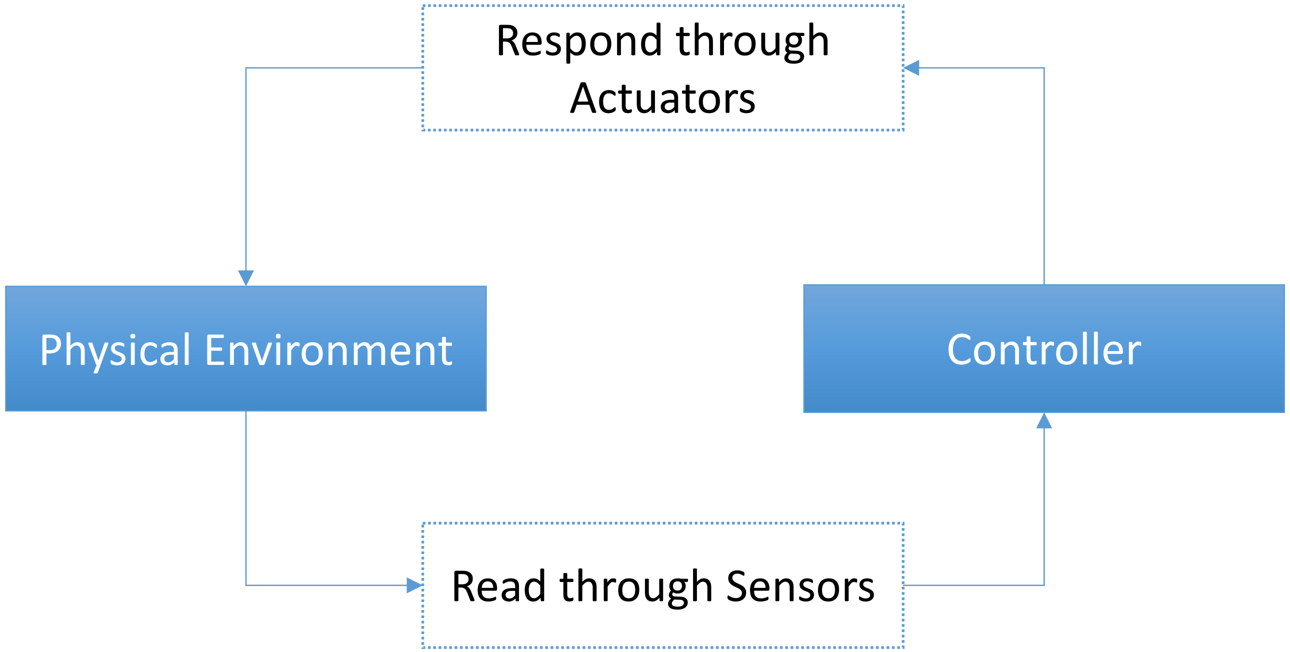
Robustness as Remedy for Model Checking Cyber-Physical Systems

Nima Roohi
University of Pennsylvania

Applications of Formal Methods to Control Theory and Dynamical Systems
June 23, 2018

Cyber-Physical Systems

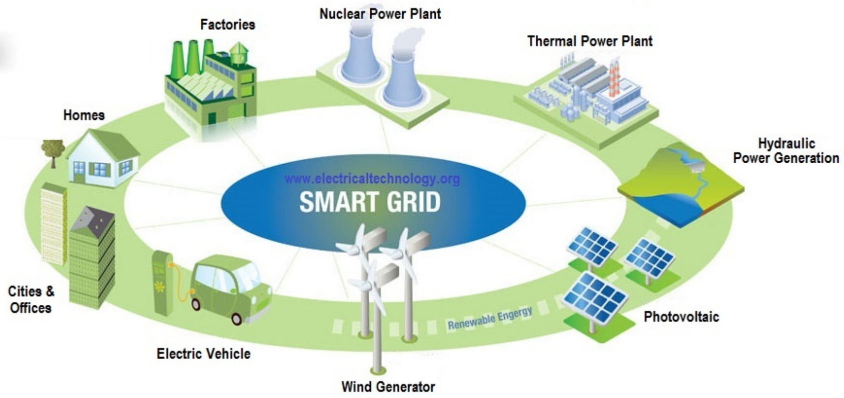
What are they? Where they are?



