\ [\mathbf{1}, \mathbf{2}] \end{cases}$

Interval Abstraction

with Symbolic Constant Propagation [Li19]



DeepPoly [Singh19]

maintain symbolic lower- and upper-bounds for each neuron + convex ReLU approximations

$$x_{i+1,j} \mapsto \begin{cases} [\sum_k c_{i,k} \cdot x_{i,k} + c,\ \sum_k d_{i,k} \cdot x_{i,k} + d] & c_{i,k}, c, d_{i,k}, d \in \mathscr{R} \\ [a,\ b] & a, b \in \mathscr{R} \end{cases}$$

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

ReLU $0 \leq a$:

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

ReLU $a < 0 \wedge 0 < b$

$b \leq -a$:

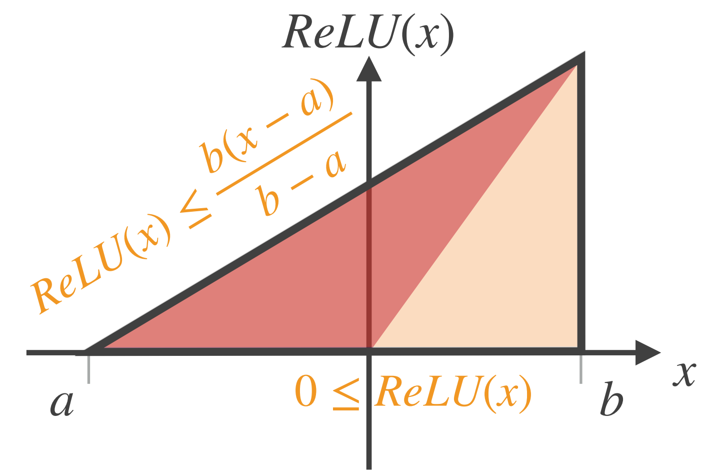
$$x_{i,j} \mapsto \begin{cases} \left[\mathbf{0}, \dfrac{\mathbf{b(x_{i,j} - a)}}{\mathbf{b - a}}\right] \\ [\mathbf{0}, \mathbf{b}] \end{cases}$$

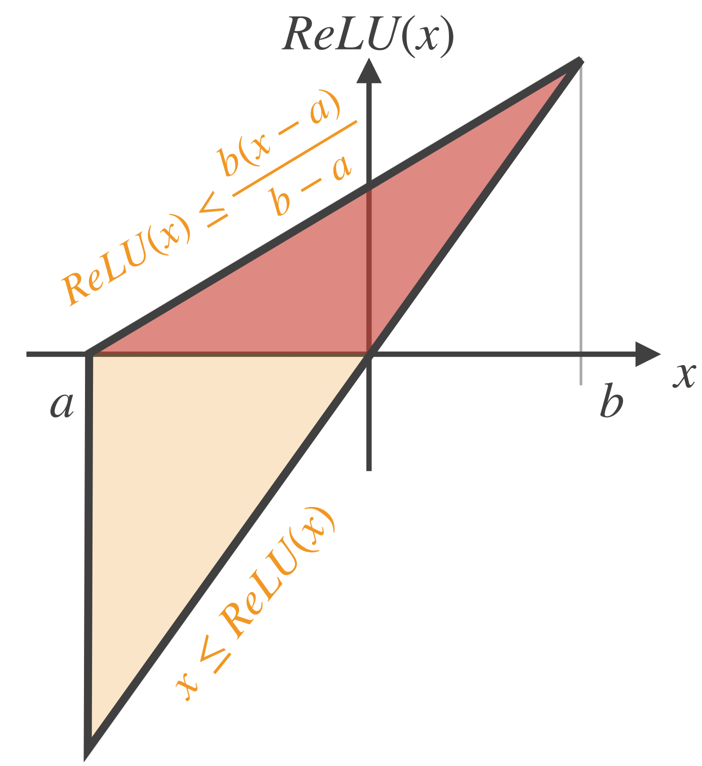
$-a < b$:

$$x_{i,j} \mapsto \begin{cases} \left[\mathbf{x_{i,j}}, \dfrac{\mathbf{b(x_{i,j} - a)}}{\mathbf{b - a}}\right] \\ [\mathbf{a}, \mathbf{b}] \end{cases}$$

ReLU $b \leq 0$:

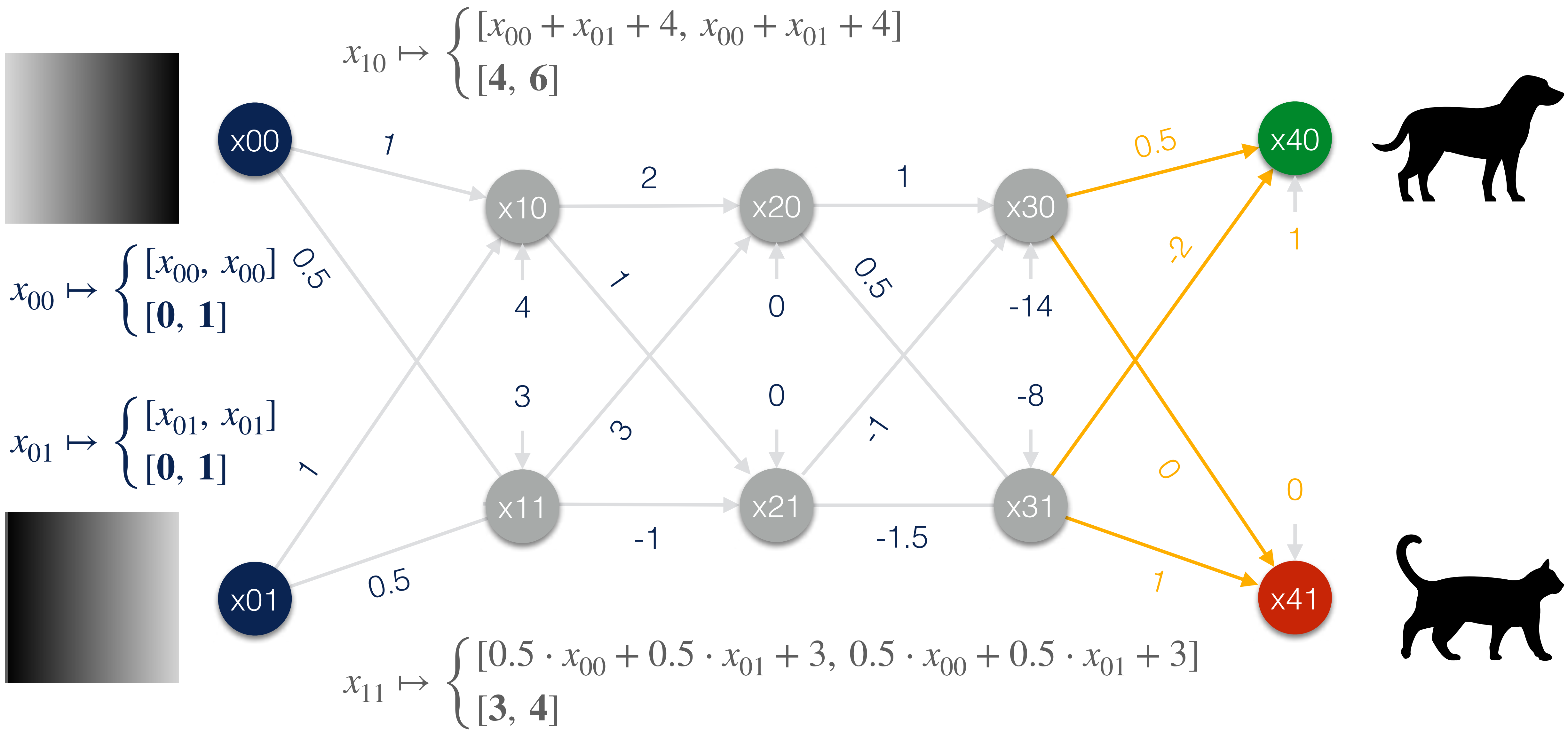
$$x_{i,j} \mapsto \begin{cases} [\mathbf{0}, \mathbf{0}] \\ [\mathbf{0}, \mathbf{0}] \end{cases}$$



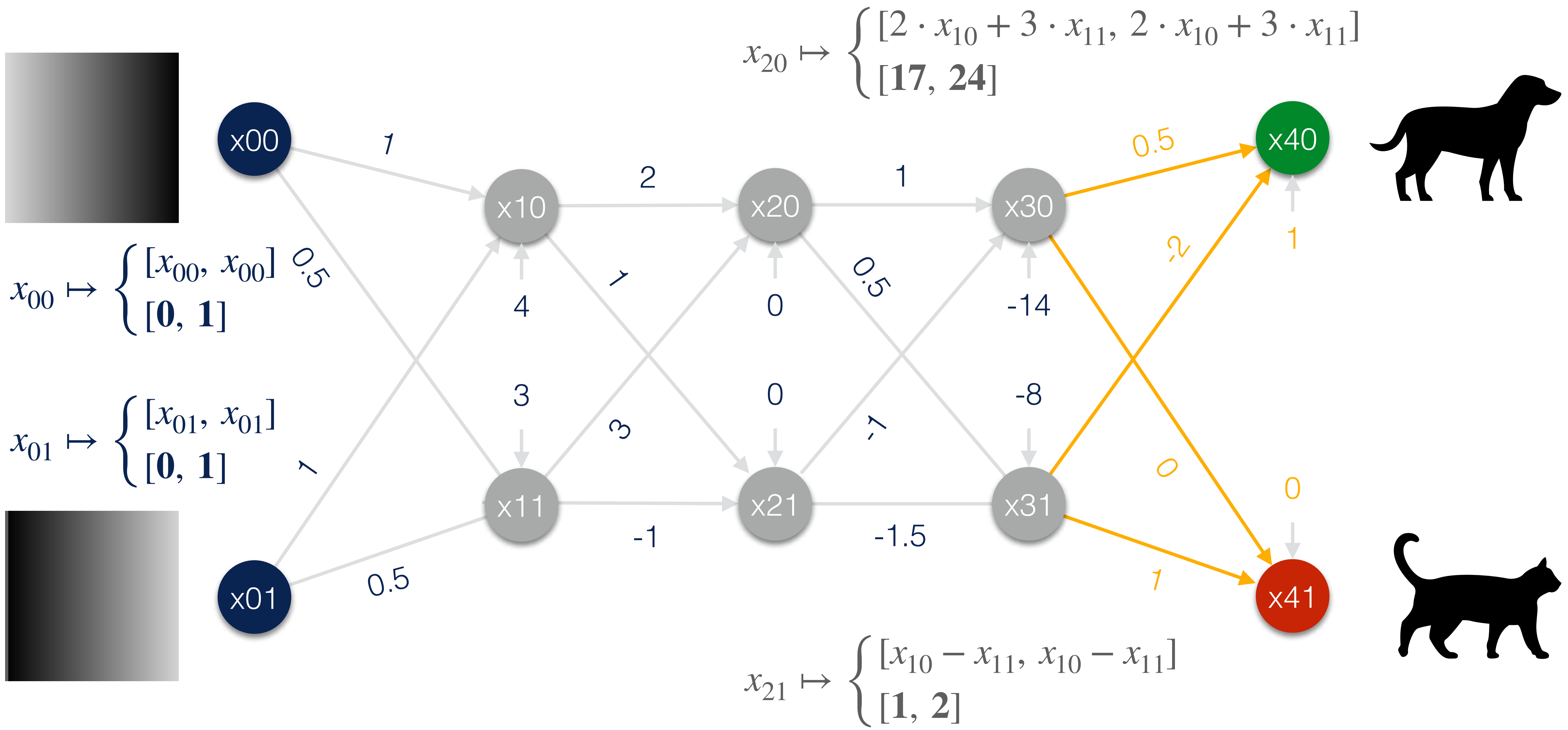


G. Singh, T. Gehr, M. Püschel, and M. Vechev - An Abstract Domain for Certifying Neural Networks (POPL 2019)

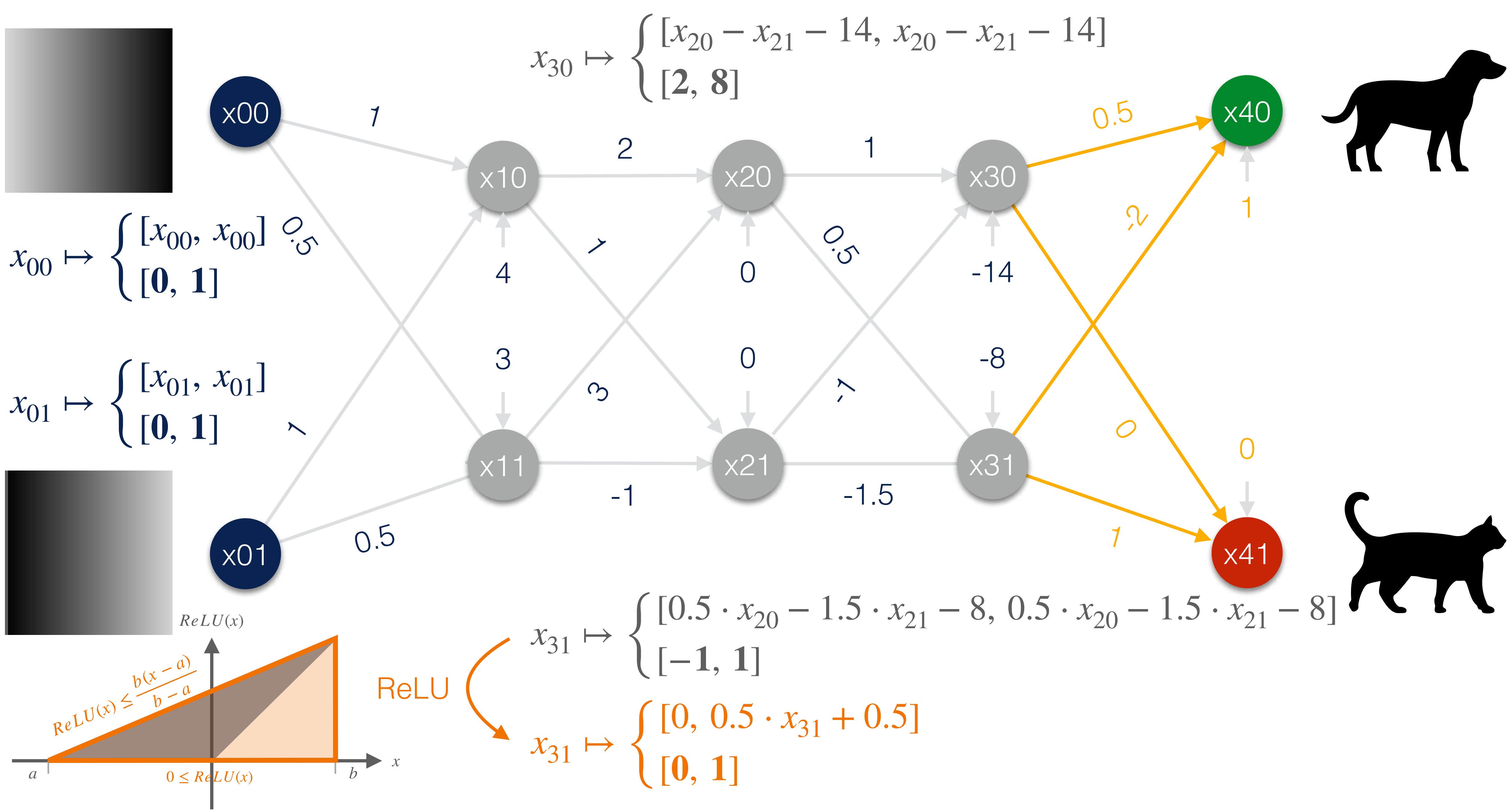
DeepPoly [Singh19]



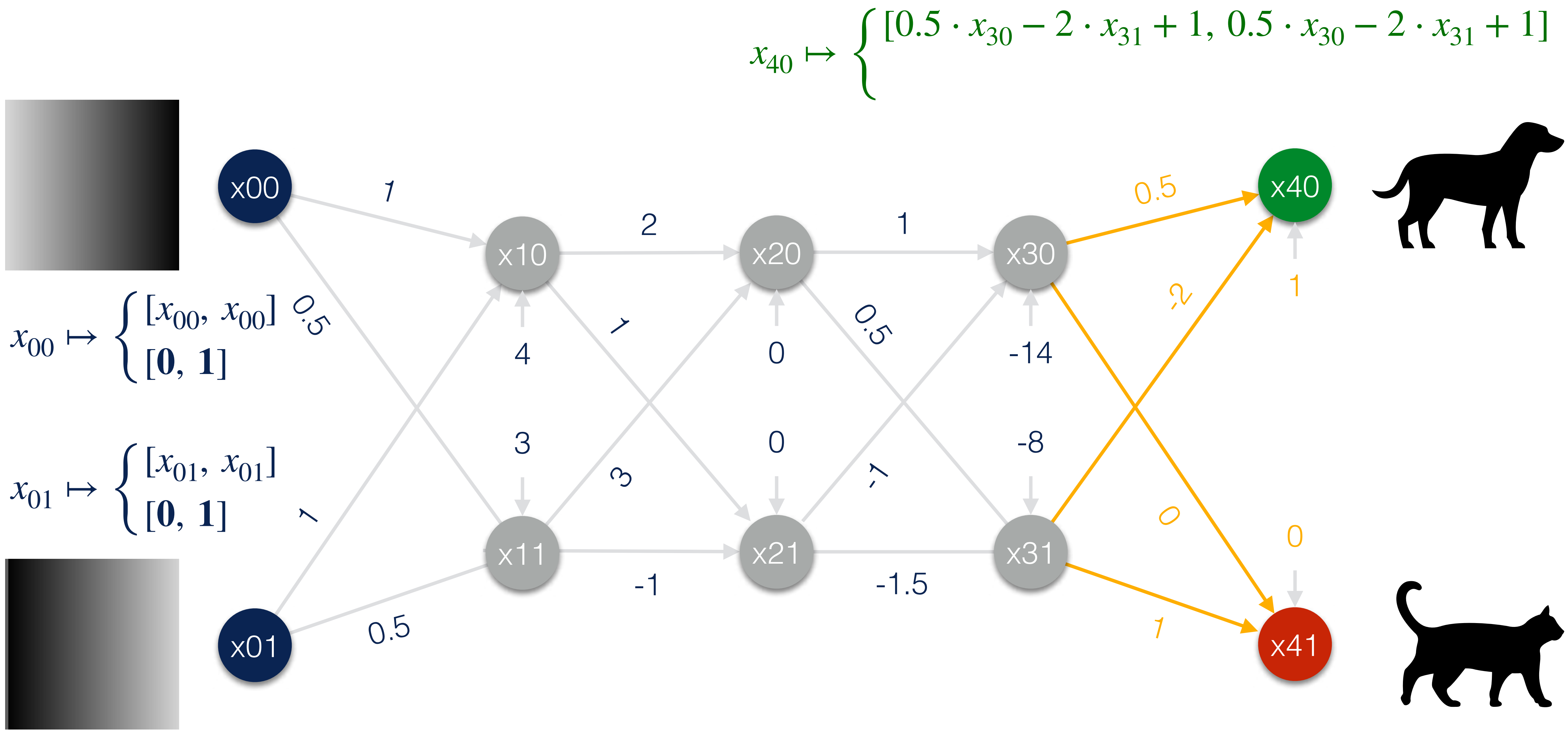
DeepPoly [Singh19]



DeepPoly [Singh19]



DeepPoly [Singh19]



DeepPoly [Singh19]

$$x_{00} \mapsto \begin{cases} [x_{00},\ x_{00}] \\ [\mathbf{0},\ \mathbf{1}] \end{cases} \qquad x_{01} \mapsto \begin{cases} [x_{01},\ x_{01}] \\ [\mathbf{0},\ \mathbf{1}] \end{cases}$$

$$x_{10} \mapsto \begin{cases} [x_{00} + x_{01} + 4,\ x_{00} + x_{01} + 4] \\ [\mathbf{4},\ \mathbf{6}] \end{cases} \qquad x_{11} \mapsto \begin{cases} [0.5 \cdot x_{00} + 0.5 \cdot x_{01} + 3,\ 0.5 \cdot x_{00} + 0.5 \cdot x_{01} + 3] \\ [\mathbf{3},\ \mathbf{4}] \end{cases}$$

$$x_{20} \mapsto \begin{cases} [2 \cdot x_{10} + 3 \cdot x_{11},\ 2 \cdot x_{10} + 3 \cdot x_{11}] \\ [\mathbf{17},\ \mathbf{24}] \end{cases} \qquad x_{21} \mapsto \begin{cases} [x_{10} - x_{11},\ x_{10} - x_{11}] \\ [\mathbf{1},\ \mathbf{2}] \end{cases}$$

$$x_{30} \mapsto \begin{cases} [x_{20} - x_{21} - 14,\ x_{20} - x_{21} - 14] \\ [\mathbf{2},\ \mathbf{8}] \end{cases} \qquad x_{31} \mapsto \begin{cases} [0,\ 0.5 \cdot (0.5 \cdot x_{20} - 1.5 \cdot x_{21} - 8) + 0.5] \\ [\mathbf{0},\ \mathbf{1}] \end{cases}$$

$$x_{40} \mapsto \begin{cases} [0.5 \cdot x_{30} - 2 \cdot x_{31} + 1,\ 0.5 \cdot x_{30} - 2 \cdot x_{31} + 1] \\ \end{cases}$$

DeepPoly [Singh19]

$$x_{00} \mapsto \begin{cases} [x_{00},\ x_{00}] \\ [\mathbf{0},\ \mathbf{1}] \end{cases} \qquad x_{01} \mapsto \begin{cases} [x_{01},\ x_{01}] \\ [\mathbf{0},\ \mathbf{1}] \end{cases}$$

$$x_{10} \mapsto \begin{cases} [x_{00} + x_{01} + 4,\ x_{00} + x_{01} + 4] \\ [\mathbf{4},\ \mathbf{6}] \end{cases} \qquad x_{11} \mapsto \begin{cases} [0.5 \cdot x_{00} + 0.5 \cdot x_{01} + 3,\ 0.5 \cdot x_{00} + 0.5 \cdot x_{01} + 3] \\ [\mathbf{3},\ \mathbf{4}] \end{cases}$$

$$x_{20} \mapsto \begin{cases} [2 \cdot x_{10} + 3 \cdot x_{11},\ 2 \cdot x_{10} + 3 \cdot x_{11}] \\ [\mathbf{17},\ \mathbf{24}] \end{cases} \qquad x_{21} \mapsto \begin{cases} [x_{10} - x_{11},\ x_{10} - x_{11}] \\ [\mathbf{1},\ \mathbf{2}] \end{cases}$$

$$x_{30} \mapsto \begin{cases} [x_{20} - x_{21} - 14,\ x_{20} - x_{21} - 14] \\ [\mathbf{2},\ \mathbf{8}] \end{cases} \qquad x_{31} \mapsto \begin{cases} [0,\ 0.5 \cdot (0.5 \cdot x_{20} - 1.5 \cdot x_{21} - 8) + 0.5] \\ [\mathbf{0},\ \mathbf{1}] \end{cases}$$

$$x_{40} \mapsto \begin{cases} [0.5 \cdot x_{30} - 2 \cdot x_{31} + 1,\ 0.5 \cdot x_{30} - 2 \cdot x_{31} + 1] \\ \end{cases}$$

$$\mapsto \begin{cases} [x_{21} + 1,\ 0.5 \cdot x_{20} - 0.5 \cdot x_{21} - 6] \\ \end{cases}$$

DeepPoly [Singh19]

$$x_{00} \mapsto \begin{cases} [x_{00}, x_{00}] \\ [\mathbf{0}, \mathbf{1}] \end{cases} \qquad x_{01} \mapsto \begin{cases} [x_{01}, x_{01}] \\ [\mathbf{0}, \mathbf{1}] \end{cases}$$

$$x_{10} \mapsto \begin{cases} [x_{00} + x_{01} + 4, x_{00} + x_{01} + 4] \\ [\mathbf{4}, \mathbf{6}] \end{cases} \qquad x_{11} \mapsto \begin{cases} [0.5 \cdot x_{00} + 0.5 \cdot x_{01} + 3, 0.5 \cdot x_{00} + 0.5 \cdot x_{01} + 3] \\ [\mathbf{3}, \mathbf{4}] \end{cases}$$

$$x_{20} \mapsto \begin{cases} [2 \cdot x_{10} + 3 \cdot x_{11}, 2 \cdot x_{10} + 3 \cdot x_{11}] \\ [\mathbf{17}, \mathbf{24}] \end{cases} \qquad x_{21} \mapsto \begin{cases} [x_{10} - x_{11}, x_{10} - x_{11}] \\ [\mathbf{1}, \mathbf{2}] \end{cases}$$

$$x_{30} \mapsto \begin{cases} [x_{20} + x_{21} - 14, x_{20} - x_{21} - 14] \\ [\mathbf{2}, \mathbf{8}] \end{cases} \qquad x_{31} \mapsto \begin{cases} [0, 0.5 \cdot (0.5 \cdot x_{20} - 1.5 \cdot x_{21} - 8) + 0.5] \\ [\mathbf{0}, \mathbf{1}] \end{cases}$$

$$x_{40} \mapsto \begin{cases} [0.5 \cdot x_{30} - 2 \cdot x_{31} + 1, 0.5 \cdot x_{30} - 2 \cdot x_{31} + 1] \\ \end{cases}$$

$$\mapsto \begin{cases} [x_{21} + 1, 0.5 \cdot x_{20} - 0.5 \cdot x_{21} - 6] \\ \end{cases}$$

$$\mapsto \begin{cases} [x_{10} - x_{11} + 1, 0.5 \cdot x_{10} + 2 \cdot x_{11} - 6] \\ \end{cases}$$

DeepPoly [Singh19]

$$x_{00} \mapsto \begin{cases} [x_{00},\ x_{00}] \\ [\mathbf{0},\ \mathbf{1}] \end{cases} \qquad x_{01} \mapsto \begin{cases} [x_{01},\ x_{01}] \\ [\mathbf{0},\ \mathbf{1}] \end{cases}$$

$$x_{10} \mapsto \begin{cases} [x_{00} + x_{01} + 4,\ x_{00} + x_{01} + 4] \\ [\mathbf{4},\ \mathbf{6}] \end{cases} \qquad x_{11} \mapsto \begin{cases} [0.5 \cdot x_{00} + 0.5 \cdot x_{01} + 3,\ 0.5 \cdot x_{00} + 0.5 \cdot x_{01} + 3] \\ [\mathbf{3},\ \mathbf{4}] \end{cases}$$

$$x_{20} \mapsto \begin{cases} [2 \cdot x_{10} + 3 \cdot x_{11},\ 2 \cdot x_{10} + 3 \cdot x_{11}] \\ [\mathbf{17},\ \mathbf{24}] \end{cases} \qquad x_{21} \mapsto \begin{cases} [x_{10} - x_{11},\ x_{10} - x_{11}] \\ [\mathbf{1},\ \mathbf{2}] \end{cases}$$

$$x_{30} \mapsto \begin{cases} [x_{20} - x_{21} - 14,\ x_{20} - x_{21} - 14] \\ [\mathbf{2},\ \mathbf{8}] \end{cases} \qquad x_{31} \mapsto \begin{cases} [0,\ 0.5 \cdot (0.5 \cdot x_{20} - 1.5 \cdot x_{21} - 8) + 0.5] \\ [\mathbf{0},\ \mathbf{1}] \end{cases}$$

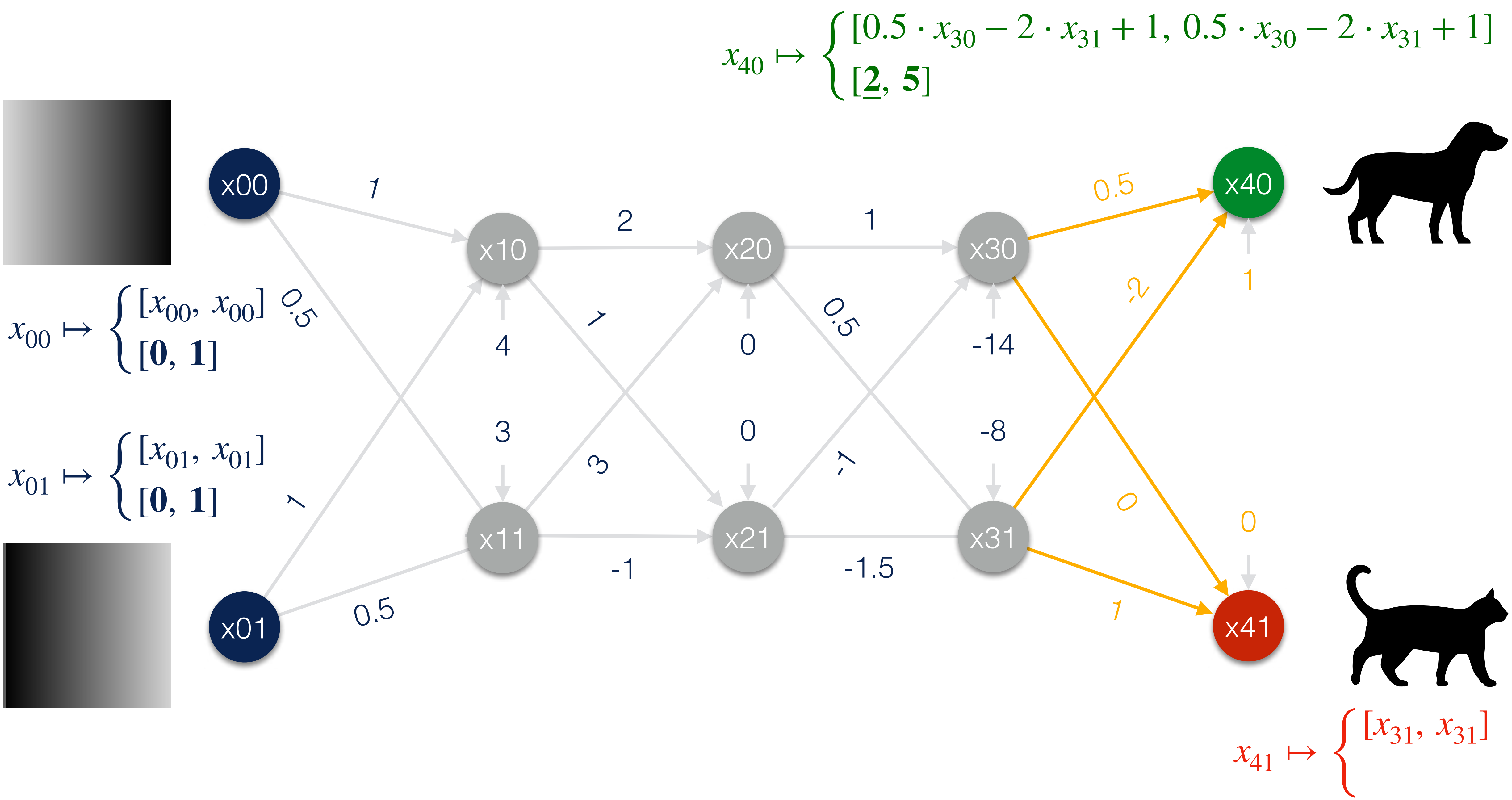
$$x_{40} \mapsto \begin{cases} [0.5 \cdot x_{30} - 2 \cdot x_{31} + 1,\ 0.5 \cdot x_{30} - 2 \cdot x_{31} + 1] \\ \end{cases}$$

$$\mapsto \begin{cases} [x_{21} + 1,\ 0.5 \cdot x_{20} - 0.5 \cdot x_{21} - 6] \\ \end{cases}$$

$$\mapsto \begin{cases} [x_{10} - x_{11} + 1,\ 0.5 \cdot x_{10} + 2 \cdot x_{11} - 6] \\ \end{cases}$$

$$\mapsto \begin{cases} [0.5 \cdot x_{00} + 0.5 \cdot x_{01} + 2,\ 1.5 \cdot x_{00} + 1.5 \cdot x_{11} + 2] \\ [\underline{\mathbf{2}},\ \mathbf{5}] \end{cases}$$

DeepPoly [Singh19]



DeepPoly [Singh19]

$x_{00} \mapsto \begin{cases} [x_{00},\ x_{00}] \\ [\mathbf{0},\ \mathbf{1}] \end{cases}$

$x_{01} \mapsto \begin{cases} [x_{01},\ x_{01}] \\ [\mathbf{0},\ \mathbf{1}] \end{cases}$

$x_{10} \mapsto \begin{cases} [x_{00} + x_{01} + 4,\ x_{00} + x_{01} + 4] \\ [\mathbf{4},\ \mathbf{6}] \end{cases}$

$x_{11} \mapsto \begin{cases} [0.5 \cdot x_{00} + 0.5 \cdot x_{01} + 3,\ 0.5 \cdot x_{00} + 0.5 \cdot x_{01} + 3] \\ [\mathbf{3},\ \mathbf{4}] \end{cases}$

$x_{20} \mapsto \begin{cases} [2 \cdot x_{10} + 3 \cdot x_{11},\ 2 \cdot x_{10} + 3 \cdot x_{11}] \\ [\mathbf{17},\ \mathbf{24}] \end{cases}$

$x_{21} \mapsto \begin{cases} [x_{10} - x_{11},\ x_{10} - x_{11}] \\ [\mathbf{1},\ \mathbf{2}] \end{cases}$

$x_{30} \mapsto \begin{cases} [x_{20} - x_{21} - 14,\ x_{20} - x_{21} - 14] \\ [\mathbf{2},\ \mathbf{8}] \end{cases}$

$x_{31} \mapsto \begin{cases} [0,\ 0.5 \cdot (0.5 \cdot x_{20} - 1.5 \cdot x_{21} - 8) + 0.5] \\ [\mathbf{0},\ \mathbf{1}] \end{cases}$

$x_{41} \mapsto \begin{cases} [x_{31},\ x_{31}] \\ \end{cases}$

DeepPoly [Singh19]

$$x_{00} \mapsto \begin{cases} [x_{00},\ x_{00}] \\ [\mathbf{0},\ \mathbf{1}] \end{cases} \qquad x_{01} \mapsto \begin{cases} [x_{01},\ x_{01}] \\ [\mathbf{0},\ \mathbf{1}] \end{cases}$$

$$x_{10} \mapsto \begin{cases} [x_{00} + x_{01} + 4,\ x_{00} + x_{01} + 4] \\ [\mathbf{4},\ \mathbf{6}] \end{cases} \qquad x_{11} \mapsto \begin{cases} [0.5 \cdot x_{00} + 0.5 \cdot x_{01} + 3,\ 0.5 \cdot x_{00} + 0.5 \cdot x_{01} + 3] \\ [\mathbf{3},\ \mathbf{4}] \end{cases}$$

$$x_{20} \mapsto \begin{cases} [2 \cdot x_{10} + 3 \cdot x_{11},\ 2 \cdot x_{10} + 3 \cdot x_{11}] \\ [\mathbf{17},\ \mathbf{24}] \end{cases} \qquad x_{21} \mapsto \begin{cases} [x_{10} - x_{11},\ x_{10} - x_{11}] \\ [\mathbf{1},\ \mathbf{2}] \end{cases}$$

$$x_{30} \mapsto \begin{cases} [x_{20} - x_{21} - 14,\ x_{20} - x_{21} - 14] \\ [\mathbf{2},\ \mathbf{8}] \end{cases} \qquad x_{31} \mapsto \begin{cases} [0,\ 0.5 \cdot (0.5 \cdot x_{20} - 1.5 \cdot x_{21} - 8) + 0.5] \\ [\mathbf{0},\ \mathbf{1}] \end{cases}$$

$$x_{41} \mapsto \begin{cases} [x_{31},\ x_{31}] \\ \end{cases}$$

$$\mapsto \begin{cases} [0,\ 0.25 \cdot x_{20} - 0.75 \cdot x_{21} - 3.5] \\ \end{cases}$$

DeepPoly [Singh19]

$$x_{00} \mapsto \begin{cases} [x_{00},\ x_{00}] \\ [\mathbf{0},\ \mathbf{1}] \end{cases} \qquad x_{01} \mapsto \begin{cases} [x_{01},\ x_{01}] \\ [\mathbf{0},\ \mathbf{1}] \end{cases}$$

$$x_{10} \mapsto \begin{cases} [x_{00} + x_{01} + 4,\ x_{00} + x_{01} + 4] \\ [\mathbf{4},\ \mathbf{6}] \end{cases} \qquad x_{11} \mapsto \begin{cases} [0.5 \cdot x_{00} + 0.5 \cdot x_{01} + 3,\ 0.5 \cdot x_{00} + 0.5 \cdot x_{01} + 3] \\ [\mathbf{3},\ \mathbf{4}] \end{cases}$$

$$x_{20} \mapsto \begin{cases} [2 \cdot x_{10} + 3 \cdot x_{11},\ 2 \cdot x_{10} + 3 \cdot x_{11}] \\ [\mathbf{17},\ \mathbf{24}] \end{cases} \qquad x_{21} \mapsto \begin{cases} [x_{10} - x_{11},\ x_{10} - x_{11}] \\ [\mathbf{1},\ \mathbf{2}] \end{cases}$$

$$x_{30} \mapsto \begin{cases} [x_{20} - x_{21} - 14,\ x_{20} - x_{21} - 14] \\ [\mathbf{2},\ \mathbf{8}] \end{cases} \qquad x_{31} \mapsto \begin{cases} [0,\ 0.5 \cdot (0.5 \cdot x_{20} - 1.5 \cdot x_{21} - 8) + 0.5] \\ [\mathbf{0},\ \mathbf{1}] \end{cases}$$

$$x_{41} \mapsto \begin{cases} [x_{31},\ x_{31}] \\ \end{cases}$$

$$\mapsto \begin{cases} [0,\ 0.25 \cdot x_{20} - 0.75 \cdot x_{21} - 3.5] \\ \end{cases}$$

$$\mapsto \begin{cases} [0,\ -0.25 \cdot x_{10} + 1.5 \cdot x_{11} - 3.5] \\ \end{cases}$$

