Neural Proofs for Sound Verification and Control of Complex Systems

Alessandro Abate

with D. Roy, A. Edwards, A. Peruffo, D. Ahmed, L. Rickard, M. Giacobbe

OXCAV - Department of Computer Science - University of Oxford

CDC - 9 Dec 2025

[references at end of deck]

Outline

1. Context
2. Neural Proofs - Formal Synthesis of Neural Certificates
3. Synthesis of Formal Abstractions

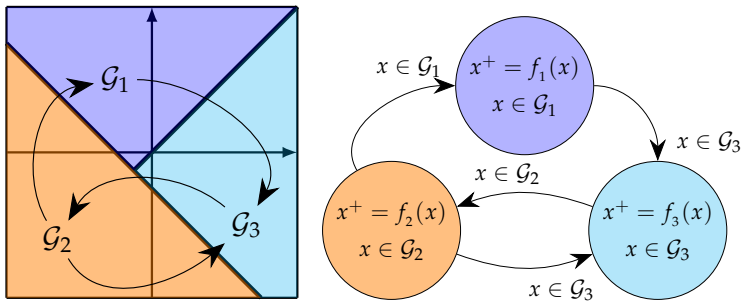
Outline

1. Context
2. Neural Proofs - Formal Synthesis of Neural Certificates
3. Synthesis of Formal Abstractions

Control vs formal verification

- dynamical models for control

$$x \in \mathbb{R}^n, \mathcal{G}_i \subset \mathbb{R}^n, i \in \{1,2,3\}$$
$$\forall x \in \mathcal{G}_i, \quad x^+ = f_i(x)$$



- safety,
  reachability,
  stability
- invariants & barrier certificates,
  reach-set computation,
  Lyapunov functions

- software programs

```
int x, c, r;
...
x = rand(1,...,10);
c = 1;
while (x > 0) {
r = Bernoulli(0.5);
if (c == 1 && r) { x = 2 * x; }
else if (c == 1 && !r) { c++; }
else { x = x - 1; }
}
```

- invariance,
  termination,
  assertion check/violation
- program and loop invariants,
  reach over CFG, abstract int,
  ranking functions and progress measures

Models

- dynamical/transition systems and reactive programs

$$x \in \mathbb{S}, \theta \in \Theta, a \in \mathbb{A}$$
$$x^+ = f(x, \theta, a)$$

- x - states/variables (possibly hybrid)
- θ - uncertainty (lack of determinism)
- a/u - action, input, decisions
  - non-environmental: adversarial (demonic) vs agential (angelic)
- general-space MDPs[1] [2] [3], or SHS[4]

[1] Bertsekas & Shreve, "Stochastic optimal control: The discrete-time case," Athena Scientific, 1996

[2] Hernández-Lerma & Lasserre, "Discrete-time Markov control processes," Springer, 1996

[3] Meyn & Tweedie, "Markov chains and stochastic stability," Springer, 1993

[4] A. Lavaei et.al., "Automated Verification and Synthesis of Stochastic Hybrid Systems: A Survey," Automatica, v 146, 2022

Cyber-Physical Systems: from code to control

- complex embedded systems
- interleaving of cyber/digital components with physical/analogue dynamics

- dynamics, computation, and control

- safety-critical applications

$\rightarrow$ sound and automated verification

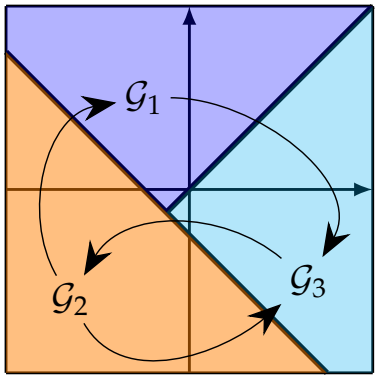
$\rightarrow$ correct-by-design control synthesis

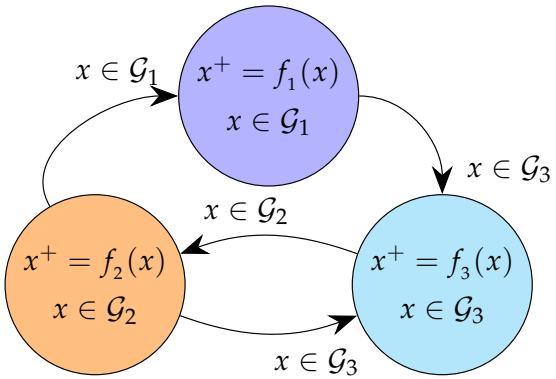
Properties: Encoding rich dynamical behaviour

- as specifications, requirements for verification (e.g., safety)
- as tasks, objectives for control design (e.g., reachability)
- without manual reward engineering - *"rewards are NOT enough"*
  useful also in model-free (e.g., RL) sequential decision making

Properties: Encoding rich dynamical behaviour

$$x \in \mathbb{R}^n$$
$$\mathcal{G}_i \subset \mathbb{R}^n, i \in \{1, \dots m\}$$
$$\forall x \in \mathcal{G}_i, \quad x^+ = f_i(x)$$

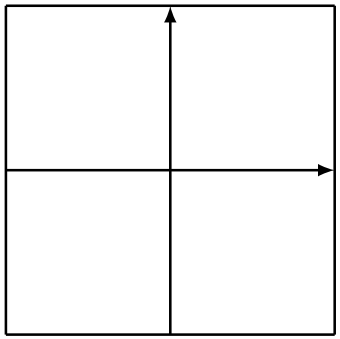




Properties: Encoding rich dynamical behaviour

$$x \in \mathbb{R}^n$$

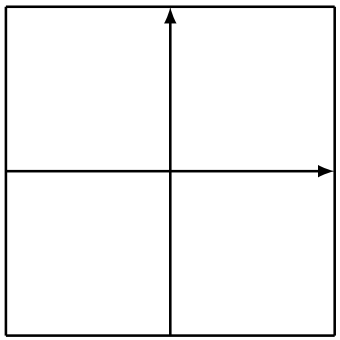
$$x^+ = f(x)$$

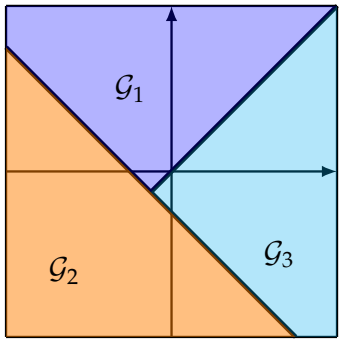


$x^+ = f(x)$

Properties: Encoding rich dynamical behaviour

$$x \in \mathbb{R}^n$$
$$\mathcal{G}_i \subset \mathbb{R}^n, i \in \{1, \ldots m\}$$
$$x^+ = f(x)$$





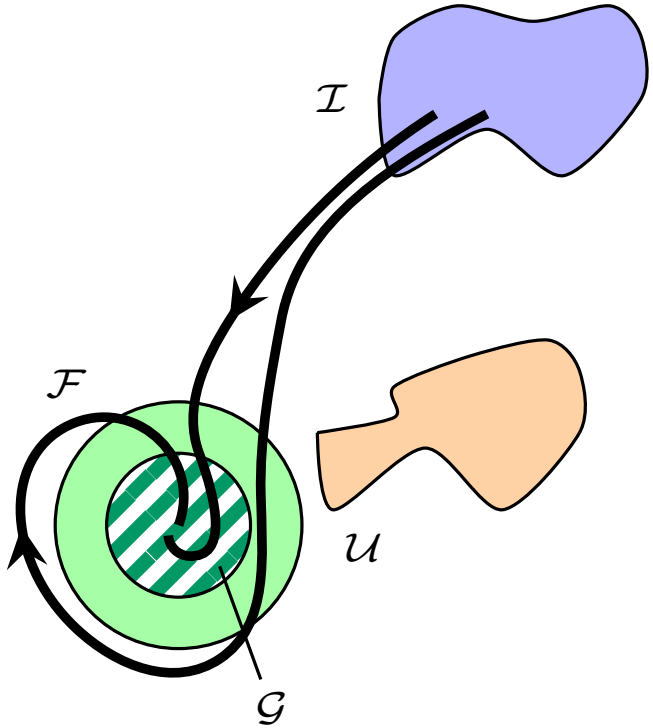
$$\begin{array}{lccccc} x: & x_0 \rightarrow & x_1 \rightarrow & x_2 \rightarrow & \ldots \\ \rho: & \mathcal{G}_{x_0} \rightarrow & \mathcal{G}_{x_1} \rightarrow & \mathcal{G}_{x_2} \rightarrow & \ldots \end{array}$$

- model's behaviour (*cf.* Jan Willems) $\leftrightarrow$ formal language

Properties: Encoding rich dynamical behaviour

- consider (class of) properties/requirements/specifications

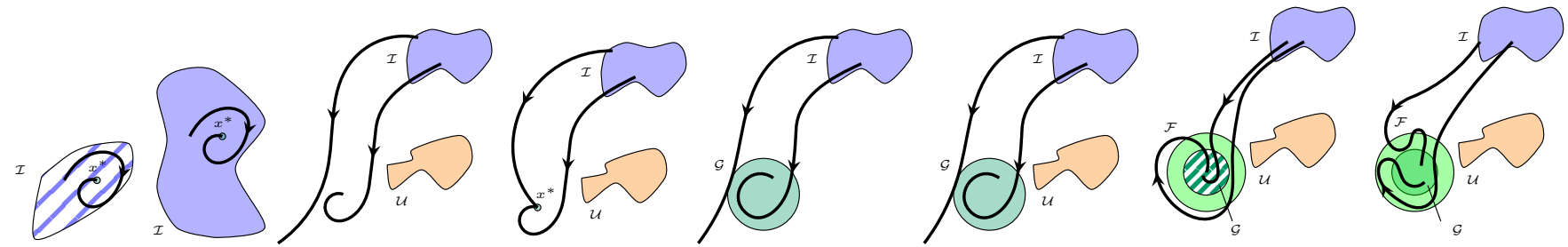
$$\forall x_0 \in \mathcal{I}, \quad \exists T \in \mathbb{N}, \quad \forall k \in \{0, 1, \ldots, T-1\}, \quad \forall \tau \geq T : \\ x_T \in \mathcal{G}, \quad x_k \notin \mathcal{U}, \quad x_\tau \in \mathcal{F}$$



Properties: Encoding rich dynamical behaviour

- consider (class of) properties/requirements/specifications

$$\forall x_0 \in \mathcal{I}, \quad \exists T \in \mathbb{N}, \quad \forall k \in \{0, 1, \ldots, T-1\}, \quad \forall \tau \geq T : \\ x_T \in \mathcal{G}, \quad x_k \notin \mathcal{U}, \quad x_\tau \in \mathcal{F}$$



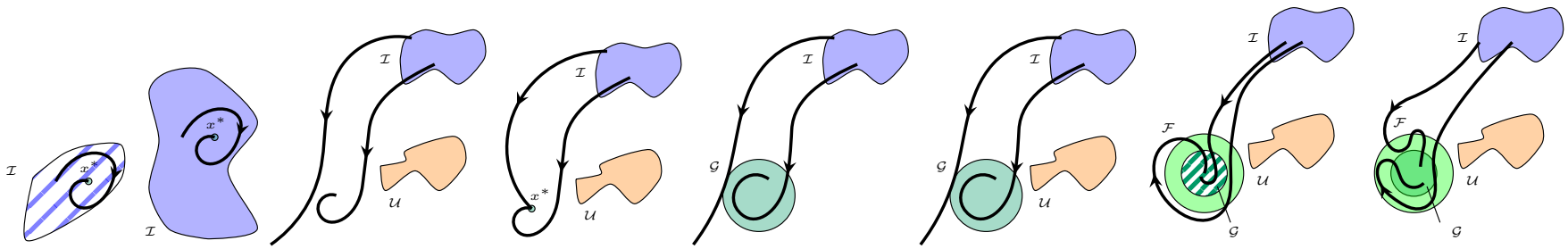
a: Stability b: ROA c: Safety d: SWA e: Reachability f: RWA g: RSWA h: RAR

- class encompasses stability, invariance, safety, reachability, reach-avoid, ... (cf. safe/co-safe fragment in linear temporal logic)

Properties: Encoding rich dynamical behaviour

- consider (class of) properties/requirements/specifications

$$\forall x_0 \in \mathcal{I}, \quad \exists T \in \mathbb{N}, \quad \forall k \in \{0, 1, \ldots, T-1\}, \quad \forall \tau \geq T :$$
$$x_T \in \mathcal{G}, \quad x_k \notin \mathcal{U}, \quad x_\tau \in \mathcal{F}$$