Cyber-Physical Systems

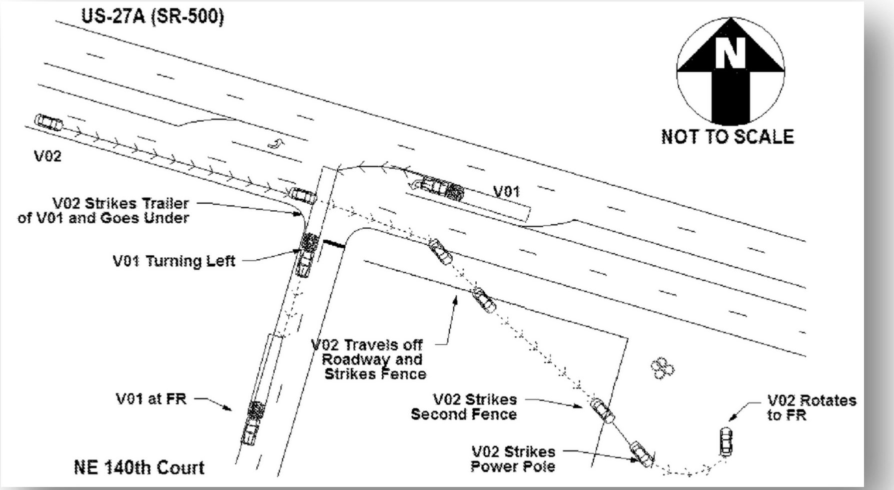
What are they? Where they are?

Respond through



Cyber-Physical Systems
What do we want?

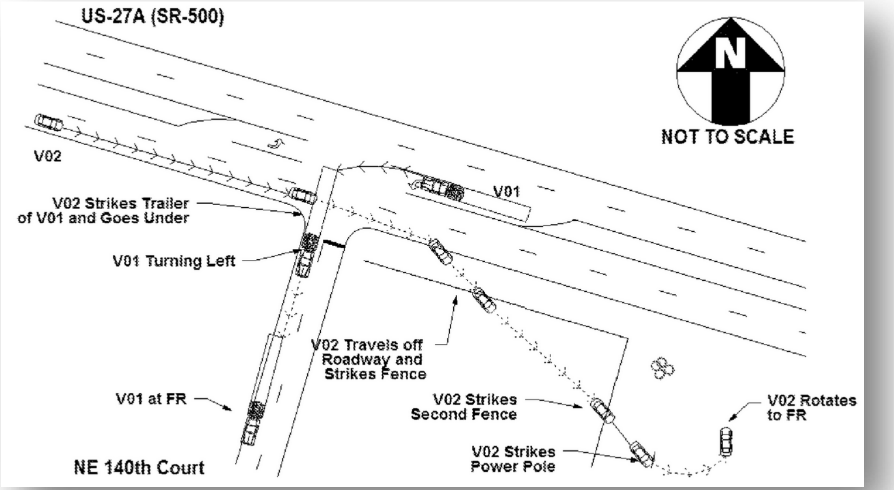
- Safety
  - Something bad never happens



Cyber-Physical Systems

What do we want?