DeepPoly [Singh19]

$$x_{00} \mapsto \begin{cases} [x_{00},\ x_{00}] \\ [\mathbf{0},\ \mathbf{1}] \end{cases} \qquad x_{01} \mapsto \begin{cases} [x_{01},\ x_{01}] \\ [\mathbf{0},\ \mathbf{1}] \end{cases}$$

$$x_{10} \mapsto \begin{cases} [x_{00} + x_{01} + 4,\ x_{00} + x_{01} + 4] \\ [\mathbf{4},\ \mathbf{6}] \end{cases} \qquad x_{11} \mapsto \begin{cases} [0.5 \cdot x_{00} + 0.5 \cdot x_{01} + 3,\ 0.5 \cdot x_{00} + 0.5 \cdot x_{01} + 3] \\ [\mathbf{3},\ \mathbf{4}] \end{cases}$$

$$x_{20} \mapsto \begin{cases} [2 \cdot x_{10} + 3 \cdot x_{11},\ 2 \cdot x_{10} + 3 \cdot x_{11}] \\ [\mathbf{17},\ \mathbf{24}] \end{cases} \qquad x_{21} \mapsto \begin{cases} [x_{10} - x_{11},\ x_{10} - x_{11}] \\ [\mathbf{1},\ \mathbf{2}] \end{cases}$$

$$x_{30} \mapsto \begin{cases} [x_{20} - x_{21} - 14,\ x_{20} - x_{21} - 14] \\ [\mathbf{2},\ \mathbf{8}] \end{cases} \qquad x_{31} \mapsto \begin{cases} [0,\ 0.5 \cdot (0.5 \cdot x_{20} - 1.5 \cdot x_{21} - 8) + 0.5] \\ [\mathbf{0},\ \mathbf{1}] \end{cases}$$

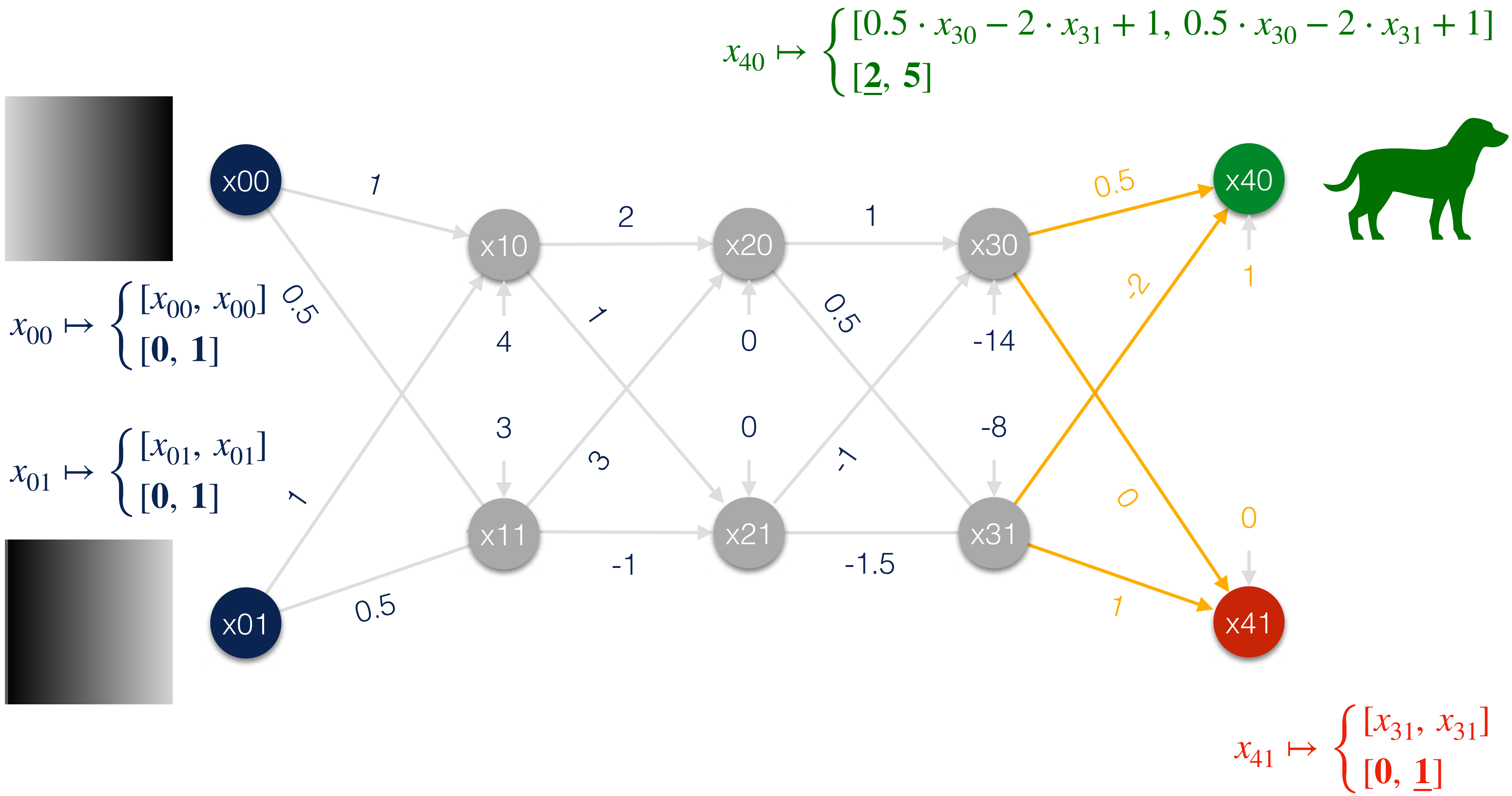
$$x_{41} \mapsto \begin{cases} [x_{31},\ x_{31}] \\ \end{cases}$$

$$\mapsto \begin{cases} [0,\ 0.25 \cdot x_{20} - 0.75 \cdot x_{21} - 3.5] \\ \end{cases}$$

$$\mapsto \begin{cases} [0,\ -0.25 \cdot x_{10} + 1.5 \cdot x_{11} - 3.5] \\ \end{cases}$$

$$\mapsto \begin{cases} [0,\ 0.5 \cdot x_{00} + 0.5 \cdot x_{01}] \\ [\mathbf{0},\ \underline{\mathbf{1}}] \end{cases}$$

DeepPoly [Singh19]



Other Static Analysis Methods

- **T. Gehr, M. Mirman, D. Drachsler-Cohen, P. Tsankov, S. Chaudhuri, and M. Vechev**. *AI2: Safety and Robustness Certification of Neural Networks with Abstract Interpretation*. In S&P, 2018.
  the first use of abstract interpretation for verifying neural networks

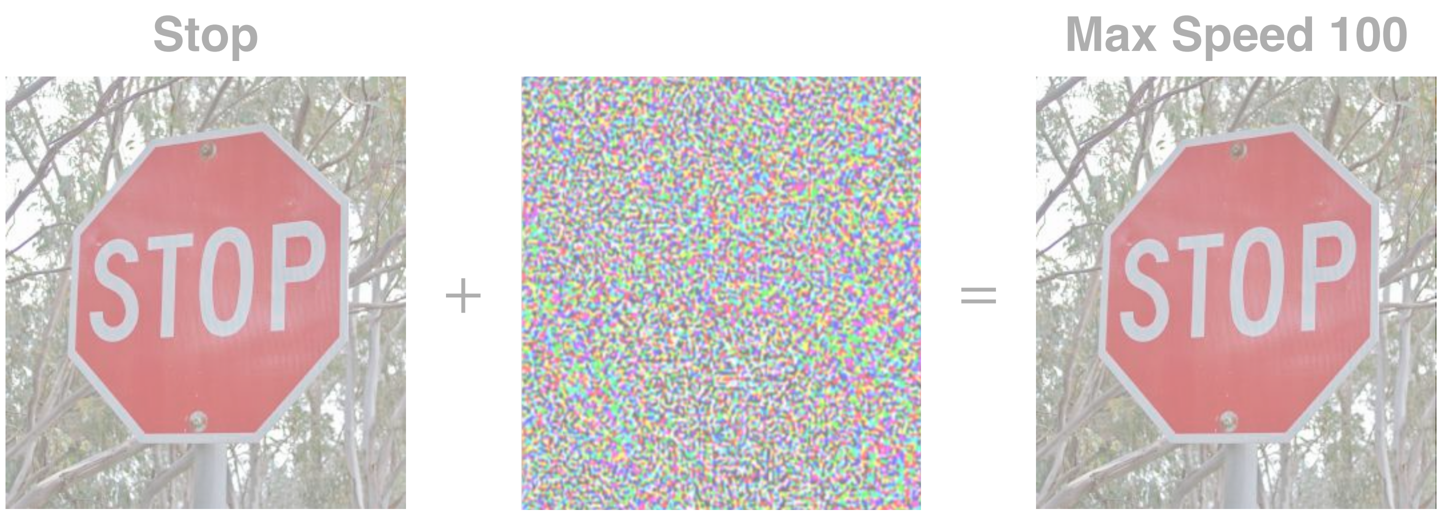
- **G. Singh, T. Gehr, M. Mirman, M. Püschel, and M. Vechev**. *Fast and Effective Robustness Certification*. In NeurIPS, 2018.
  a custom zonotope domain for certifying neural networks

- **G. Singh, R. Ganvir, M. Püschel, and M. Vechev**. *Beyond the Single Neuron Convex Barrier for Neural Network Certification*. In NeurIPS, 2019.
  a framework to jointly approximate k ReLU activations

- **M. N. Müller, G. Makarchuk, G. Singh, M. Püschel, and M. Vechev**. *PRIMA: General and Precise Neural Network Certification via Scalable Convex Hull Approximations*. In POPL, 2022.
  a multi-neuron abstraction via a convex-hull approximation algorithm

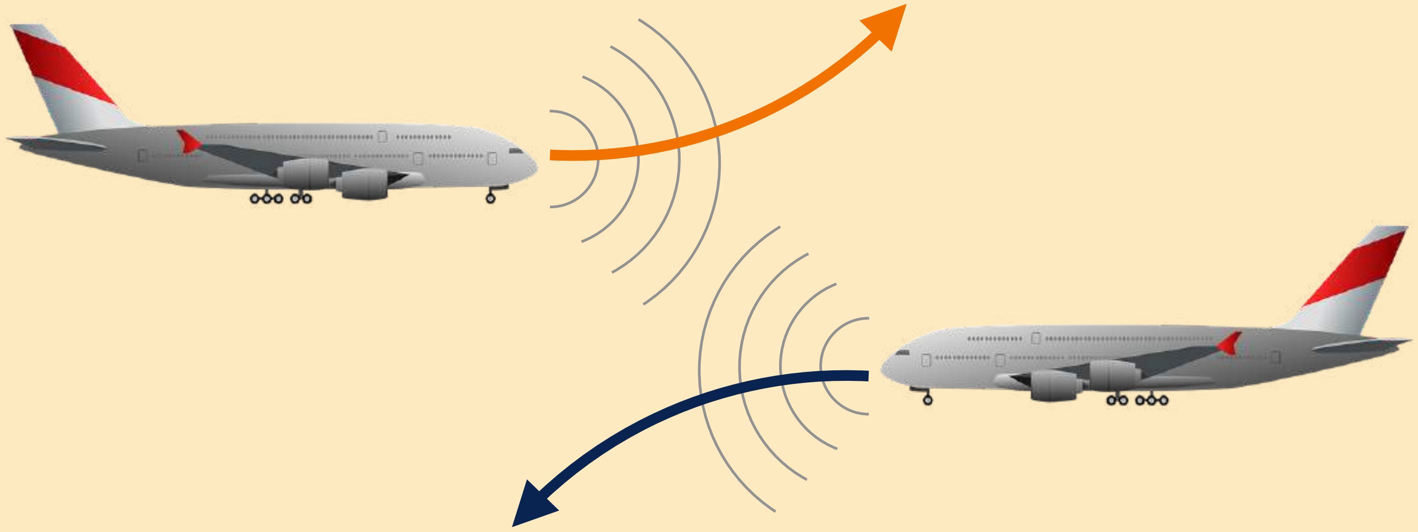
Stability

Goal G3 in [Kurd03]

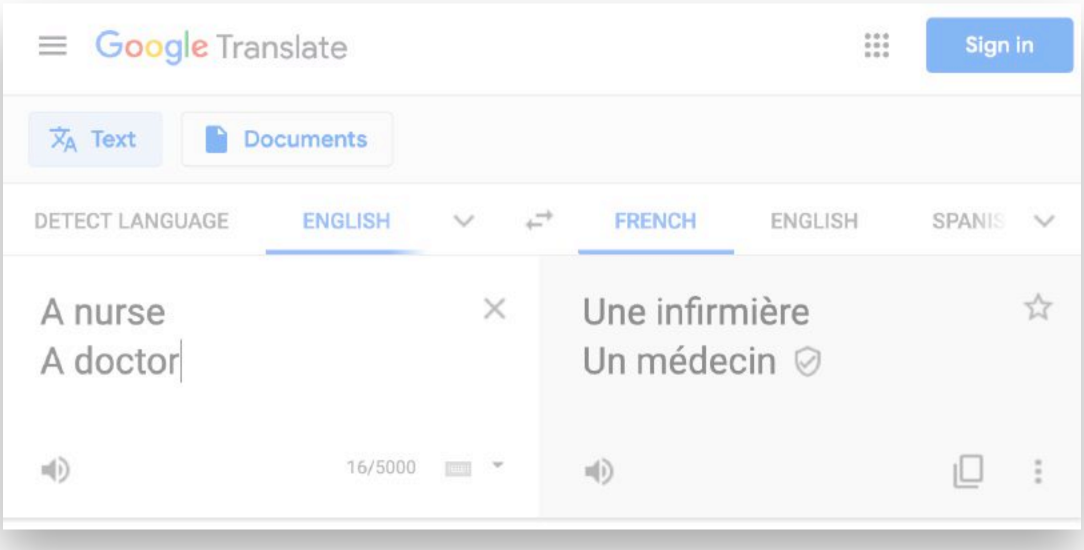


Safety

Goal G4 in [Kurd03]



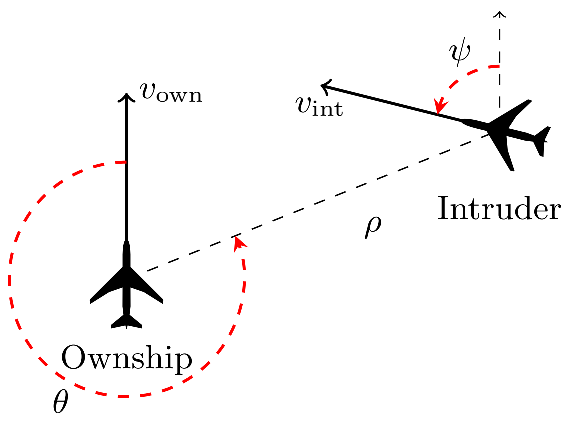
Fairness

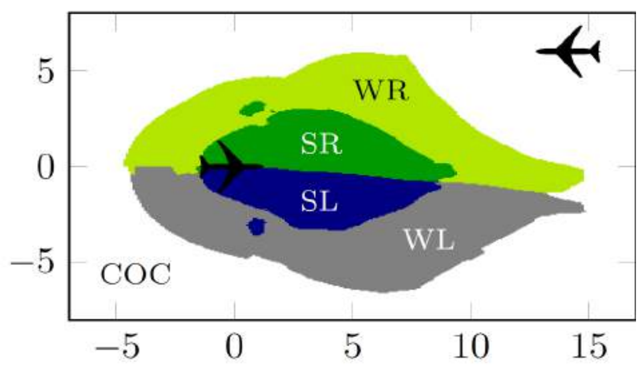


ACAS Xu [Julian16][Katz17]

Airborne Collision Avoidance System for Unmanned Aircraft

implemented using **45 feed-forward fully-connected ReLU networks**





5 input **sensor measurements**

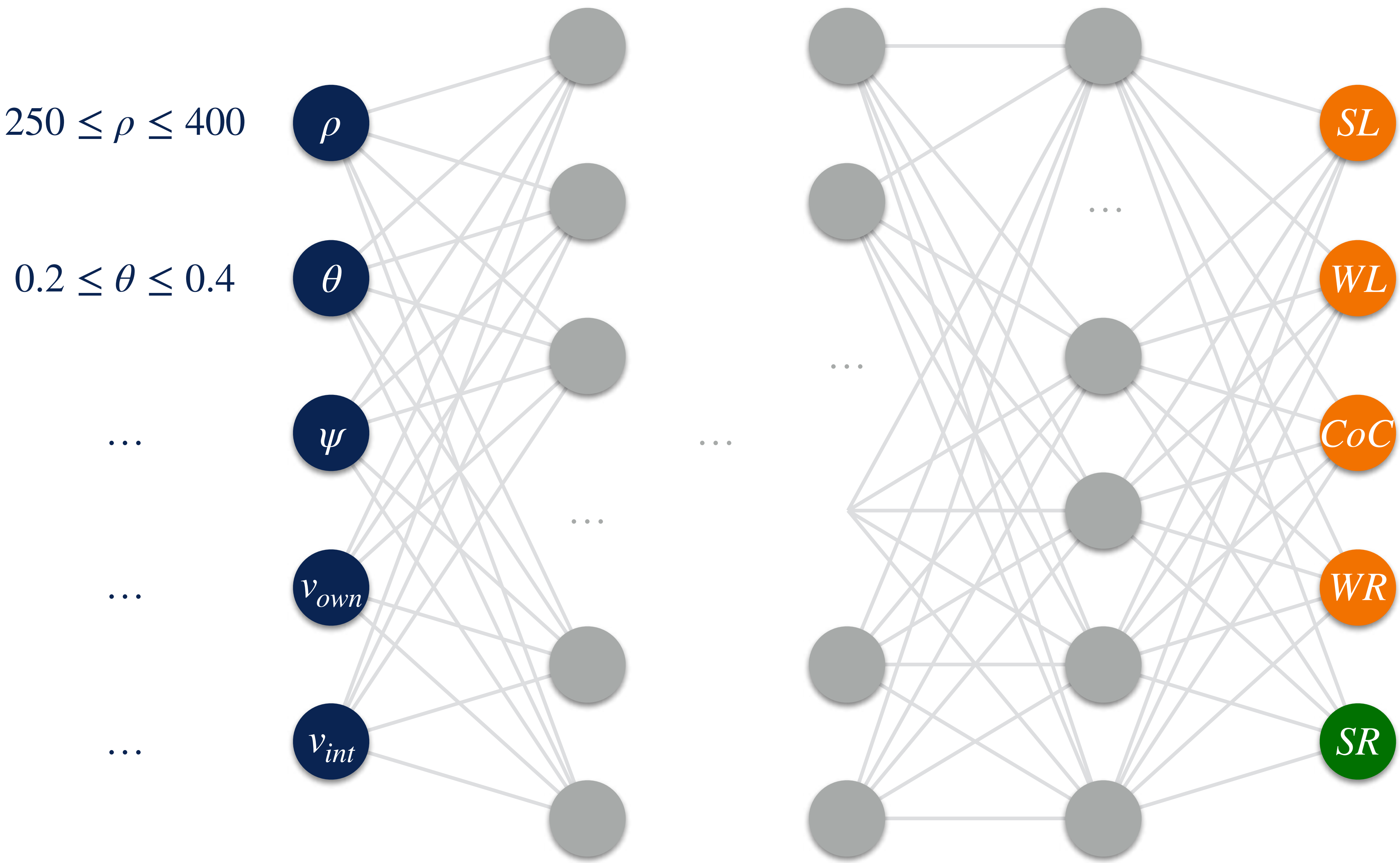
- ρ: distance from ownship to intruder
- θ: angle to intruder relative to ownship heading direction
- ψ: heading angle to intruder relative to ownship heading direction
- v_{own}: speed of ownship
- v_{int}: speed of intruder

5 output **horizontal advisories**

- Strong Left
- Weak Left
- Clear of Conflict
- Weak Right
- Strong Right

ACAS Xu Properties [Katz17]

Example: “if intruder is **near** and approaching **from the left**, go **Strong Right**”



Safety

Input-Output Properties

$\mathbf{I}$: input specification

$\mathbf{O}$: output specification

$$\mathcal{S}^{\mathbf{I}}_{\mathbf{O}} \stackrel{\text{def}}{=} \{[\![M]\!] \in \mathscr{P}(\Sigma^*) \mid \text{SAFE}^{\mathbf{I}}_{\mathbf{O}}([\![M]\!])\}$$

$\mathcal{S}^{\mathbf{I}}_{\mathbf{O}}$ is the set of all neural networks M (or, rather, their semantics $[\![M]\!]$) that **satisfy** the input and output specification $\mathbf{I}$ and $\mathbf{O}$

$$\text{SAFE}^{\mathbf{I}}_{\mathbf{O}}([\![M]\!]) \stackrel{\text{def}}{=} \forall t \in [\![M]\!] : t_0 \models \mathbf{I} \Rightarrow t_\omega \models \mathbf{O}$$

Theorem

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

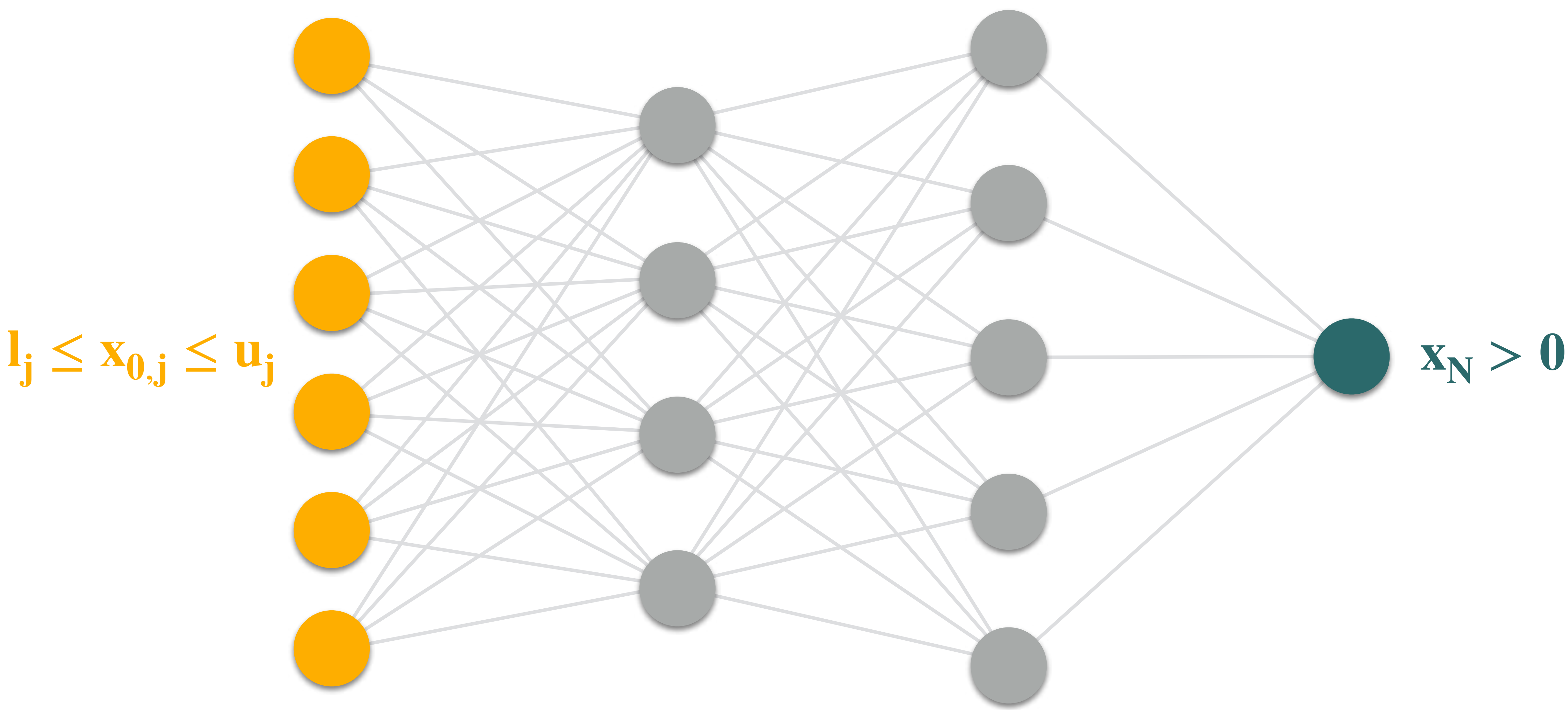
Corollary

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

Model Checking Methods

Safety

Example



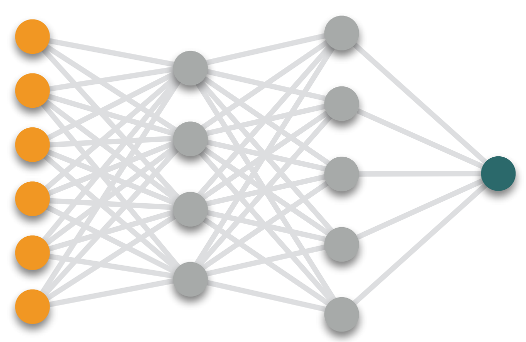
SMT-Based Methods

Verification Reduced to Constraint Satisfiability

$\mathbf{l_j \leq x_{0,j} \leq u_j}$ $\quad j \in \{0, \ldots, |\mathbf{X}_0|\}$ **input specification**

$$\hat{x}_{i+1,j} = \sum_{k=0}^{|\mathbf{X}_i|} w^i_{j,k} \cdot x_{i,k} + b_{i,j} \qquad i \in \{0, \ldots, n-1\}$$

$$x_{i,j} = \max\{0, \hat{x}_{i,j}\} \qquad i \in \{1, \ldots, n-1\}, \; j \in \{0, \ldots, |\mathbf{X}_i|\}$$



$\mathbf{x_N \leq 0}$

(negation of)
output specification

satisfiable → ✗ **counterexample**

otherwise → ✓ **safe**

Planet

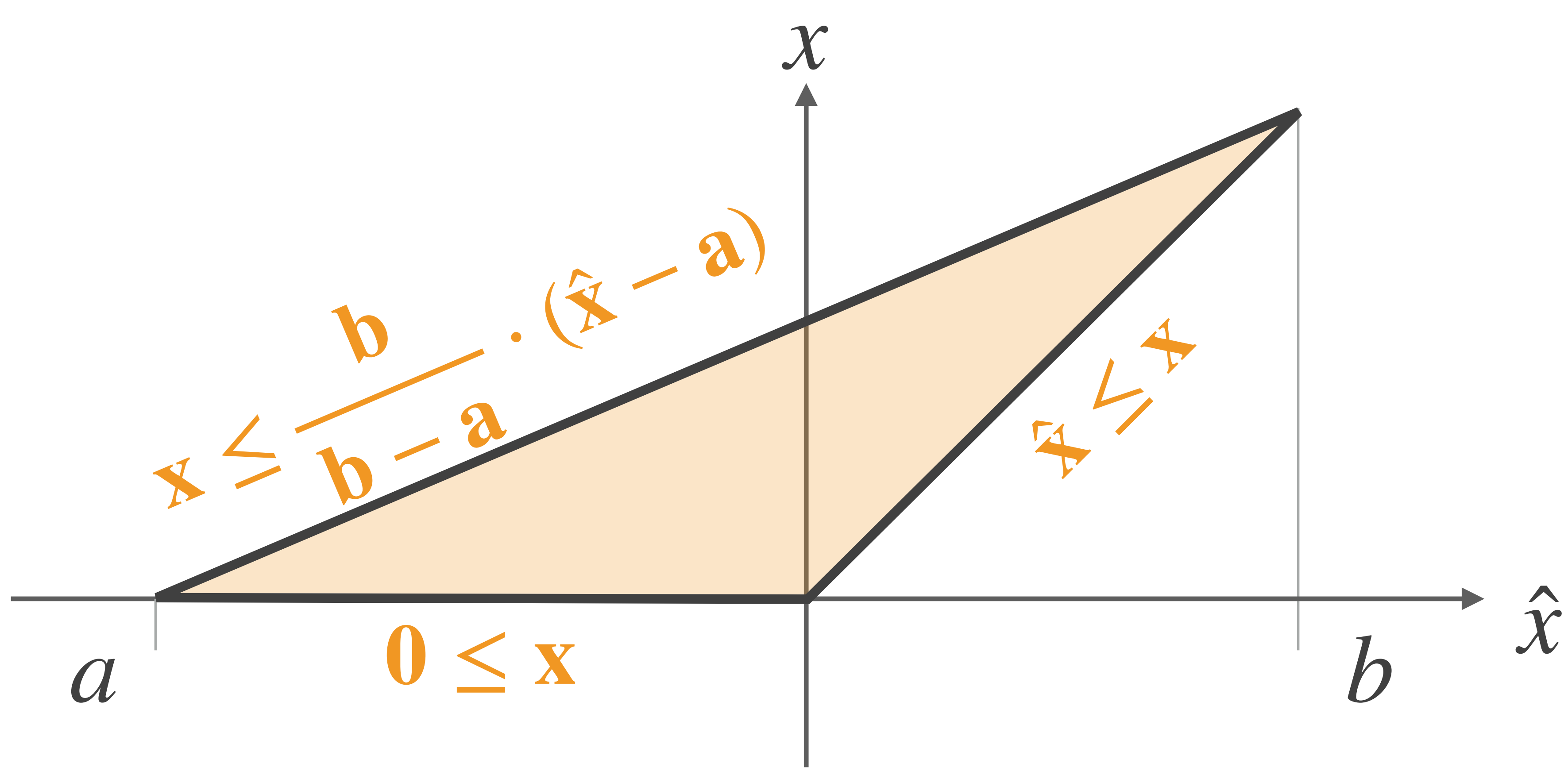
use **approximations** to reduce the solution search space

$$x_{i,j} = \max\{0, \hat{x}_{i,j}\}$$

$$\mathbf{0 \leq x_{i,j}}$$

$$\mathbf{\hat{x}_{i,j} \leq x_{i,j}}$$

$$\mathbf{x_{i,j} \leq \frac{b_{i,j}}{b_{i,j} - a_{i,j}} \cdot (\hat{x}_{i,j} - a_{i,j})}$$



R. Ehlers - Formal Verification of Piece-Wise Linear Feed-Forward Neural Networks (ATVA 2017)

Reluplex

based on the simplex algorithm extended to support ReLUs

Variable	Value
$\mathbf{x}_{00}$	v_{00}
$\dots$	$\dots$
$\hat{\mathbf{x}}_{\mathbf{ij}}$	$\hat{v}_{ij}$
$\mathbf{x}_{\mathbf{ij}}$	v_{ij}
$\dots$	$\dots$
$\mathbf{x}_{\mathbf{N}}$	v_N

Variable	Value
$\mathbf{x}_{00}$	v_{00}
$\dots$	$\dots$
$\hat{\mathbf{x}}_{\mathbf{ij}}$	$\hat{v}'_{ij}$
$\mathbf{x}_{\mathbf{ij}}$	v_{ij}
$\dots$	$\dots$
$\mathbf{x}_{\mathbf{N}}$	v_N

Variable	Value
$\mathbf{x}_{00}$	v_{00}
$\dots$	$\dots$
$\hat{\mathbf{x}}_{\mathbf{ij}}$	$\hat{v}'_{ij}$
$\mathbf{x}_{\mathbf{ij}}$	$\hat{v}'_{ij}$
$\dots$	$\dots$
$\mathbf{x}_{\mathbf{N}}$	v_N

Variable	Value
$\mathbf{x}_{00}$	v_{00}
$\dots$	$\dots$
$\hat{\mathbf{x}}_{\mathbf{ij}}$	$\hat{v}'_{ij}$
$\mathbf{x}_{\mathbf{ij}}$	0
$\dots$	$\dots$
$\mathbf{x}_{\mathbf{N}}$	v_N

G. Katz et al. - Reluplex: An Efficient SMT Solver for Verifying Deep Neural Networks (CAV 2017)

Reluplex

based on th
extended

Follow-up Work

G. Katz et al. - The Marabou Framework for Verification and Analysis of Deep Neural Networks (CAV 2019)

Variable	Value
$\mathbf{x}_{00}$	v_{00}
$\dots$	$\dots$
$\hat{\mathbf{x}}_{\mathbf{ij}}$	$\hat{v}_{ij}$
$\mathbf{x}_{\mathbf{ij}}$	v_{ij}
$\dots$	$\dots$
$\mathbf{x}_{\mathrm{N}}$	v_N

Variable	Value
$\mathbf{x}_{00}$	v_{00}
$\dots$	$\dots$
$\hat{\mathbf{x}}_{\mathbf{ij}}$	$\hat{v}'_{ij}$
$\mathbf{x}_{\mathbf{ij}}$	v_{ij}
$\dots$	$\dots$
$\mathbf{x}_{\mathrm{N}}$	v_N

Variable	
$\mathbf{x}_{00}$	v_{00}
$\dots$	$\dots$
$\hat{\mathbf{x}}_{\mathbf{ij}}$	$\hat{v}'_{ij}$
$\mathbf{x}_{\mathbf{ij}}$	$\hat{v}'_{ij}$
$\dots$	$\dots$
$\mathbf{x}_{\mathrm{N}}$	v_N

Variable	Value
$\mathbf{x}_{00}$	v_{00}
$\dots$	$\dots$
$\hat{\mathbf{x}}_{\mathbf{ij}}$	$\hat{v}'_{ij}$
$\mathbf{x}_{\mathbf{ij}}$	0
$\dots$	$\dots$
$\mathbf{x}_{\mathrm{N}}$	v_N

G. Katz et al. - Reluplex: An Efficient SMT Solver for Verifying Deep Neural Networks (CAV 2017)

Other SMT-Based Methods

- **L. Pulina and A. Tacchella**. *An Abstraction-Refinement Approach to Verification of Artificial Neural Networks*. In CAV, 2010.
  the first formal verification method for neural networks

- **O. Bastani, Y. Ioannou, L. Lampropoulos, D. Vytiniotis, A. Nori, and A. Criminisi**. *Measuring Neural Net Robustness with Constraints*. In NeurIPS, 2016.
  an approach for finding the nearest adversarial example according to the L_∞ distance

- **X. Huang, M. Kwiatkowska, S. Wang, and M. Wu**. *Safety Verification of Deep Neural Networks*. In CAV, 2017.
  an approach for proving local robustness to adversarial perturbations

- **N. Narodytska, S. Kasiviswanathan, L. Ryzhyk, M. Sagiv, and T. Walsh**. *Verifying Properties of Binarized Deep Neural Networks*. In AAAI, 2018.
  C. H. Cheng, G. Nührenberg, C. H. Huang, and H. Ruess. *Verification of Binarized Neural Networks via Inter-Neuron Factoring*. In VSTTE, 2018.
  approaches focusing on binarized neural networks

MILP-Based Methods

Verification Reduced to Mixed Integer Linear Program

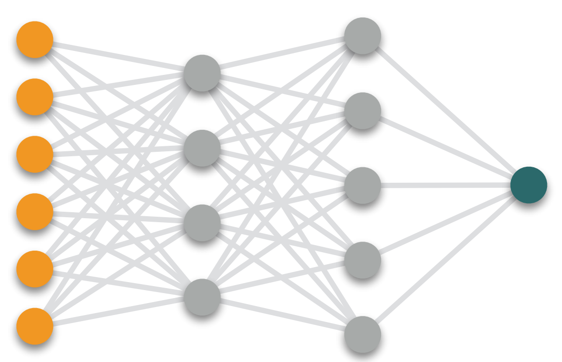
$\mathbf{l_j} \leq \mathbf{x_{0,j}} \leq \mathbf{u_j}$ $\quad j \in \{0, \ldots, |\mathbf{X}_0|\}$ **input specification**

$$\hat{x}_{i+1,j} = \sum_{k=0}^{|\mathbf{X}_i|} w^i_{j,k} \cdot x_{i,k} + b_{i,j} \qquad i \in \{0, \ldots, n-1\}$$

$$x_{i,j} = \delta_{\mathbf{i,j}} \cdot \hat{x}_{i,j} \qquad \delta_{\mathbf{i,j}} \in \{\mathbf{0}, \mathbf{1}\}$$

$$\delta_{\mathbf{i,j}} = 1 \Rightarrow \hat{x}_{i,j} \geq 0 \qquad i \in \{1, \ldots, n-1\}$$

$$\delta_{\mathbf{i,j}} = 0 \Rightarrow \hat{x}_{i,j} < 0 \qquad j \in \{0, \ldots, |\mathbf{X}_i|\}$$



min $\mathbf{x_N}$ **objective function**

min $\mathbf{x_N} \leq \mathbf{0} \rightarrow$ ✗ **counterexample**

otherwise $\rightarrow$ ✓ **safe**

MILP-Based Methods

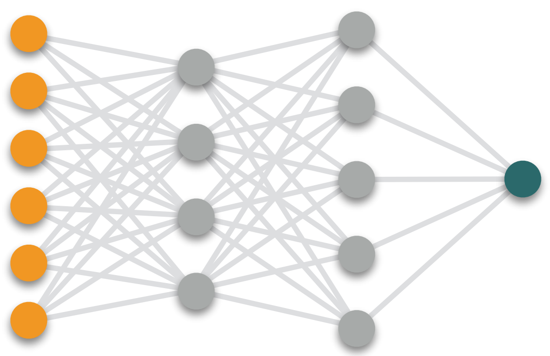
Bounded Encoding with Symmetric Bounds

$$\hat{x}_{i+1,j} = \sum_{k=0}^{|\mathbf{X}_i|} w^i_{j,k} \cdot x_{i,k} + b_{i,j} \qquad i \in \{0, \ldots, n-1\}$$

$$0 \leq x_{i,j} \leq \mathbf{M_{i,j}} \cdot \delta_{i,j} \qquad \delta_{\mathbf{i,j}} \in \{\mathbf{0}, \mathbf{1}\}$$

$$\hat{x}_{i,j} \leq x_{i,j} \leq \hat{x}_{i,j} - \mathbf{M_{i,j}} \cdot (1 - \delta_{i,j}) \qquad i \in \{1, \ldots, n-1\}$$

$$\mathbf{M_{i,j}} = \mathbf{max}\{-\mathbf{l_i}, \mathbf{u_i}\} \qquad j \in \{0, \ldots, |\mathbf{X}_i|\}$$