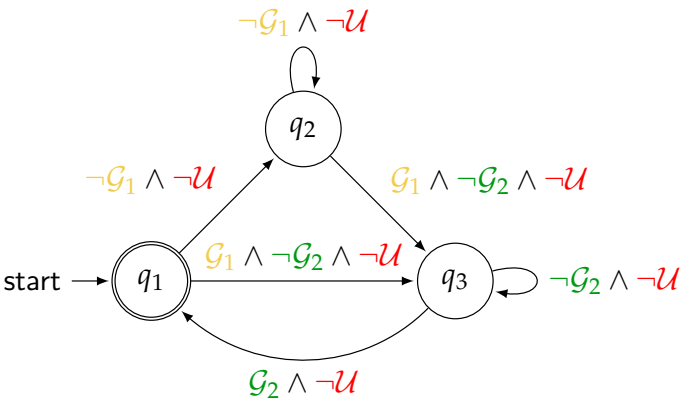


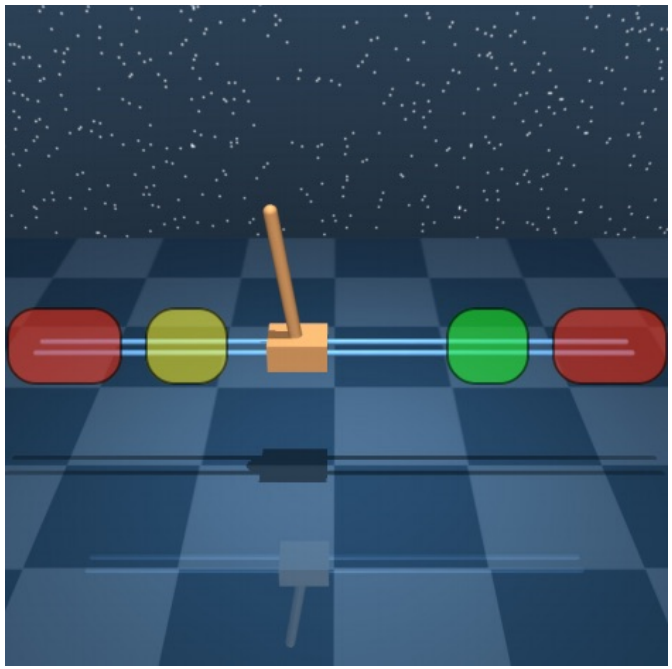
a: Stability b: ROA c: Safety d: SWA e: Reachability f: RWA g: RSWA h: RAR

- connections to:
  - temporal logics
  - formal languages
  - automata theory
- beyond obligation, to full reactivity (∞-horizon specs)

Properties: Encoding rich dynamical behaviour

$$\Box\Diamond\mathcal{G}_1 \wedge \Box\Diamond\mathcal{G}_2 \wedge \Box\neg\mathcal{U}$$





Synthesis for verification and design

from verification/analysis

to model design, or synthesis of controller (scheduler, policy, strategy)

Synthesis for verification and design

from verification/analysis

to model design, or synthesis of controller (scheduler, policy, strategy)

this line of research:
consider broad model class (∞-state g-MDP's)
propose two general, automated, and formal approaches:

Synthesis for verification and design

from verification/analysis

to model design, or synthesis of controller (scheduler, policy, strategy)

this line of research:
consider broad model class (∞-state g-MDP's)
propose two general, automated, and formal approaches:

1. synthesis of certificates for verification, and of controllers+certificates
2. synthesis of abstractions, for verification and design [ThB03, FrB02]

Synthesis for verification and design

from verification/analysis

to model design, or synthesis of controller (scheduler, policy, strategy)

this line of research:
consider broad model class (∞-state g-MDP's)
propose two general, automated, and formal approaches:

Neural Proofs - Formal Synthesis of Neural Certificates

Outline

1. Context
2. Neural Proofs - Formal Synthesis of Neural Certificates
3. Synthesis of Formal Abstractions

Outline

1 Context

2 Neural Proofs - Formal Synthesis of Neural Certificates

3 Synthesis of Formal Abstractions

Decision problems: SAT and SMT

- SAT is a decision problem (yes/no question)
- problem: to find satisfying assignment of Boolean function
- e.g., assume Boolean x_i, check

$$\exists x_1, x_2, x_3 : \quad (x_1 \vee \neg x_2) \wedge (\neg x_1 \vee x_2 \vee x_3) \wedge \neg x_1$$

Decision problems: SAT and SMT

- SAT is a decision problem (yes/no question)
- problem: to find satisfying assignment of Boolean function
- e.g., assume Boolean x_i, check

$$\exists x_1, x_2, x_3 : \quad (x_1 \vee \neg x_2) \wedge (\neg x_1 \vee x_2 \vee x_3) \wedge \neg x_1$$

- SMT is a decision problem for a logical formula within a theory
- instance: theory of non-linear arithmetics over real closed fields
- e.g., assume reals $x_i \in \mathbb{R}$, check

$$\exists x_1, x_2 : \quad x_1 \geq 0 \Rightarrow 3x_1 + 2x_2 + 1 > 0$$

Decision problems: SAT and SMT

- SAT is a decision problem (yes/no question)
- problem: to find satisfying assignment of Boolean function
- e.g., assume Boolean x_i, check

$$\exists x_1, x_2, x_3 : \quad (x_1 \vee \neg x_2) \wedge (\neg x_1 \vee x_2 \vee x_3) \wedge \neg x_1$$

- SMT is a decision problem for a logical formula within a theory
- consider harder problem (2nd-order logic): assume reals $x_i \in \mathbb{R}$, seek function $F : \mathbb{R} \times \mathbb{R} \to \mathbb{R}$, s.t.

$$\exists F(x_1, x_2), \forall x_1, x_2 :$$
$$F(x_1, x_2) \geq x_1 \wedge F(x_1, x_2) \geq x_2 \wedge (F(x_1, x_2) = x_1 \vee F(x_1, x_2) = x_2)$$

(spoiler: $F(x_1, x_2) = \max\{x_1, x_2\}$)

Neural proofs and certificates

neural proofs = 1) proof rules + 2) certificates synthesis

1) proof rules = spec-dependent algebraic conditions on model's dynamics
2) certificate synthesis

Neural proofs and certificates

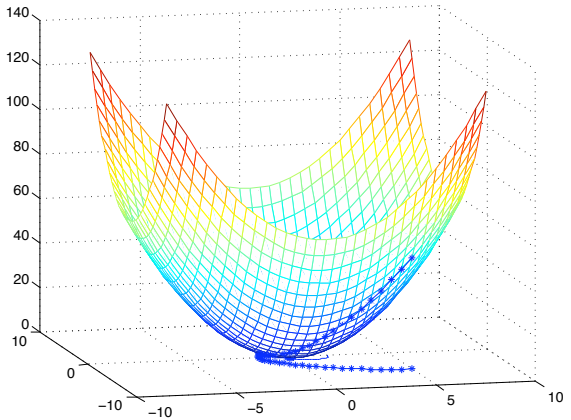
neural proofs = 1) proof rules + 2) certificates synthesis

1) proof rules = spec-dependent algebraic conditions on model's dynamics
2) certificate synthesis, 2a) deductive vs 2b) inductive
   - 2a) deductive, i.e. via logic, SMT
   - 2b) inductive: neural networks + SMT

Lyapunov functions for stability

- consider $x^+ = f(x)$, assume $x_e \in \mathbb{R}^n$ is an equilibrium, $f(x_e) = x_e$
- ensure (asymptotic) stability of $x_e \in \mathcal{D} \subseteq \mathbb{R}^n$ (assume $\mathcal{D}$ given)
- by finding $\mathbb{R}$-valued Lyapunov function $V(x)$, satisfying

1. lower bound:
$$V(x_e) = 0 \tag{1}$$
2. positive definiteness:
$$V(x) > 0,\ \forall x \in \mathcal{D} \setminus \{x_e\} \tag{2}$$
3. negative decrease:
$$V(f(x)) - V(x) < -\epsilon,\ \forall x \in \mathcal{D} \setminus \{x_e\} \quad (\epsilon > 0) \tag{3}$$



Lyapunov functions for stability

- consider $x^+ = f(x)$, assume $x_e \in \mathbb{R}^n$ is an equilibrium, $f(x_e) = x_e$
- ensure (asymptotic) stability of $x_e \in \mathcal{D} \subseteq \mathbb{R}^n$ (assume $\mathcal{D}$ given)
- by finding $\mathbb{R}$-valued Lyapunov function $V(x)$, satisfying

1. lower bound:
$$V(x_e) = 0 \tag{1}$$
2. positive definiteness:
$$V(x) > 0,\ \forall x \in \mathcal{D} \setminus \{x_e\} \tag{2}$$
3. negative decrease:
$$V(f(x)) - V(x) < -\epsilon,\ \forall x \in \mathcal{D} \setminus \{x_e\} \quad (\epsilon > 0) \tag{3}$$

- that is, solve following synthesis problem:

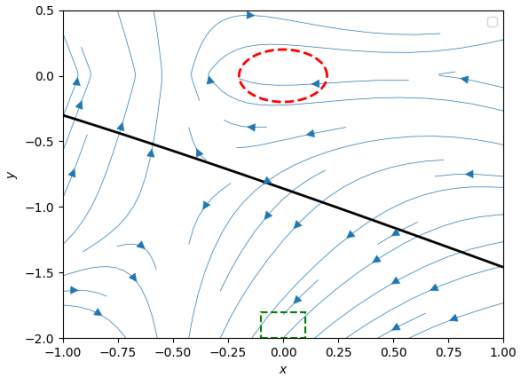
$$\exists V : \mathcal{D} \to \mathbb{R} \quad s.t. \ \ \forall x \in \mathcal{D}, \quad \text{conditions (1)} \wedge \text{(2)} \wedge \text{(3) hold}$$

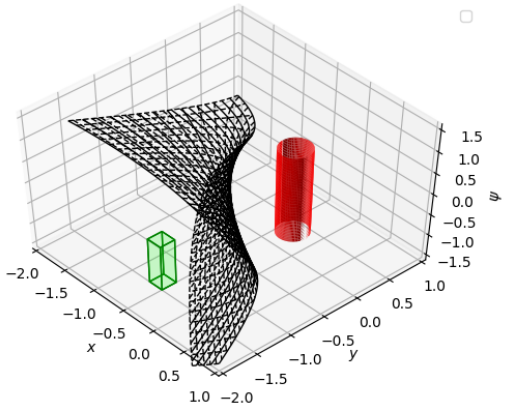
- feed 2nd-order logical formula above to SMT solver

Barrier certificates for safety

- consider $x^+ = f(x)$, along with sets $\mathcal{I}$ (initial) and $\mathcal{U}$ (unsafe)
- ensure there exists no trajectory starting in $\mathcal{I}$ ever entering $\mathcal{U}$

Barrier certificates for safety





Barrier certificates for safety

- consider $x^+ = f(x)$, along with sets $\mathcal{I}$ (initial) and $\mathcal{U}$ (unsafe)
- ensure there exists no trajectory starting in $\mathcal{I}$ ever entering $\mathcal{U}$
- by finding barrier function $B(x)$, satisfying

1. negativity within initial set $\mathcal{I}$:

$$B(x) \leq 0, \ \forall x \in \mathcal{I}$$

2. positivity within unsafe set $\mathcal{U}$:

$$B(x) > 0, \ \forall x \in \mathcal{U}$$

3. set invariance property via negative decrease:

$$B(f(x)) - B(x) < 0, \ \forall x \in (\mathcal{U} \cup \mathcal{I})^C$$

- again, feed a 2nd-order logical formula to SMT solver

Ranking functions for termination/halting/reachability

- consider probabilistic program $\mathtt{x}^+ = f(\mathtt{x}, \mathtt{w})$:

```
while (x > 0):
if prob(0.75) {
x-- }
```

Ranking functions for termination/halting/reachability

- consider probabilistic program $\mathtt{x}^+ = f(\mathtt{x}, \mathtt{w})$:

```
while (x > 0):
if prob(0.75) {
x-- }
```

- ensure that, for any $\mathtt{x} \geq 0$, $\mathsf{F}(\mathtt{x} \leq 0)$ holds almost surely (with probability 1)
- by finding ranking function $V(\mathtt{x})$, satisfying

1. $V(0) = 0$
2. $\mathtt{x} > 0 \rightarrow V(\mathtt{x}) > 0$
3. $\mathtt{x} > 0 \rightarrow \mathbb{E}[V(f(\mathtt{x}))] \leq V(\mathtt{x}) - \epsilon$

   namely, if $\mathtt{x} > 0$, $V(\mathtt{x})$ decreases in expectation by ϵ (positive constant)