Sherlock

Output Range Analysis

use **local search** to speed up the MILP solver

$$\mathbf{l_j} \leq \mathbf{x_{0,j}} \leq \mathbf{u_j}$$

$$\hat{x}_{i+1,j} = \sum_{k=0}^{|\mathbf{X}_i|} w^i_{j,k} \cdot x_{i,k} + b_{i,j}$$

$$0 \leq x_{i,j} \leq \mathbf{M_{i,j}} \cdot \delta_{i,j}$$

$$\hat{x}_{i,j} \leq x_{i,j} \leq \hat{x}_{i,j} - \mathbf{M_{i,j}} \cdot (1 - \delta_{i,j})$$

$$\mathbf{M_{i,j}} = \textbf{max}\{-\mathbf{l_i}, \mathbf{u_i}\}$$

$$\textbf{min } \mathbf{x_N}$$

S. Dutta et al. - Output Range Analysis for Deep Feedforward Neural Networks (NFM 2018)

Sherlock

Output Range Analysis

use **local search** to speed up the MILP solver

$$\mathbf{l_j} \leq \mathbf{x_{0,j}} \leq \mathbf{u_j}$$

$$\hat{x}_{i+1,j} = \sum_{k=0}^{|\mathbf{X}_i|} w_{j,k}^{i} \cdot x_{i,k} + b_{i,j}$$

$$0 \leq x_{i,j} \leq \mathbf{M_{i,j}} \cdot \delta_{i,j}$$

$$\hat{x}_{i,j} \leq x_{i,j} \leq \hat{x}_{i,j} - \mathbf{M_{i,j}} \cdot (1 - \delta_{i,j})$$

$$\mathbf{M_{i,j}} = \mathbf{max}\{-\mathbf{l_i}, \mathbf{u_i}\}$$

$$\mathbf{x_N} < \mathbf{L}$$

sample random input $\mathbf{X}$
and evaluate output $\mathbf{L}$

S. Dutta et al. - Output Range Analysis for Deep Feedforward Neural Networks (NFM 2018)

Sherlock

Output Range Analysis

use **local search** to speed up the MILP solver

$\mathbf{l_j \leq x_{0,j} \leq u_j}$

$$\hat{x}_{i+1,j} = \sum_{k=0}^{|\mathbf{X}_i|} w^i_{j,k} \cdot x_{i,k} + b_{i,j}$$

$$0 \leq x_{i,j} \leq \mathbf{M_{i,j}} \cdot \delta_{i,j}$$

$$\hat{x}_{i,j} \leq x_{i,j} \leq \hat{x}_{i,j} - \mathbf{M_{i,j}} \cdot (1 - \delta_{i,j})$$

$$\mathbf{M_{i,j} = \max\{-l_i, u_i\}}$$

$$\mathbf{x_N < L}$$

find another input $\hat{\mathbf{X}}$ such that $\hat{\mathbf{L}} \leq \mathbf{x_N}$

S. Dutta et al. - Output Range Analysis for Deep Feedforward Neural Networks (NFM 2018)

Sherlock

Output Range Analysis

use **local search** to speed up the MILP solver

$$\mathbf{l_j} \leq \mathbf{x_{0,j}} \leq \mathbf{u_j}$$

$$\hat{x}_{i+1,j} = \sum_{k=0}^{|\mathbf{X}_i|} w^i_{j,k} \cdot x_{i,k} + b_{i,j}$$

$$0 \leq x_{i,j} \leq \mathbf{M_{i,j}} \cdot \delta_{i,j}$$

$$\hat{x}_{i,j} \leq x_{i,j} \leq \hat{x}_{i,j} - \mathbf{M_{i,j}} \cdot (1 - \delta_{i,j})$$

$$\mathbf{M_{i,j}} = \mathbf{max}\{-\mathbf{l_i}, \mathbf{u_i}\}$$

$$\mathbf{x_N} < \hat{\mathbf{L}}$$

find another input $\hat{\mathbf{X}}$ such that $\hat{\mathbf{L}} \leq \mathbf{x_N}$

S. Dutta et al. - Output Range Analysis for Deep Feedforward Neural Networks (NFM 2018)

Sherlock

Output Range Analysis

use **local search** to speed up the MILP solver

$\mathbf{l_j \leq x_{0,j} \leq u_j}$

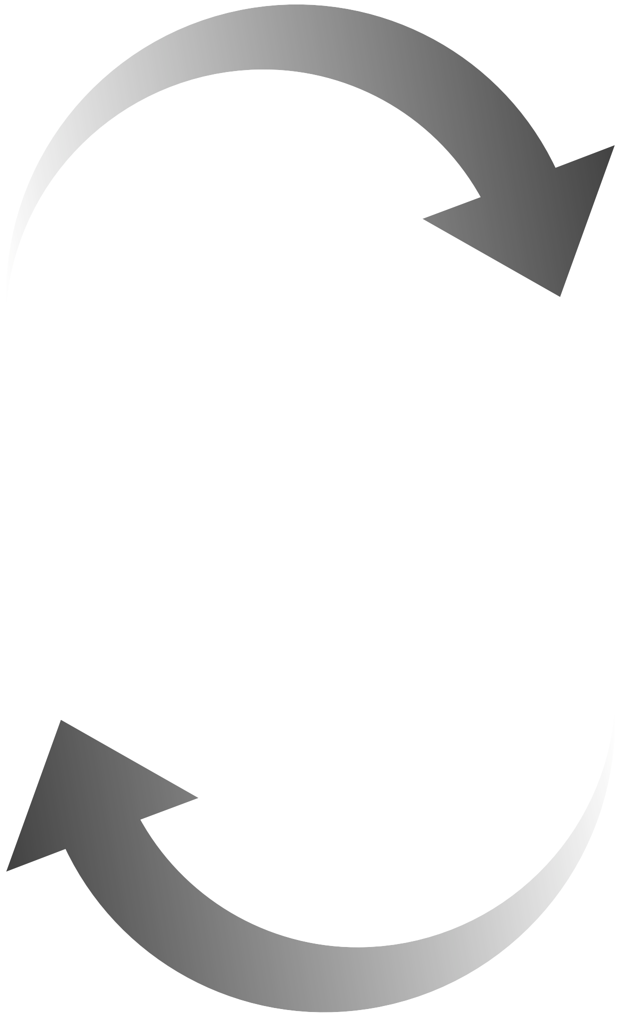
$$\hat{x}_{i+1,j} = \sum_{k=0}^{|\mathbf{X}_i|} w^i_{j,k} \cdot x_{i,k} + b_{i,j}$$

$$0 \leq x_{i,j} \leq \mathbf{M_{i,j}} \cdot \delta_{i,j}$$

$$\hat{x}_{i,j} \leq x_{i,j} \leq \hat{x}_{i,j} - \mathbf{M_{i,j}} \cdot (1 - \delta_{i,j})$$

$$\mathbf{M_{i,j} = \max\{-l_i, u_i\}}$$

$$\mathbf{x_N} < \hat{\mathbf{L}}$$



find another input $\hat{\mathbf{X}}$ such that $\hat{\mathbf{L}} \leq \mathbf{x_N}$

S. Dutta et al. - Output Range Analysis for Deep Feedforward Neural Networks (NFM 2018)

MILP-Based Methods

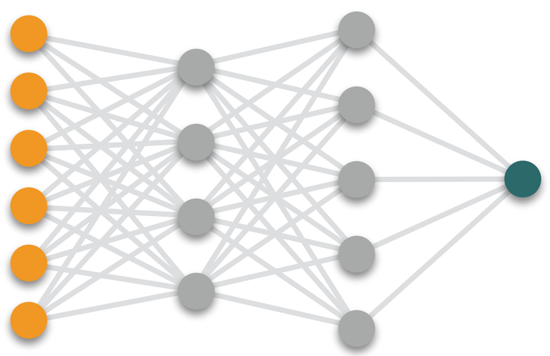
Bounded Encoding with Asymmetric Bounds

$$\hat{x}_{i+1,j} = \sum_{k=0}^{|\mathbf{X}_i|} w^i_{j,k} \cdot x_{i,k} + b_{i,j} \qquad i \in \{0, \ldots, n-1\}$$

$$\delta_{\mathbf{i,j}} \in \{\mathbf{0}, \mathbf{1}\}$$

$$0 \leq x_{i,j} \leq \mathbf{u_{i,j}} \cdot \delta_{i,j} \qquad i \in \{1, \ldots, n-1\}$$

$$\hat{x}_{i,j} \leq x_{i,j} \leq \hat{x}_{i,j} - \mathbf{l_{i,j}} \cdot (1 - \delta_{i,j}) \qquad j \in \{0, \ldots, |\mathbf{X}_i|\}$$



MIPVerify

Finding Nearest Adversarial Example

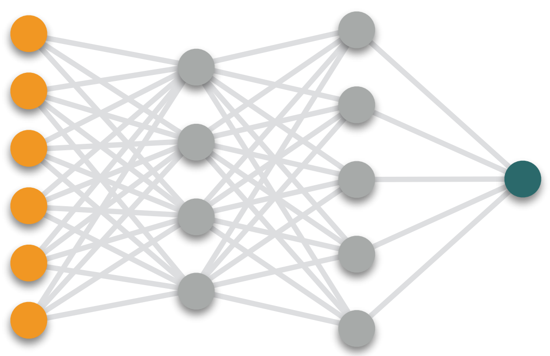
$\min_{\mathbf{X}'} \mathbf{d}(\mathbf{X}, \mathbf{X}')$

$$\hat{x}_{i+1,j} = \sum_{k=0}^{|\mathbf{X}_i|} w^i_{j,k} \cdot x_{i,k} + b_{i,j} \qquad i \in \{0, \ldots, n-1\}$$

$$\delta_{\mathbf{i,j}} \in \{\mathbf{0}, \mathbf{1}\}$$

$$0 \leq x_{i,j} \leq \mathbf{u_{i,j}} \cdot \delta_{i,j} \qquad i \in \{1, \ldots, n-1\}$$

$$\hat{x}_{i,j} \leq x_{i,j} \leq \hat{x}_{i,j} - \mathbf{l_{i,j}} \cdot (1 - \delta_{i,j}) \qquad j \in \{0, \ldots, |\mathbf{X}_i|\}$$



$$\mathbf{x_N} \neq \mathbf{O}$$

V. Tjeng et al. - Evaluating Robustness of Neural Networks with Mixed Integer Programming (ICLR 2019)

Other MILP-Based Methods

- **R. Bunel, I. Turkaslan, P. H. S. Torr, P. Kohli, and M. P. Kumar**. *A Unified View of Piecewise Linear Neural Network Verification*. In NeurIPS, 2018.
  a unifying verification framework for piecewise-linear ReLU neural networks

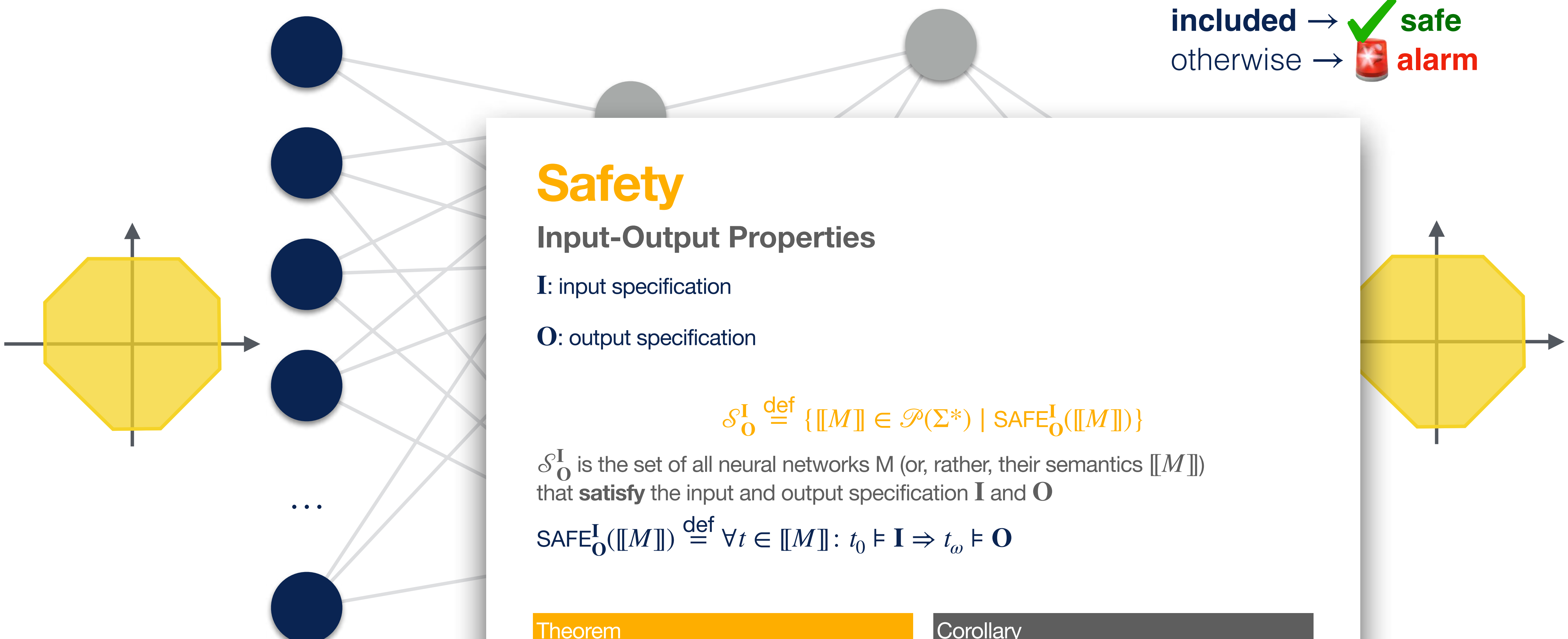
- **C.-H. Cheng, G. Nührenberg, and H. Ruess**. *Maximum Resilience of Artificial Neural Networks*. In ATVA, 2017.
  an approach for finding a lower bound on robustness to adversarial perturbations

- **M. Fischetti and J. Jo**. *Deep Neural Networks and Mixed Integer Linear Optimization*. 2018.
  an approach for feature visualization and building adversarial examples

Static Analysis Methods

Forward Analysis

② check output for **inclusion** in **output specification $\mathbf{O}$**:
included → ✔ **safe**
otherwise → 🚨 **alarm**



Safety

Input-Output Properties

$\mathbf{I}$: input specification

$\mathbf{O}$: output specification

$$\mathcal{S}^{\mathbf{I}}_{\mathbf{O}} \overset{\text{def}}{=} \{[\![M]\!] \in \mathcal{P}(\Sigma^*) \mid \mathsf{SAFE}^{\mathbf{I}}_{\mathbf{O}}([\![M]\!])\}$$

$\mathcal{S}^{\mathbf{I}}_{\mathbf{O}}$ is the set of all neural networks M (or, rather, their semantics $[\![M]\!]$) that **satisfy** the input and output specification $\mathbf{I}$ and $\mathbf{O}$

$$\mathsf{SAFE}^{\mathbf{I}}_{\mathbf{O}}([\![M]\!]) \overset{\text{def}}{=} \forall t \in [\![M]\!] \colon t_0 \models \mathbf{I} \Rightarrow t_\omega \models \mathbf{O}$$

Theorem	Corollary
$M \models \mathcal{S}^{\mathbf{I}}_{\mathbf{O}} \Leftrightarrow \{[\![M]\!]\} \subseteq \mathcal{S}^{\mathbf{I}}_{\mathbf{O}}$	$M \models \mathcal{S}^{\mathbf{I}}_{\mathbf{O}} \Leftrightarrow [\![M]\!] \subseteq \bigcup \mathcal{S}^{\mathbf{I}}_{\mathbf{O}}$

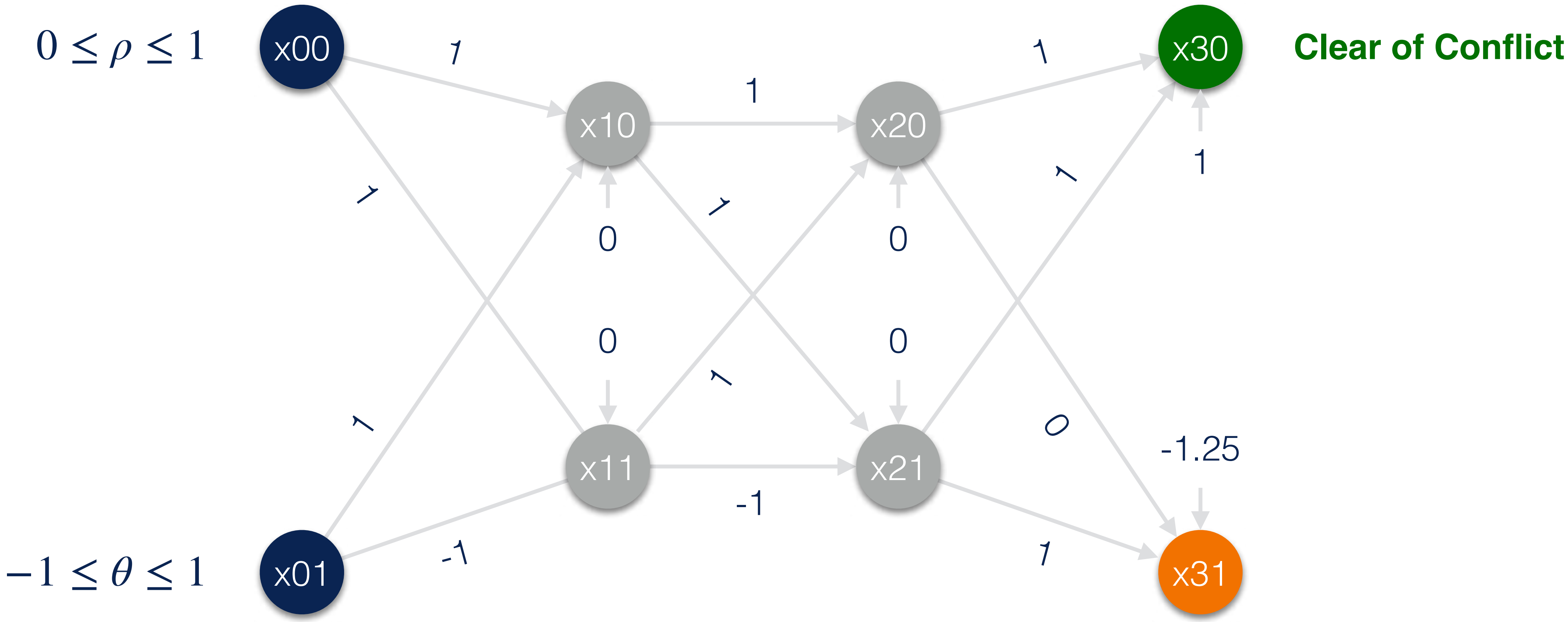
53

Theorem

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

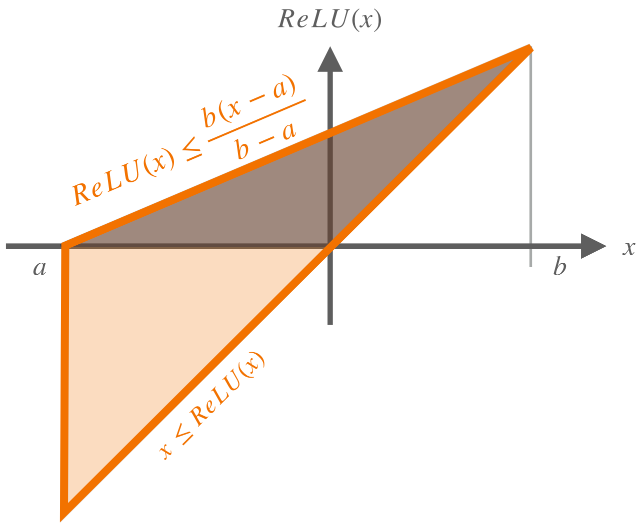
① proceed **forwards from an abstraction** of the input specification $\mathbf{I}$

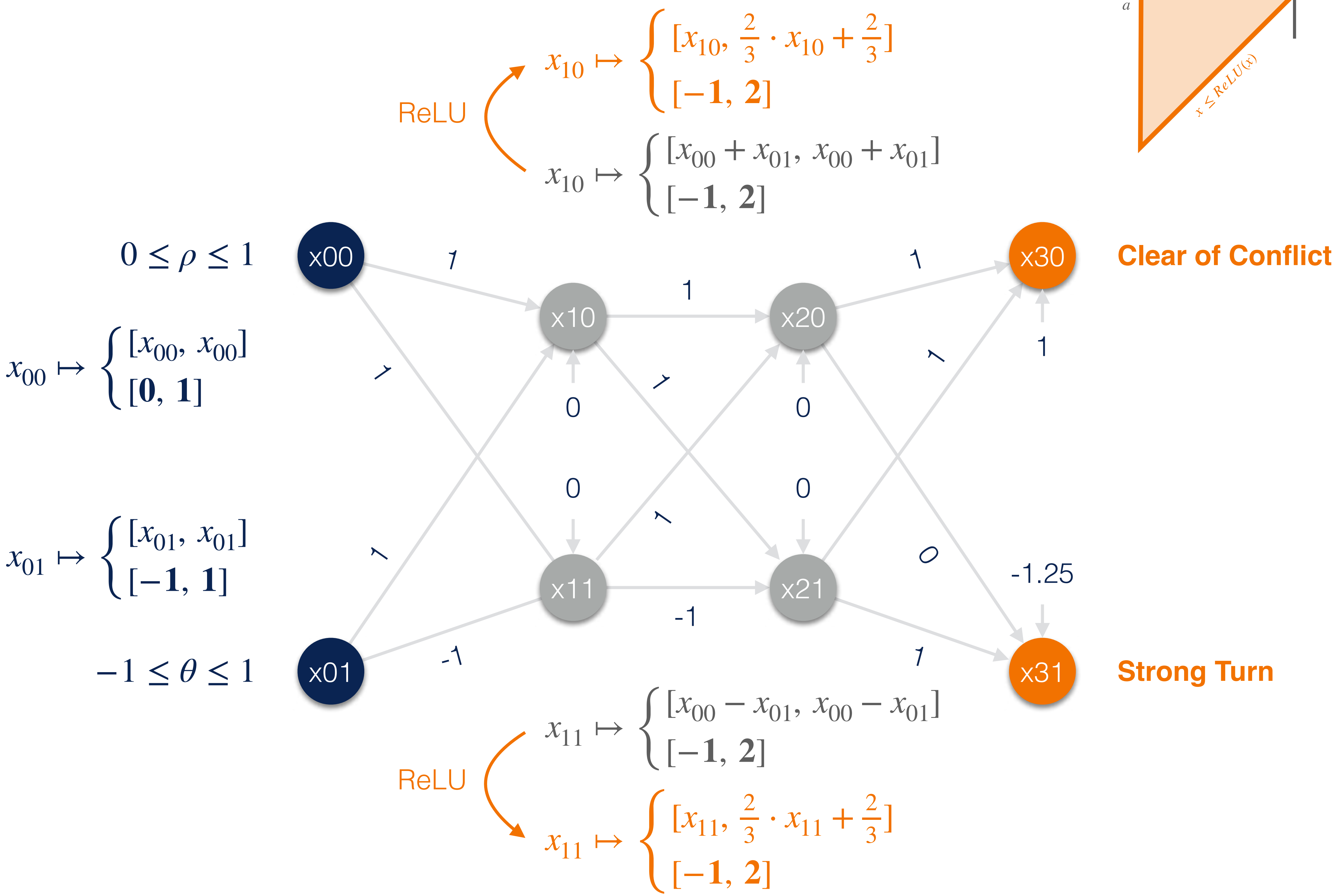
Example



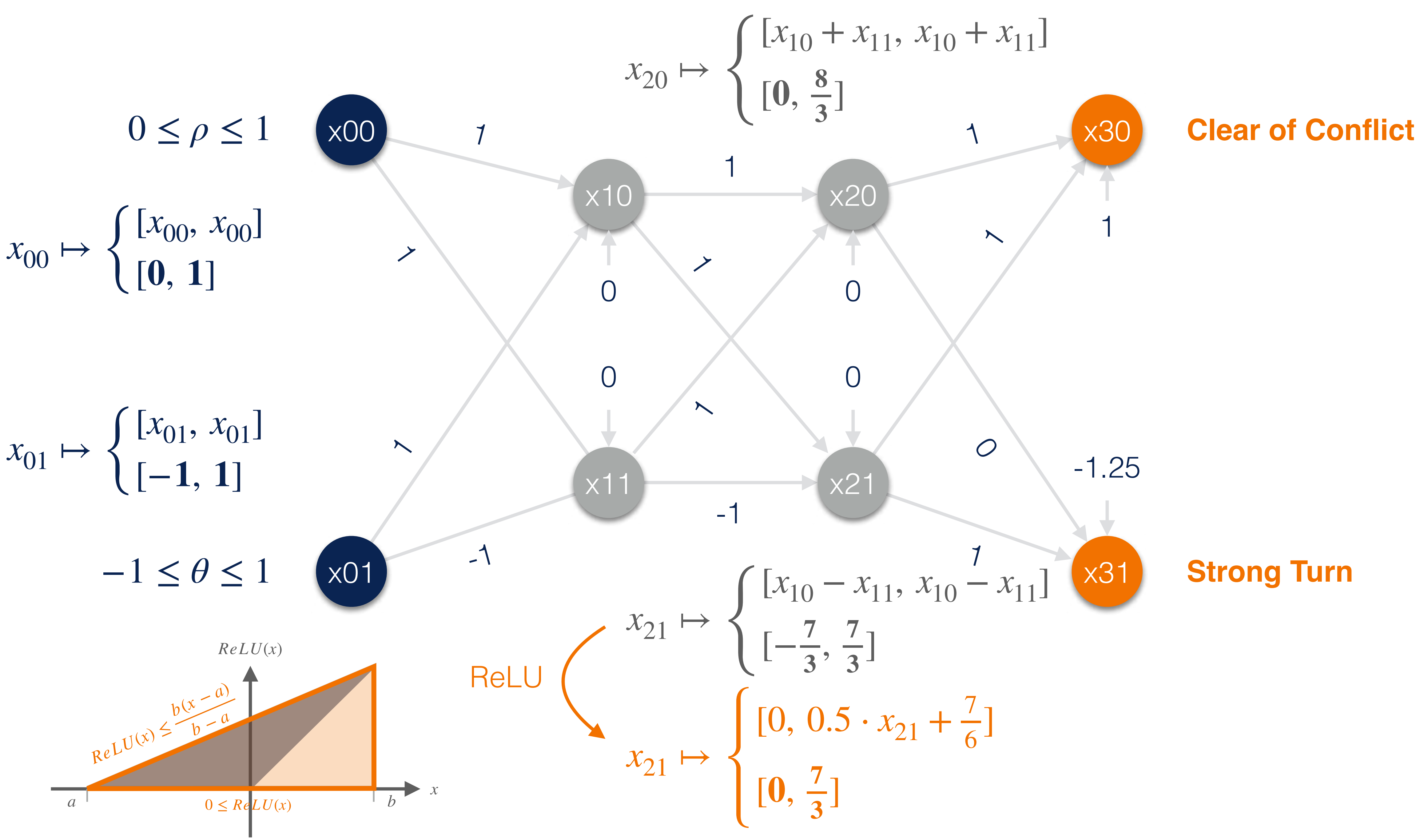
$0 \leq \rho \leq 1$
$-1 \leq \theta \leq 1$
x00
x01
x10
x11
x20
x21
x30
x31
Clear of Conflict
1
1
1
-1
1
1
-1
0
0
0
0
1
1
1
1
1
0
1
-1.25

DeepPoly[Singh19]





DeepPoly[Singh19]

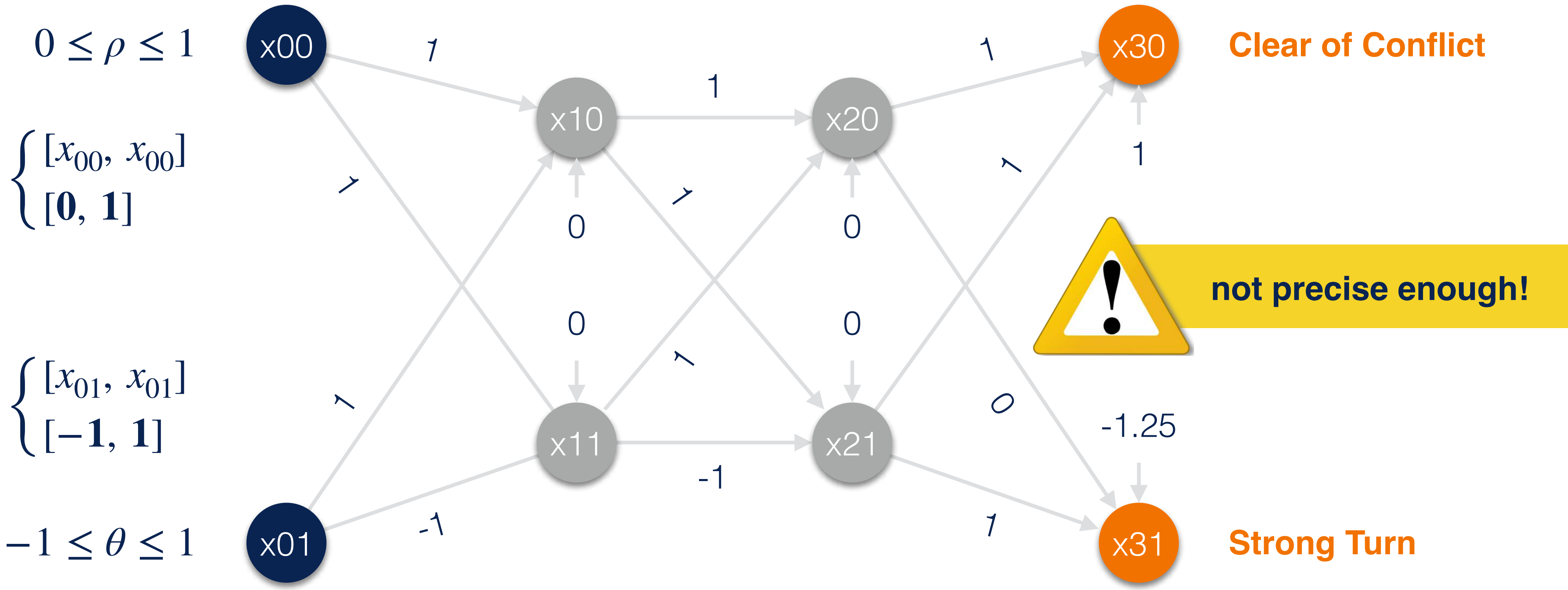


DeepPoly[Singh19]

$$x_{30} \mapsto \begin{cases} [x_{20} + x_{21} + 1,\ x_{20} + x_{21} + 1] \\ [\underline{\mathbf{1}},\ \mathbf{5.5}] \end{cases}$$

$$x_{00} \mapsto \begin{cases} [x_{00},\ x_{00}] \\ [\mathbf{0},\ \mathbf{1}] \end{cases}$$

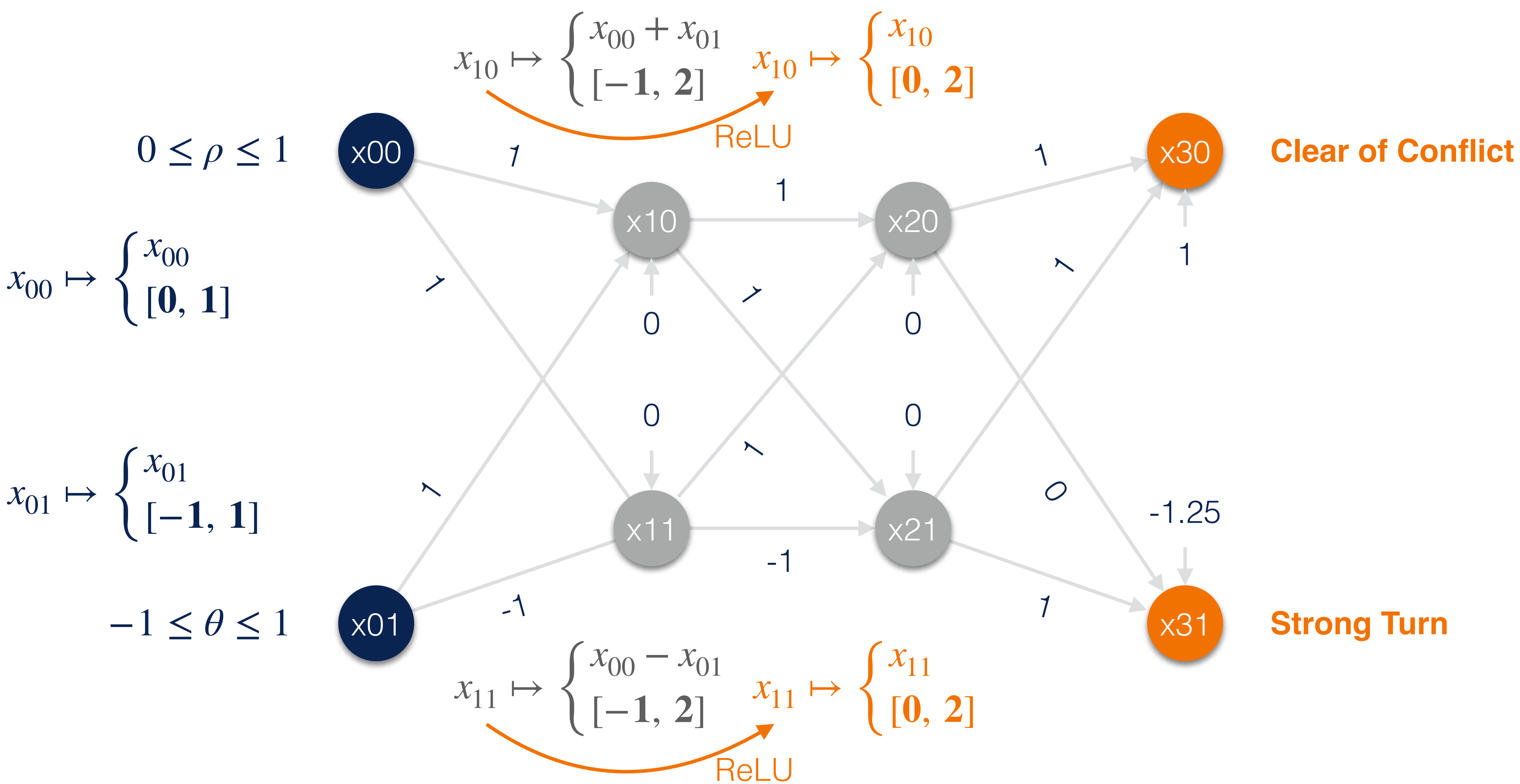
$$x_{01} \mapsto \begin{cases} [x_{01},\ x_{01}] \\ [\mathbf{-1},\ \mathbf{1}] \end{cases}$$



$$x_{31} \mapsto \begin{cases} [x_{21} - 1.25,\ x_{21} - 1.25] \\ [\mathbf{-1.25},\ \underline{\mathbf{\tfrac{13}{12}}}] \end{cases}$$

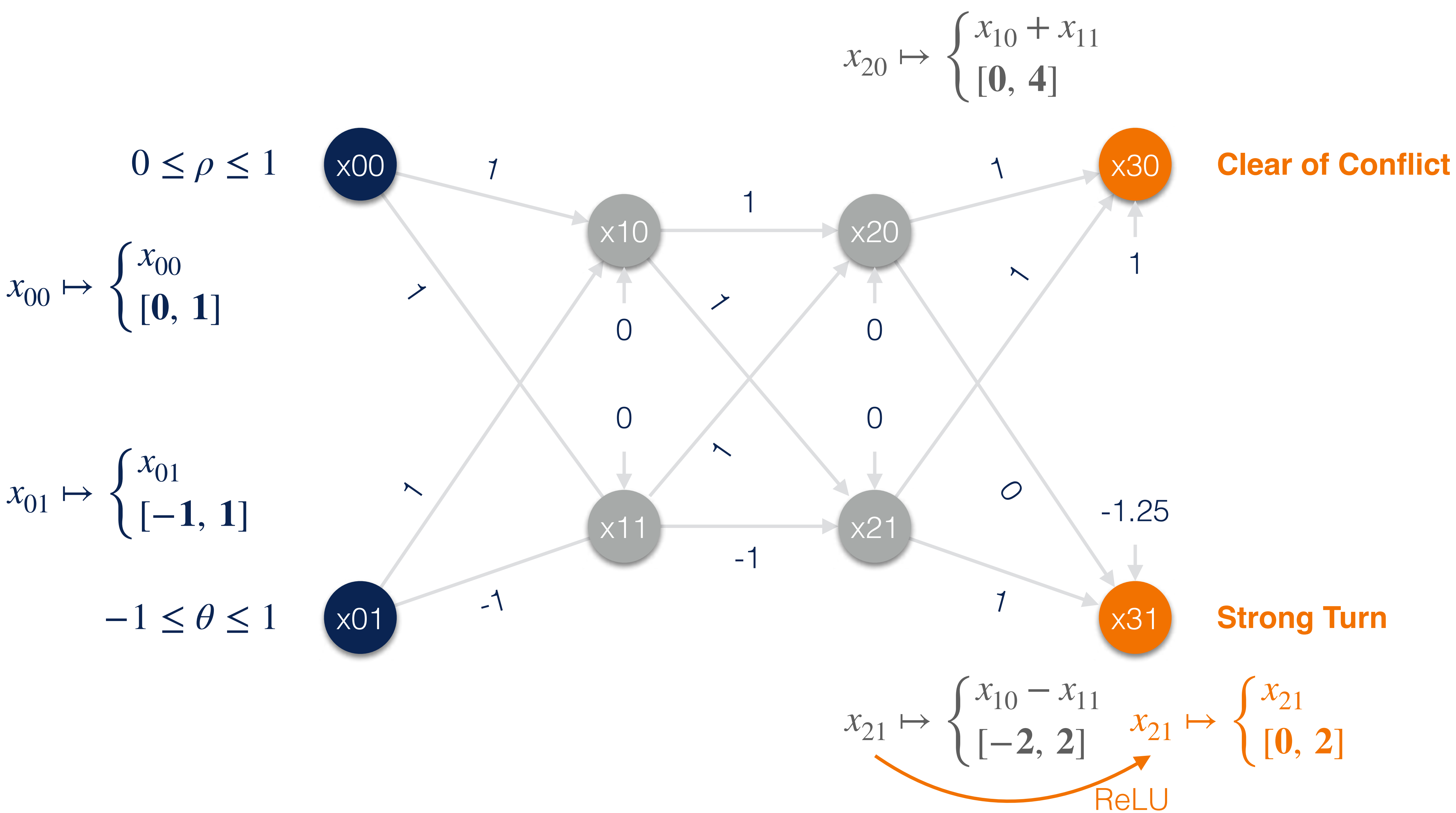
Interval Abstraction

with Symbolic Constant Propagation [Li19]



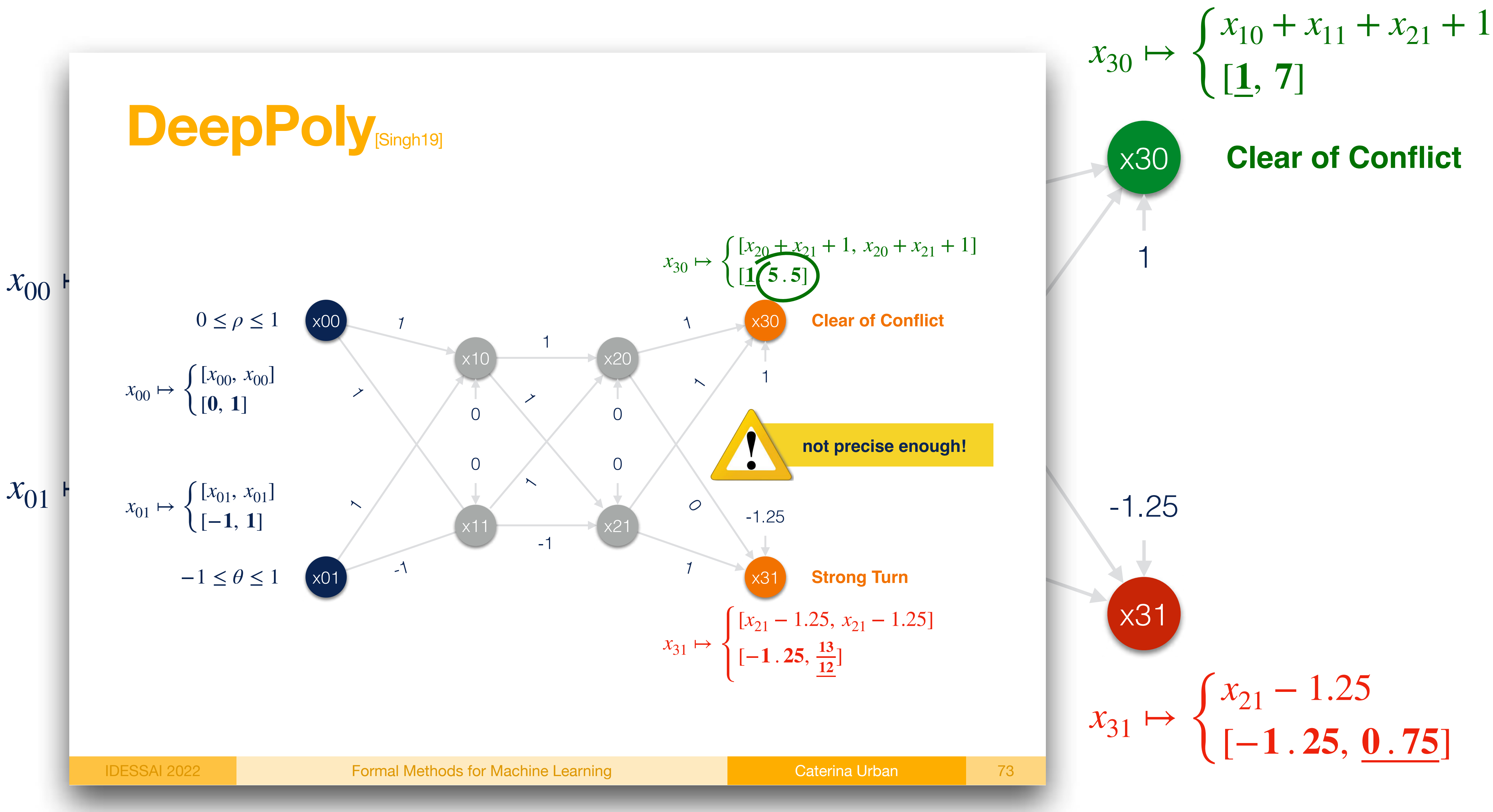
Interval Abstraction

with Symbolic Constant Propagation [Li19]



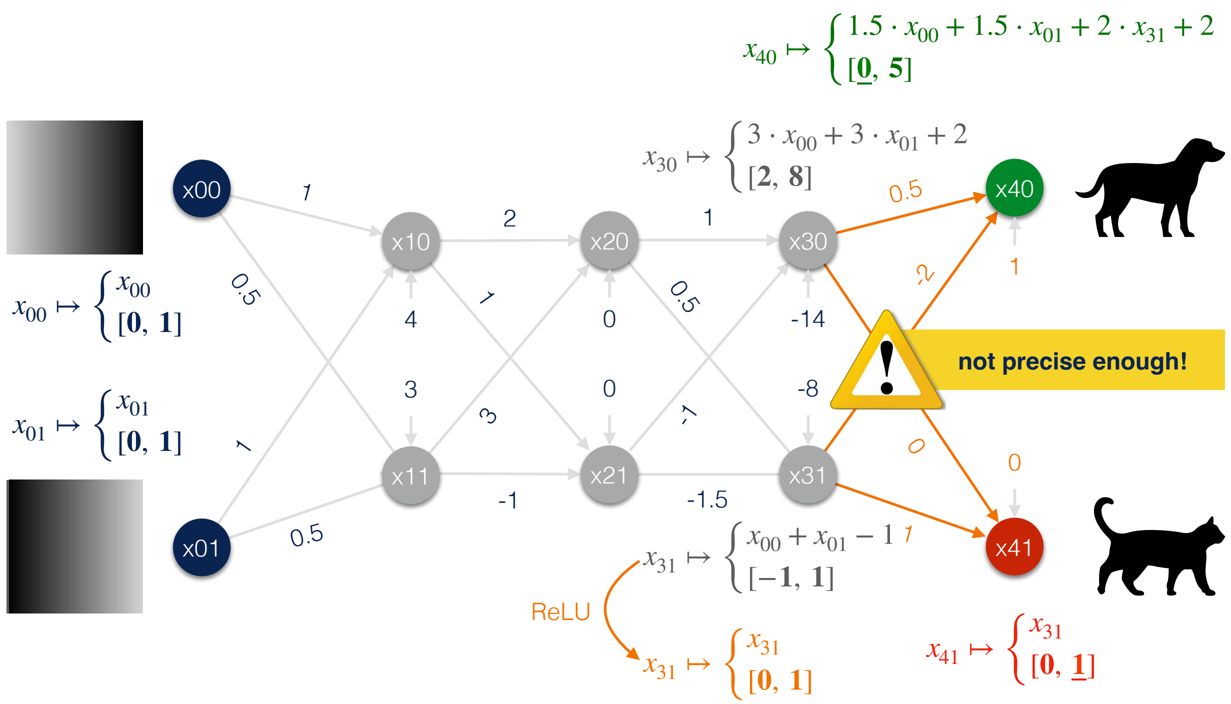
Interval Abstraction

with Symbolic Constant Propagation [Li19]

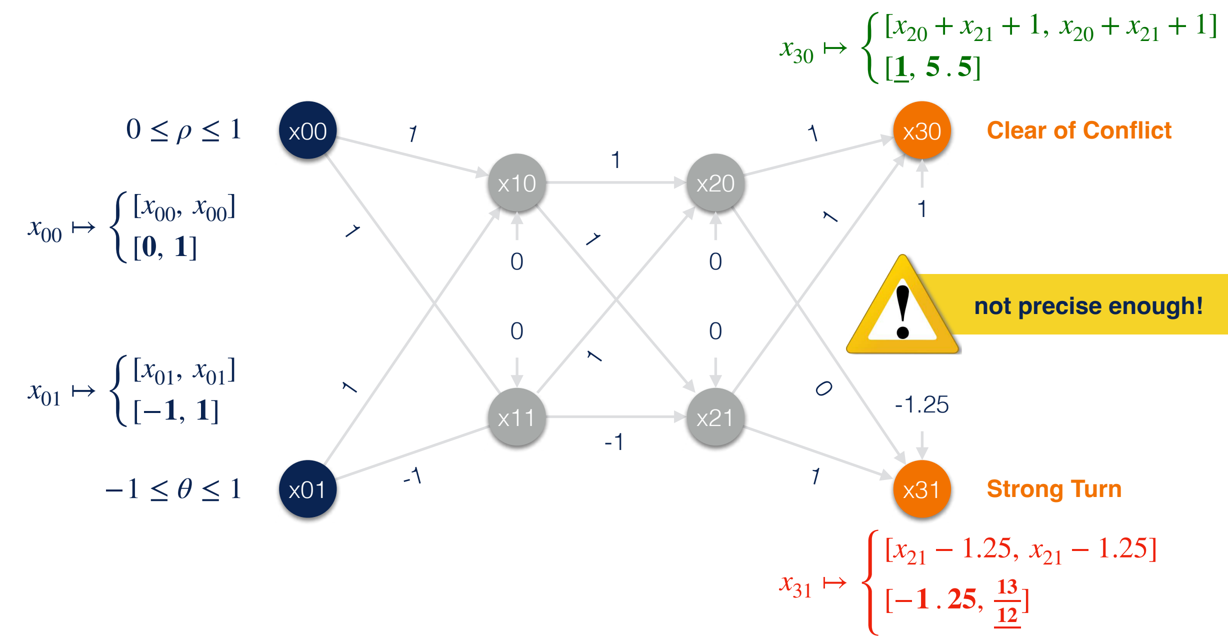


Interval Abstraction

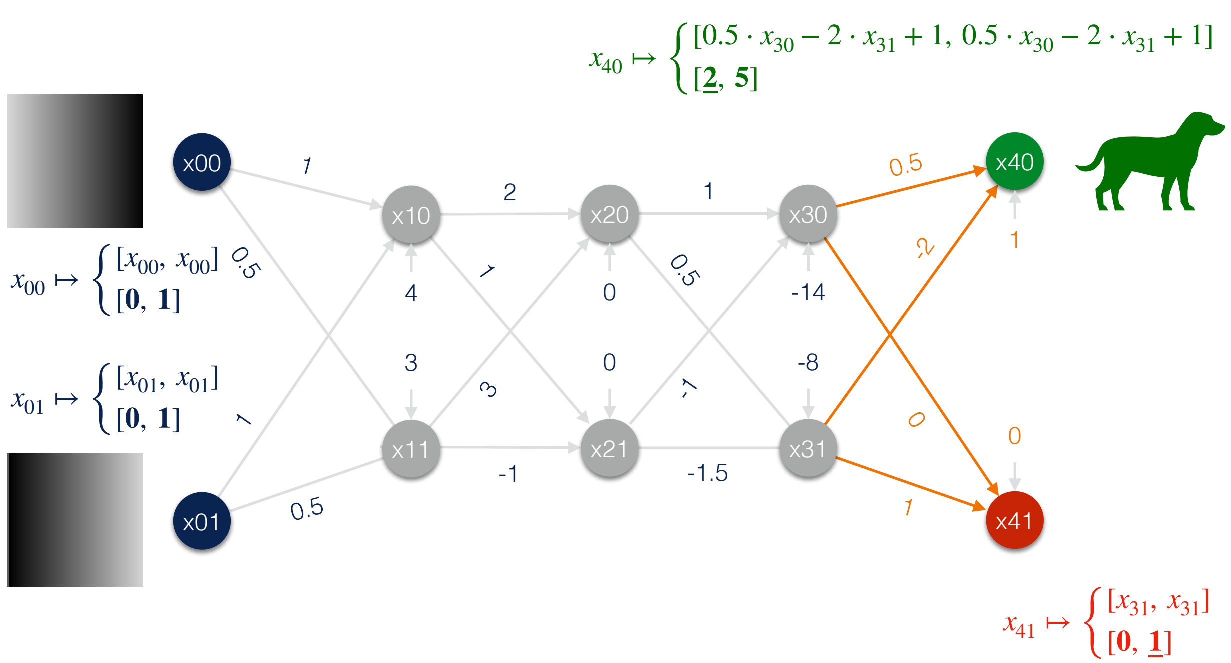
with Symbolic Constant Propagation [Li19]



DeepPoly [Singh19]

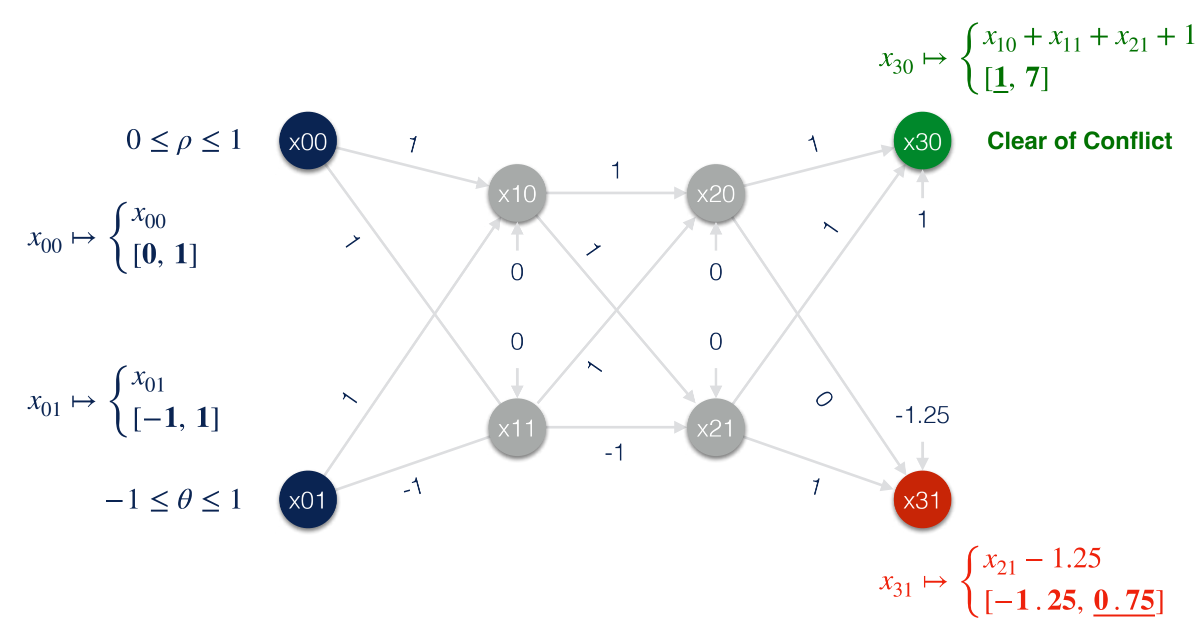


DeepPoly [Singh19]



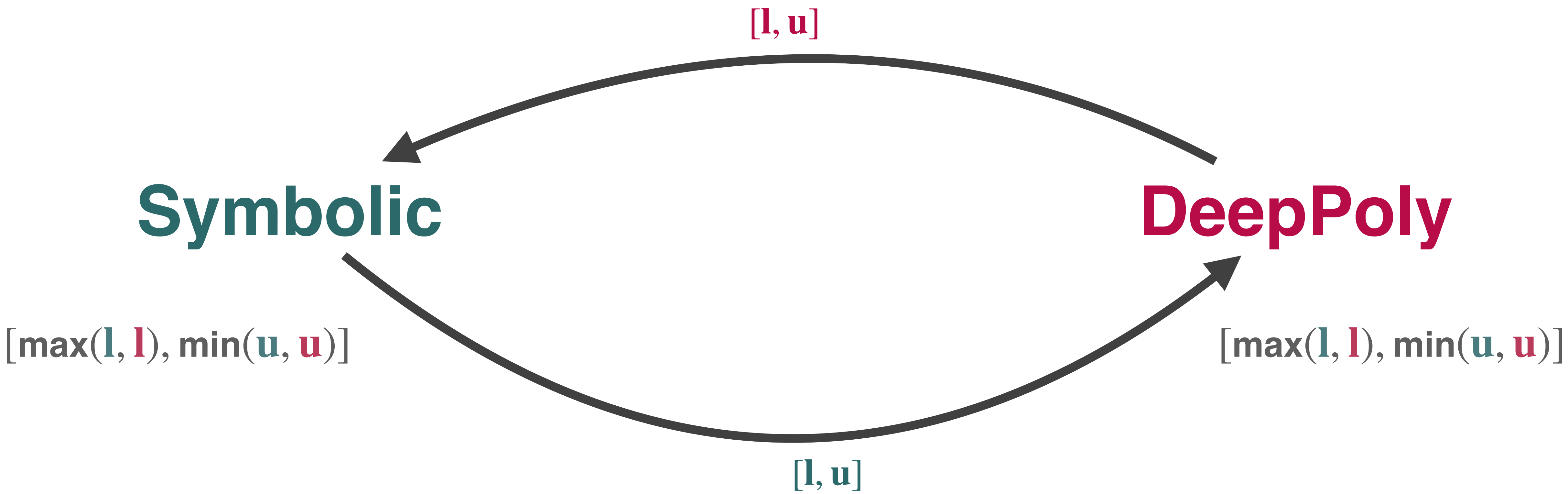
Interval Abstraction

with Symbolic Constant Propagation [Li19]

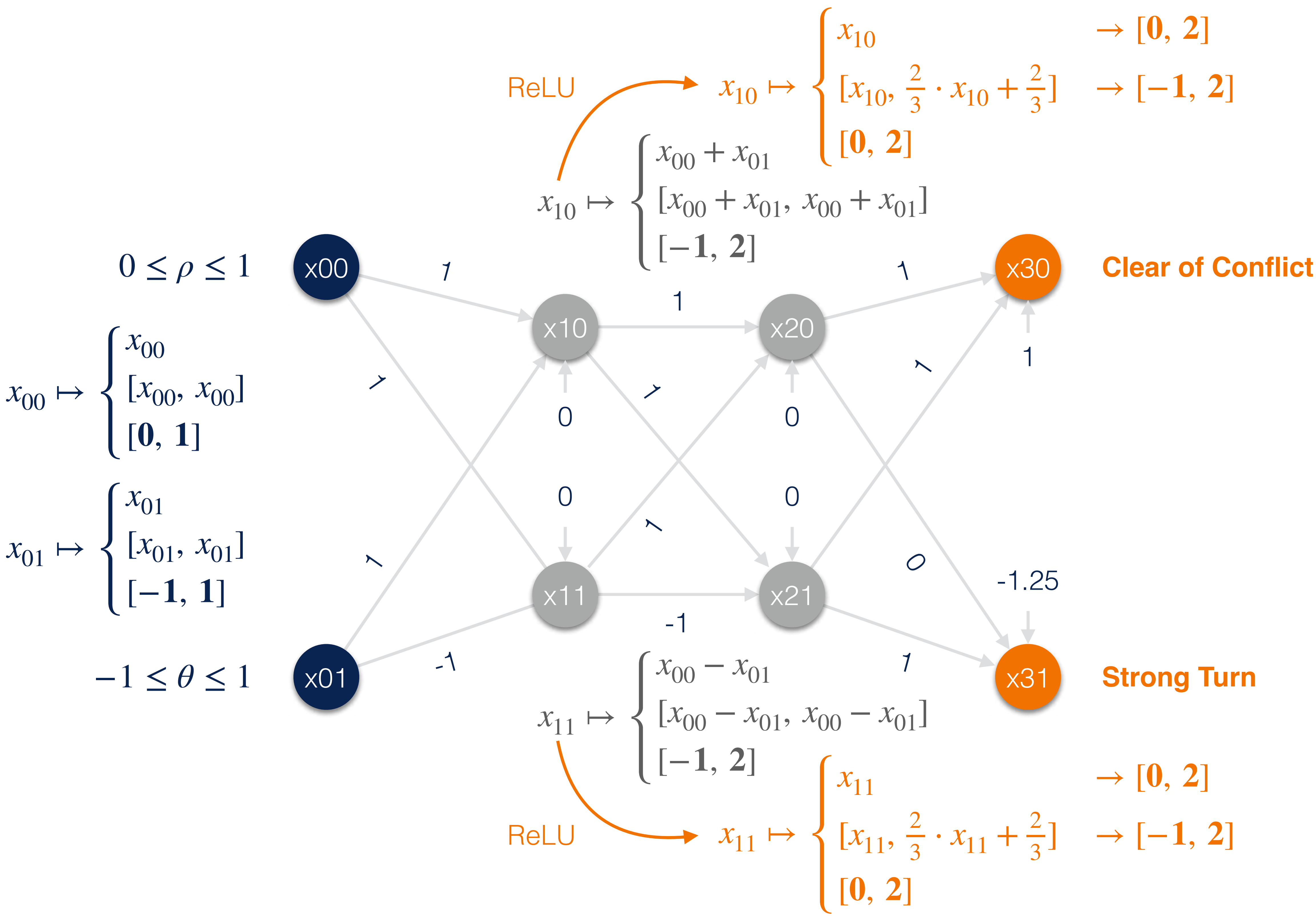


Product Domain [Mazzucato21]

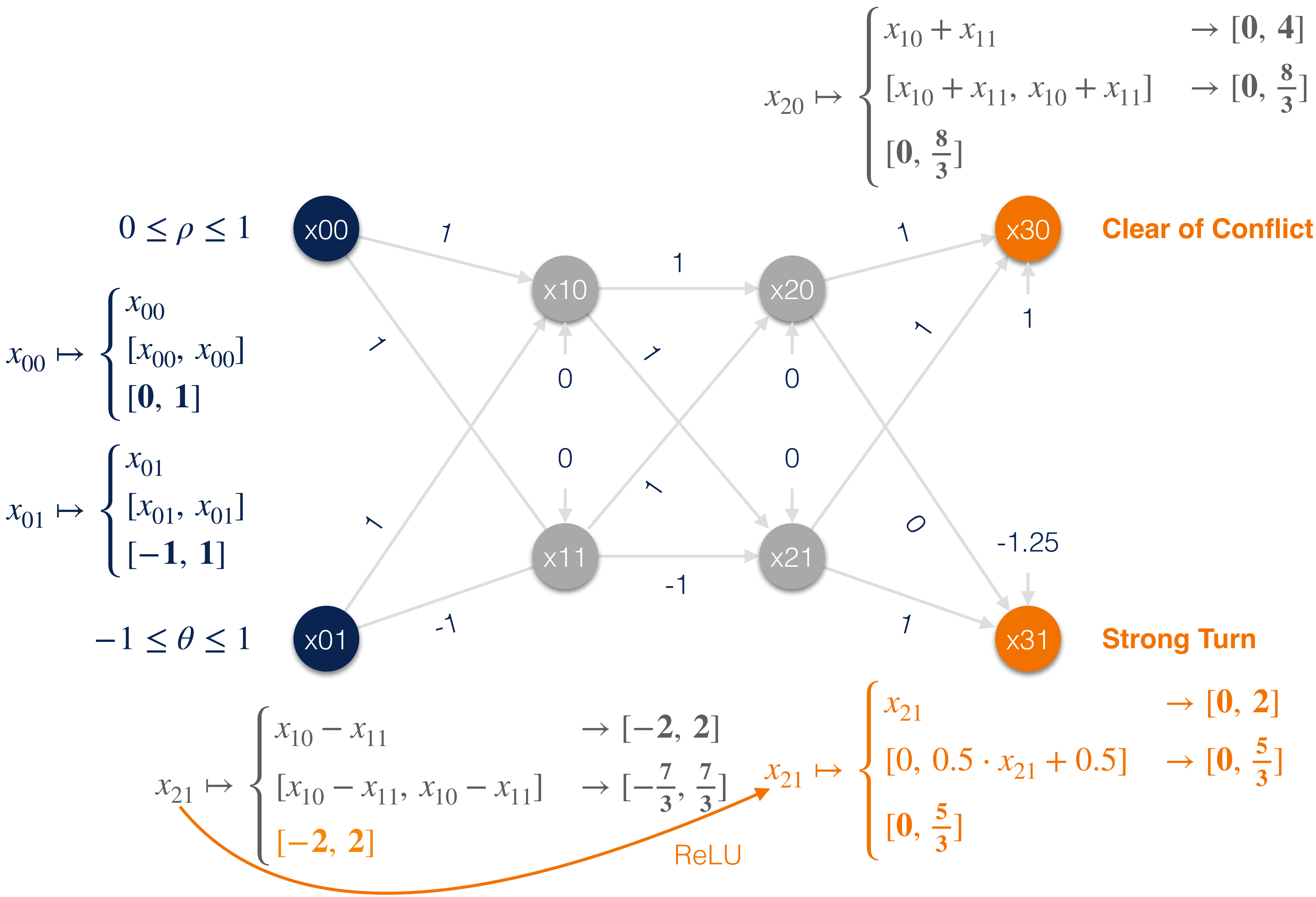
DeepPoly with Symbolic Constant Propagation



Product Domain [Mazzucato21]



Product Domain [Mazzucato21]



Product Domain [Mazzucato21]

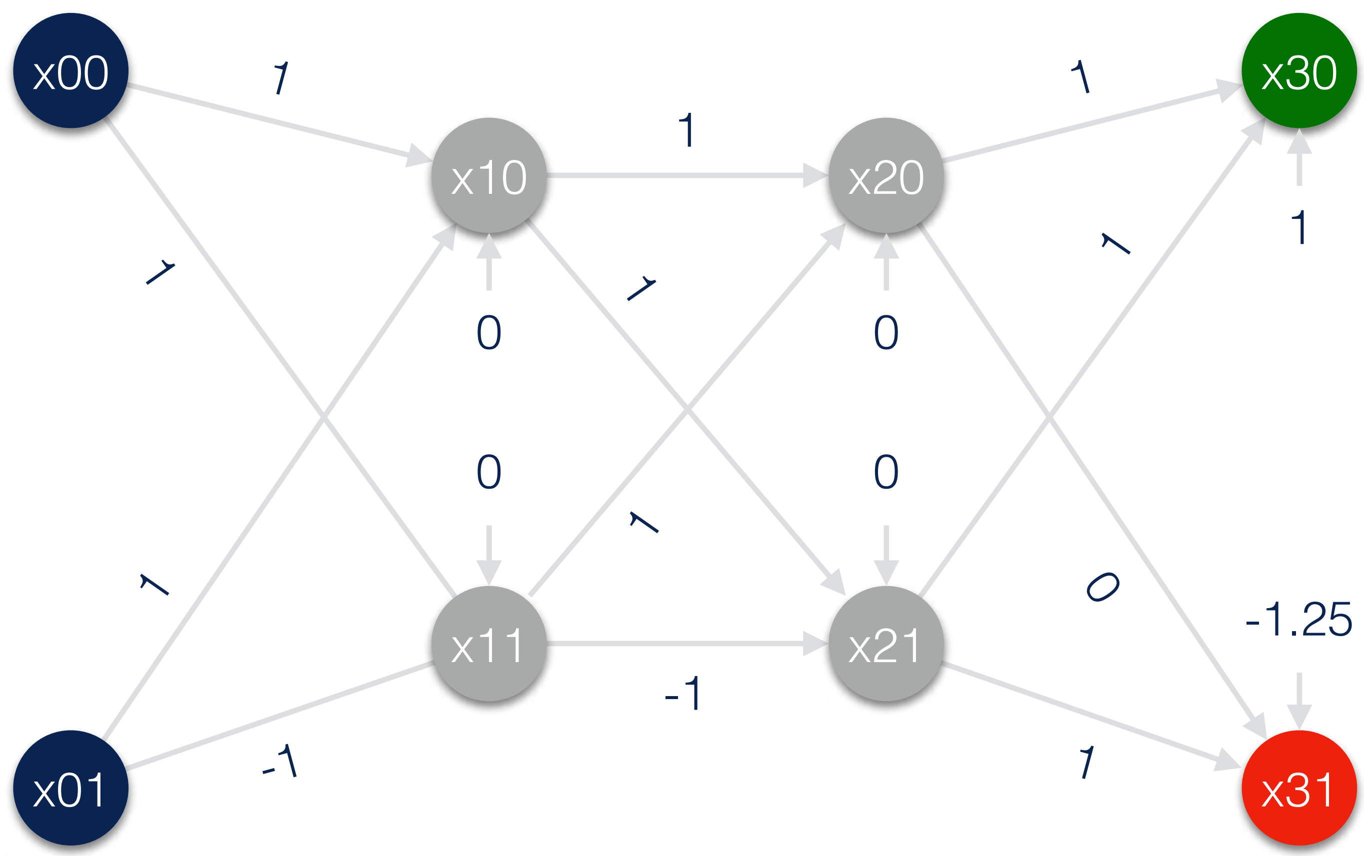
$$x_{30} \mapsto \begin{cases} x_{10} + x_{11} + x_{21} + 1 & \to [\mathbf{1},\ \tfrac{\mathbf{20}}{\mathbf{3}}] \\ [x_{20} + x_{21} + 1,\ x_{20} + x_{21} + 1] & \to [\mathbf{1},\ \mathbf{4.5}] \\ [\underline{\mathbf{1}},\ \mathbf{4.5}] & \end{cases}$$

$0 \leq \rho \leq 1$

$$x_{00} \mapsto \begin{cases} x_{00} \\ [x_{00},\ x_{00}] \\ [\mathbf{0},\ \mathbf{1}] \end{cases}$$

$$x_{01} \mapsto \begin{cases} x_{01} \\ [x_{01},\ x_{01}] \\ [-\mathbf{1},\ \mathbf{1}] \end{cases}$$

$-1 \leq \theta \leq 1$



Clear of Conflict

$$x_{31} \mapsto \begin{cases} x_{21} - 1.25 & \to [-\mathbf{1.25},\ \tfrac{\mathbf{5}}{\mathbf{12}}] \\ [x_{21} - 1.25,\ x_{21} - 1.25] & \to [-\mathbf{1.25},\ \tfrac{\mathbf{5}}{\mathbf{12}}] \\ [-\mathbf{1.25},\ \underline{\tfrac{\mathbf{5}}{\mathbf{12}}}] & \end{cases}$$

Other Complete Methods

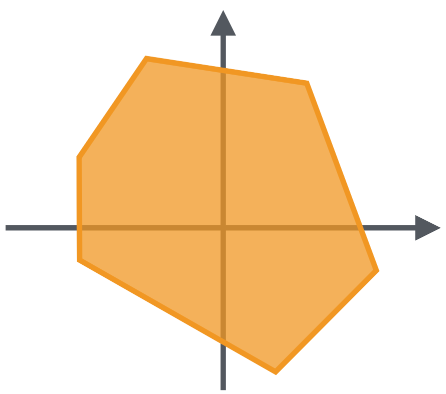
use union of efficient representations of bounded convex polyhedra

Star Sets

Exact Static Analysis Method

$$\Theta \stackrel{\text{def}}{=} \langle c, V, P \rangle$$

$c \in \mathcal{R}^n$: center
$V = \{v_1, \ldots, v_m\}$: basis vectors in $\mathcal{R}^n$
$P : \mathcal{R}^m \to \{ \bot, \top \}$: predicate



$$[\![\Theta]\!] = \{x \mid x = c + \sum_{i=1}^{m} \alpha_i v_i \text{ such that } P(\alpha_1, \ldots, \alpha_m) = \top \}$$

- fast and cheap **affine mapping operations** → neural network layers
- inexpensive **intersections with half-spaces** → ReLU activations

H.-D. Tran et al. - Star-Based Reachability Analysis of Deep Neural Networks (FM 2018)

Star Sets

Exact Static Analysis Method

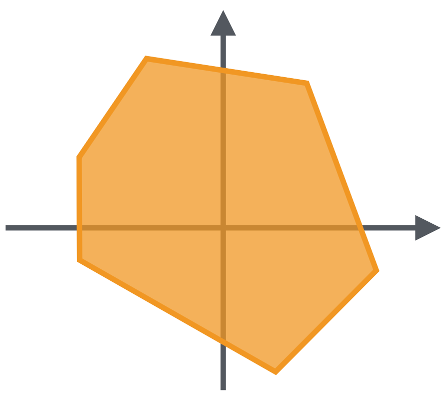
us
efficient
of bounded

Follow-up Work

H.-D. Tran et al. - Verification of Deep Convolutional Neural Networks Using ImageStars (CAV 2020)

$\Theta \stackrel{\text{def}}{=} \langle c, V, P \rangle$

$c \in \mathscr{R}^n$: center
$V = \{v_1, \ldots, v_m\}$: basis vectors in $\mathscr{R}^n$
$P : \mathscr{R}^m \rightarrow \{ \perp, \top \}$: predicate



$$[\![\Theta]\!] = \{ x \mid x = c + \sum_{i=1}^{m} \alpha_i v_i \text{ such that } P(\alpha_1, \ldots, \alpha_m) = \top \}$$

- fast and cheap **affine mapping operations** → neural network layers
- inexpensive **intersections with half-spaces** → ReLU activations

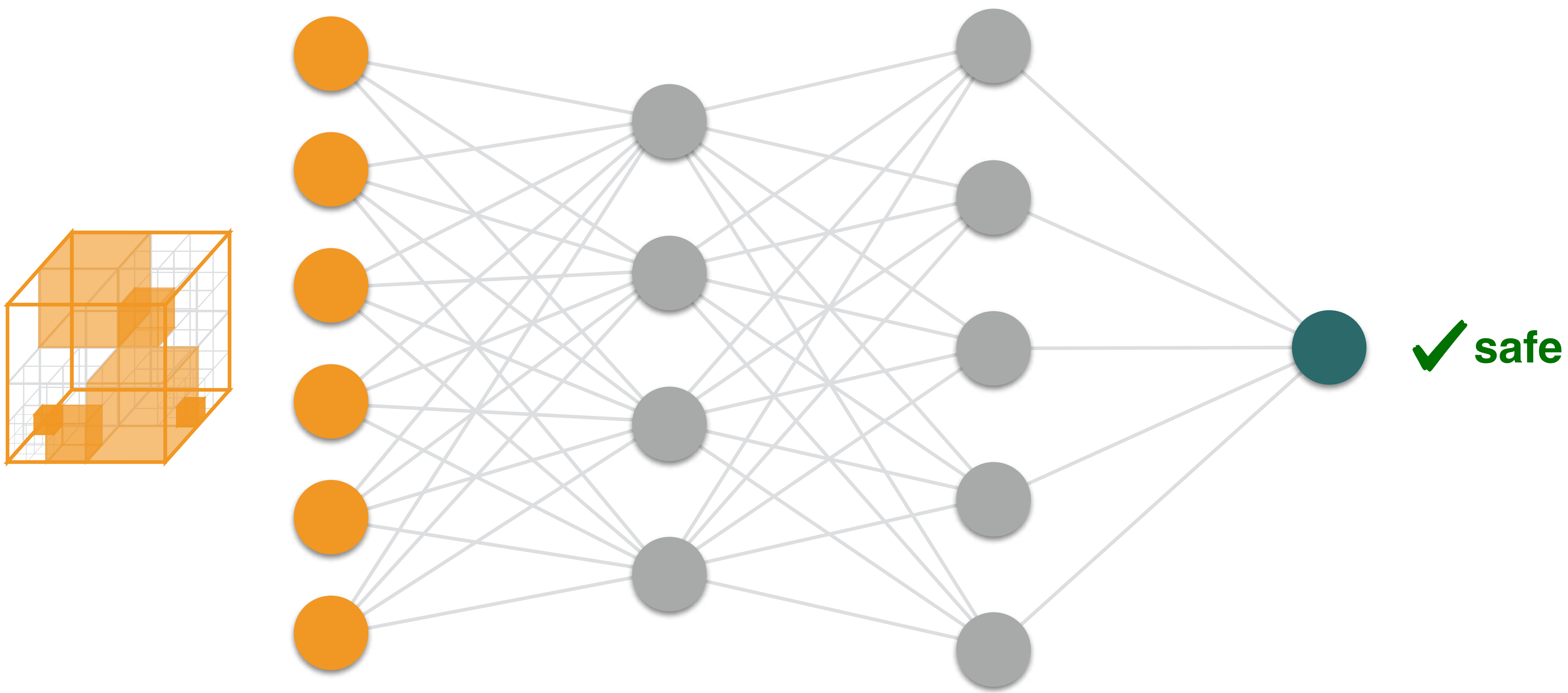
---

H.-D. Tran et al. - Star-Based Reachability Analysis of Deep Neural Networks (FM 2018)

ReluVal

Asymptotically Complete Method

use symbolic propagation
+ iterative input refinement



S. Wang et al. - Formal Security Analysis of Neural Networks Using Symbolic Intervals (USENIX Security 2018)

Neurify

Asymptotically Complete Method

use symbolic propagation + convex ReLU approximation + iterative input/ReLU refinement

$$x_{i,j} \mapsto \begin{cases} [\sum_k c_{0,k} \cdot x_{0,k} + c, \ \sum_k d_{0,k} \cdot x_{0,k} + d] & c_{0,k}, c, d_{0,k}, d \in \mathscr{R} \\ [a, \ b] & a, b \in \mathscr{R} \end{cases}$$

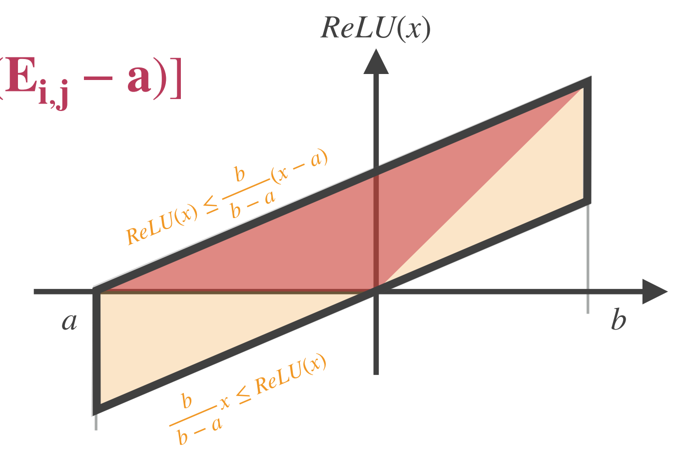
$$x_{i,j} \mapsto \begin{cases} [\mathbf{E_{i,j}}, \mathbf{E_{i,j}}] \\ [\mathbf{a}, \mathbf{b}] \end{cases}$$