Ranking functions for termination/halting/reachability

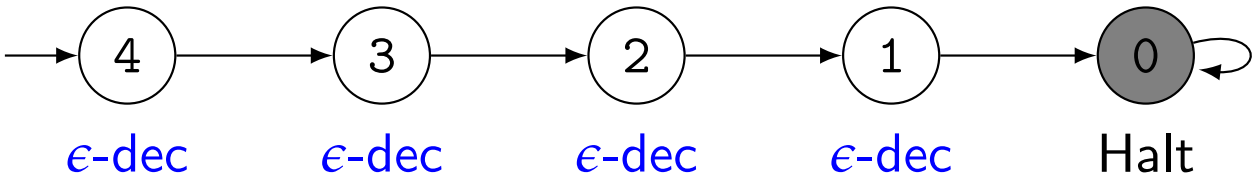
- consider probabilistic program $\mathtt{x}^+ = f(\mathtt{x}, \mathtt{w})$:

```
while (x > 0):
if prob(0.75) {
x-- }
```

- ensure that, for any $\mathtt{x} \geq 0$, $\mathsf{F}(\mathtt{x} \leq 0)$ holds almost surely (with probability 1)
- by finding ranking function $V(\mathtt{x})$, satisfying

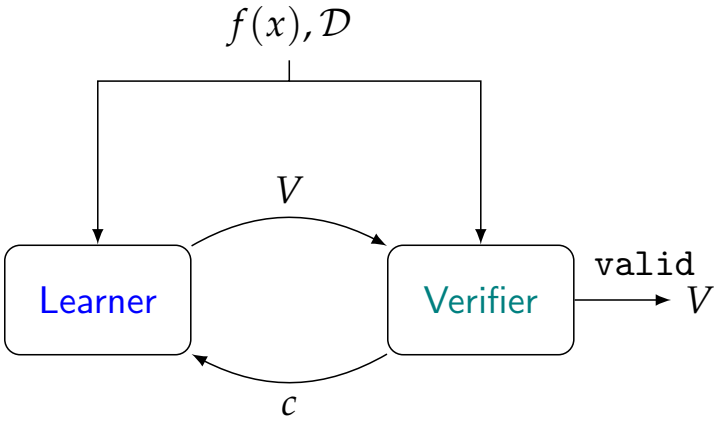
1. $V(0) = 0$
2. $\mathtt{x} > 0 \rightarrow V(\mathtt{x}) > 0$
3. $\mathtt{x} > 0 \rightarrow \mathbb{E}[V(f(\mathtt{x}))] \leq V(\mathtt{x}) - \epsilon$

   namely, if $\mathtt{x} > 0$, $V(\mathtt{x})$ decreases in expectation by ϵ (positive constant)



e.g., $V(\mathtt{x}) = \mathtt{x} \geq 0$ & decreases in expectation by say $\epsilon = 3/4$

Counterexample-guided Inductive Synthesis (CEGIS)



1. Learner

   generates candidates V over finite set S

2. Verifier

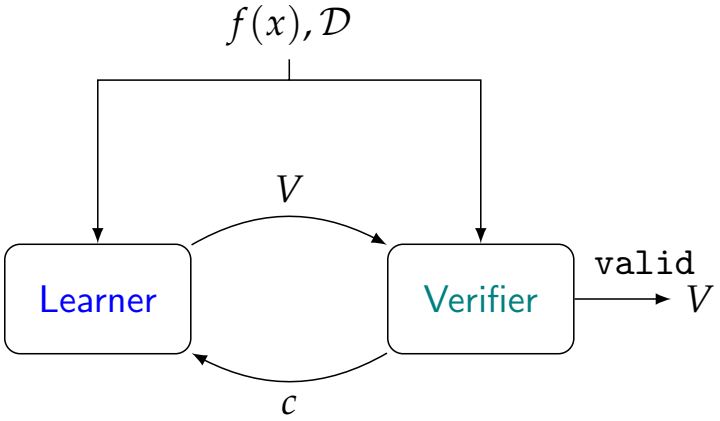
   certifies validity on $\mathcal{D}$, or provides counterexample(s) c

- inductive synthesis loop

  1. sample (finite) set $S \subset \mathcal{D}$
  2. Learner generates $V(\theta, \cdot)$ via query SMT solver on formula: $\exists \theta \in \Theta$: proof rule holds on points $s \in S$
  3. Verifier checks either $V(\cdot, x)$ valid over dense $\mathcal{D}$, or counterexample c : query SMT solver on formula $\exists c \in \mathcal{D}$: proof rule does not hold on c
  4. $S \leftarrow S \cup c$, loop back to 2

- sound, but not complete: infinite search space ($\theta \in \Theta$) and domain ($x \in \mathcal{D}$)

Neural Inductive Synthesis



1. Learner

   generates candidates V over finite set S
2. Verifier

   certifies validity on $\mathcal{D}$, or provides counterexample(s) c

- inductive synthesis loop

  1. sample (finite) set $S \subset \mathcal{D}$
  2. Learner trains neural $V(\theta, \cdot)$ via loss function L;
     L penalises deviations from conditions in proof rules
  3. Verifier checks either $V(\cdot, x)$ valid over dense $\mathcal{D}$, or counterexample c :
     query SMT solver on formula $\exists c \in \mathcal{D}$: proof rule does not hold on c
  4. $S \leftarrow S \cup c$, loop back to 2

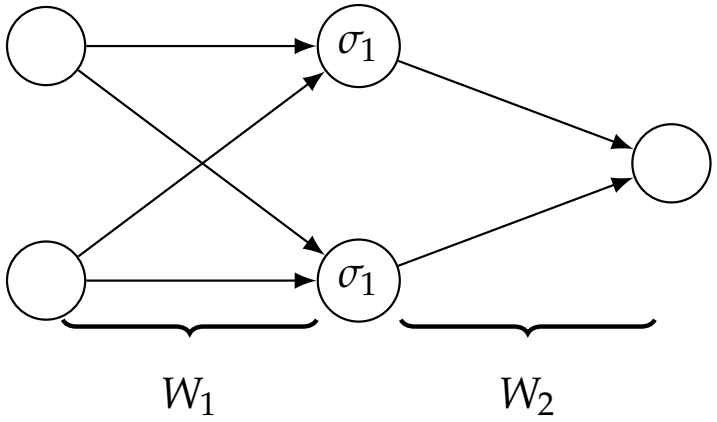
- sound, but not complete: infinite search space ($\theta \in \Theta$) and domain ($x \in \mathcal{D}$)

Neural networks as Lyapunov functions

- neural nets are general and flexible (universal function approximators)
- Learner trains shallow neural network

$$V(x) = W_2 \cdot \sigma_1(W_1 x + b_1)$$

(W_i weights, (σ_1) activation fcns)

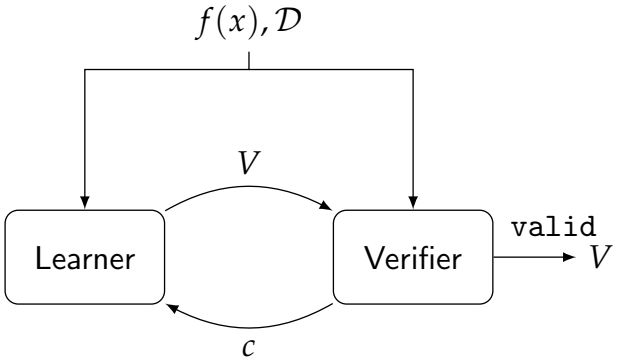


- loss function enforces Lyapunov conditions in (1)+(2) and (3) on points in S:

$$L(S) = \mid V(0) \mid + \sum_{s \in S} \max\{0, -V(s)\} + \sum_{s \in S} \max\{0, V(f(s)) - V(s) + \epsilon\}$$

- loss function L is *"pretty good"* proxy of synthesis formula

Neural networks as Lyapunov functions



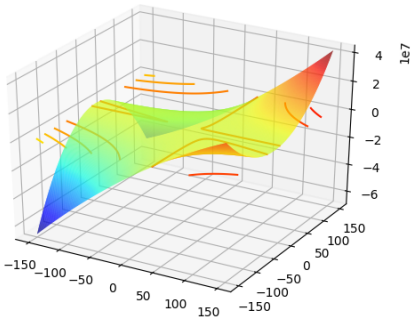
- surprisingly effective! Communication *Learner* $\leftrightarrow$ *Verifier* is crucial

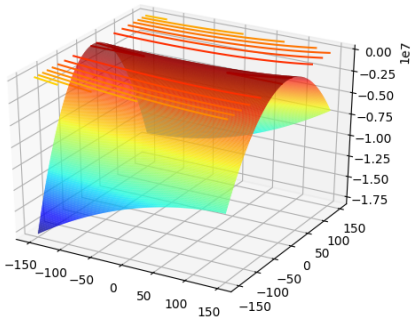
- loss function enforces Lyapunov conditions in (1)+(2) and (3) on points in S:

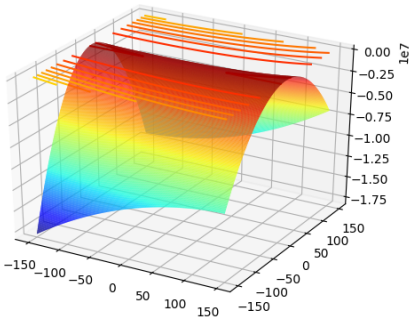
$$L(S) = \mid V(0) \mid + \sum_{s \in S} \max\{0, -V(s)\} + \sum_{s \in S} \max\{0, V(f(s)) - V(s) + \epsilon\}$$

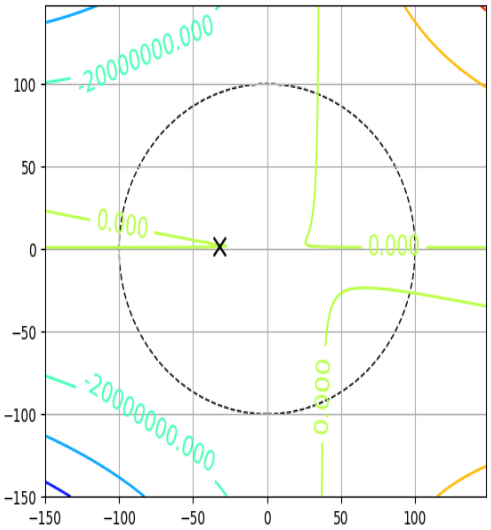
- loss function L is *"pretty good"* proxy of synthesis formula

Synthesis of Lyapunov functions - example

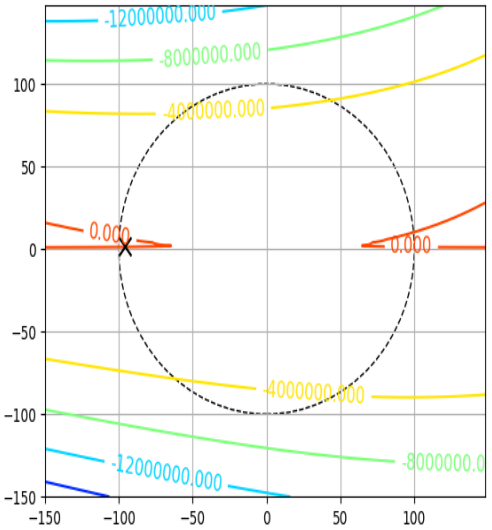




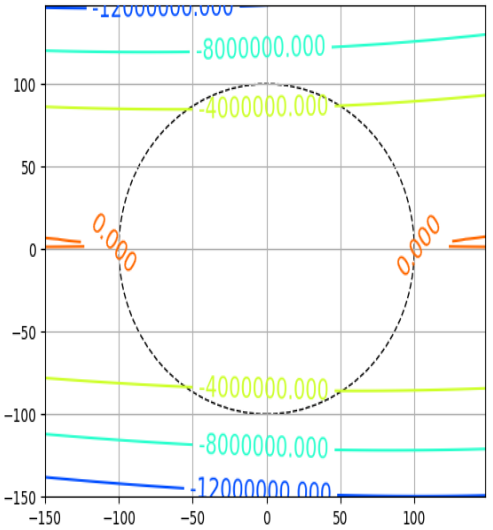




x c-ex

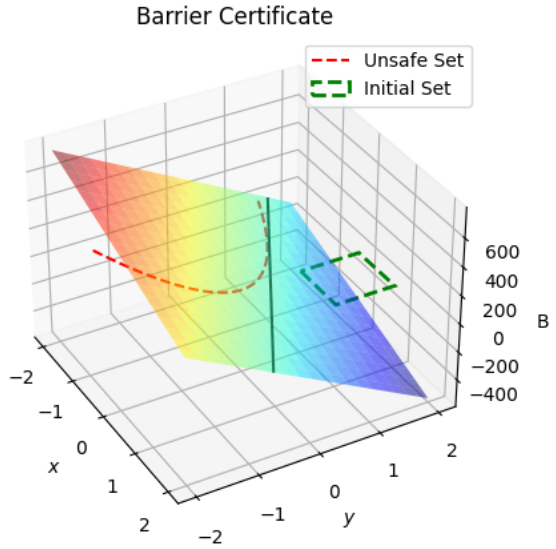


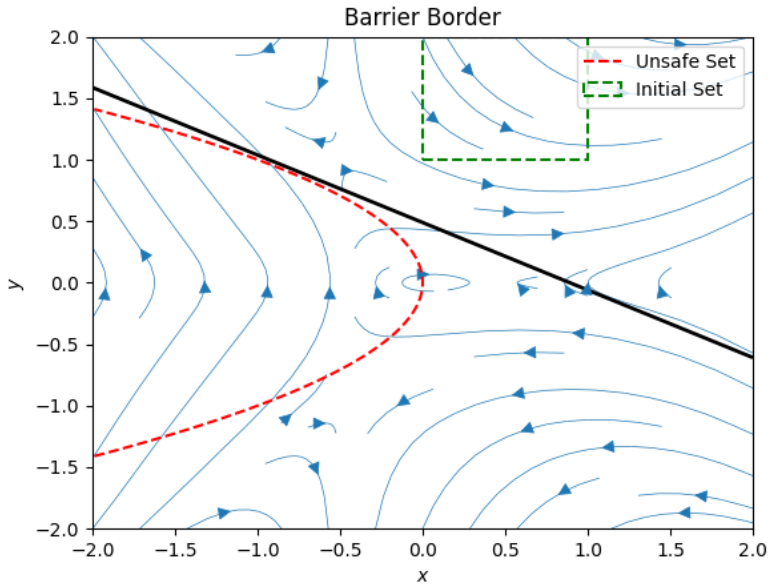
x c-ex



NO c-ex

Synthesis of barrier certificates - examples

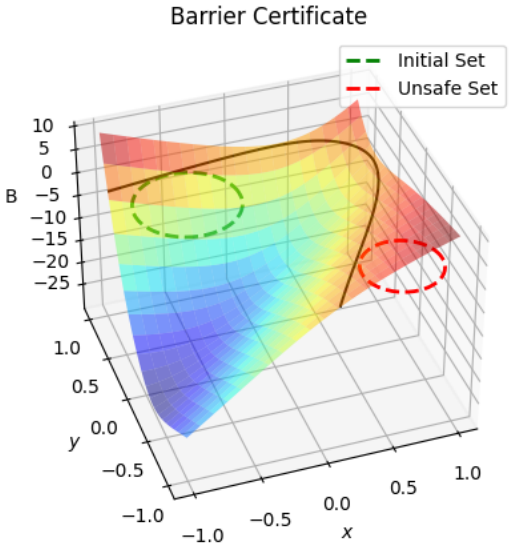


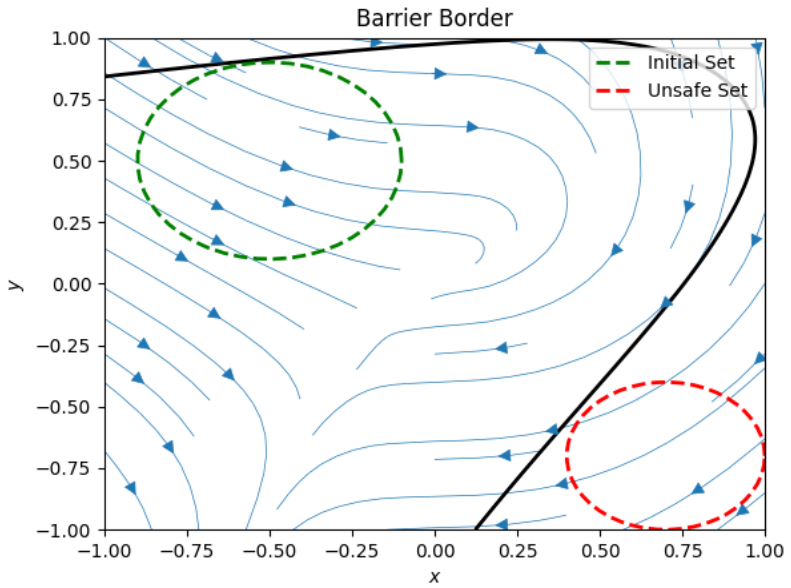


$$\begin{cases} x^+ &= y + 2xy \\ y^+ &= -x + 2x^2 - y^2 \end{cases}$$

[10] · Linear

Synthesis of barrier certificates - examples

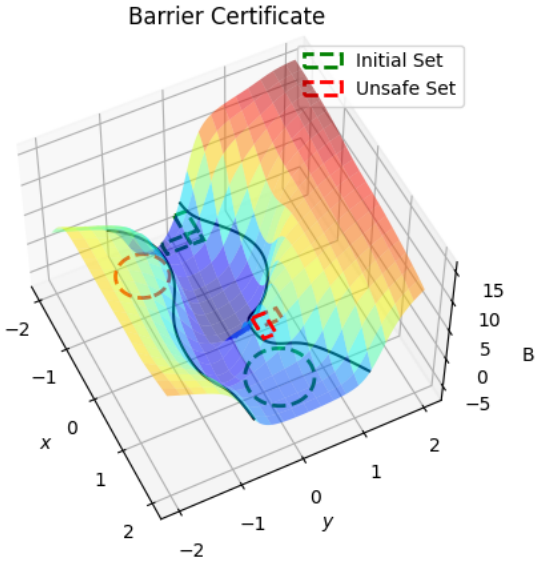


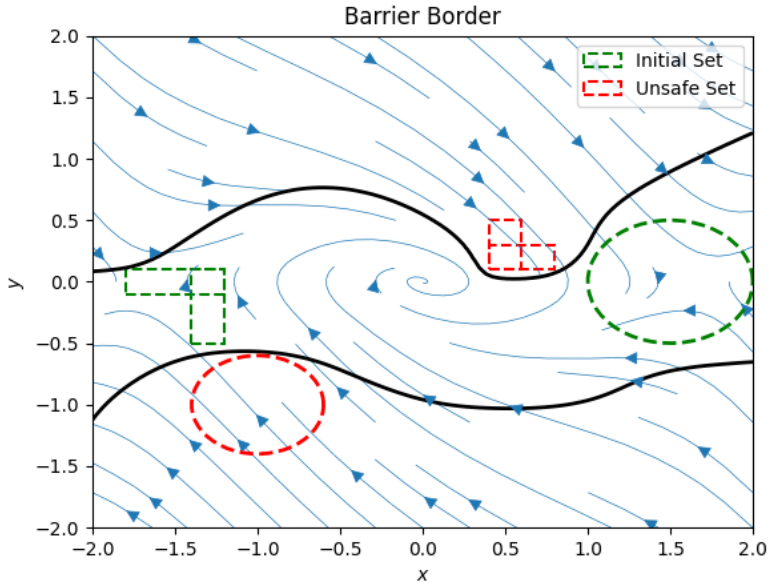


$$\begin{cases} x^+ &= \exp(-x) + y - 1 \\ y^+ &= -\sin(x)^2 \end{cases}$$

[20] · Softplus

Synthesis of barrier certificates - examples





$$\begin{cases} x^+ &= y \\ y^+ &= -x - y + \frac{1}{3}x^3 \end{cases}$$

$[20, 20] \cdot \text{Sigmoid}, \text{Sigmoid}$

Synthesis of control certificates for complex tasks

- dynamical models with inputs (a.k.a., external non-determinism)

$$x^+ = f(x, u)$$

$\rightarrow$ synthesis of "control certificates"

- modify known synthesis problem:

$$\exists V : \mathcal{D} \to \mathbb{R} \quad s.t. \quad \forall x \in \mathcal{D} \quad \text{conditions (1)} \wedge \text{(2)} \wedge \text{(3) hold}$$

Synthesis of control certificates for complex tasks

- dynamical models with inputs (a.k.a., external non-determinism)

$$x^+ = f(x, u)$$

$\rightarrow$ synthesis of "control certificates"

- approach:
  1. control policies are also NN-templated
  2. concurrent synthesis controls and corresponding certificates

Synthesis of control certificates for complex tasks

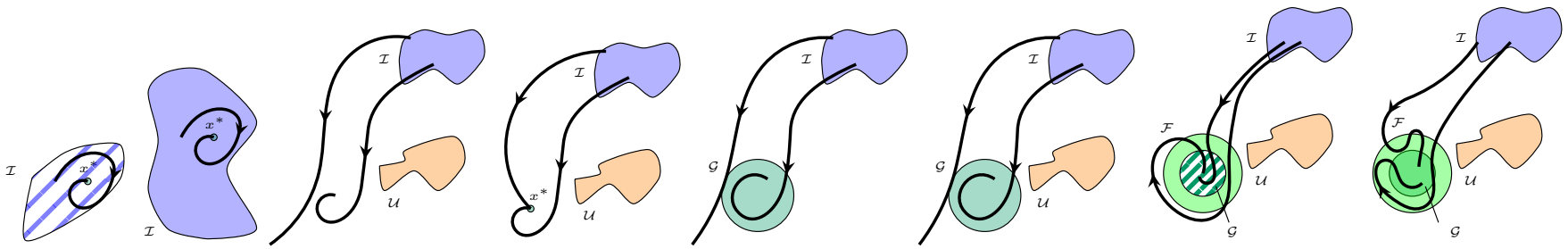
- dynamical models with inputs (a.k.a., external non-determinism)

$$x^+ = f(x, u)$$

$\rightarrow$ synthesis of "control certificates"

- (back to) broad class of properties/requirements

$$\forall x_0 \in \mathcal{I}, \quad \exists T \in \mathbb{N}, \quad \forall t \in \{0, \ldots, T-1\}, \quad \forall \tau \geq T : \\ x_T \in \mathcal{G}, \quad x_t \notin \mathcal{U}, \quad x_\tau \in \mathcal{F}$$



a: Stability b: ROA c: Safety d: SWA e: Reachability f: RWA g: RSWA h: RAR

Synthesis of control certificates for complex tasks

- dynamical models with inputs (a.k.a., external non-determinism)

$$x^+ = f(x, u)$$

→ synthesis of "control certificates"