ReLU

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

ReLU

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



ReLU

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

S. Wang et al. - Formal Security Analysis of Neural Networks Using Symbolic Intervals (USENIX Security 2018)

Further Complete Methods

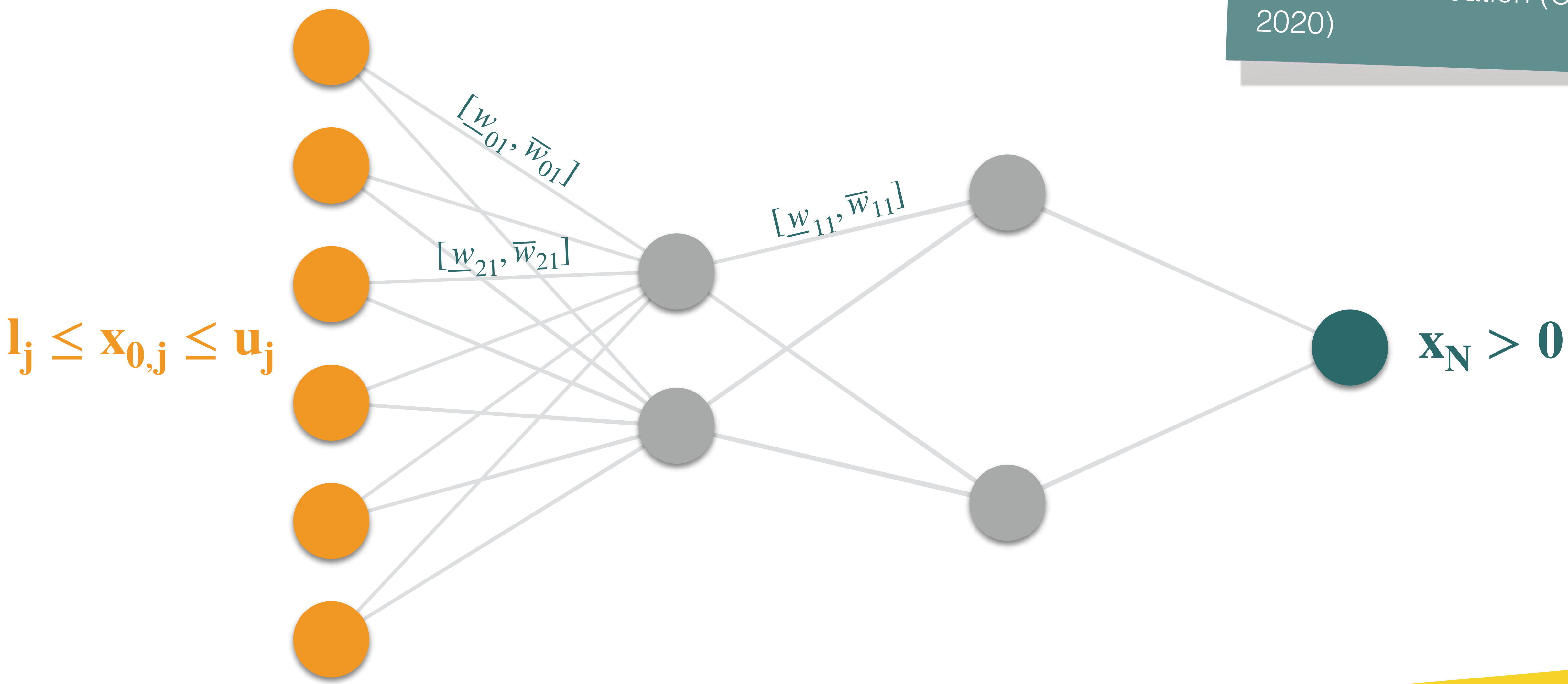
- **W. Ruan, X. Huang, and M. Kwiatkowska**. *Reachability Analysis of Deep Neural Networks with Provable Guarantees*. In IJCAI, 2018.
  a global optimization-based approach for verifying Lipschitz continuous neural networks

- **G. Singh, T. Gehr, M. Püschel, and M. Vechev**. *Boosting Robustness Certification of Neural Networks*. In ICLR, 2019.
  an approach combining abstract interpretation and (mixed integer) linear programming

Other Incomplete Methods

Interval Neural Networks

Abstraction-Based Method



Related Work

Y. Y. Elboher et al. - An Abstraction-Based Framework for Neural Network Verification (CAV 2020)

merge neurons layer-wise based on partitioning strategy + replace weights with intervals

P. Prabhakar and Z. R. Afza - Abstraction based Output Range Analysis for Neural Networks (NeurIPS 2019)

Further Incomplete Methods

- **W. Xiang, H.-D. Tran, and T. T. Johnson**. *Output Reachable Set Estimation and Verification for Multi-Layer Neural Networks*. 2018.
  an approach combining simulation and linear programming

- **K. Dvijotham, R. Stanforth, S. Gowal, T. Mann, and P. Kohli**. *A Dual Approach to Scalable Verification of Deep Networks*. In UAI, 2018.
  an approach based on duality for verifying neural networks

Further Incomplete Methods

- **E. Wong and Z. Kolter**. *Provable Defenses Against Adversarial Examples via the Convex Outer Adversarial Polytope*. In ICML, 2018.
  A. Raghunathan, J. Steinhardt, and P. Liang. *Certified Defenses against Adversarial Examples*. In ICML, 2018.
  T.-W. Weng, H. Zhang, H. Chen, Z. Song, C.-J. Hsieh, L. Daniel, D. Boning, and I. Dhillon. *Towards Fast Computation of Certified Robustness for ReLU Networks*. In ICML, 2018.
  H. Zhang, T.-W. Weng, P.-Y. Chen, C.-J. Hsieh, and L. Daniel. Efficient *Neural Network Robustness Certification with General Activation Functions*. In NeurIPS, 2018.
  approaches for finding a lower bound on robustness to adversarial perturbations

Further Incomplete Methods

- **A. Boopathy, T.-W. Weng, P.-Y. Chen, S. Liu, and L. Daniel**. *CNN-Cert: An Efficient Framework for Certifying Robustness of Convolutional Neural Networks*. In AAAI, 2019.
  approach focusing on convolutional neural networks

- **C.-Y. Ko, Z. Lyu, T.-W. Weng, L. Daniel, N. Wong, and D. Lin**. *POPQORN: Quantifying Robustness of Recurrent Neural Networks*. In ICML, 2019.
  H. Zhang, M. Shinn, A. Gupta, A. Gurfinkel, N. Le, and N. Narodytska. *Verification of Recurrent Neural Networks for Cognitive Tasks via Reachability Analysis*. In ECAI, 2020.
  approaches focusing on recurrent neural networks

- **D. Gopinath, H. Converse, C. S. Pasareanu, and A. Taly**. *Property Inference for Deep Neural Networks*. In ASE, 2019.
  an approach for inferring safety properties of neural networks

Complete Methods

Advantages

sound and **complete**

Disadvantages

soundness not typically guaranteed
with respect to **floating-point arithmetic**

do not scale to large models

often **limited** to certain
model **architectures**

Incomplete Methods

Advantages

able to scale to large models

sound often also with respect to
floating-point arithmetic

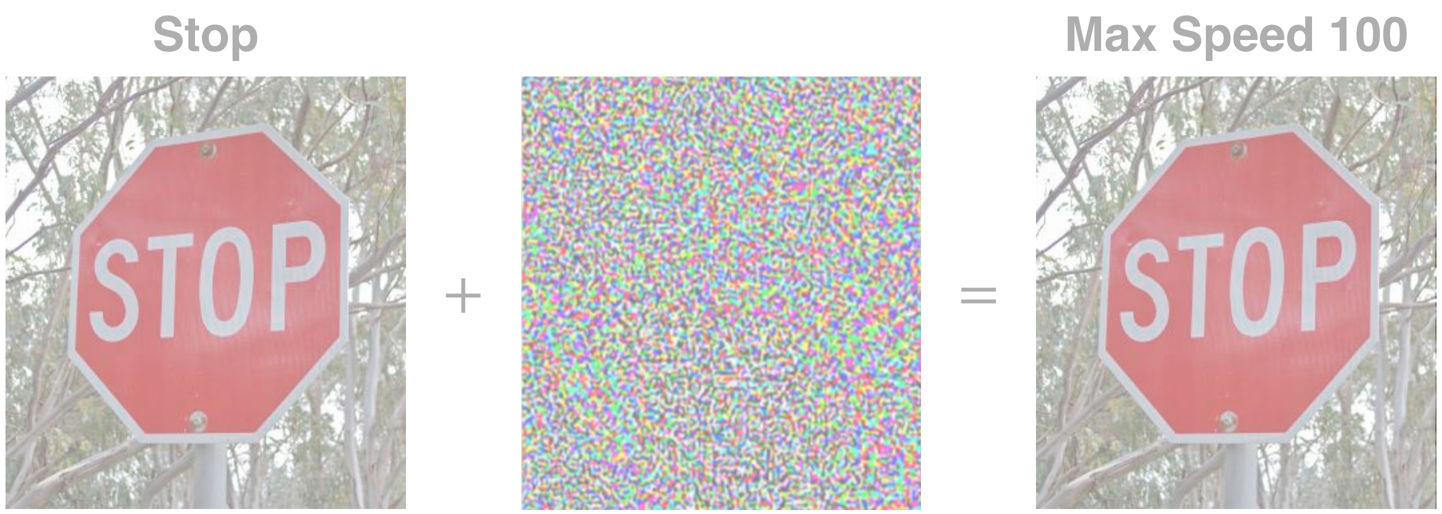
less limited to certain
model **architectures**

Disadvantages

suffer from **false positives**

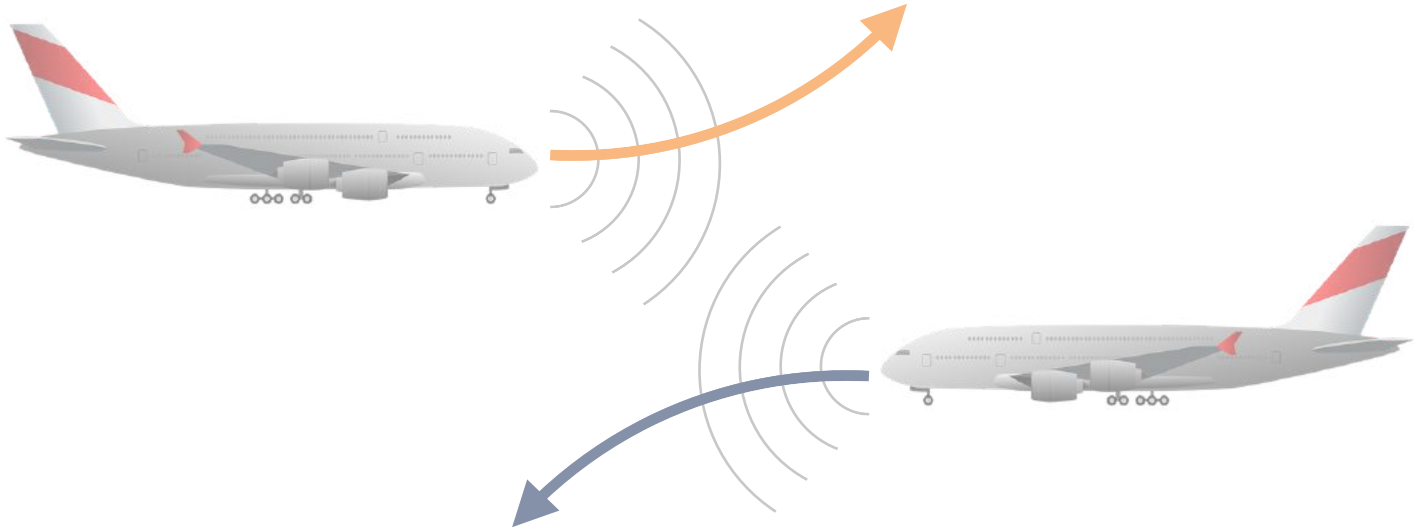
Stability

Goal G3 in [Kurd03]



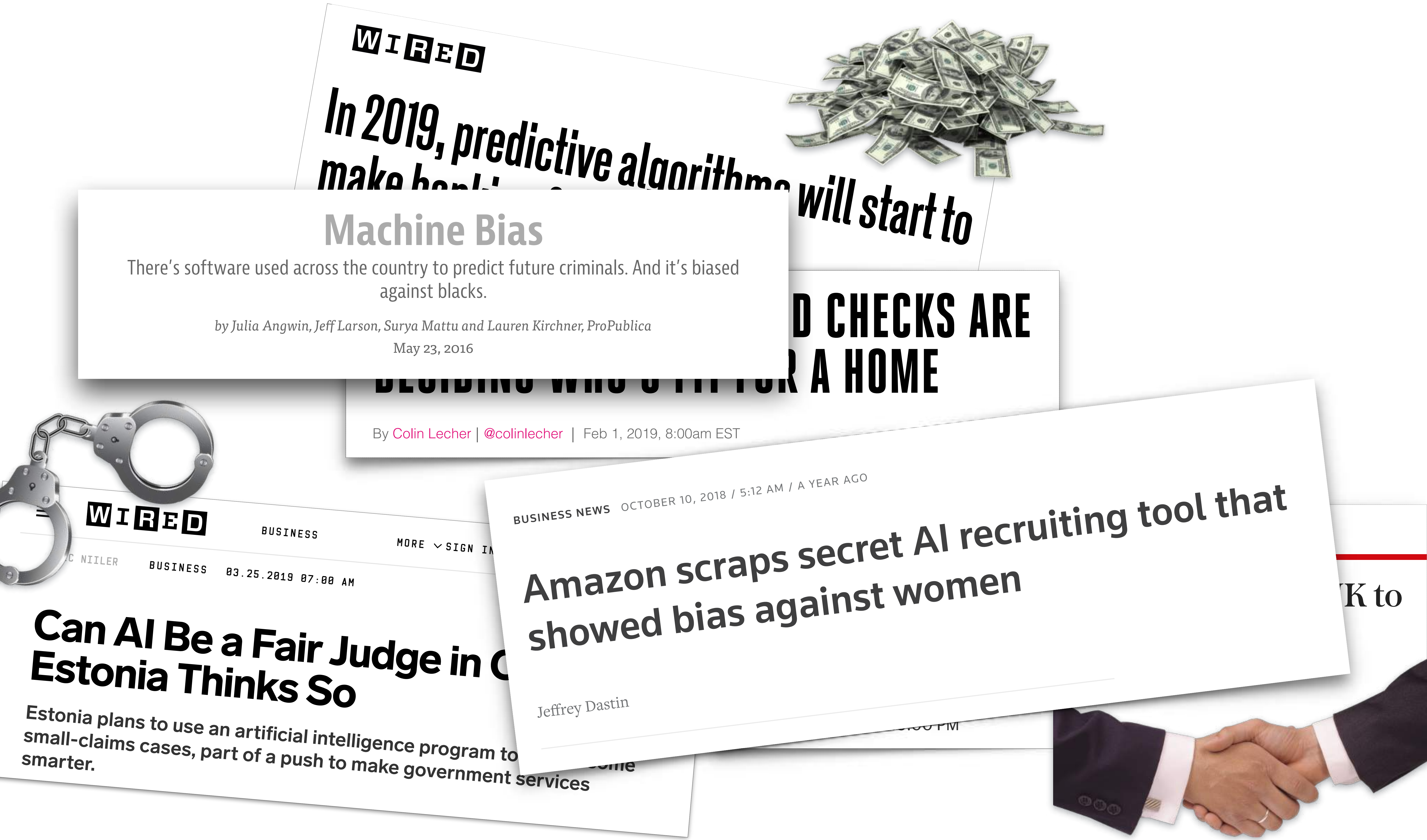
Safety

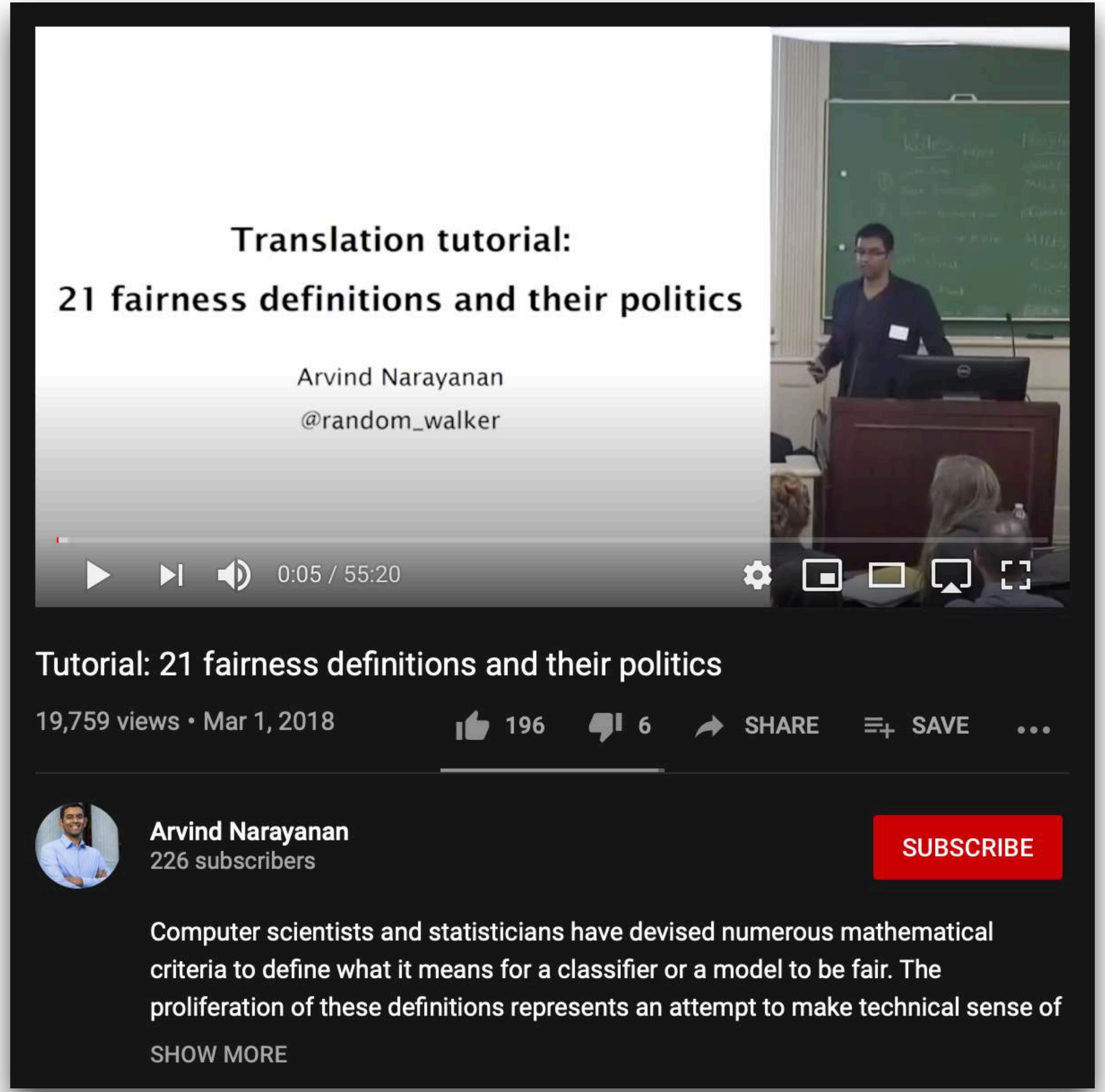
Goal G4 in [Kurd03]



Fairness

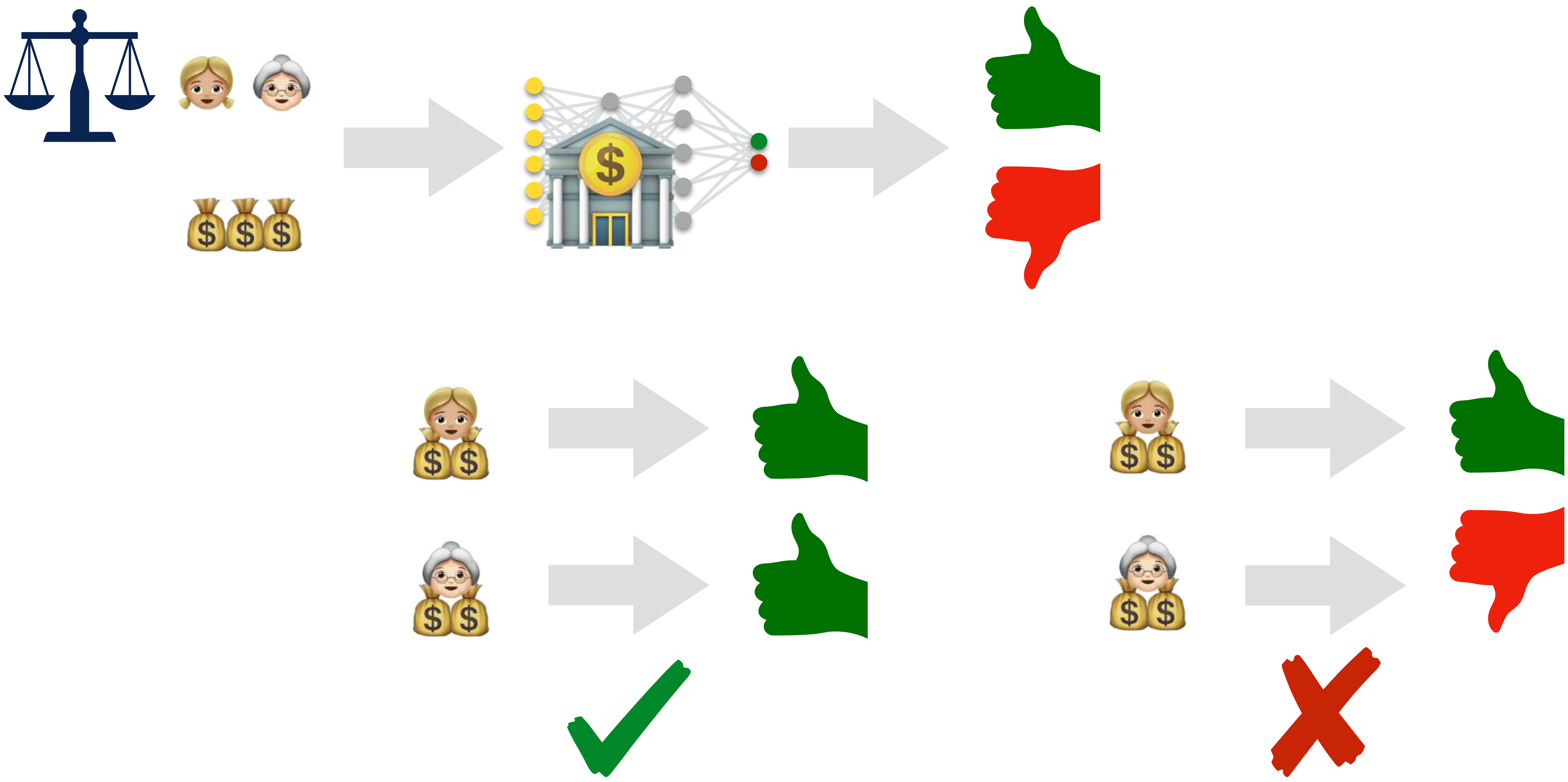
ML Impacts Our Society




Translation tutorial:
21 fairness definitions and their politics
Arvind Narayanan
@random_walker
0:05 / 55:20
Tutorial: 21 fairness definitions and their politics
19,759 views • Mar 1, 2018
196
6
SHARE
SAVE
Arvind Narayanan
226 subscribers
SUBSCRIBE
Computer scientists and statisticians have devised numerous mathematical criteria to define what it means for a classifier or a model to be fair. The proliferation of these definitions represents an attempt to make technical sense of
SHOW MORE

Dependency Fairness [Galhotra17]

The **classification** is **independent of** the values of the **sensitive inputs**



Dependency Fairness

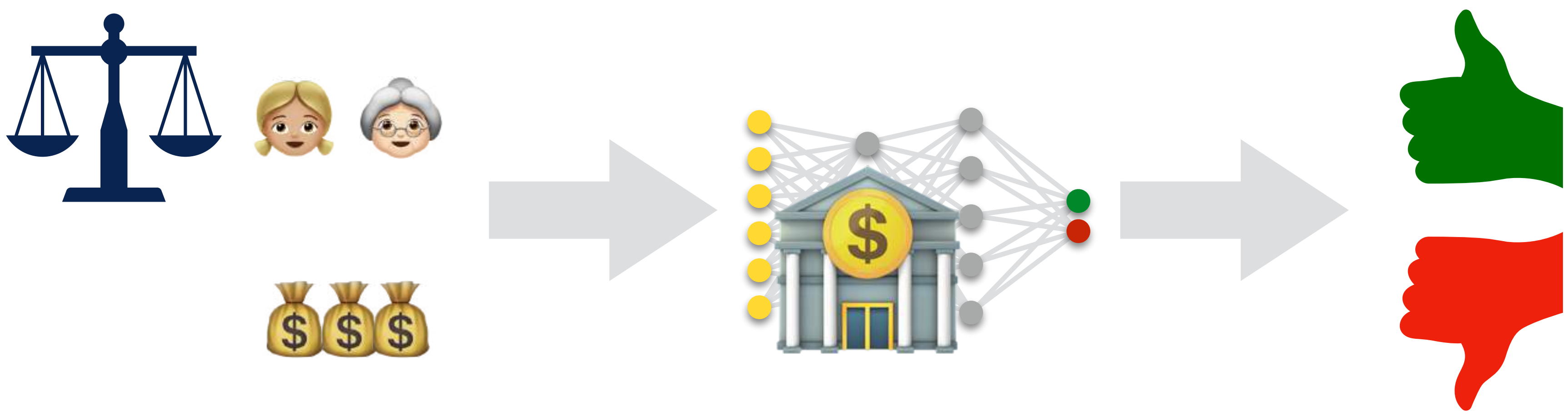
$$\mathcal{F}_i \overset{\text{def}}{=} \{[\![M]\!] \in \mathcal{P}(\Sigma^*) \mid \text{UNUSED}_i([\![M]\!])\}$$

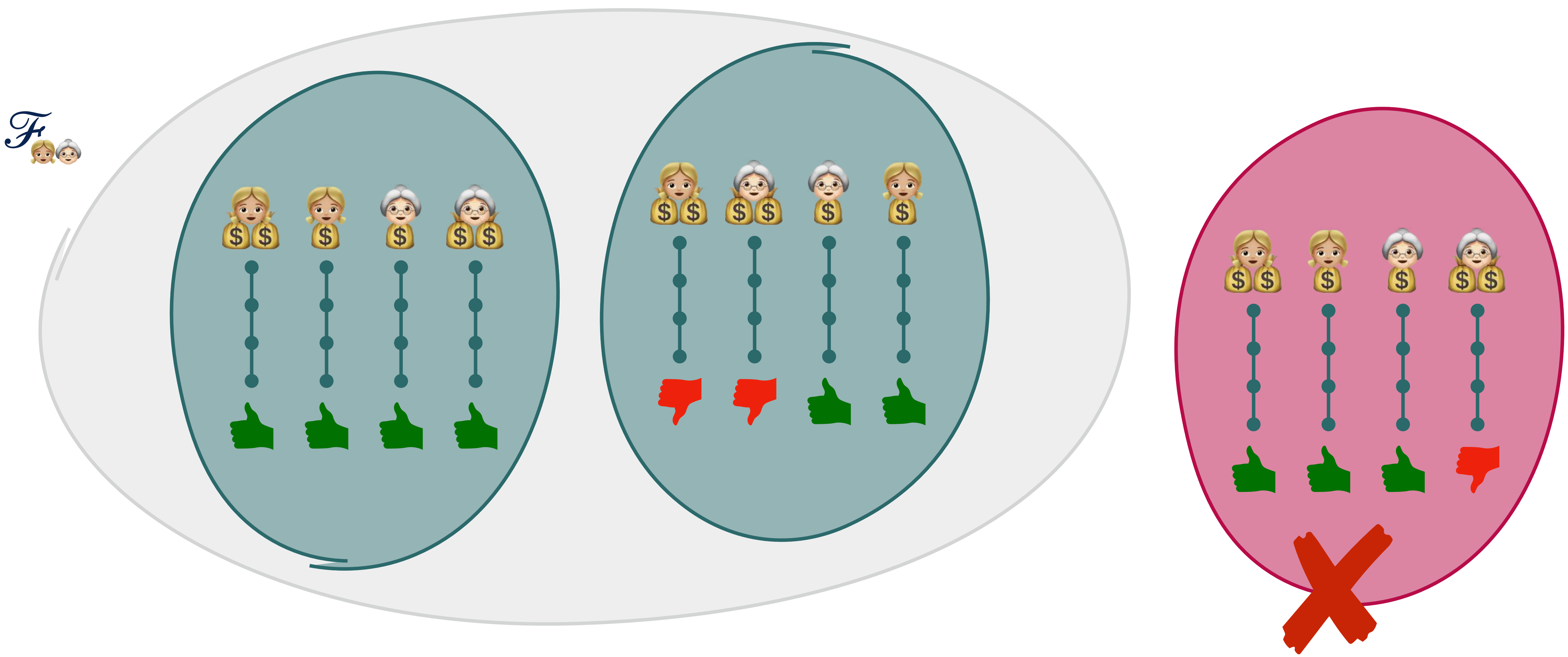
$\mathcal{F}_i$ is the set of all neural networks M (or, rather, their semantics $[\![M]\!]$) that **do not use** the value of the sensitive input node $x_{0,i}$ for classification

$$\begin{aligned}\text{UNUSED}_i([\![M]\!]) \overset{\text{def}}{=} \; & \forall t \in [\![M]\!], v \in \mathcal{R} : t_0(x_{0,i}) \neq v \Rightarrow \exists t' \in [\![M]\!] : \\ & (\forall 0 \leq j \leq |L_0| : j \neq i \Rightarrow t_0(x_{0,j}) = t'_0(x_{0,j})) \\ & \wedge t'_0(x_{0,i}) = v \\ & \wedge \max_j t_\omega(x_{N,j}) = \max_j t'_\omega(x_{N,j})\end{aligned}$$

Intuitively: **any possible classification outcome** is possible **from any value of the sensitive input** node $x_{0,i}$

Dependency Fairness





Dependency Fairness

$$\mathcal{F}_i \stackrel{\text{def}}{=} \{[\![M]\!] \in \mathcal{P}(\Sigma^*) \mid \text{UNUSED}_i([\![M]\!])\}$$

$\mathcal{F}_i$ is the set of all neural networks M (or, rather, their semantics $[\![M]\!]$) that **do not use** the value of the sensitive input node $x_{0,i}$ for classification

$$\begin{aligned}\text{UNUSED}_i([\![M]\!]) \stackrel{\text{def}}{=} \; & \forall t \in [\![M]\!], v \in \mathcal{R} : t_0(x_{0,i}) \neq v \Rightarrow \exists t' \in [\![M]\!] : \\ & (\forall 0 \leq j \leq |L_0| : j \neq i \Rightarrow t_0(x_{0,j}) = t'_0(x_{0,j})) \\ & \wedge t'_0(x_{0,i}) = v \\ & \wedge \max_j t_\omega(x_{N,j}) = \max_j t'_\omega(x_{N,j})\end{aligned}$$

Intuitively: **any possible classification outcome** is possible **from any value of the sensitive input** node $x_{0,i}$

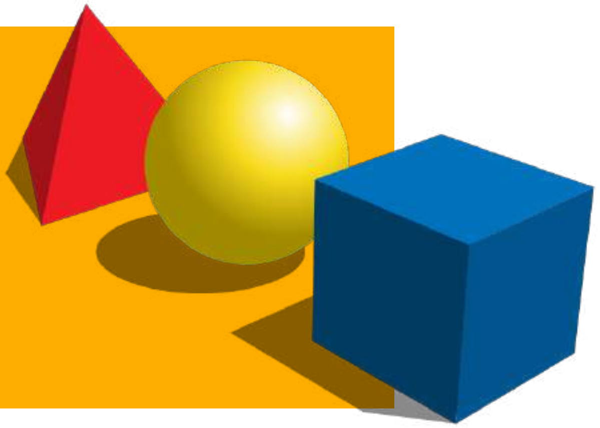
Theorem

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

Abstract Interpretation Recipe

practical tools
targeting specific programs

algorithmic approaches
to decide program properties

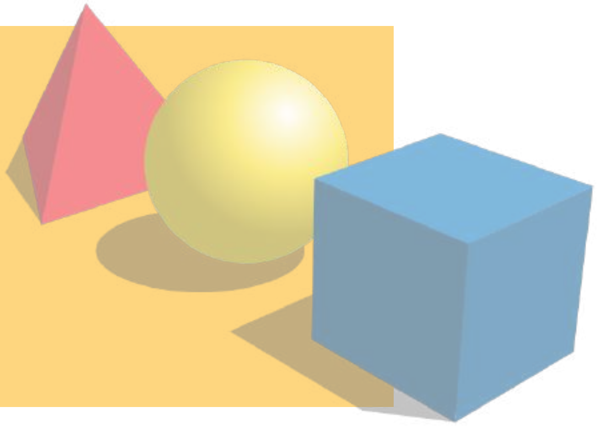


mathematical models
of the program behavior

Abstract Interpretation Recipe

practical tools
targeting specific programs

algorithmic approaches
to decide program properties

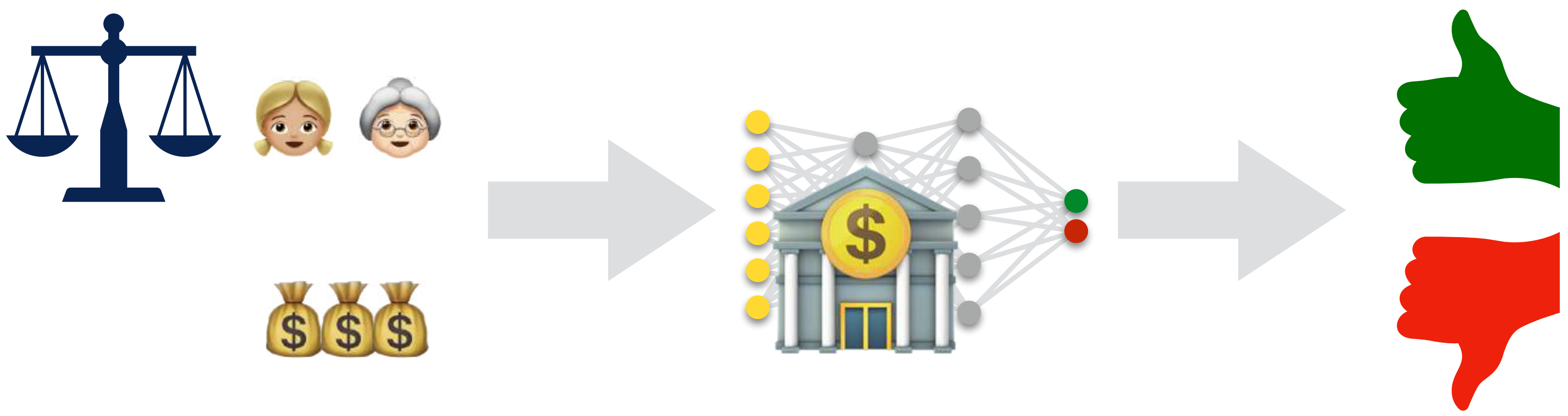


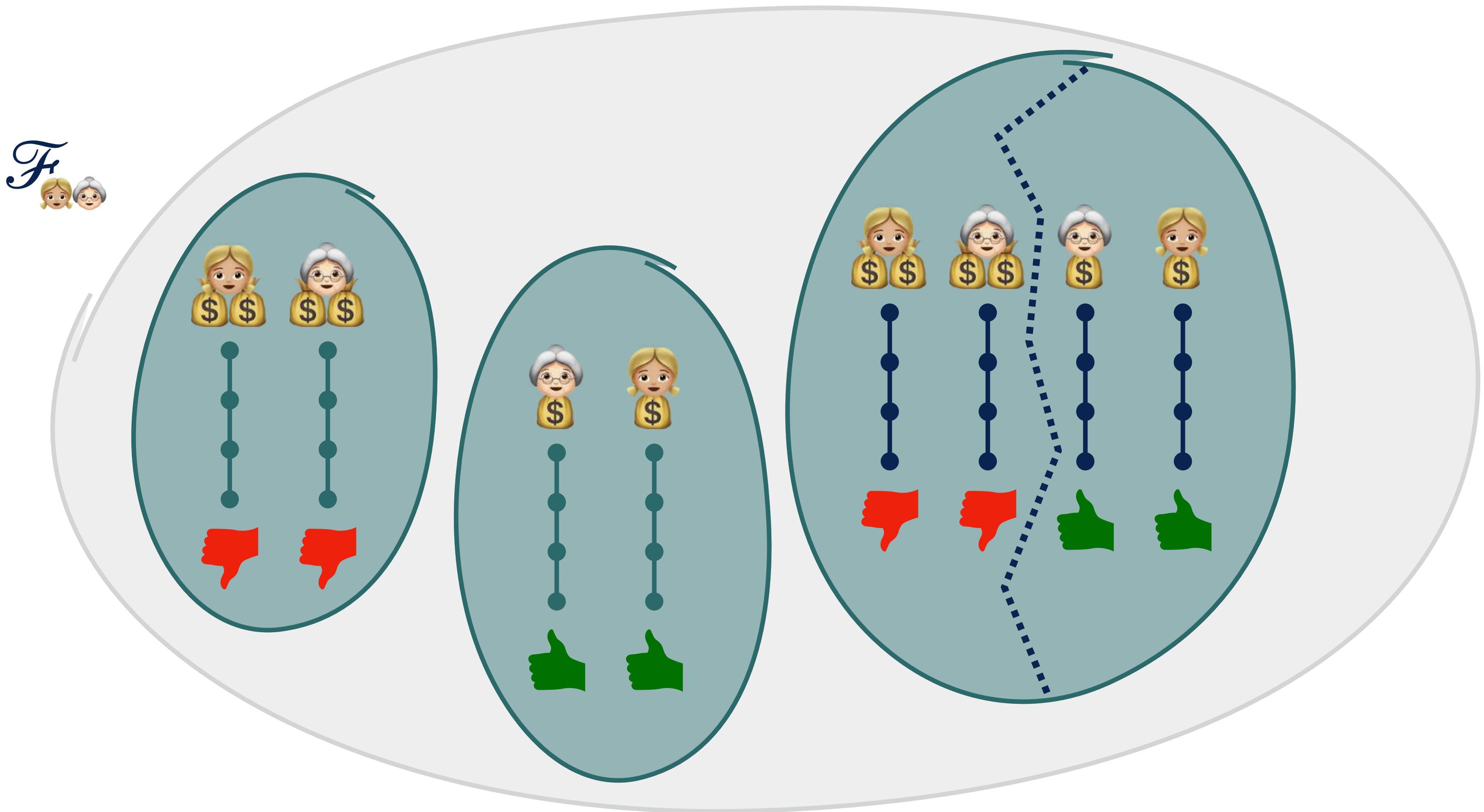
mathematical models
of the program behavior