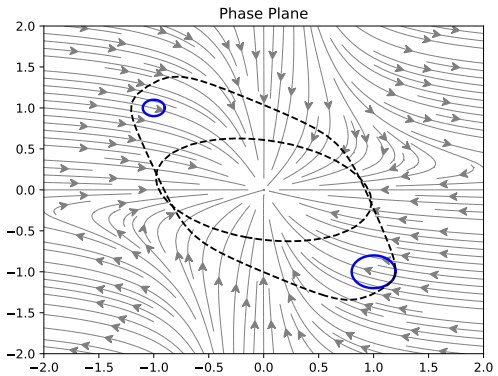
	N_s	N_u	Property	Neurons	Activations	T (s)			Success (%)
						min	μ	max	S
1	2	0	Stability	[6]	$[\varphi_2]$	0.01 ($\approx$ 0.00)	0.16 (0.15)	1.50 (1.48)	100
2	3	0	Stability	[8]	$[\varphi_2]$	0.28 ($\approx$ 0.00)	2.22 (0.45)	12.57 (3.31)	100
3	2	2	Stability	[4]	$[\varphi_2]$	0.07 (0.01)	0.19 (0.02)	0.47 (0.04)	100
4	2	2	Stability	[5]	$[\varphi_2]$	0.09 (0.01)	0.26 (0.02)	0.54 (0.03)	100
5	2	0	ROA	[5]	$[\sigma_{\text{soft}}]$	0.21 (0.12)	14.09 (12.59)	25.32 (22.13)	40
6	3	3	ROA	[8]	$[\varphi_2]$	1.24 (0.02)	39.08 (0.03)	287.89 (0.04)	100
7	2	0	Safety	[15]	$[\sigma_{\text{t}}]$	0.44 (0.35)	3.36 (2.90)	7.61 (7.11)	100
9	8	0	Safety	[10]	$[\varphi_1]$	12.63 (7.71)	51.97 (32.75)	70.59 (44.66)	70
10	3	1	Safety	[15]	$[\sigma_{\text{t}}]$	1.57 (0.19)	11.87 (2.50)	51.08 (7.52)	90
11	3	0	SWA	[6], [5]	$[\varphi_2]$, $[\sigma_{\text{t}}]$	0.19 (0.05)	2.46 (0.100)	12.10 (0.20)	90
12	2	0	SWA	[5], [5, 5]	$[\varphi_2]$, $[\sigma_{\text{sig}},\varphi_2]$	0.13 (0.06)	0.27 (0.14)	0.39 (0.20)	100
13	2	1	SWA	[8], [5]	$[\varphi_2]$, $[\varphi_2]$	0.06 (0.03)	0.20 (0.10)	0.58 (0.24)	90
14	3	1	SWA	[10], [8]	$[\varphi_2]$, $[\sigma_{\text{t}}]$	4.06 (0.87)	19.81 (2.73)	103.49 (7.23)	90
15	2	0	RWA	[4]	$[\varphi_2]$	0.14 (0.09)	1.81 (1.75)	4.70 (4.63)	100
16	3	0	RWA	[16]	$[\varphi_2]$	1.36 (0.09)	14.10 (0.14)	72.97 (0.20)	90
17	2	1	RWA	[4, 4]	$[\sigma_{\text{sig}},\varphi_2]$	0.59 (0.27)	6.82 (3.32)	20.07 (11.46)	100
18	3	1	RWA	[5]	$[\varphi_2]$	0.46 (0.11)	16.06 (5.81)	72.47 (44.64)	80
19	2	2	RWA	[5]	$[\sigma_{\text{sig}}]$	0.69 (0.40)	1.38 (0.94)	2.14 (1.90)	100
20	2	0	RSWA	[4]	$[\varphi_2]$	0.19 (0.03)	1.29 (1.04)	3.79 (3.37)	100
21	3	0	RSWA	[16]	$[\varphi_2]$	4.81 (0.13)	27.14 (0.19)	80.95 (0.25)	100
22	2	0	RSWA	[5, 5]	$[\sigma_{\text{sig}},\varphi_2]$	1.52 (0.06)	4.45 (0.19)	10.97 (0.35)	100
23	2	1	RSWA	[8]	$[\varphi_2]$	0.21 (0.05)	0.67 (0.25)	1.19 (0.91)	100
24	2	2	RSWA	[5, 5]	$[\sigma_{\text{sig}},\varphi_2]$	0.98 (0.16)	1.23 (0.28)	1.61 (0.46)	100
25	2	0	RAR	[6], [6]	$[\sigma_{\text{soft}}]$, $[\varphi_2]$	6.65 (1.08)	24.74 (6.46)	77.80 (15.06)	100
26	2	2	RAR	[6, 6], [6, 6]	$[\sigma_{\text{sig}},\varphi_2]$, $[\sigma_{\text{sig}},\varphi_2]$	5.13 (1.34)	26.99 (9.90)	101.23 (60.14)	100

Synthesis of control certificates for complex tasks

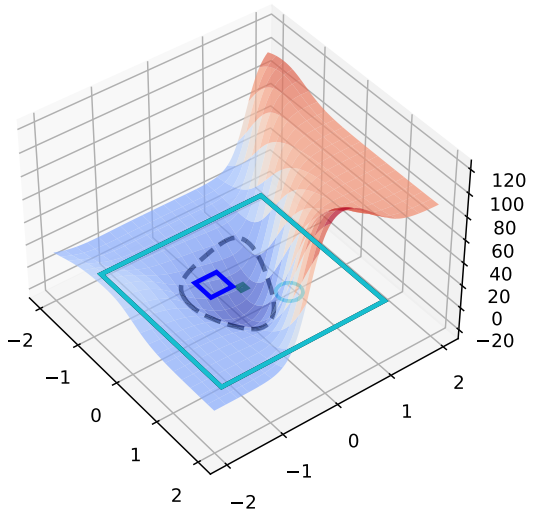
- dynamical models with inputs (a.k.a., external non-determinism)

$$x^+ = f(x, u)$$

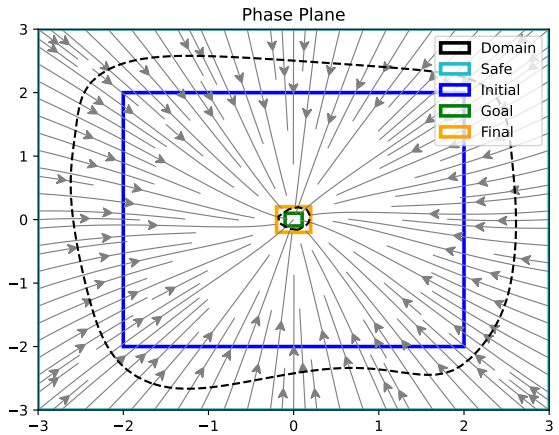
$\rightarrow$ synthesis of "control certificates"



ROA for NL model,
non-poly Lyapunov,
2 disjoint initial sets



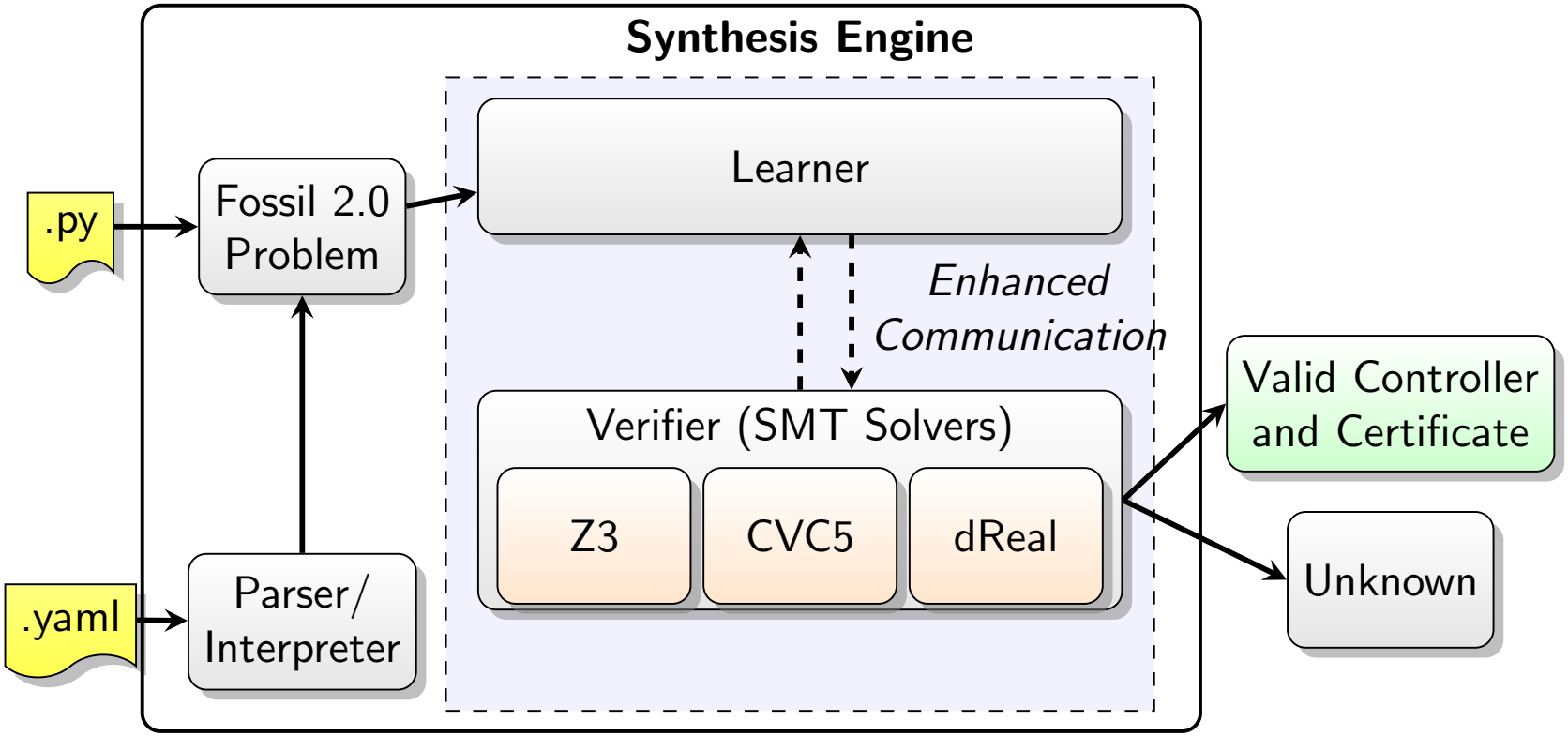
RWA: reach-while-avoid



RAR certificate for closed-loop NL model

dashed lines: level sets; dark blue: $\mathcal{I}$; light blue: $\mathcal{S}$; green: $\mathcal{G}$; orange: $\mathcal{F}$

Software for Neural Synthesis - Fossil 2.0



`github.com/oxford-oxcav/fossil`

Ranking certificates as supermartingales

- almost-sure (w.p. 1) countable-state program termination
- can be extended to
  - continuous state spaces
  - control synthesis
  - quantitative termination (e.g., via supermartingale inequalities)

Ranking certificates as supermartingales

- almost-sure (w.p. 1) countable-state program termination
- can be extended to
  - continuous state spaces
  - control synthesis
  - quantitative termination (e.g., via supermartingale inequalities)

. . . and to whole-LTL quantitative model checking and design of gMDP

Proof rule for almost-sure Streett acceptance

- consider Markov chain $\mathtt{x}^+ = f(\mathtt{x}, \mathtt{w})$, along with Streett pair:

$$\mathsf{GF}(A) \implies \mathsf{GF}(B) \qquad\qquad A, B \subseteq \mathsf{State}$$

Streett Supermartingale Proof Rule [AA et al, CAV24]

$$\exists \text{ function } V : \mathsf{State} \to \mathbb{R}_{\geq 0},$$

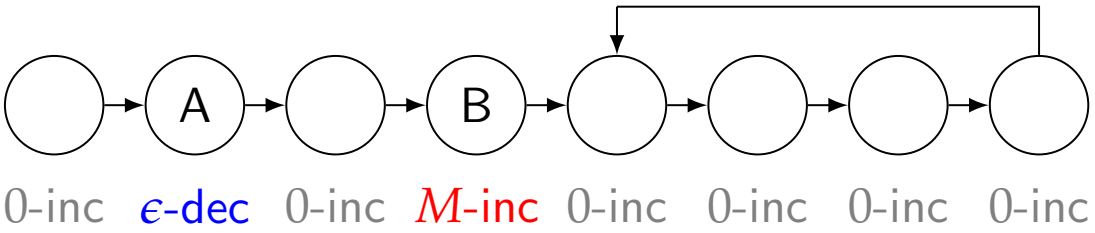
- **In B:** allow V to increase in expectation by M (positive const)
- **In $A \setminus B$:** decrease V in expectation by ϵ (positive const)
- **Otherwise:** ensure V does not increase in expectation

V is **non-negative** and **almost** a supermartingale

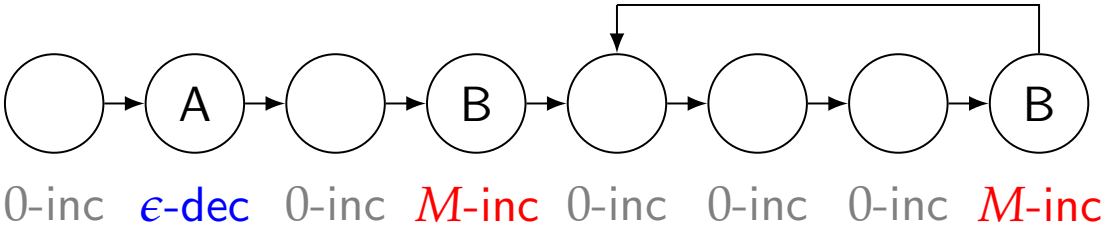
allow increase in B

Proof rule for almost-sure Streett acceptance

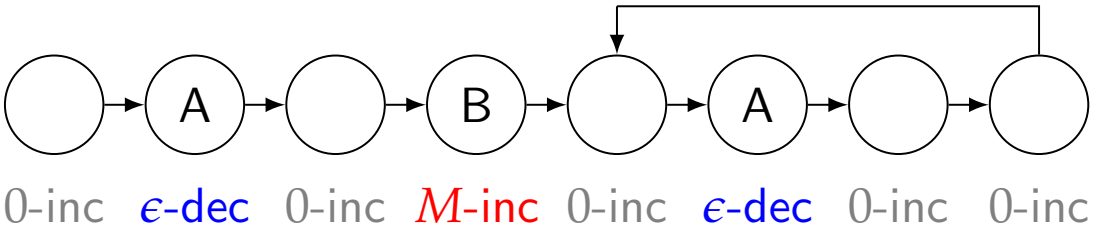
$$\mathrm{GF}(\mathrm{A}) \implies \mathrm{GF}(\mathrm{B})?$$



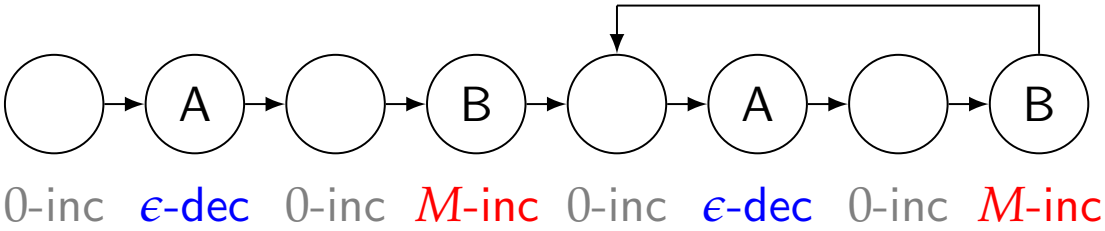
✓ finite A's, finite B's



✓ finite A's, infinite B's



✗ infinite A's, finite B's



✓ infinite A's, infinite B's

Experiments

- I - invariance generation
- V - verification (certificate synthesis)
- C - control synthesis

Benchmark	ω-Regular Specification	Output	Time [s]
`EvenOrNegative`	$\mathsf{GF}(x \text{ even}) \vee \mathsf{FG}(x < 0)$	V	0.09
`SafeRWalk1`	$\mathsf{G}(x < 100)$	VIC	1.09
`PersistRW`	$\mathsf{FG}(x \leq 10)$	VI	1.16
`RecurRW`	$\mathsf{GF}(x > 100)$	VI	1.49
`SafeRWalk2`	$\mathsf{G}(x \geq 10)$	VIC	1.09
`GuaranteeRW`	$\mathsf{G}(x \geq -10) \rightarrow \mathsf{F}(x \geq 10^3)$	VI	5.61
`Temperature1`	$\mathsf{FG}(\neg \mathsf{Hot} \wedge \neg \mathsf{Cold})$	VIC	4.11
`Temperature2`	$\mathsf{GF}(x \leq 30) \wedge \mathsf{G}(x \leq 60)$	VI	28.93
`Temperature3`	$\mathsf{G}(\mathsf{Safe}) \wedge [\mathsf{GF}(\mathsf{Cold}) \rightarrow \mathsf{GF}(\mathsf{Hot})]$	VIC	28.58
`Temperature4`	$\mathsf{G}(\mathsf{Safe}) \wedge [\mathsf{GF}(\mathsf{Cold}) \rightarrow \mathsf{GF}(\mathsf{Hot})]$	VC	4.64
`FinMemoryControl`	$\mathsf{GF}(x \leq 0) \rightarrow \mathsf{GF}(x \geq 100)$	VIC	16.73

Outline

1. Context
2. Neural Proofs - Formal Synthesis of Neural Certificates
3. Synthesis of Formal Abstractions

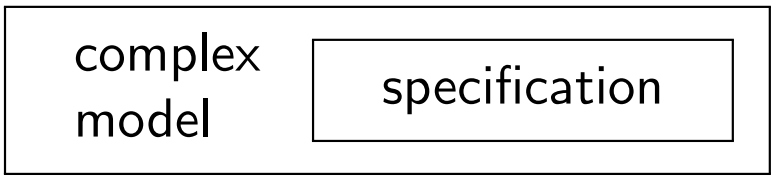
Outline

1 Context

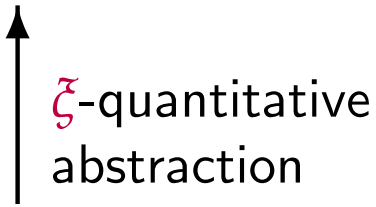
2 Neural Proofs - Formal Synthesis of Neural Certificates