Hierarchy of Semantics

$$
\begin{tikzcd}
\text{parallel semantics} & \{[M]\}^{\parallel}_{\rightsquigarrow} & & \\
& & [\![M]\!]_{\rightsquigarrow} \arrow[ul, "\alpha_{\parallel}"'] & \text{dependency semantics} \\
& \{[M]\}^{\parallel}_{\bullet} \arrow[uu, "\alpha_{\rightsquigarrow}"'] & & \\
& & [\![M]\!]_{\bullet} \arrow[ul, "\alpha_{\parallel}"'] \arrow[uu, "\alpha_{\rightsquigarrow}"'] & \text{outcome semantics} \\
& \{[M]\}^{\parallel} \arrow[uu, "\alpha_{\bullet}"'] & & \\
& & \{[\![M]\!]\} \arrow[ul, "\alpha_{\parallel}"'] \arrow[uu, "\alpha_{\bullet}"'] & \text{collecting semantics}
\end{tikzcd}
$$

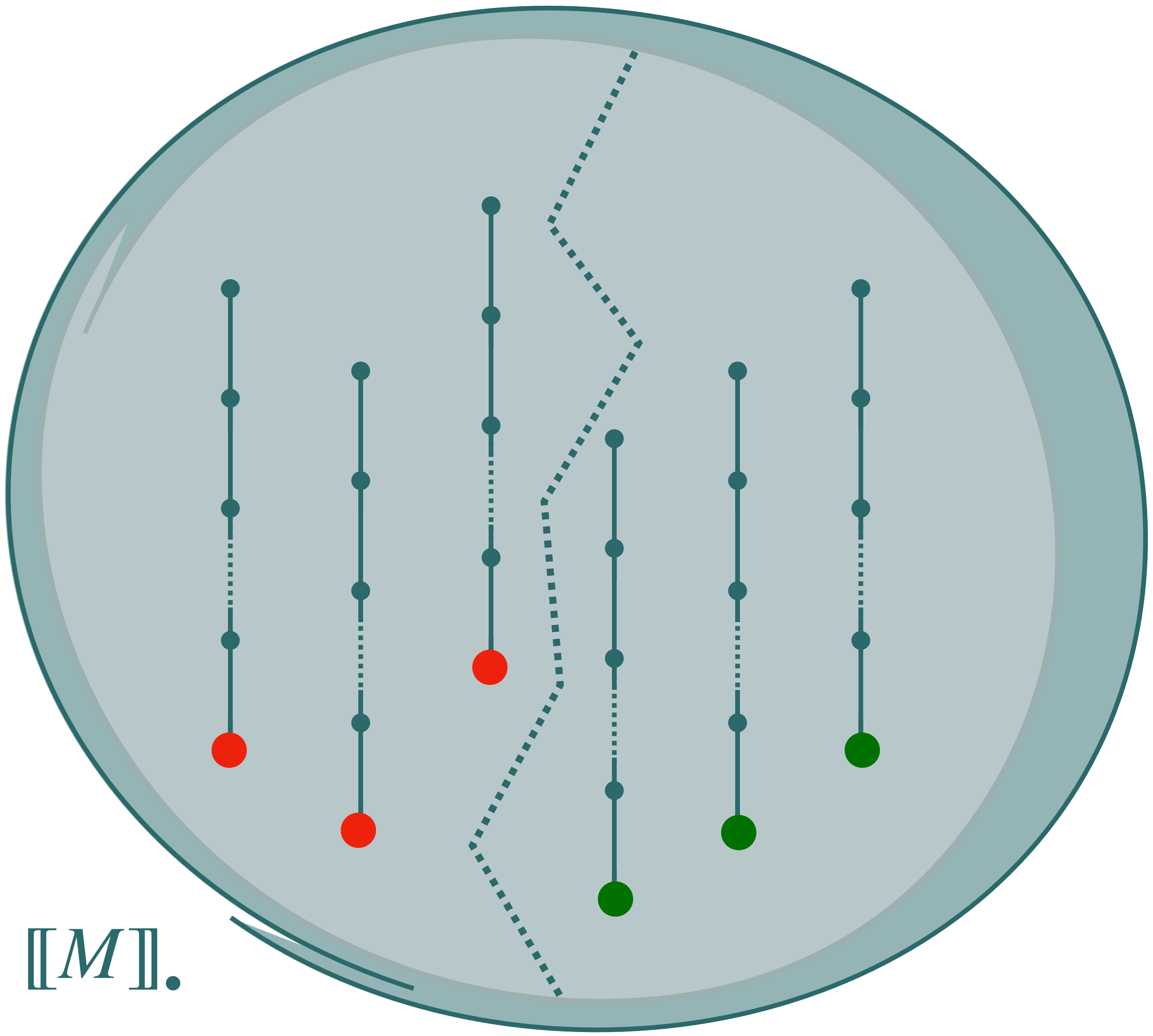
Outcome Semantics





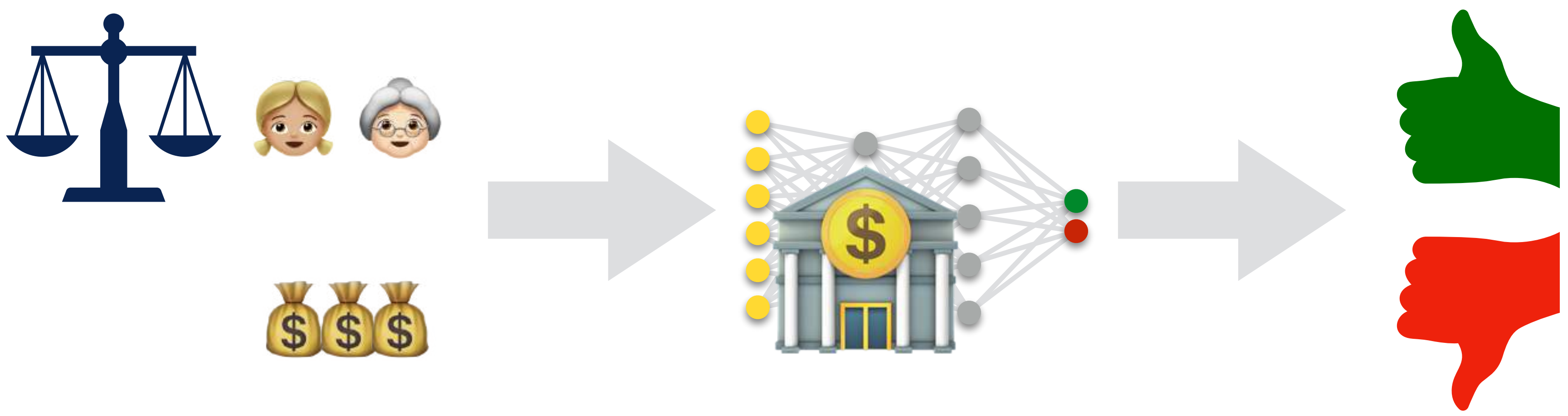
💡 **partitioning** a set of traces that satisfies dependency fairness **with respect to the program outcome** yields sets of traces that also satisfy dependency fairness

Outcome Semantics

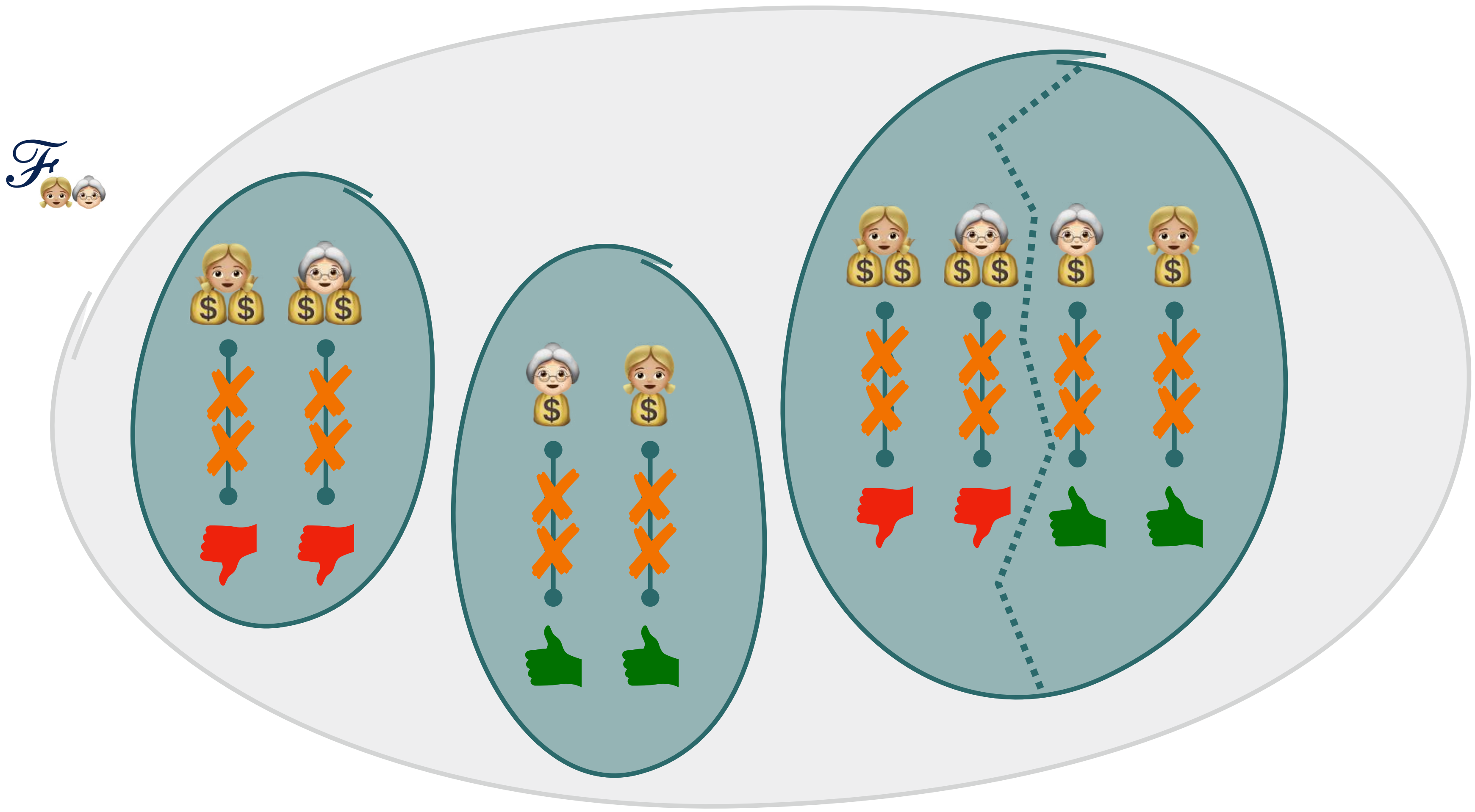


💡 **partitioning** a set of traces that satisfies dependency fairness **with respect to the program outcome** yields sets of traces that also satisfy dependency fairness

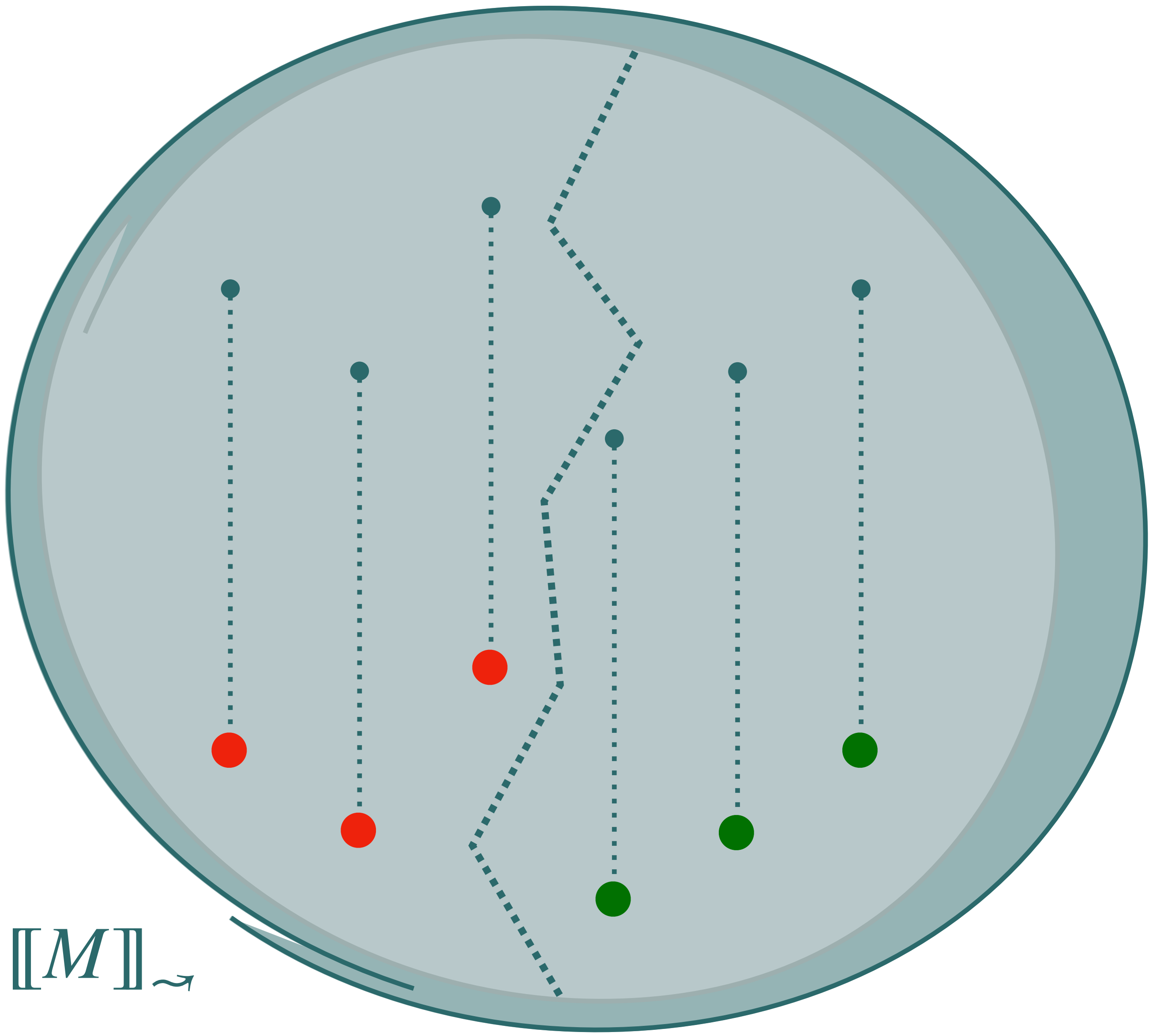
Dependency Semantics



💡 to reason about dependency fairness **we do not need to consider all intermediate computations** between the initial and final states of a trace (if any)

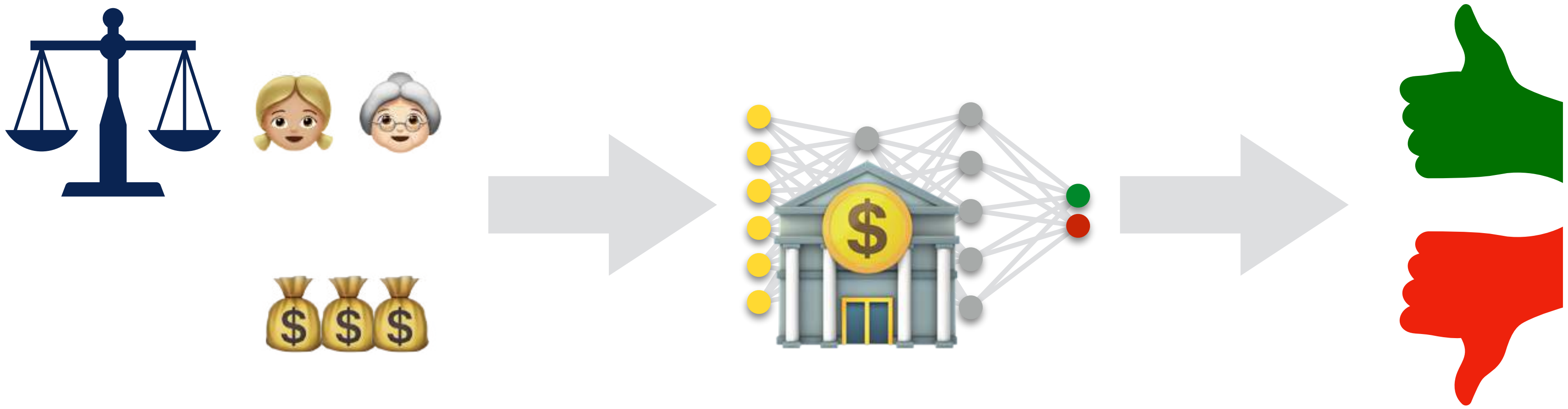


Dependency Semantics

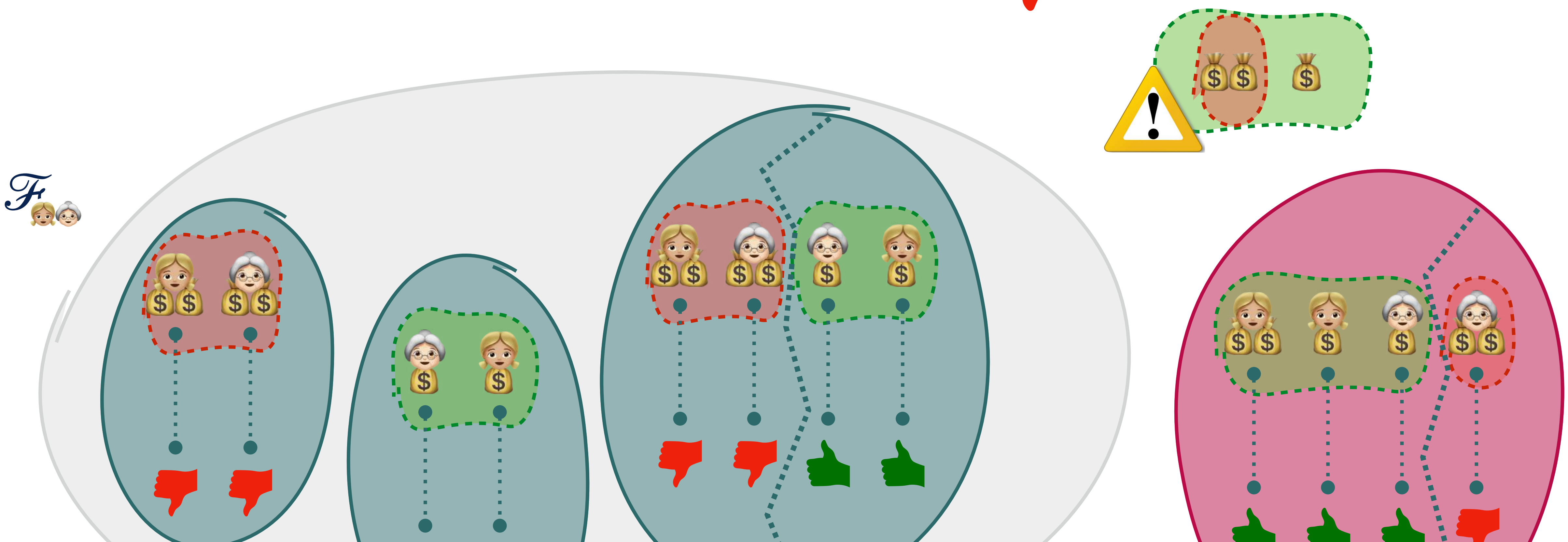


💡 to reason about dependency fairness **we do not need to consider all intermediate computations** between the initial and final states of a trace (if any)

Dependency Semantics



💡 **partitioning with respect to the outcome** classification **induces a partition of the** space of **values** of the input nodes ***used*** **for classification**



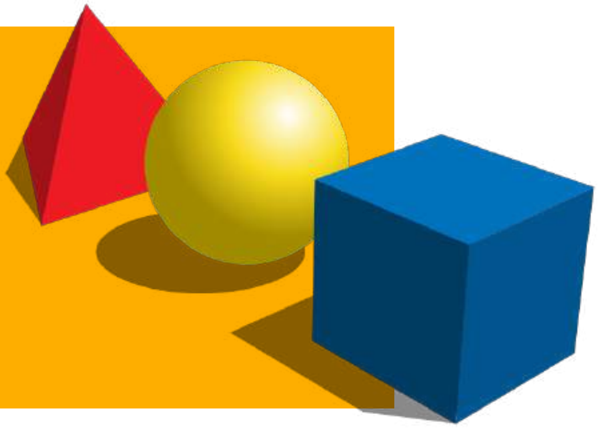
Lemma

$$M \models \mathscr{F}_i \Leftrightarrow \forall A, B \in [\![M]\!]_{\rightsquigarrow} : (A_\omega \neq B_\omega \Rightarrow A_0|_{\neq i} \cap B_0|_{\neq i} = \emptyset)$$

Abstract Interpretation Recipe

practical tools
targeting specific programs

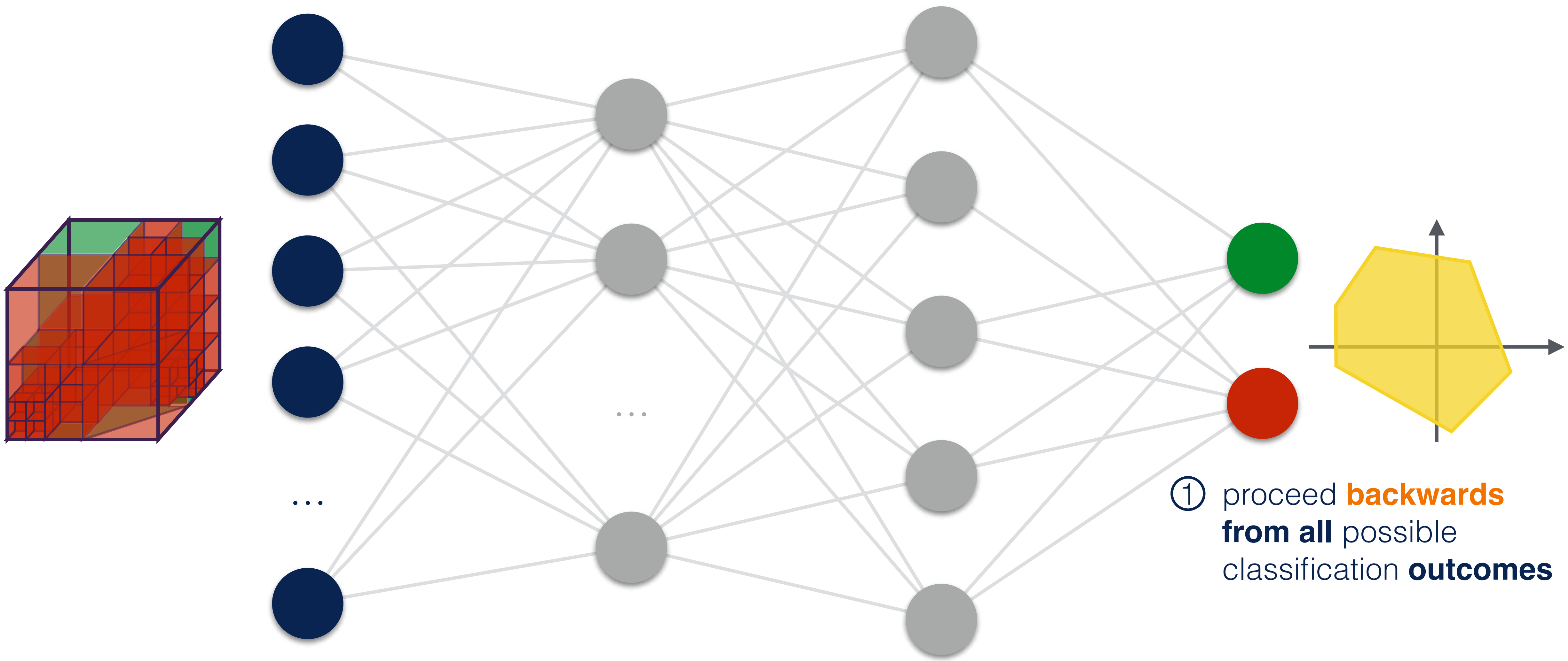
algorithmic approaches
to decide program properties



mathematical models
of the program behavior

Naïve Backward Analysis

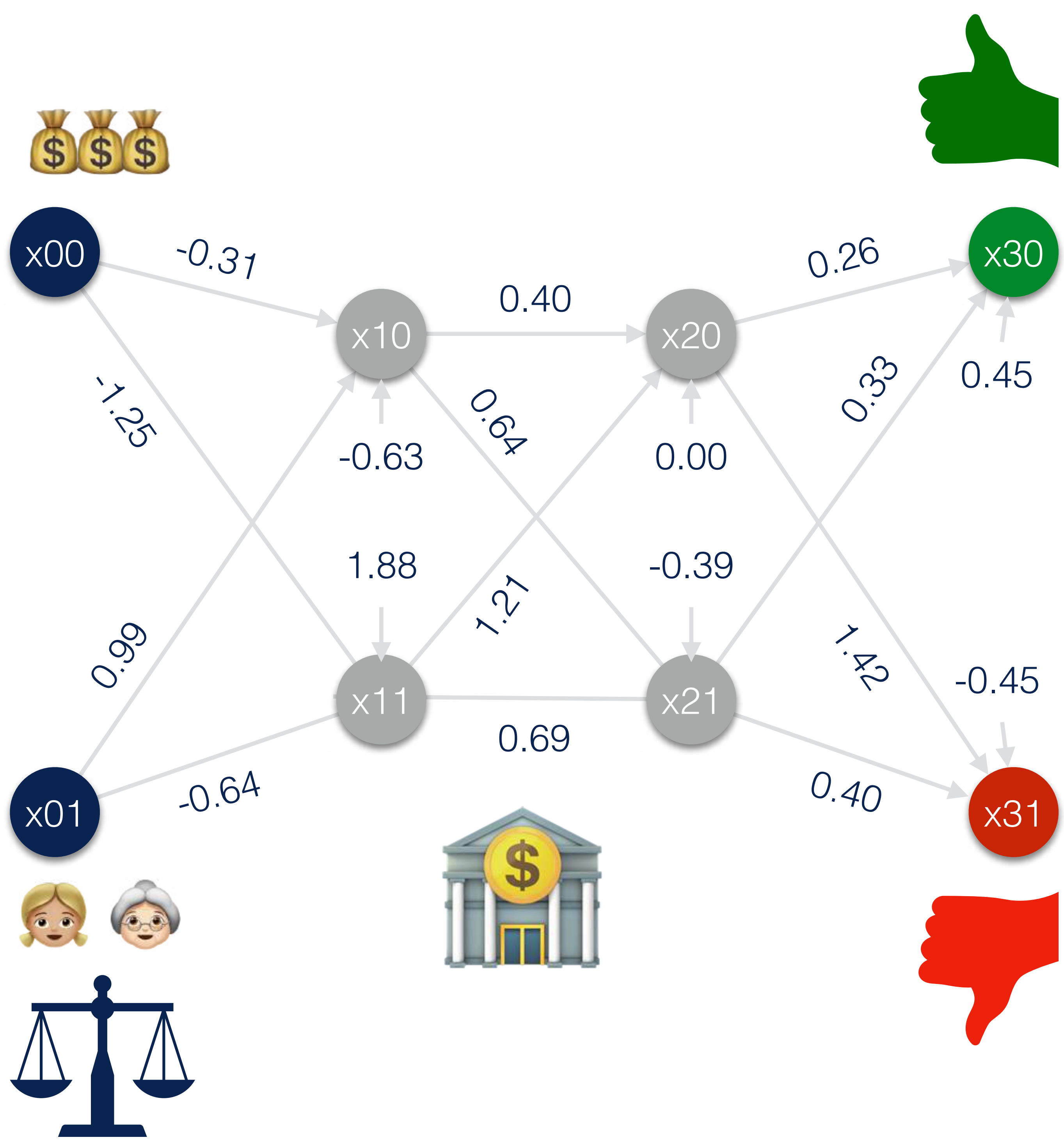
② **forget** the values of the **sensitive input** nodes



① proceed **backwards from all** possible classification **outcomes**

③ check for **intersection**:
empty → ✓ **fair**
otherwise → 🚨 **alarm**

Naïve Backward Analysis



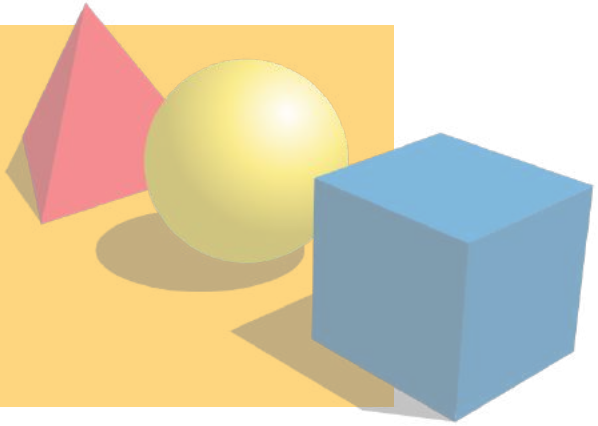

```
x00 = input()
x01 = input()
x10 = -0.31 * x00 + 0.99 * x01 + (-0.63)
x11 = -1.25 * x00 + (-0.64) * x01 + 1.88
x10 = 0 if x10 < 0 else x10
x11 = 0 if x11 < 0 else x11
x20 = 0.40 * x10 + 1.21 * x11 + 0.00
x21 = 0.64 * x10 + 0.69 * x11 + (-0.39)
x20 = 0 if x20 < 0 else x20
x21 = 0 if x21 < 0 else x21
1.16 * x20 + 0.07 * x21 ≤ 0.90    1.16 * x20 + 0.07 * x21 ≥ 0.90
x30 = 0.26 * x20 + 0.33 * x21 + 0.45
X31 = 1.42 * x20 + 0.40 * x21 + (-0.45)
x30 ≥ x31    x31 ≥ x30
return ‘ ’ if x31 < 30 else ‘ ’
```

too many disjunctions!

Abstract Interpretation Recipe

practical tools
targeting specific programs

algorithmic approaches
to decide program properties



mathematical models
of the program behavior

Hierarchy of Semantics

parallel semantics

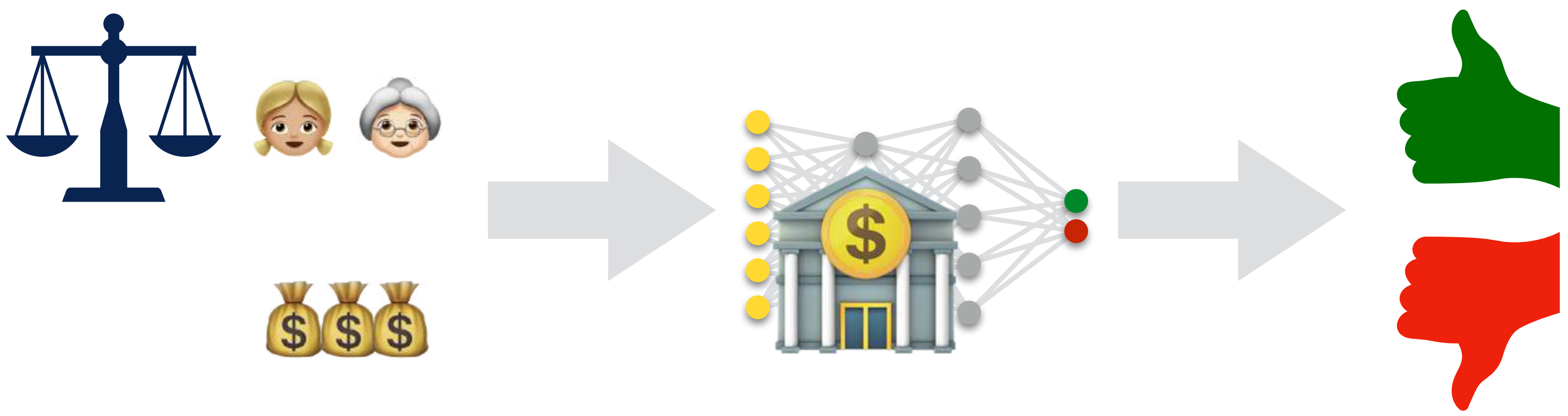
$$\begin{tikzcd}
\{[M]\}^{\square}_{\rightsquigarrow} & \\
& [\![M]\!]_{\rightsquigarrow} \arrow[ul, "\alpha_{\square}"'] \\
\{[M]\}^{\square}_{\bullet} \arrow[uu, "\alpha_{\rightsquigarrow}"] & \\
& [\![M]\!]_{\bullet} \arrow[uu, "\alpha_{\rightsquigarrow}"] \arrow[ul, "\alpha_{\square}"'] \\
\{[M]\}^{\square} \arrow[uu, "\alpha_{\bullet}"] & \\
& \{[\![M]\!]\} \arrow[uu, "\alpha_{\bullet}"] \arrow[ul, "\alpha_{\square}"']
\end{tikzcd}$$