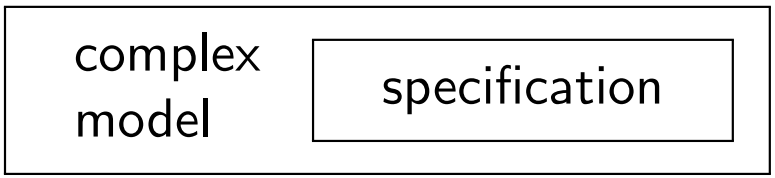
3 Synthesis of Formal Abstractions

Formal abstractions

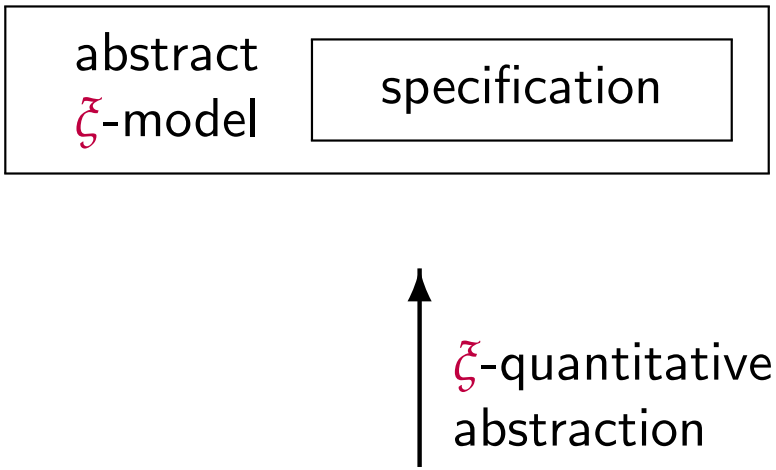


Formal abstractions

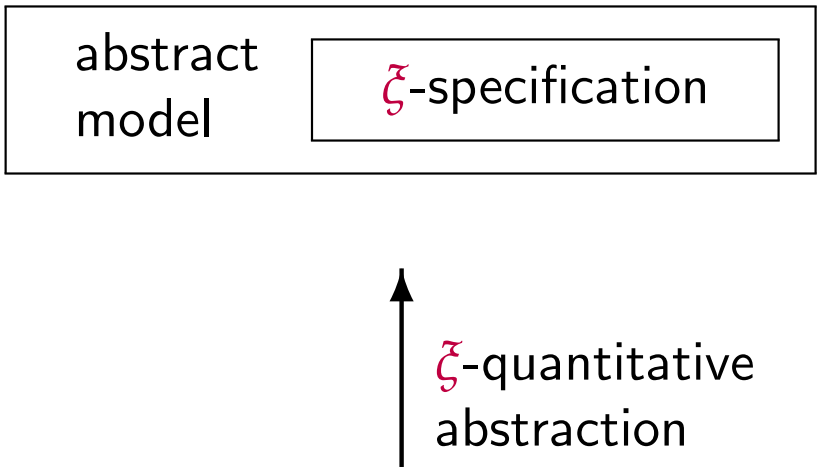




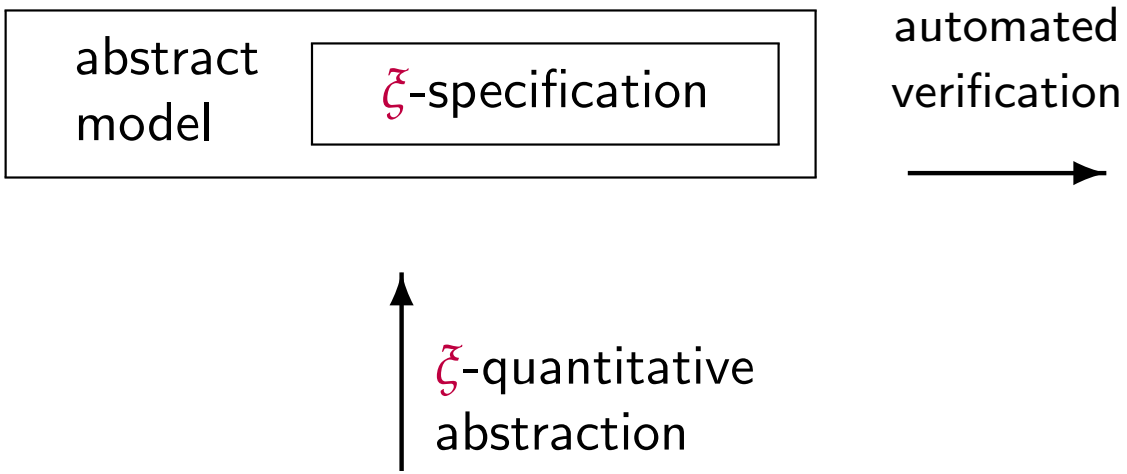
Formal abstractions



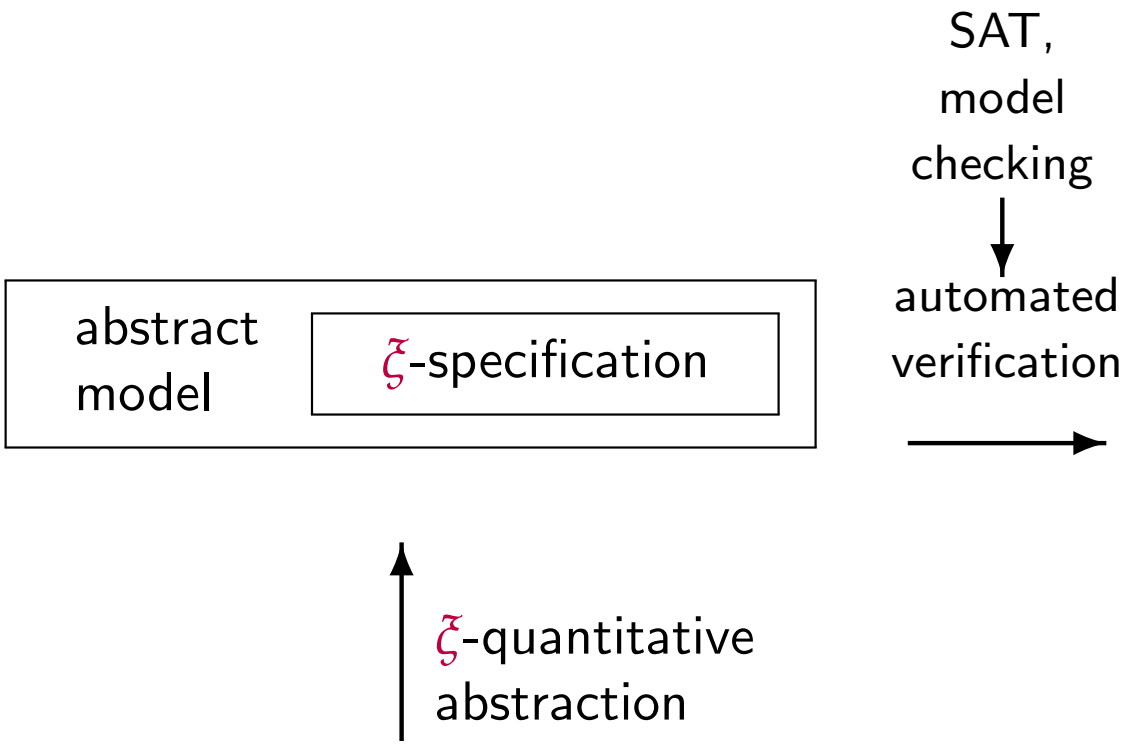
Formal abstractions



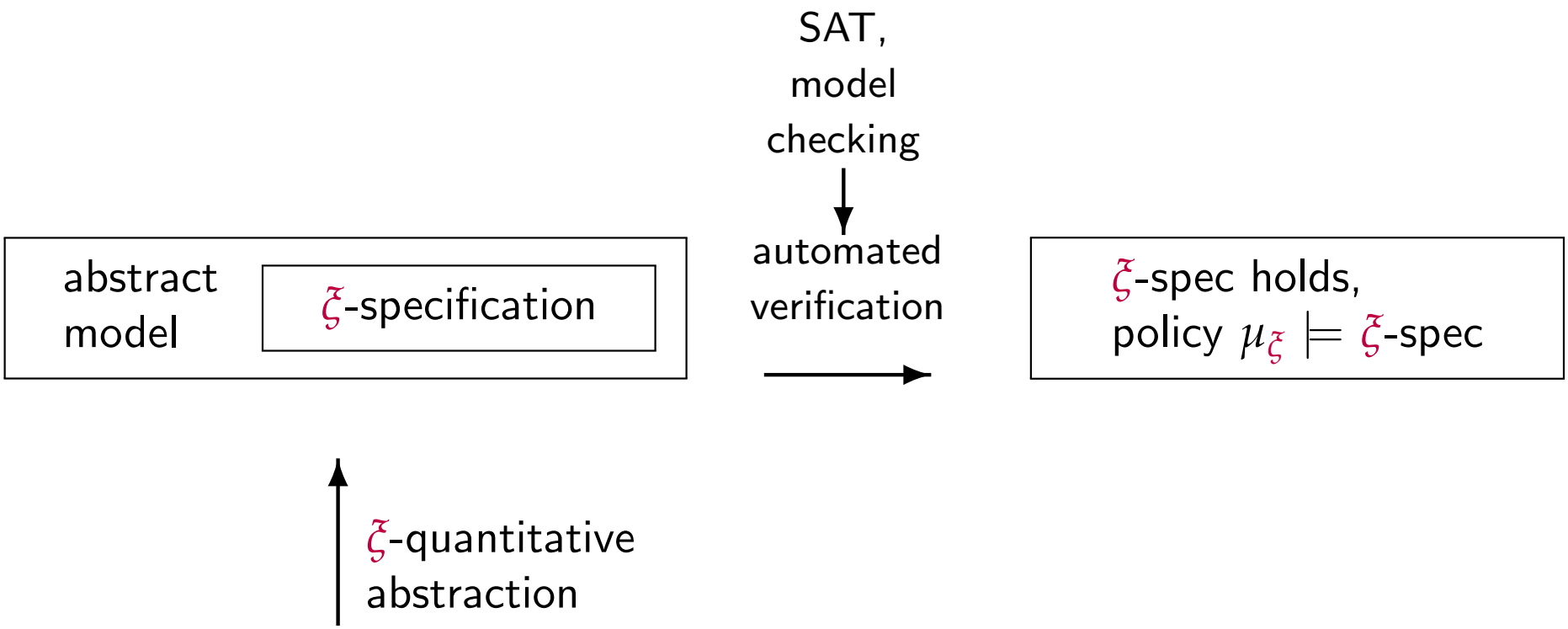
Formal abstractions



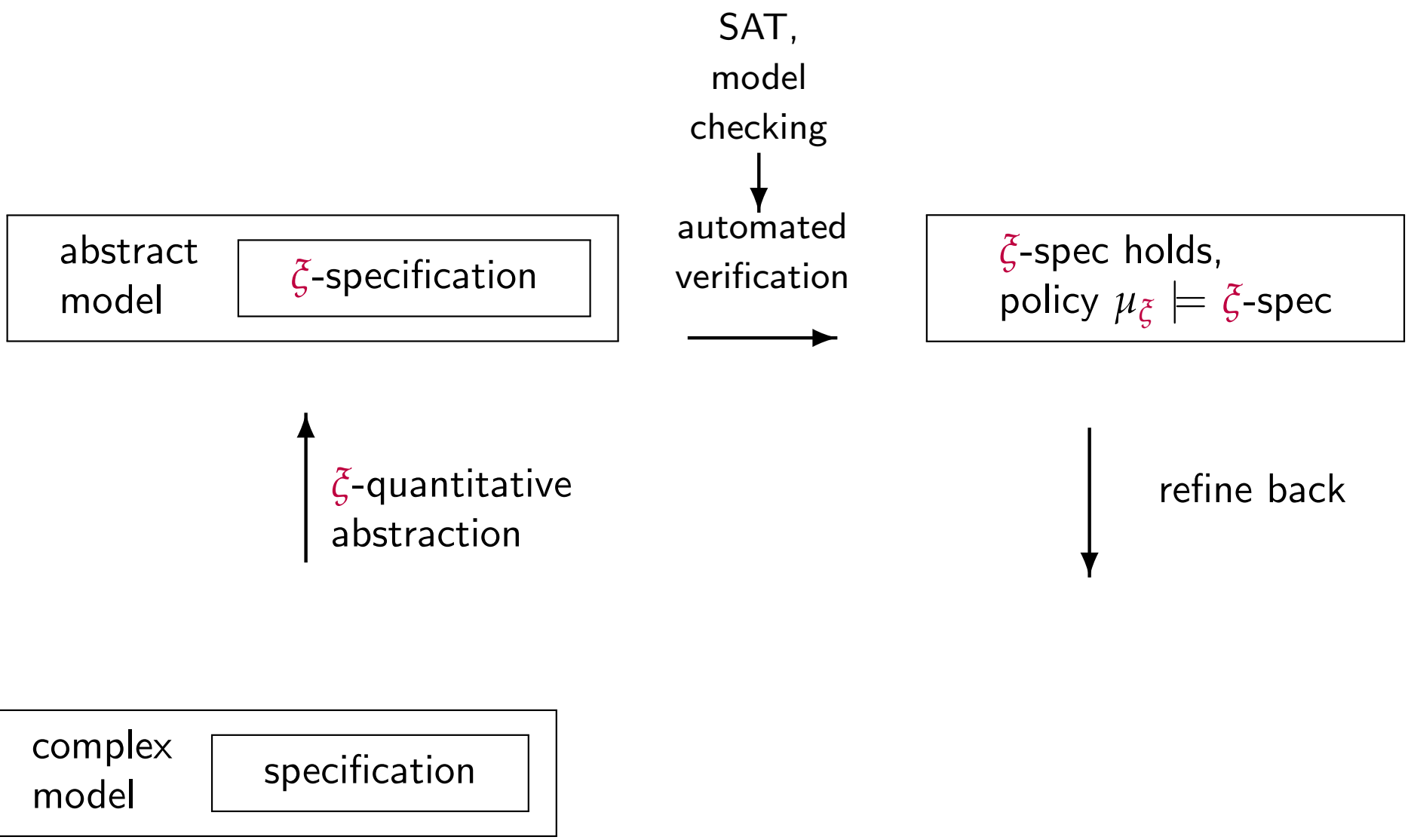
Formal abstractions



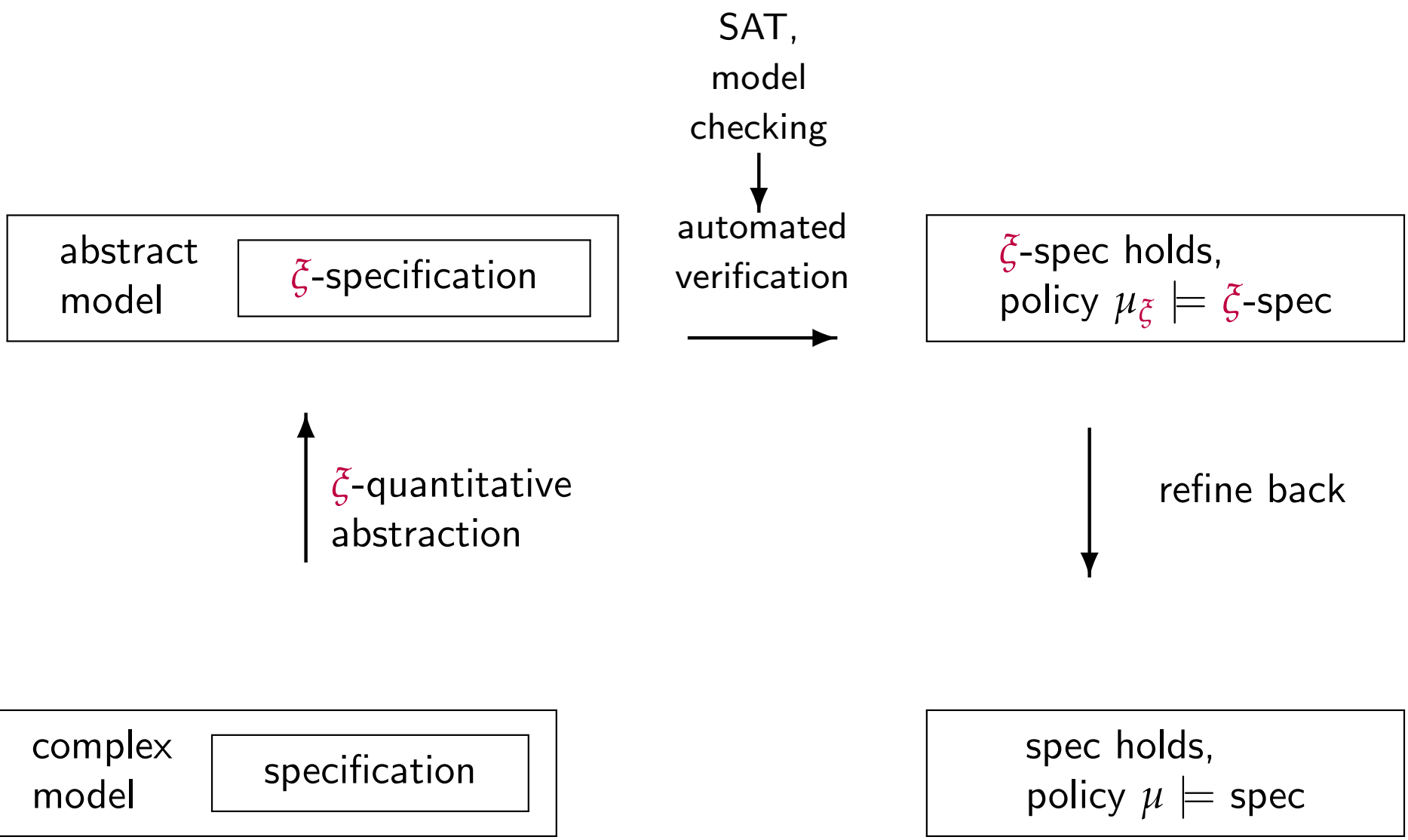
Formal abstractions



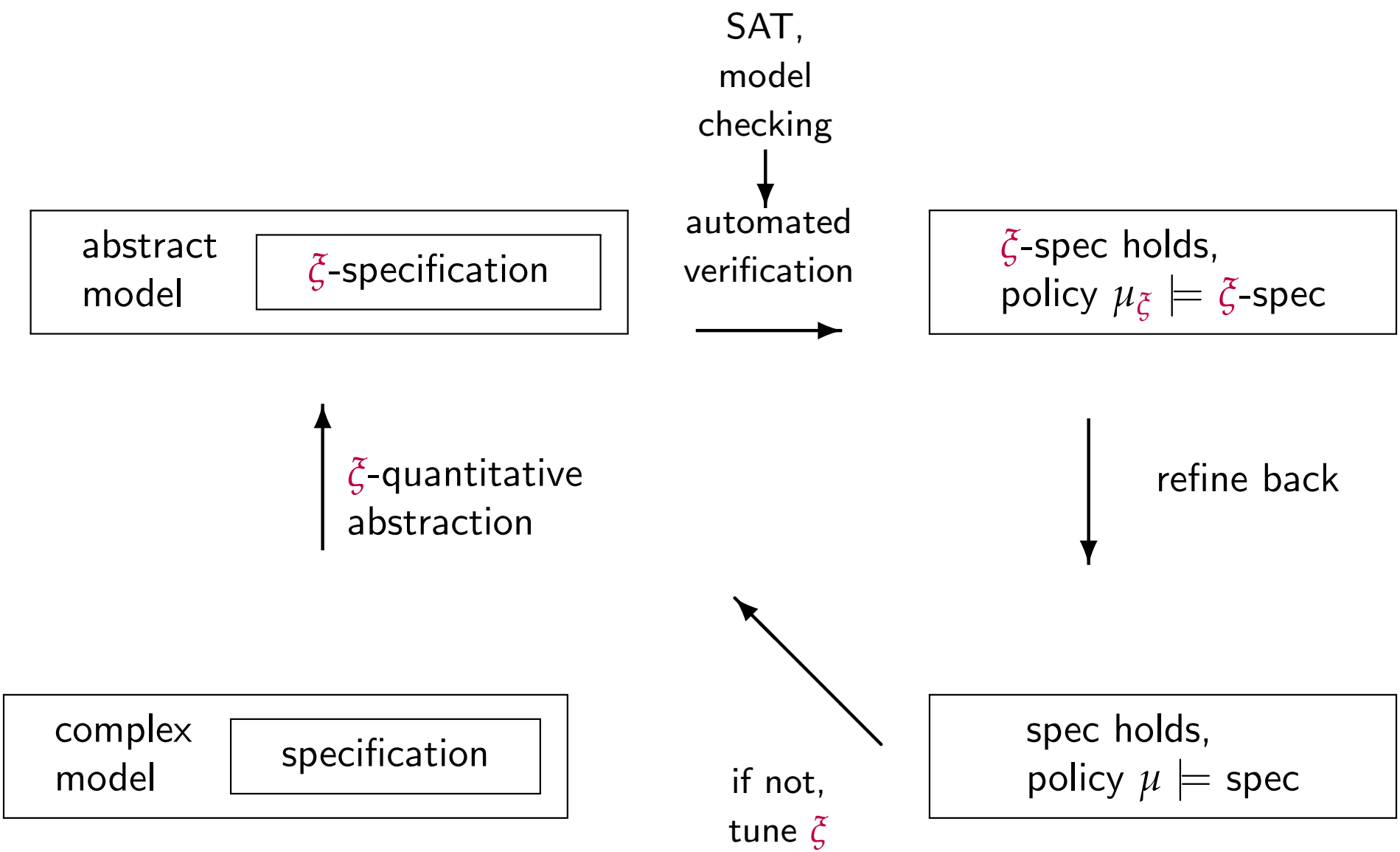
Formal abstractions



Formal abstractions



Formal abstractions



1. Context

2. Neural Proofs - Formal Synthesis of Neural Certificates

3. Synthesis of Formal Abstractions

Thank you for your attention

oxcav.web.ox.ac.uk

All images used are under Wikimedia CCAS license, or by author

Selected References on Inductive (SMT and Neural) Synthesis (of Certificates/Controllers/Models)

L. Rickard, A. Abate, K. Margellos, "Data-Driven Neural Certificate Synthesis," arXiv:2502.05510, 2025.

A. Edwards, A. Peruffo and A. Abate, "Fossil 2.0: Design, Usage and Impact of a Software Tool for Verification and Control of Dynamical Models," Science of Computer Programming, v. 247, Jan 2026.