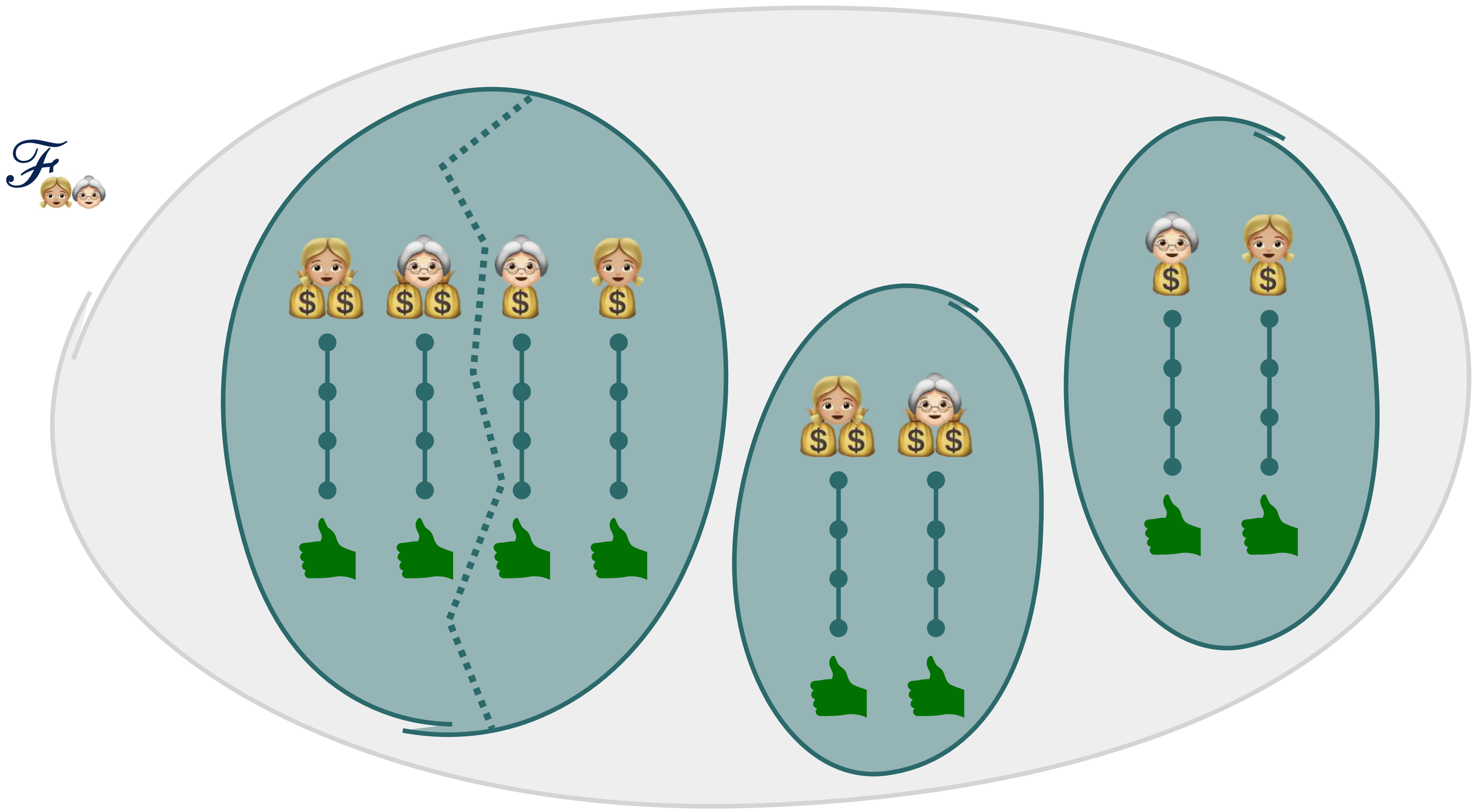
dependency semantics

outcome semantics

collecting semantics

Parallel Semantics

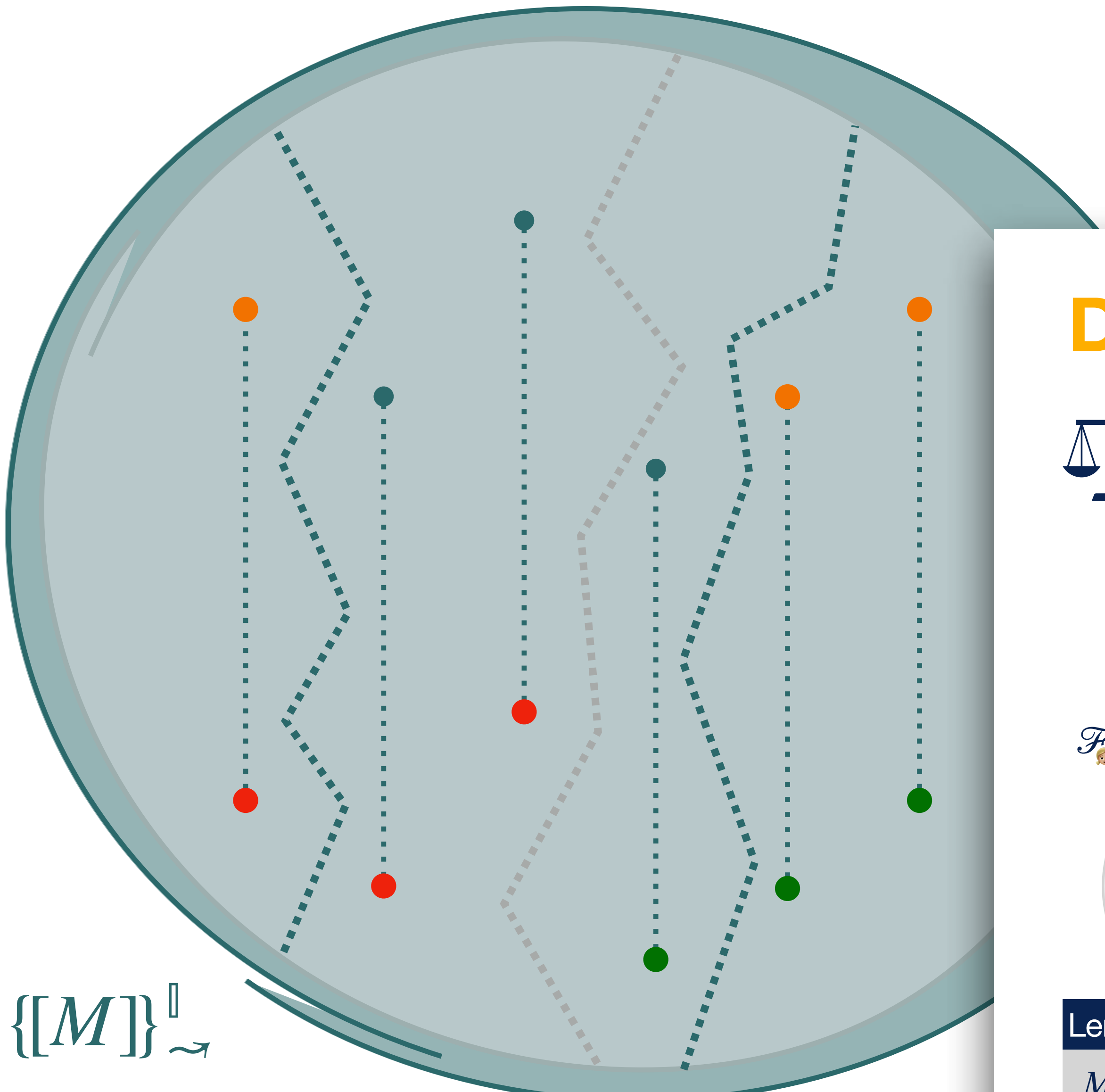


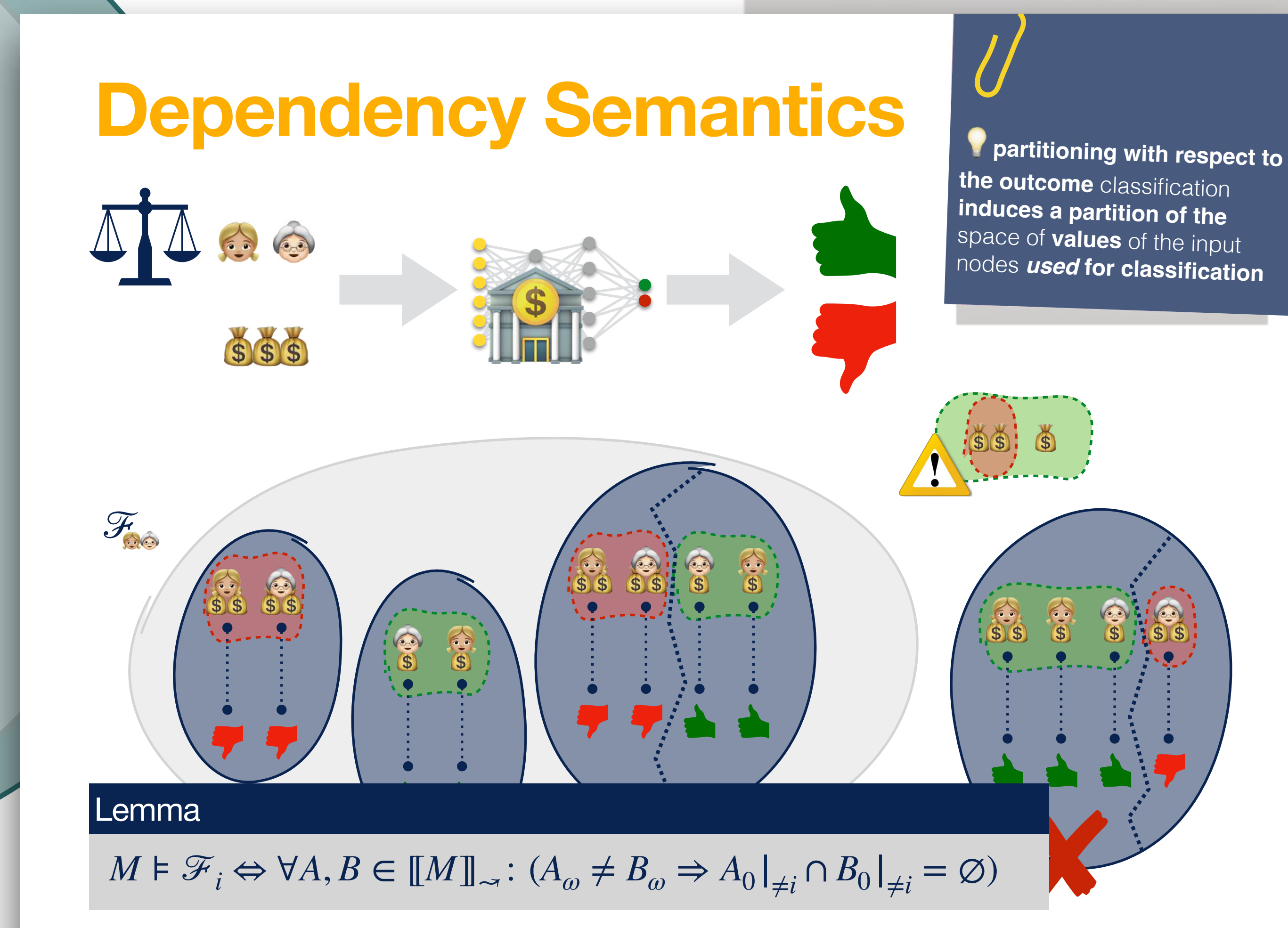


💡 **partitioning** a set of traces that satisfies dependency fairness **with respect to the non-sensitive inputs** yields sets of traces that also satisfy dependency fairness

Parallel Semantics

💡 **partitioning** a set of traces that satisfies dependency fairness **with respect to the non-sensitive inputs** yields sets of traces that also satisfy dependency fairness





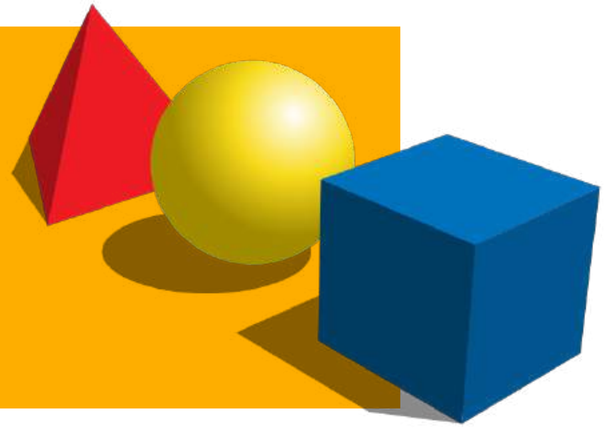
Lemma

$$M \models \mathcal{F}_i \Leftrightarrow \forall I \in \mathbb{I}: \forall A, B \in \{[M]\}^{\mathbb{I}}_{\rightsquigarrow}: (A^I_\omega \neq B^I_\omega \Rightarrow A^I_0|_{\neq i} \cap B^I_0|_{\neq i} = \emptyset)$$

Abstract Interpretation Recipe

practical tools
targeting specific programs

algorithmic approaches
to decide program properties



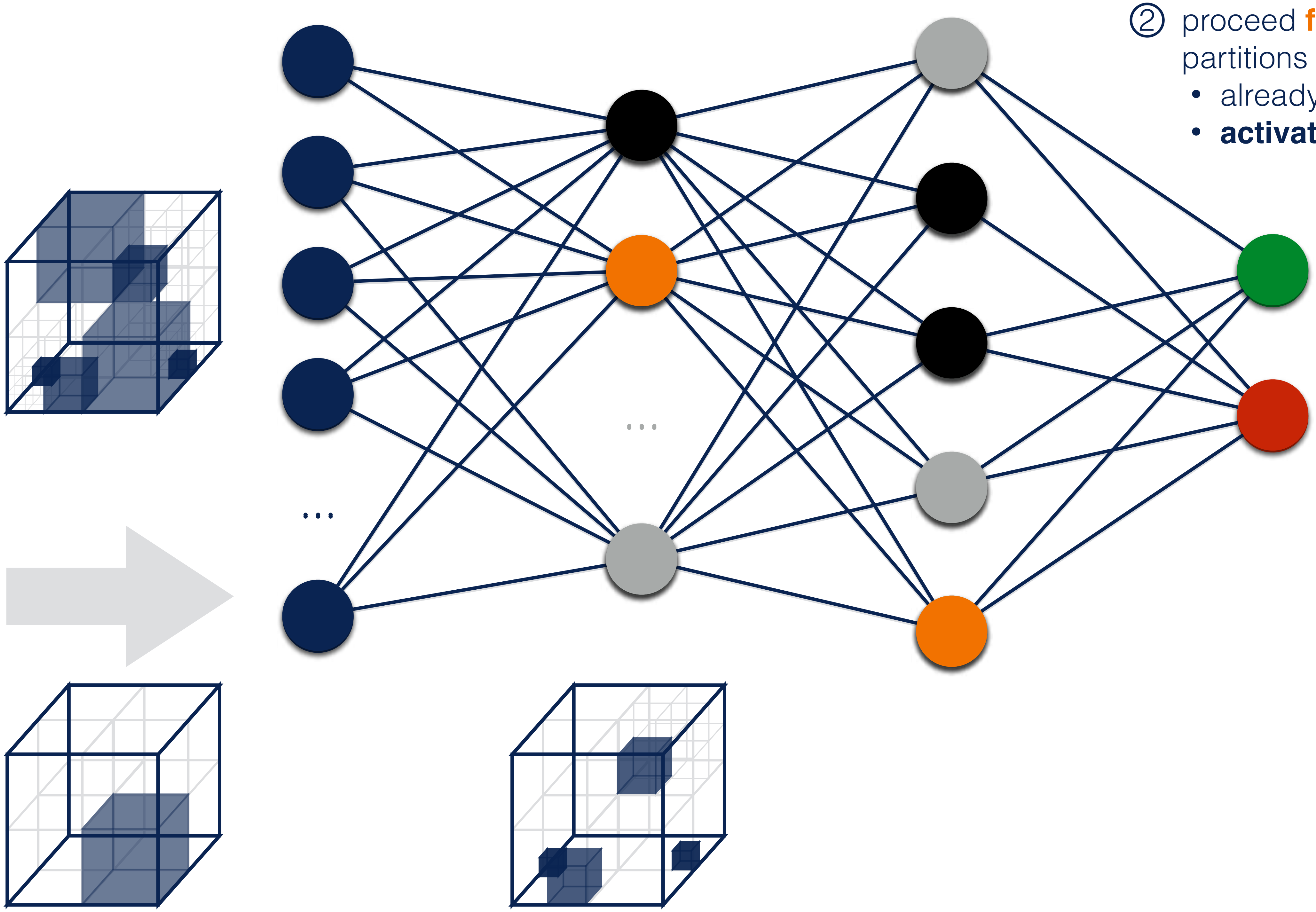
mathematical models
of the program behavior

Forward and Backward Analysis

① **partition** the space of values of the **non-sensitive input** nodes

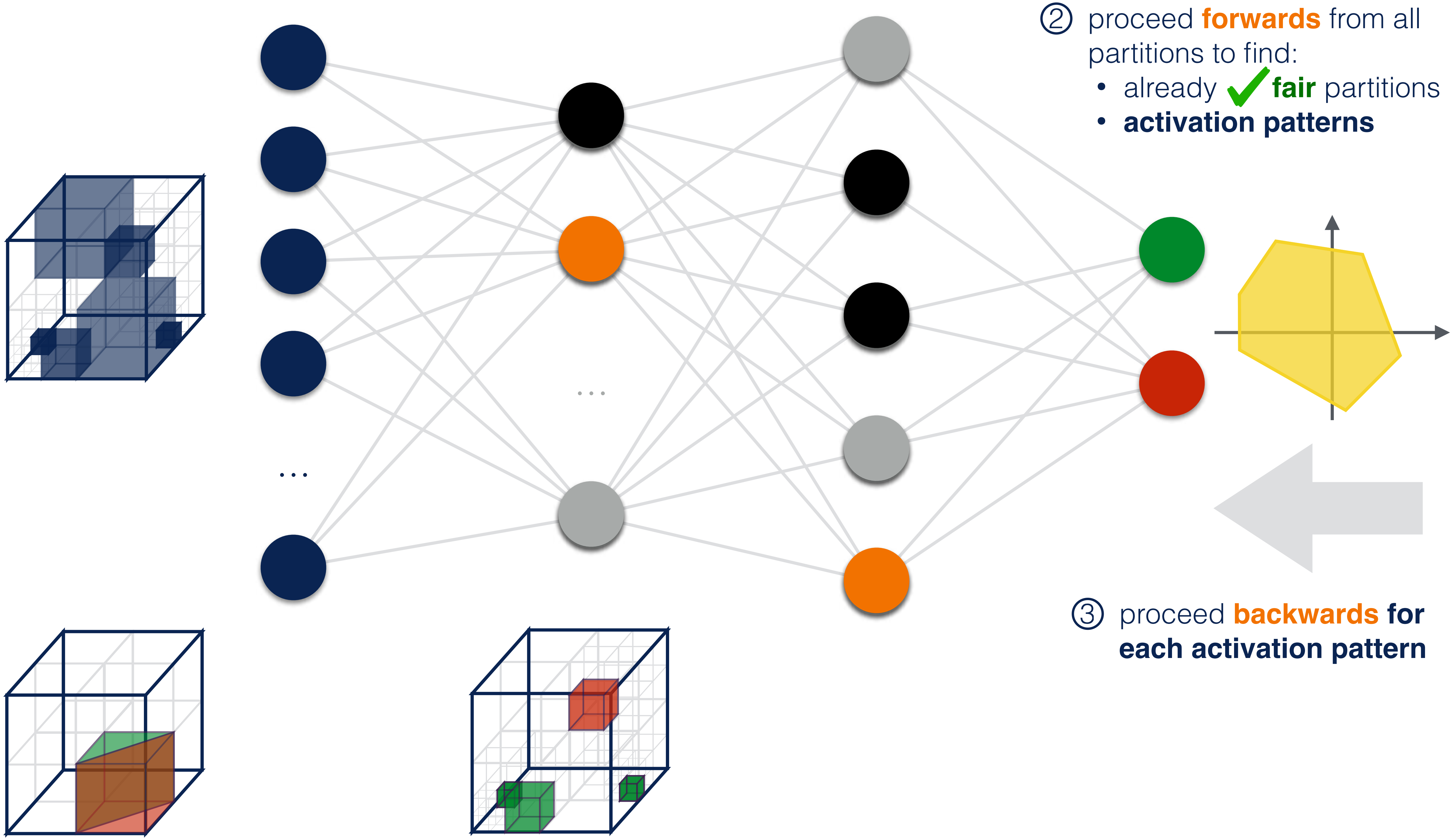
② proceed **forwards** from all partitions to find:

- already ✓ **fair** partitions
- **activation patterns**



Forward and Backward Analysis

① **partition** the space of values of the **non-sensitive input** nodes



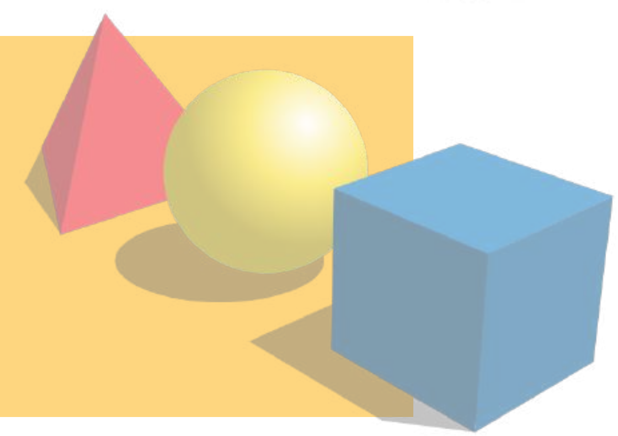
② proceed **forwards** from all partitions to find:
- already ✓ **fair** partitions
- **activation patterns**

③ proceed **backwards for each activation pattern**

Abstract Interpretation Recipe

practical tools
targeting specific programs

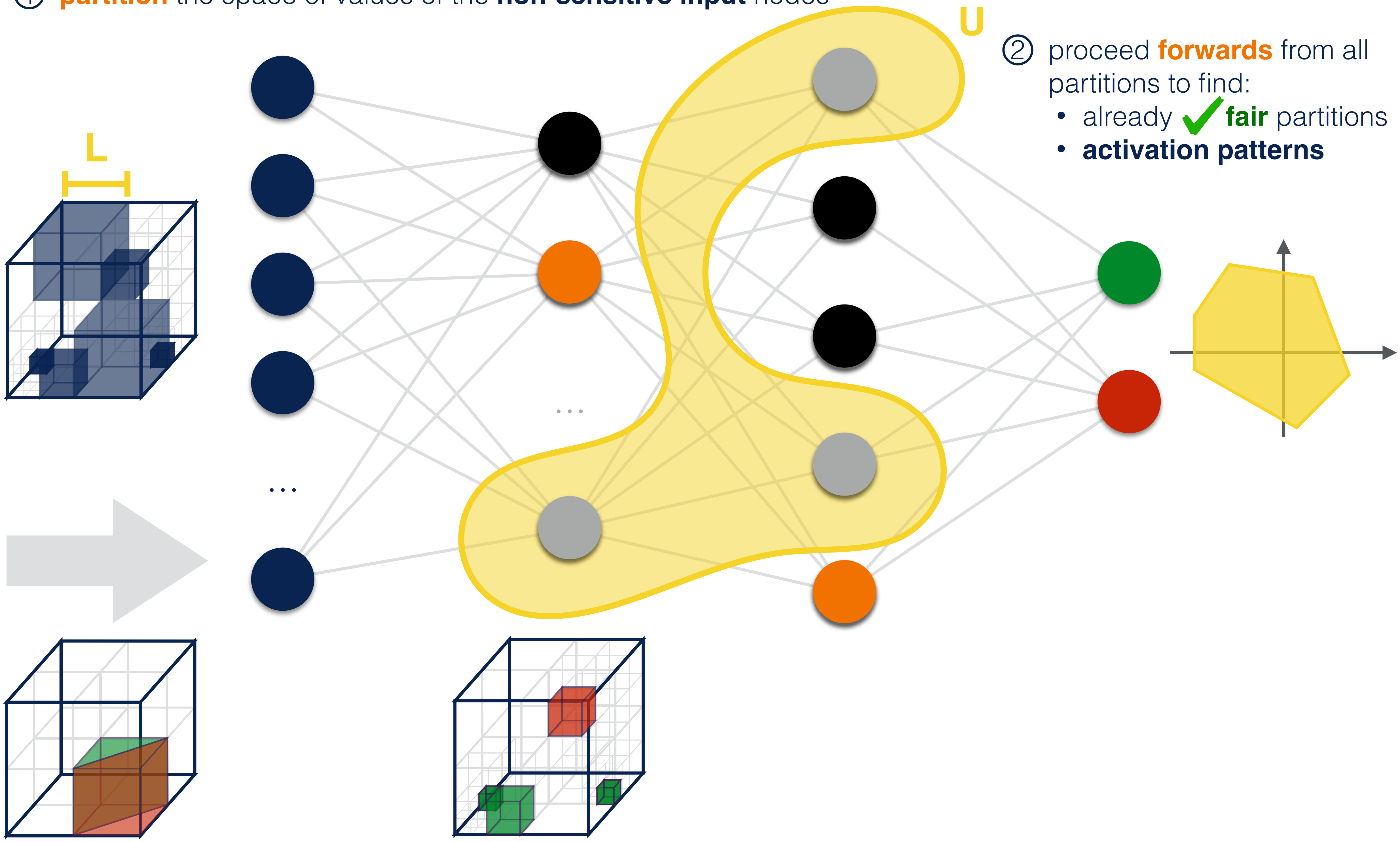
algorithmic approaches
to decide program properties



mathematical models
of the program behavior

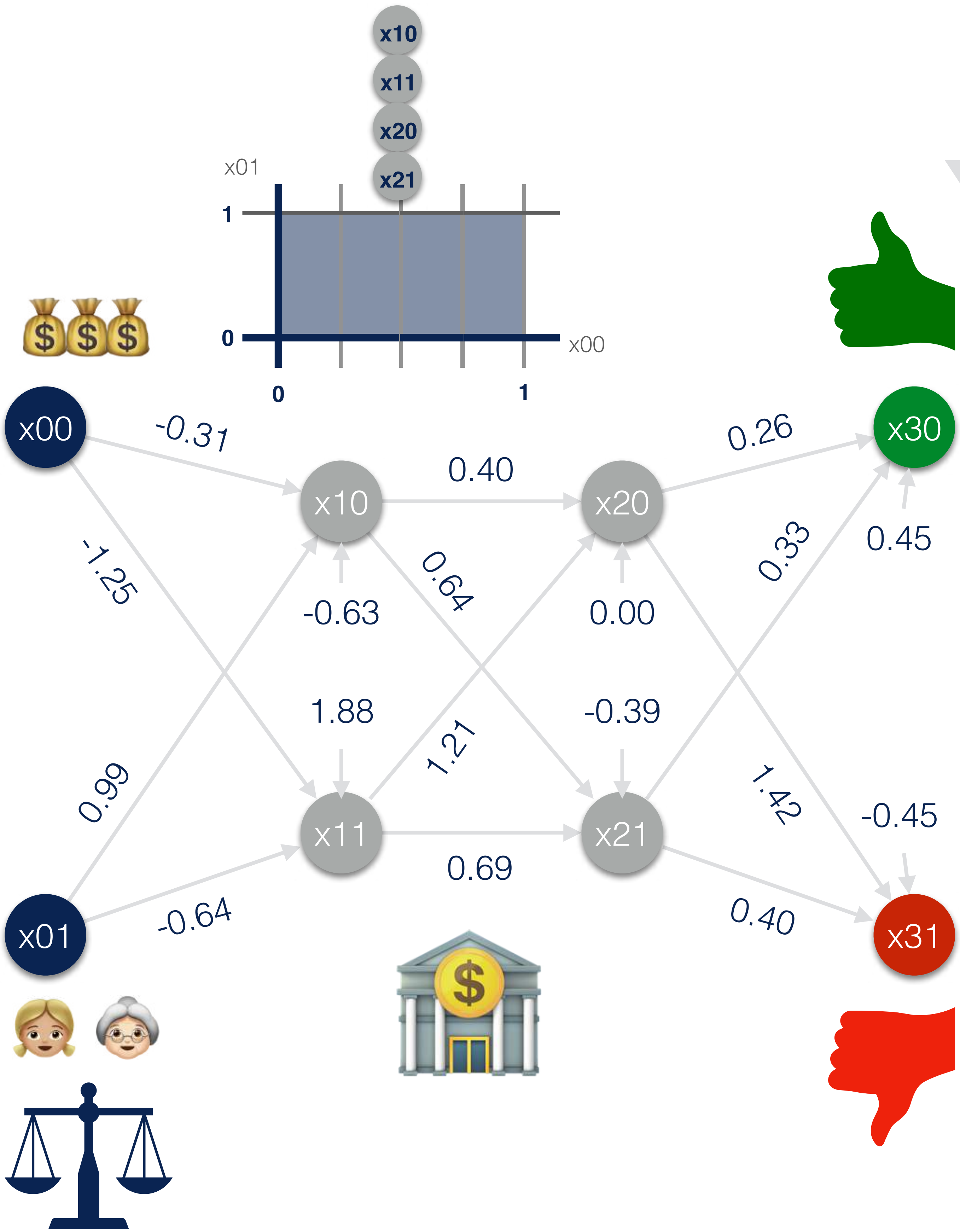
Iterative Forward Analysis

① **partition** the space of values of the **non-sensitive input** nodes



② proceed **forwards** from all partitions to find:

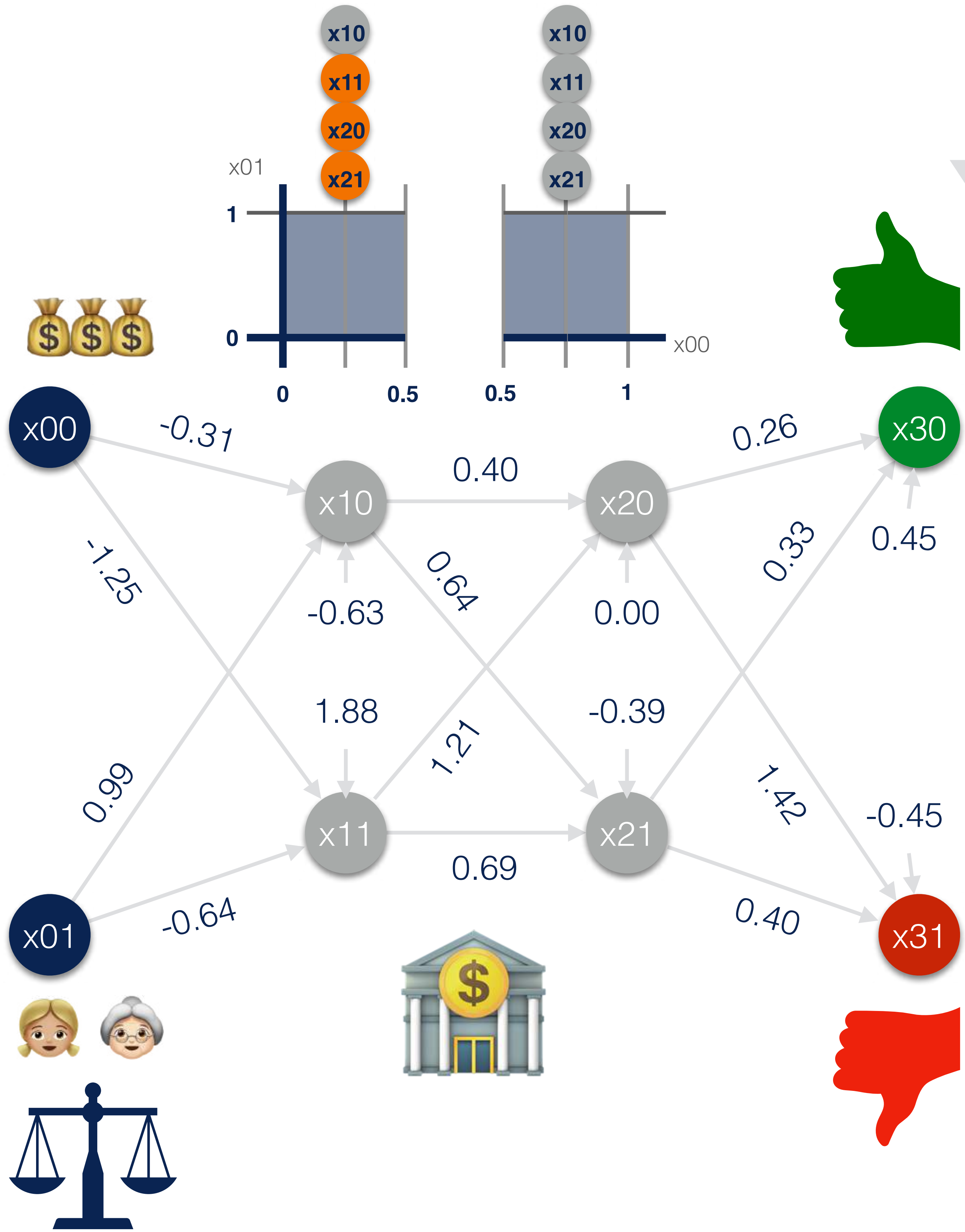
- already ✓ **fair** partitions
- **activation patterns**



L = 0.25

U = 2

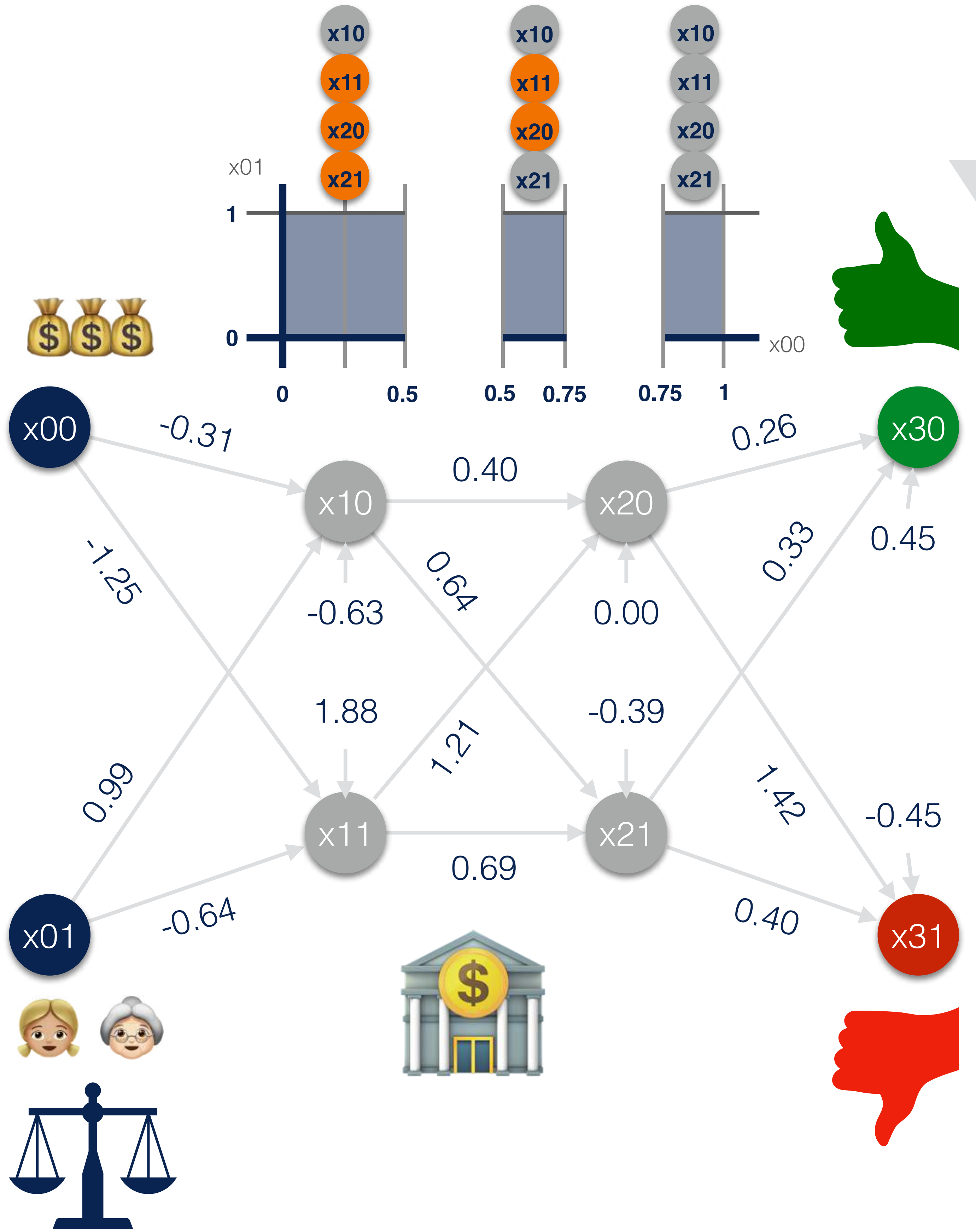
```
x00 = input()
x01 = input()

x10 = -0.31 * x00 + 0.99 * x01 + (-0.63)
x11 = -1.25 * x00 + (-0.64) * x01 + 1.88

x10 = 0 if x10 < 0 else x10
x11 = 0 if x11 < 0 else x11

x20 = 0.40 * x10 + 1.21 * x11 + 0.00
x21 = 0.64 * x10 + 0.69 * x11 + (-0.39)

x20 = 0 if x20 < 0 else x20
x21 = 0 if x21 < 0 else x21

x30 = 0.26 * x20 + 0.33 * x21 + 0.45
X31 = 1.42 * x20 + 0.40 * x21 + (-0.45)

return '👍' if x31 < 30 else '👎'
```



L = 0.25

U = 2

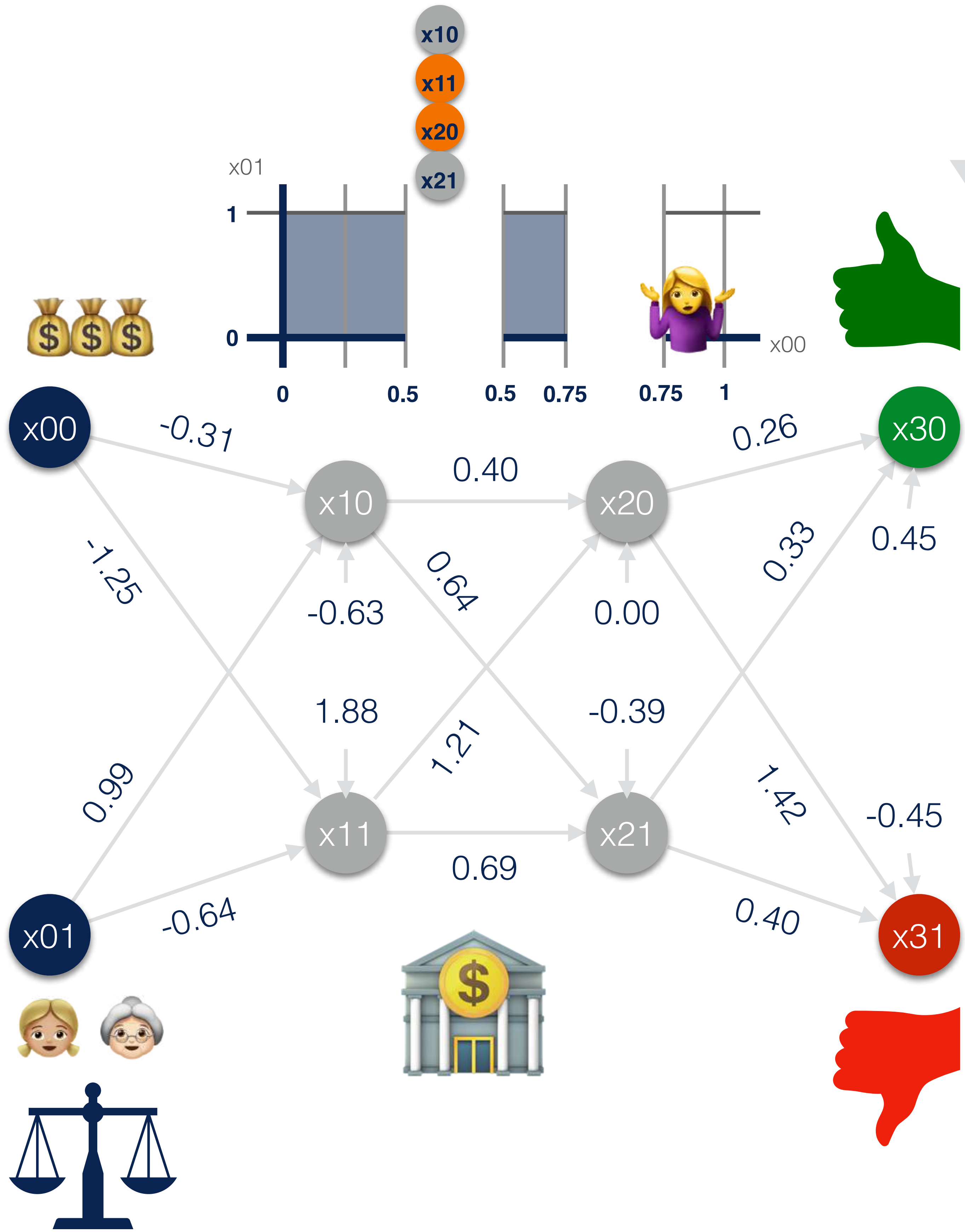
```
x00 = input()
x01 = input()

x10 = -0.31 * x00 + 0.99 * x01 + (-0.63)
x11 = -1.25 * x00 + (-0.64) * x01 + 1.88

x10 = 0 if x10 < 0 else x10
x11 = 0 if x11 < 0 else x11

x20 = 0.40 * x10 + 1.21 * x11 + 0.00
x21 = 0.64 * x10 + 0.69 * x11 + (-0.39)

x20 = 0 if x20 < 0 else x20
x21 = 0 if x21 < 0 else x21

x30 = 0.26 * x20 + 0.33 * x21 + 0.45
X31 = 1.42 * x20 + 0.40 * x21 + (-0.45)

return '👍' if x31 < 30 else '👎'
```