A. Abate, M. Giacobbe, and D. Roy, "Quantitative Supermartingale Certificates," CAV25, LNCS 15934, pp. 3-28, 2025.

A. Edwards, A. Peruffo and A. Abate, "A General Verification Framework for Dynamical and Control Models via Certificate Synthesis," arXiv:2309.06090, 2024.

A. Abate, M. Giacobbe, and D. Roy, "Stochastic Omega-Regular Verification and Control with Supermartingales," CAV24, LNCS 14683, pp. 395-419, 2024.

A. Edwards, A. Peruffo and A. Abate, "Fossil 2.0: Formal Certificate Synthesis for the Verification and Control of Dynamical Models," HSCC24, In Press, 2024.

A. Abate, A. Edwards, M. Giacobbe, H. Punchihewa, and D. Roy, "Quantitative Neural Verification of Probabilistic Programs," CONCUR23, arXiv:2301.06136, 2023.

D. Roy, M. Giacobbe, and A. Abate, "Learning Probabilistic Termination Proofs," CAV21, LNCS 12760, pp. 3–26, 2021.

A. Abate, D. Ahmed, A. Edwards, M. Giacobbe and A. Peruffo, "FOSSIL: A Software Tool for the Formal Synthesis of Lyapunov Functions and Barrier Certificates using Neural Networks," HSCC21, pp. 1-11, 2021.

A. Abate, D. Ahmed and A. Peruffo, "Automated Formal Synthesis of Neural Barrier Certificates for Dynamical Models," TACAS21, LNCS 12651, pp. 370–388, 2021.

D. Ahmed, A. Peruffo and A. Abate, "Automated and Sound Synthesis of Lyapunov Functions with SMT Solvers," TACAS20, LNCS 12078, pp. 97-114, 2020.

A. Abate, D. Ahmed, M. Giacobbe and A. Peruffo "Automated Formal Synthesis of Lyapunov Neural Networks," IEEE Control Systems Letters, 5 (3), 773-778, 2020.

A. Edwards, M. Giacobbe, and A. Abate, "On the Trade-off Between Efficiency and Precision of Neural Abstraction," QEST23, LNCS 14287, pp. 152-171, 2023.

A. Abate, A. Edwards, and M. Giacobbe, "Neural Abstractions," NeurIPS22, Advances in Neural Information Processing Systems 35, 26432-26447, 2022.

A. Abate, H. Barbosa, C. Barrett. C. David, P. Kesseli, D. Kroening, E. Polgreen, A. Reynolds, C. Tinelli, "Synthesising programs with non-trivial constants," Journal of Automated Reasoning (JAR), 67(2):19, 2023.

A. Abate, I. Bessa, D. Cattaruzza, L. Cordeiro, C. David, P. Kesseli, D. Kroening and E. Polgreen, "Automated Formal Synthesis of Provably Safe Digital Controllers for Continuous Plants," Acta Informatica, 57(3), 2020.

Selected Journal References on Model- and Sample-Based Formal Abstractions (Not discussed in this talk)

T. Badings, L Romao, A. Abate, D. Parker, H. Poonawala, M. Stoelinga and N. Jansen, "Robust Control for Dynamical Systems with Non-Gaussian Noise via Formal Abstractions," JAIR, vol 76, pp.341-391, 2023.

T.S. Badings, A. Abate, N. Jansen, D. Parker, H.A. Poonawala, and M. Stoelinga, "Sampling-Based Robust Control of Autonomous Systems with Non-Gaussian Noise," AAAI22, 36 (9), pp. 9669-9678, 2022.

A. Lavaei, S. Soudjani, A. Abate, and M. Zamani, "Automated Verification and Synthesis of Stochastic Hybrid Systems: A Survey," Automatica, vol. 146, Dec. 2022.

L. Laurenti, M. Lahijanian, A. Abate, L. Cardelli and M. Kwiatkowska, "Formal and Efficient Control Synthesis for Continuous-Time Stochastic Processes," IEEE Transactions on Automatic Control, vol. 66, no. 1, pp. 17-32, Jan 2021.

S. Haesaert, S.E.Z. Soudjani, and A. Abate, "Verification of general Markov decision processes by approximate similarity relations and policy refinement," SIAM Journal on Control and Optimisation, vol. 55, nr. 4, pp. 2333-2367, 2017.

I. Tkachev, A. Mereacre, J.-P. Katoen, and A. Abate, "Quantitative Model Checking of Controlled Discrete-Time Markov Processes," Information and Computation, vol. 253, nr. 1, pp. 1–35, 2017.

S. Haesaert, N. Cauchi and A. Abate, "Certified policy synthesis for general Markov decision processes: An application in building automation systems," Performance Evaluation, vol. 117, pp. 75-103, 2017.

S.E.Z. Soudjani and A. Abate, "Aggregation and Control of Populations of Thermostatically Controlled Loads by Formal Abstractions," IEEE Transactions on Control Systems Technology. vol. 23, nr. 3, pp. 975–990, 2015.

S.E.Z. Soudjani and A. Abate, "Quantitative Approximation of the Probability Distribution of a Markov Process by Formal Abstractions," Logical Methods in Computer Science, Vol. 11, nr. 3, Oct. 2015.

M. Zamani, P. Mohajerin Esfahani, R. Majumdar, A. Abate, and J. Lygeros, "Symbolic control of stochastic systems via approximately bisimilar finite abstractions," IEEE Transactions on Automatic Control, vol. 59 nr. 12, pp. 3135-3150, Dec. 2014.

I. Tkachev and A. Abate, "Characterization and computation of infinite horizon specifications over Markov processes," Theoretical Computer Science, vol. 515, pp. 1-18, 2014.

S. Soudjani and A. Abate, "Adaptive and Sequential Gridding for Abstraction and Verification of Stochastic Processes," SIAM Journal on Applied Dynamical Systems, vol. 12, nr. 2, pp. 921-956, 2013.

A. Abate, J.P Katoen, J. Lygeros and M. Prandini, "Approximate Model Checking of Stochastic Hybrid Systems," European Journal of Control, 16(6), 624-641, 2010.

A. Abate, M. Prandini, J. Lygeros and S. Sastry, "Probabilistic Reachability and Safety Analysis of Controlled Discrete-Time Stochastic Hybrid Systems," Automatica, 44(11), 2724-2734, Nov. 2008.

Almost-sure proof rules

so far we have seen that:

- ranking supermartingales for almost-sure reachability **(Chakarov/Sankaranarayanan, CAV'13)**
- almost sure persistence & recurrence **(Chakarov/Voronin/Sankaranarayanan, TACAS'16)**
- Streett supermartingales for almost-sure reactivity conditions **(AA et. al., CAV'24)**

and indeed **many other similar proof rules** ...

Almost-sure proof rules

so far we have seen that:

- ranking supermartingales for almost-sure reachability **(Chakarov/Sankaranarayanan, CAV'13)**
- almost sure persistence & recurrence **(Chakarov/Voronin/Sankaranarayanan, TACAS'16)**
- Streett supermartingales for almost-sure reactivity conditions **(AA et. al., CAV'24)**

  and indeed **many other similar proof rules** ...
- restrict state space to **supporting invariant** $I \subseteq$ State, so that:

$$\boxed{\mathbb{P}\left(\text{ Specification } \mid \mathbf{G}\ I\ \right) = 1}$$

Almost-sure proof rules

so far we have seen that:

- ranking supermartingales for almost-sure reachability **(Chakarov/Sankaranarayanan, CAV'13)**
- almost sure persistence & recurrence **(Chakarov/Voronin/Sankaranarayanan, TACAS'16)**
- Streett supermartingales for almost-sure reactivity conditions

  (AA et. al., CAV'24)

  and indeed **many other similar proof rules** ...
- restrict state space to **supporting invariant** $I \subseteq$ State, so that:

$$\boxed{\mathbb{P}\left(\text{ Specification } \mid \mathbf{G}\ I\ \right) = 1}$$

- now, we can generalise any **almost-sure proof rule** to **quantitative** supermartingale certificates:

$$\boxed{\mathbb{P}\left(\text{ Specification }\right) \geq p}$$

- this is done via a "decomposition"

  (AA et. al., CAV'25)

Decomposing Quantitative Probabilistic Verification

- soundness of decomposition - suppose:
  1. $\mathbb{P}(\text{ Specification } \mid \mathbf{G}\ I\ ) = 1$
  2. $\mathbb{P}(\ \mathbf{G}\ I\ ) \geq p$
- then $\mathbb{P}(\text{ Specification }) \geq p$

Decomposing Quantitative Probabilistic Verification

- soundness of decomposition - suppose:
  1. $\mathbb{P}(\text{ Specification } | \mathbf{G}\ I\ ) = 1$
  2. $\mathbb{P}(\ \mathbf{G}\ I\ ) \geq p$
- then $\mathbb{P}(\text{ Specification }) \geq p$

- special case:
  1. if $\mathbb{P}(\text{ Streett acceptance } |\ \mathbf{G}\ I\ ) = 1$
     (e.g., via Streett supermartingale & supporting invariant, [AA et. al., CAV'24])
  2. if $\mathbb{P}(\ \mathbf{G}\ I\ ) \geq p$
     (e.g., via stochastic invariant (supermartingale) [Kushner, 1965] or repulsing supermartingales [Chatterjee et.al., POPL'17][Takisaka et.a., ATVA'18])
- then $\mathbb{P}(\text{ Streett acceptance }) \geq p$

Completeness of Decomposition

- **question:** does there always exist a region $I \subseteq$ State such that:
  1. $\mathbb{P}(\text{ Specification } \mid \mathbf{G}\ I\ ) = 1$,
  2. $\mathbb{P}(\ \mathbf{G}\ I\ ) = \mathbb{P}(\text{ Specification })$

[5] A language is invariant under addition/deletion of any finite prefix to an infinite trajectory

Completeness of Decomposition

- **question:** does there always exist a region $I \subseteq \text{State}$ such that:
  1. $\mathbb{P}(\ \text{Specification} \mid \mathbf{G}\ I\ ) = 1$,
  2. $\mathbb{P}(\ \mathbf{G}\ I\ ) = \mathbb{P}(\ \text{Specification}\ )$
- **poositive answer:** suppose L is a shift-invariant specification[5], then the following holds:

Theorem (ε-completeness for gMC)

For every $\varepsilon > 0$, there exists a region I such that

1. $\mathbb{P}(\ L \mid \mathbf{G}\ I\ ) = 1$
2. $\mathbb{P}(\ \mathbf{G}\ I\ ) \geq \mathbb{P}(\ L\ ) - \varepsilon$

Theorem (Completeness for finite MC)

There exists a region I such that

1. $\mathbb{P}(\ L \mid \mathbf{G}\ I\ ) = 1$
2. $\mathbb{P}(\ \mathbf{G}\ I\ ) = \mathbb{P}(\ L\ )$

[5] A language is invariant under addition/deletion of any finite prefix to an infinite trajectory