L = 0.25

U = 2

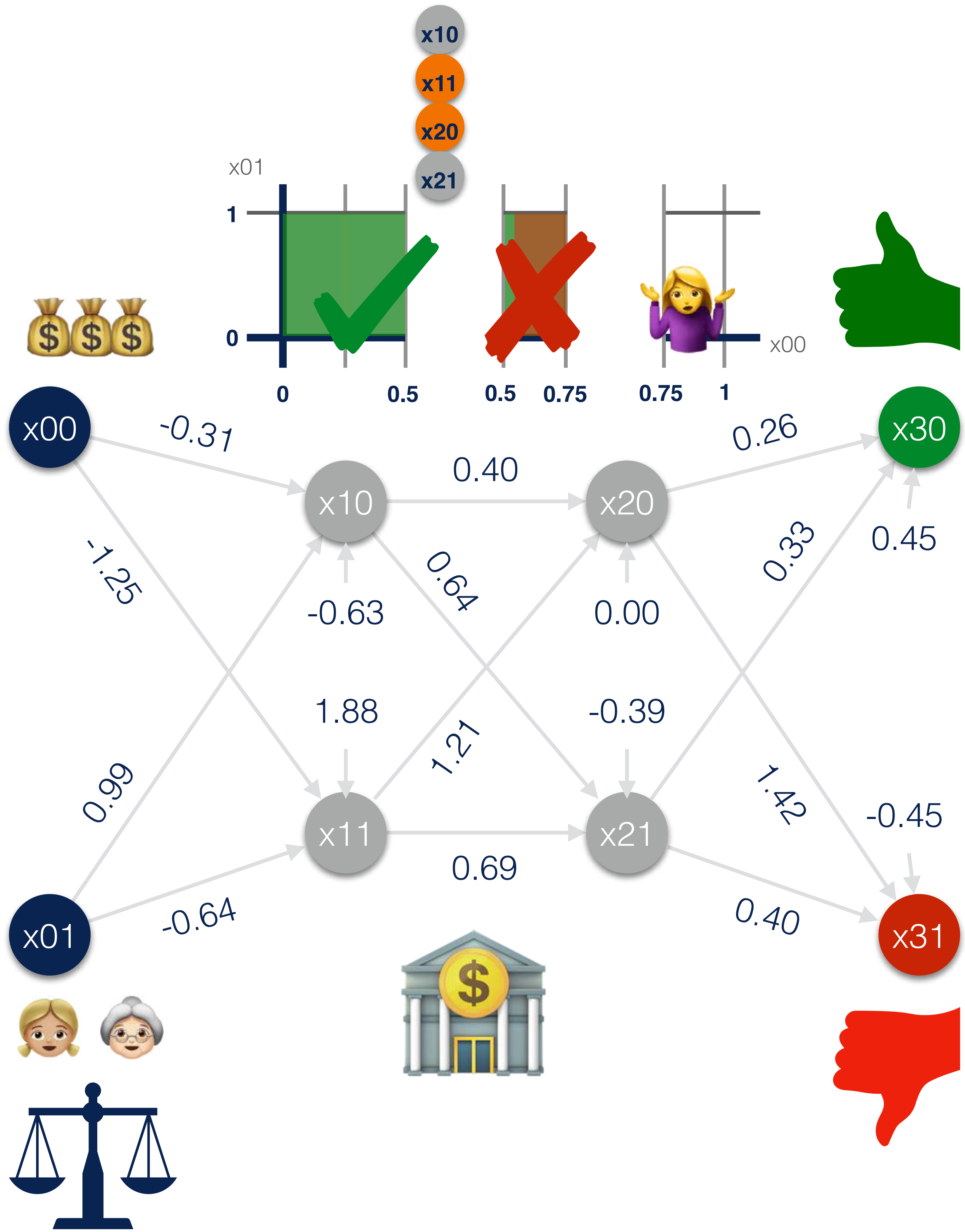
```
x00 = input()
x01 = input()

x10 = -0.31 * x00 + 0.99 * x01 + (-0.63)
x11 = -1.25 * x00 + (-0.64) * x01 + 1.88

x10 = 0 if x10 < 0 else x10
x11 = 0 if x11 < 0 else x11

x20 = 0.40 * x10 + 1.21 * x11 + 0.00
x21 = 0.64 * x10 + 0.69 * x11 + (-0.39)

x20 = 0 if x20 < 0 else x20
x21 = 0 if x21 < 0 else x21

x30 = 0.26 * x20 + 0.33 * x21 + 0.45
X31 = 1.42 * x20 + 0.40 * x21 + (-0.45)

return '👍' if x31 < 30 else '👎'
```



L = 0.25

U = 2

```
x00 = input()
x01 = input()

x10 = -0.31 * x00 + 0.99 * x01 + (-0.63)
x11 = -1.25 * x00 + (-0.64) * x01 + 1.88

x10 = 0 if x10 < 0 else x10
x11 = 0 if x11 < 0 else x11

x20 = 0.40 * x10 + 1.21 * x11 + 0.00
x21 = 0.64 * x10 + 0.69 * x11 + (-0.39)

x20 = 0 if x20 < 0 else x20
x21 = 0 if x21 < 0 else x21

x30 = 0.26 * x20 + 0.33 * x21 + 0.45
X31 = 1.42 * x20 + 0.40 * x21 + (-0.45)

return '👍' if x31 < 30 else '👎'
```

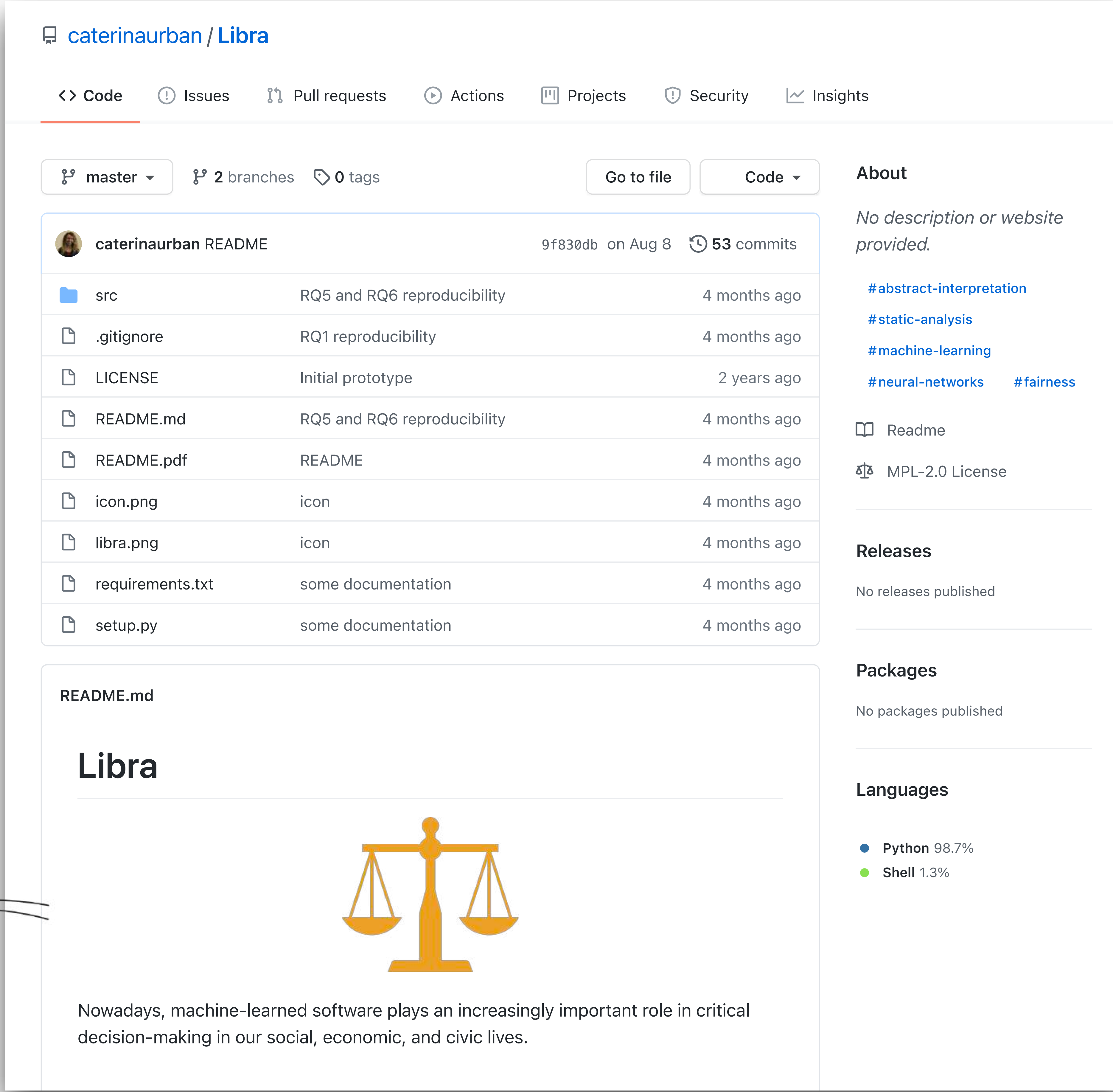


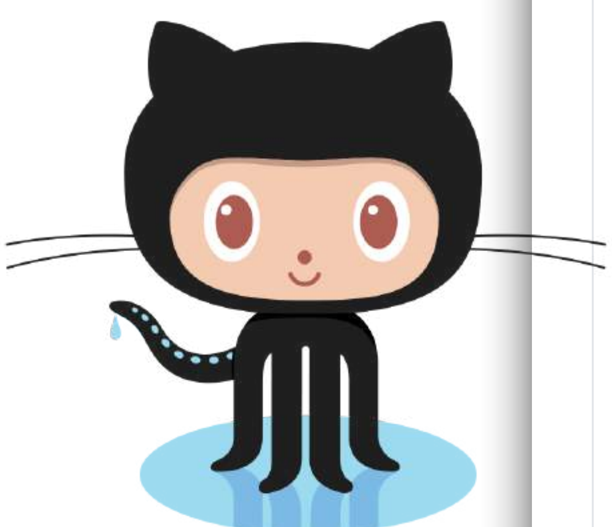
L = 0.25

U = 2

```
x00 = input()
x01 = input()

x10 = -0.31 * x00 + 0.99 * x01 + (-0.63)
x11 = -1.25 * x00 + (-0.64) * x01 + 1.88

x10 = 0 if x10 < 0 else x10
x11 = 0 if x11 < 0 else x11

x20 = 0.40 * x10 + 1.21 * x11 + 0.00
x21 = 0.64 * x10 + 0.69 * x11 + (-0.39)

x20 = 0 if x20 < 0 else x20
x21 = 0 if x21 < 0 else x21
1.16 * x20 + 0.07 * x21 ≤ 0.90    1.16 * x20 + 0.07 * x21 ≥ 0.90
x30 = 0.26 * x20 + 0.33 * x21 + 0.45
X31 = 1.42 * x20 + 0.40 * x21 + (-0.45)
x30 ≥ x31    x31 ≥ x30
return '👍' if x31 < 30 else '👎'
```

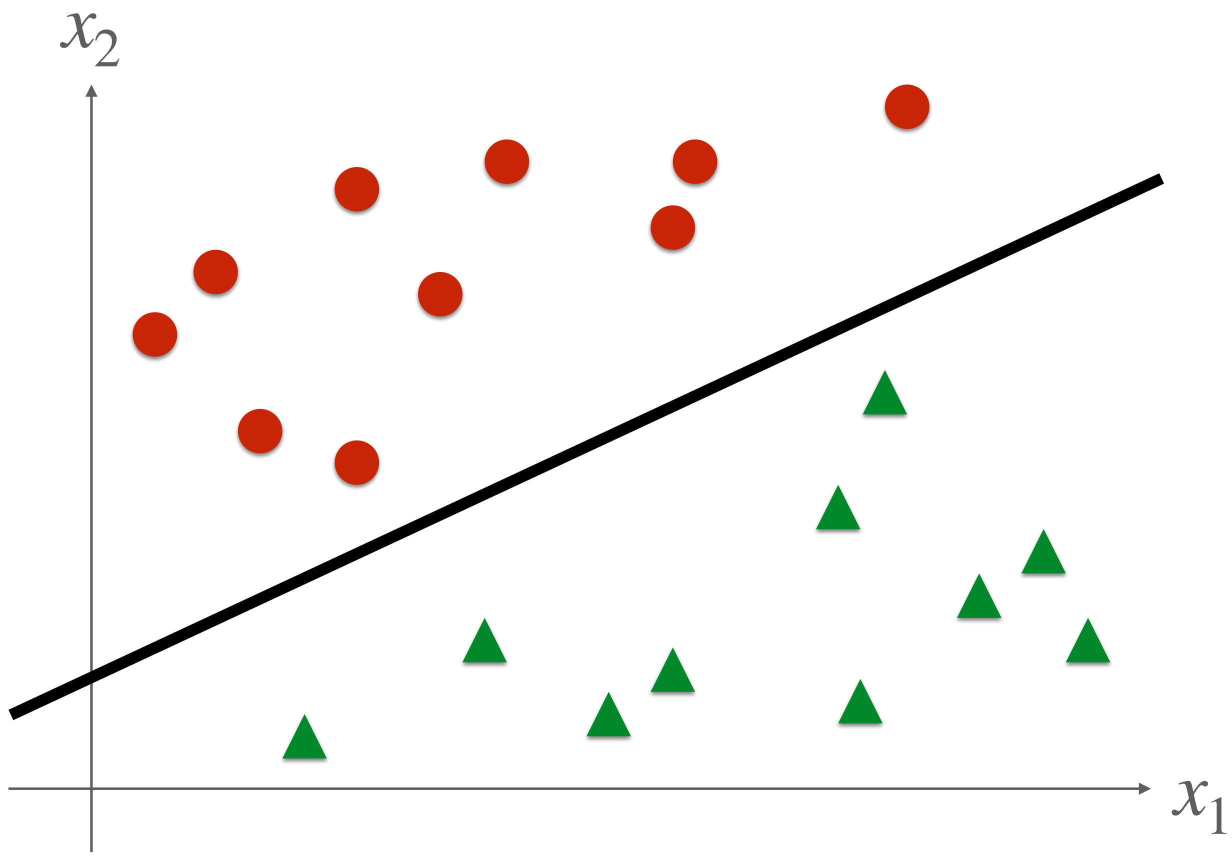
Libra


caterinaurban / Libra
Code
Issues
Pull requests
Actions
Projects
Security
Insights
master
2 branches
0 tags
Go to file
Code
caterinaurban README
9f830db on Aug 8
53 commits
src
RQ5 and RQ6 reproducibility
4 months ago
.gitignore
RQ1 reproducibility
4 months ago
LICENSE
Initial prototype
2 years ago
README.md
RQ5 and RQ6 reproducibility
4 months ago
README.pdf
README
4 months ago
icon.png
icon
4 months ago
libra.png
icon
4 months ago
requirements.txt
some documentation
4 months ago
setup.py
some documentation
4 months ago
README.md
Libra
Nowadays, machine-learned software plays an increasingly important role in critical decision-making in our social, economic, and civic lives.
About
No description or website provided.
#abstract-interpretation
#static-analysis
#machine-learning
#neural-networks
#fairness
Readme
MPL-2.0 License
Releases
No releases published
Packages
No packages published
Languages
Python 98.7%
Shell 1.3%



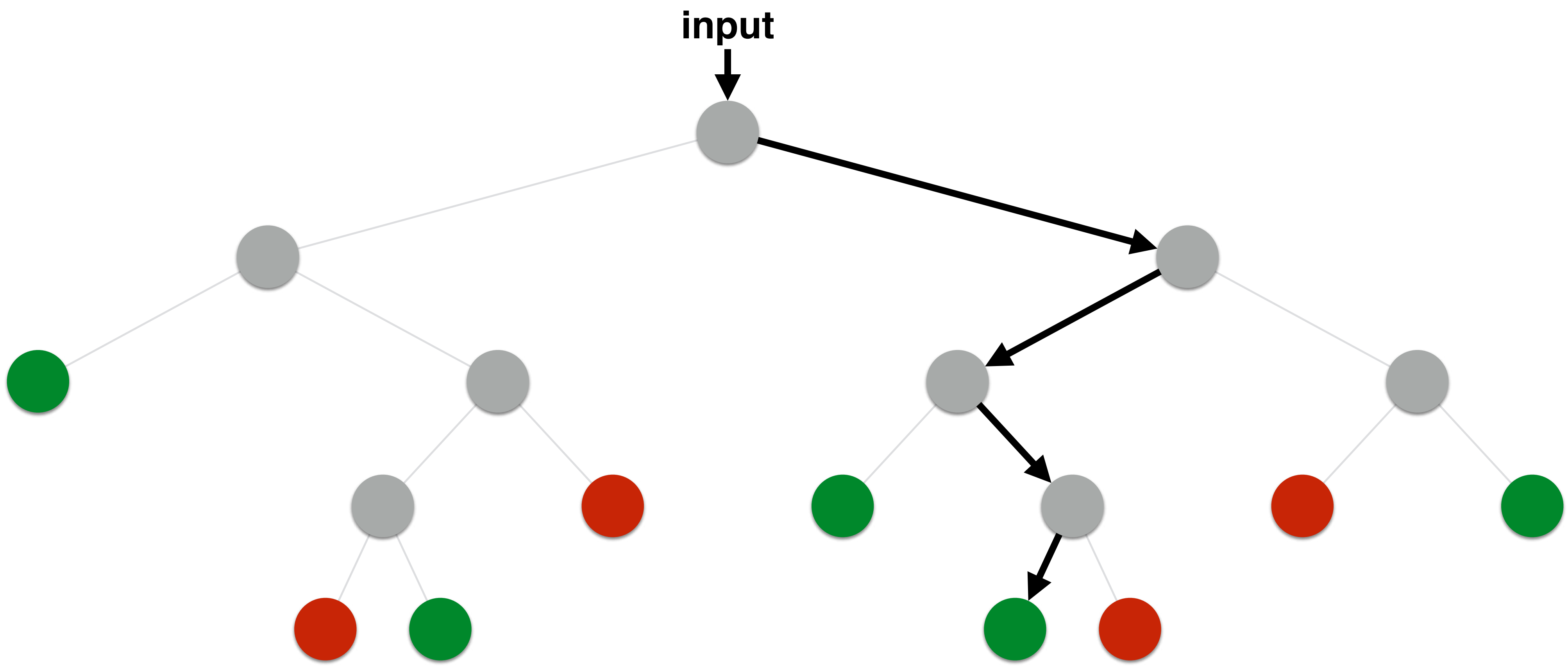
Other ML Models

Support Vector Machines



- **F. Ranzato and M. Zanella**. *Robustness Verification of Support Vector Machines.* In SAS, 2019.
  an approach for proving local robustness to adversarial perturbations

Decision Tree Ensembles



- **A. Kantchelian, J. D. Tygar, and A. Joseph**. *Evasion and Hardening of Tree Ensemble Classifiers*. In ICML 2016.
  H. Chen, H. Zhang, S. Si, Y. Li, D. Boning, and C.-J. Hsieh. *Robustness Verification of Tree-based Models*. In NeurIPS 2019.
  approaches for finding the nearest adversarial example

Decision Tree Ensembles

- **N. Sato, H. Kuruma, Y. Nakagawa, and H. Ogawa**. *Formal Verification of Decision-Tree Ensemble Model and Detection of its Violating-Input-Value Ranges*. 2020.
  approach for safety verification

- **G. Einziger, M. Goldstein, Y. Sa'ar, and I. Segall.** *Verifying Robustness of Gradient Boosted Models*. In AAAI 2019.
  SMT-based approach for local robustness

- **J. Törnblom and S. Nadjm-Tehrani**. *Formal Verification of Input-Output Mappings of Tree Ensembles*. 2020.
  F. Ranzato and M. Zanella. *Abstract Interpretation of Decision Tree Ensemble Classifiers*. In AAAI 2020.
  S. Calzavara, P. Ferrara, and C. Lucchese. *Certifying Decision Trees Against Evasion Attacks by Program Analysis*. In ESORICS 2020.
  abstract interpretation-based approaches for local robustness

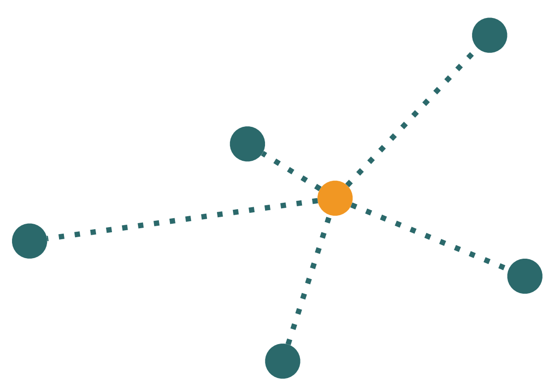
Formal Methods for Model Training

Robust Training

Minimizing the Worst-Case Loss for Each Input

Adversarial Training

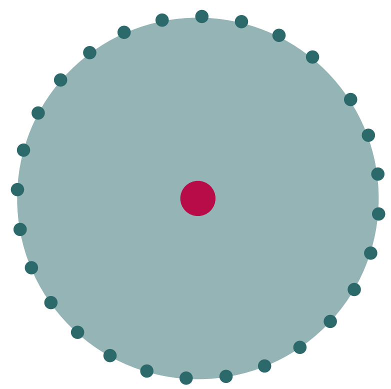
Minimizing a Lower Bound on the Worst-Case Loss for Each Input



generate adversarial inputs
and use them as training data

Certified Training

Minimizing an Upper Bound on the Worst-Case Loss for Each Input



use upper bound as regularizer
to encourage robustness

Certified Training

- **M. Andriushchenko, and M. Hein**. *Provably Robust Boosted Decision Stumps and Trees Against Adversarial Attacks*. In NeurIPS 2019.
  approach targeting decision trees

- **M. Hein and M. Andriushchenko**. *Formal Guarantees on the Robustness of a Classifier Against Adversarial Manipulation*. In NeurIPS 2017.
  E. Wong and Z. Kolter. *Provable Defenses Against Adversarial Examples via the Convex Outer Adversarial Polytope*. In ICML, 2018.
  A. Raghunathan, J. Steinhardt, and P. Liang. *Certified Defenses against Adversarial Examples*. In ICML, 2018.
  approaches targeting neural networks

Certified Training

- **M. Mirman, T. Gehr, and M. Vechev**. *Differentiable Abstract Interpretation for Provably Robust Neural Networks* In ICML 2018.
  abstract interpretation-based approach targeting neural networks

- **F. Ranzato and M. Zanella**. *Genetic Adversarial Training of Decision Trees*. In GECCO 2021.
  F. Ranzato, CU, and M. Zanella. *Fairness-Aware Training of Decision Trees by Abstract Interpretation*. In CIKM 2021.
  abstract interpretation-based approaches targeting decision trees

Formal Methods for Data Preparation

Jupyter Notebooks

localhost

jupyter Gradebook Last Checkpoint: a few seconds ago (autosaved)

Logout

File Edit View Insert Cell Kernel Help

Trusted Python 3

Run Code

In [1]:
```
import pandas as pd
```

In [2]:
```
df = pd.read_csv('Grades.csv', index_col=0)
df.head()
```

Out[2]:

ID	Name	Q1	Q2	Q3
2394	Alice	A	A	A
4583	Bob	F	B	B
3956	Carol	F	A	C
9578	David	D	F	C

In [3]:
```
grade2gpa = { 'A': 4.0, 'B': 3.0, 'C': 2.0, 'D': 1.0, 'F': 0.0 }
df.iloc[:, df.columns.str.startswith('Q')] = df.iloc[:, df.columns.str.startswith('Q')].applymap(grade2gpa.get)
```

In [4]:
```
df['Mean'] = df.iloc[:, df.columns.str.startswith('Q')].mean(axis=1)
```

In [5]:
```
es = pd.read_csv('Emails.csv', index_col=0)
```

In [6]:
```
un = df.join(es)
```

In [7]:
```
res = un[["Email", "Mean"]]
res.head()
```

Out[7]:

ID	Email	Mean
2394	alice@uni.eu	4.0
4583	bob@uni.eu	2.0
3956	carol@uni.eu	2.0
9578	david@uni.eu	1.0

Jupyter Notebooks

UNUSED DATA

```
[1] d = genfromtxt('data.csv')

[2] selector = SelectKBest(k=25)
    x = selector.fit_transform(d)

[3] x = genfromtxt('data2.csv')

[4] x_train, x_test, y_train, y_test =
      train_test_split(x, ...)

[5] lr = LogisticRegression()
    lr.fit(x_train, y_train)
    y_pred = lr.predict(x_test)
```

P. Subotić et al. - A Static Analysis Framework for Data Science Notebooks (ICSE 2022)

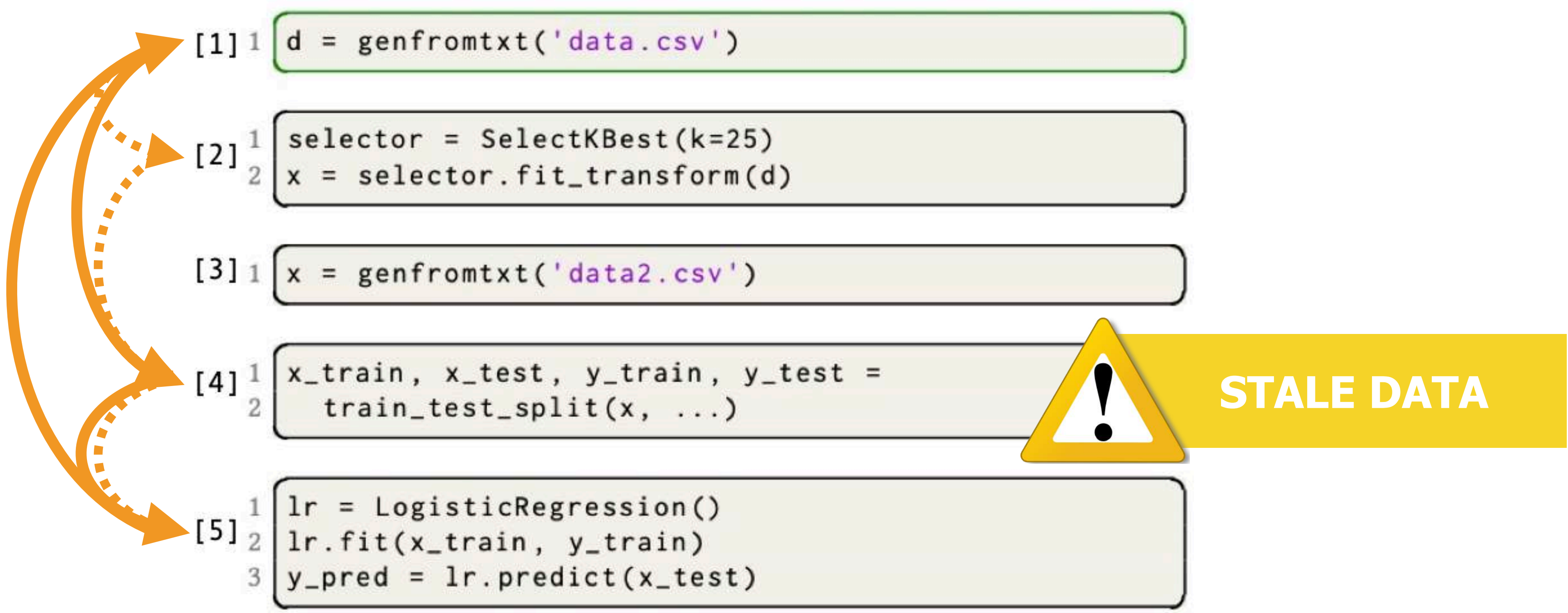
Jupyter Notebooks

```
[1] d = genfromtxt('data.csv')

[2] selector = SelectKBest(k=25)
    x = selector.fit_transform(d)

[3] x = genfromtxt('data2.csv')

[4] x_train, x_test, y_train, y_test =
      train_test_split(x, ...)

[5] lr = LogisticRegression()
    lr.fit(x_train, y_train)
    y_pred = lr.predict(x_test)
```

DATA LEAK

P. Subotić et al. - A Static Analysis Framework for Data Science Notebooks (ICSE 2022)

Jupyter Notebooks



P. Subotić et al. - A Static Analysis Framework for Data Science Notebooks (ICSE 2022)

Anomalously Unused Data

The Reinhart-Rogoff Paper

American Economic Review: Papers & Proceedings 100 (May 2010): 573–578
http://www.aeaweb.org/articles.php?doi=10.1257/aer.100.2.573

Growth in a Time of Debt

By Carmen M. Reinhart and Kenneth S. Rogoff*

In this paper, we exploit a new multi-country historical dataset on public (government) debt to search for a systemic relationship between high public debt levels, growth and inflation.[1] Our main result is that whereas the link between growth and debt seems relatively weak at "normal" debt levels, median growth rates for countries with public debt over roughly 90 percent of GDP are about one percent lower than otherwise; average (mean) growth rates are several percent lower. Surprisingly, the relationship between public debt and growth is remarkably similar across emerging markets and advanced economies. This is not the case for inflation. We find no systematic relationship between high debt levels and inflation for advanced economies as a group (albeit with individual country exceptions including the United States). By contrast, in emerging market countries, high public debt levels coincide with higher inflation.

Our topic would seem to be a timely one. Public debt has been soaring in the wake of the recent global financial maelstrom, especially in the epicenter countries. This should not be surprising, given the experience of earlier severe financial crises.[2] Outsized deficits and epic bank bailouts may be useful in fighting a downturn, but what is the long-run macroeconomic impact, especially against the backdrop of graying populations and rising social insurance costs? Are sharply elevated public debts ultimately a manageable policy challenge?

Our approach here is decidedly empirical, taking advantage of a broad new historical dataset on public debt (in particular, central government debt) first presented in Carmen M. Reinhart and Kenneth S. Rogoff (2008, 2009b). Prior to this dataset, it was exceedingly difficult to get more than two or three decades of public debt data even for many rich countries, and virtually impossible for most emerging markets. Our results incorporate data on 44 countries spanning about 200 years. Taken together, the data incorporate over 3,700 annual observations covering a wide range of political systems, institutions, exchange rate and monetary arrangements, and historic circumstances.

We also employ more recent data on external debt, including debt owed both by governments and by private entities. For emerging markets, we find that there exists a significantly more severe threshold for total gross external debt (public and private)—which is almost exclusively denominated in a foreign currency—than for total public debt (the domestically issued component of which is largely denominated in home currency). When gross external debt reaches 60 percent of GDP, annual growth declines by about two percent; for levels of external debt in excess of 90 percent of GDP, growth rates are roughly cut in half. We are not in a position to calculate separate total external debt thresholds (as opposed to public debt thresholds) for advanced countries. The available time-series is too recent, beginning only in 2000. We do note, however, that external debt levels in advanced countries now average nearly 200 percent of GDP, with external debt levels being particularly high across Europe.

The focus of this paper is on the longer-term

* Reinhart: Department of Economics, 4115 Tydings Hall, University of Maryland, College Park, MD 20742 (e-mail: creinhar@umd.edu); Rogoff: Economics Department, 216 Littauer Center, Harvard University, Cambridge MA 02138–3001 (e-mail: krogoff@harvard.edu). The authors would like to thank Olivier Jeanne and Vincent R. Reinhart for helpful comments.

[1] In this paper "public debt" refers to gross central government debt. "Domestic public debt" is government debt issued under domestic legal jurisdiction. Public debt does not include debts carrying a government guarantee. Total gross external debt includes the external debts of *all* branches of government as well as private debt that is issued by domestic private entities under a foreign jurisdiction.

[2] Reinhart and Rogoff (2009a, b) demonstrate that the

	B	C	I	J	K	L	M
2			Real GDP growth				
3			Debt/GDP				
4	Country	Coverage	30 or less	30 to 60	60 to 90	90 or above	30 or less
26			3.7	3.0	3.5	1.7	5.5
27	Minimum		1.6	0.3	1.3	-1.8	0.8
28	Maximum		5.4	4.9	10.2	3.6	13.3
29							
30	US	1946-2009	n.a.	3.4	3.3	-2.0	n.a.
31	UK	1946-2009	n.a.	2.4	2.5	2.4	n.a.
32	Sweden	1946-2009	3.6	2.9	2.7	n.a.	6.3
33	Spain	1946-2009	1.5	3.4	4.2	n.a.	9.9
34	Portugal	1952-2009	4.8	2.5	0.3	n.a.	7.9
35	New Zealand	1948-2009	2.5	2.9	3.9	-7.9	2.6
36	Netherlands	1956-2009	4.1	2.7	1.1	n.a.	6.4
37	Norway	1947-2009	3.4	5.1	n.a.	n.a.	5.4
38	Japan	1946-2009	7.0	4.0	1.0	0.7	7.0
39	Italy	1951-2009	5.4	2.1	1.8	1.0	5.6
40	Ireland	1948-2009	4.4	4.5	4.0	2.4	2.9
41	Greece	1970-2009	4.0	0.3	2.7	2.9	13.3
42	Germany	1946-2009	3.9	0.9	n.a.	n.a.	3.2
43	France	1949-2009	4.9	2.7	3.0	n.a.	5.2
44	Finland	1946-2009	3.8	2.4	5.5	n.a.	7.0
45	Denmark	1950-2009	3.5	1.7	2.4	n.a.	5.6
46	Canada	1951-2009	1.9	3.6	4.1	n.a.	2.2
47	Belgium	1947-2009	n.a.	4.2	[illegible]	2.6	n.a.
48	Austria	1948-2009	5.2	3.3	-3.8	n.a.	5.7
49	Australia	1951-2009	3.2	4.9	4.0	n.a.	5.9
50							
51			4.1	2.8	2.8	=AVERAGE(L30:L44)	

data excluded from the analysis

The Reinhart-Rogoff Paper

FAQ: Reinhart, Rogoff, and the Excel Error That Changed History

By Peter Coy | April 18, 2013

The Excel Depression

By PAUL KRUGMAN
Published: April 18, 2013 | 470 Comments

FACEBOOK
TWITTER
GOOGLE+
SAVE
EMAIL
SHARE
PRINT
REPRINTS

In this age of information, math errors can lead to disaster. NASA's Mars Orbiter crashed because engineers forgot to convert to metric measurements; JPMorgan Chase's "London Whale" venture went bad in part because modelers divided by a sum instead of an average. So, did an Excel coding error destroy the economies of the Western world?

Enlarge This Image

The story so far: At the beginning of 2010, two Harvard economists, Carmen Reinhart and Kenneth Rogoff, circulated a paper, "Growth in a Time of Debt," that purported to identify a critical "threshold," a tipping point, for government indebtedness. Once debt exceeds 90 percent of gross domestic product, they claimed, economic growth drops off sharply.

Ms. Reinhart and Mr. Rogoff had credibility thanks to a

England Covid-19 Cases Error

SCIENCE | US & WORLD | TECH

Excel spreadsheet error blamed for UK's 16,000 missing coronavirus cases

The case went missing after the spreadsheet hit its filesize limit

By James Vincent | Oct 5, 2020, 9:41am EDT

NEWS

Check for updates

The BMJ

Cite this as: *BMJ* 2020;371:m3891

http://dx.doi.org/10.1136/bmj.m3891

Published: 06 October 2020

Covid-19: Only half of 16 000 patients missed from England's official figures have been contacted

Elisabeth Mahase

Details of nearly 16 000 cases of covid-19 were not transferred to England's NHS Test and Trace service and were missed from official figures because of an error in the process for updating the data.

England's health and social care secretary, Matt Hancock, told the House of Commons on Monday 5 October that after the error was discovered on Friday 2 October "6500 hours of extra contact tracing" had been carried out over the weekend. But as at Monday morning only half (51%) of the people had been reached by contact tracers.

In response, Labour's shadow health secretary, ... said, "Thousands of people are ... exposed to covid, ...

data and furthermore have issued guidance on validation and risk management for these products if they are to be used in such a safety critical manner."

The error came as the Labour Party's leader, Keir Starmer, said that the prime minister had "lost control" of covid-19, with no clear strategy for beating it. Speaking to the *Observer*, Starmer set out his five point plan for covid-19, which starts with publishing the criteria for local restrictions, as the German government did. Secondly, he said public health messaging should be improved by adding a feature to the NHS covid-19 app so people can search their postcode and find out their local restrictions.

Starmer has also said he would fix the contact tracing system by investing in NHS and university ... to expand testing and at the same time ... in charge of contact ... in high

BMJ: first published as 10.1136/bmj.m3891 on 6 October 2020. Do

Data Usage Static Analysis

practical tools
targeting specific programs

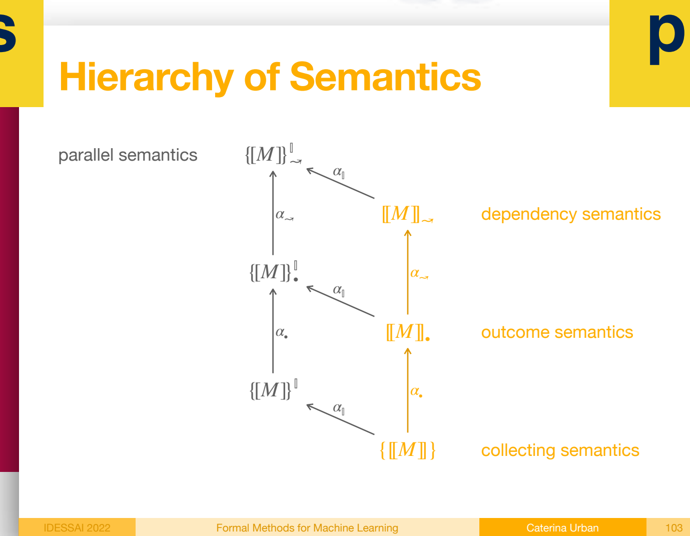
algorithmic approaches
to decide program properties

secure information flow

strongly-live variable analysis

program slicing

mathematical models
of the program behavior



CU and P. Müller - An Abstract Interpretation Framework for Data Usage (ESOP 2018)

Bibliography

[Kurd03] **Zeshan Kurd, Tim Kelly**. Establishing Safety Criteria for Artificial Neural Networks. In KES, pages 63-169, 2003.

[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.

[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.

[Mazzucato21] **Denis Mazzucato and Caterina Urban**. Reduced Products of Abstract Domains for Fairness Certification of Neural Networks. In SAS, 2021.

Bibliography

[Julian16] **Kyle D. Julian, Jessica Lopez, Jeffrey S. Brush, Michael P. Owen, Mykel J. Kochenderfer**. Policy Compression for Aircraft Collision Avoidance Systems. In DASC, pages 1–10, 2016.

[Katz17] **Guy Katz, Clark W. Barrett, David L. Dill, Kyle Julian, Mykel J. Kochenderfer**. Reluplex: An Efficient SMT Solver for Verifying Deep Neural Networks. In CAV, pages 97–117, 2017.

[Galhotra17] **Sainyam Galhotra, Yuriy Brun, and Alexandra Meliou**. Fairness Testing: Testing Software for Discrimination. In FSE, pages 498–510, 2017.

[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.

[Urban21] **Caterina Urban and Antoine Miné**. A Review of Formal Methods applied to Machine Learning. https://arxiv.org/abs/2104.02466, 2021.