- Safety
  - Something bad never happens
- Liveness
  - Something good will eventually happen



Cyber-Physical Systems
Formal Verification is a Necessity

- System failures are very expensive
  - Automakers recalled a record of 51.2 million vehicles over 868 separate recalls in 2015 for safety defects (USA TODAY January 21, 2016)
  - Study in University of Michigan shows self deriving cars has five times bigger accident rate (USA TODAY October 31, 2015)
  - Tesla and Uber had fatalities (2016 and 2018 - https://en.wikipedia.org/wiki/List_of_autonomous_car_fatalities)

Cyber-Physical Systems
Formal Verification is a Necessity

- System failures are very expensive
  - Automakers recalled a record of 51.2 million vehicles over 868 separate recalls in 2015 for safety defects (USA TODAY January 21, 2016)
  - Study in University of Michigan shows self deriving cars has five times bigger accident rate (USA TODAY October 31, 2015)
  - Tesla and Uber had fatalities (2016 and 2018 - https://en.wikipedia.org/wiki/List_of_autonomous_car_fatalities)
- Testing is Not Enough
  - It is required, good, but not enough!

Cyber-Physical Systems
Formal Verification is a Necessity

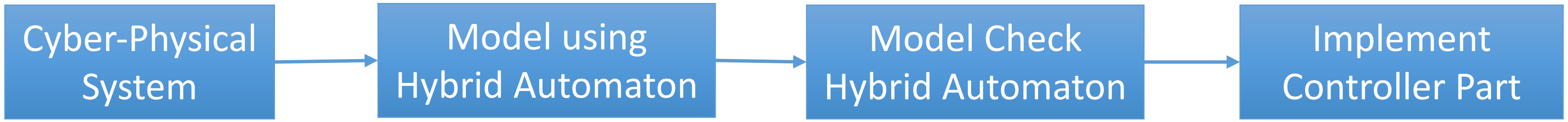
- System failures are very expensive
  - Automakers recalled a record of 51.2 million vehicles over 868 separate recalls in 2015 for safety defects (USA TODAY January 21, 2016)
  - Study in University of Michigan shows self deriving cars has five times bigger accident rate (USA TODAY October 31, 2015)
  - Tesla and Uber had fatalities (2016 and 2018 - https://en.wikipedia.org/wiki/List_of_autonomous_car_fatalities)
- Testing is Not Enough
  - It is required, good, but not enough!
- We need proof of correctness
  - Cyber-Physical Systems do not compute an answer
  - They are assumed to run infinitely long
    - Executing all possible paths is not even possible in theory

Cyber-Physical Systems

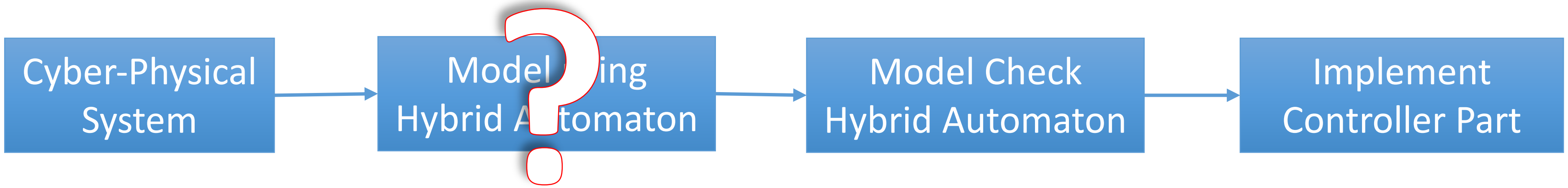
Formal Verification is a Necessity

- System failures are very expensive
  - Automakers recalled a record of 51.2 million vehicles over 868 separate recalls in 2015 for safety defects (USA TODAY January 21, 2016)
  - Study in University of Michigan shows self deriving cars has five times bigger accident rate (USA TODAY October 31, 2015)
  - Tesla and Uber had fatalities (2016 and 2018 - https://en.wikipedia.org/wiki/List_of_autonomous_car_fatalities)
- Testing is Not Enough
  - It is required, good, but not enough!
- We need proof of correctness
  - Cyber-Physical Systems do not compute an answer
  - They are assumed to run infinitely long
    - Executing all possible paths is not even possible in theory
- Hybrid automata are used to model a cyber-physical system
  - Mathematical Model
  - Mathematical Proof

Formal Verification of Cyber-Physical Systems



Formal Verification of Cyber-Physical Systems

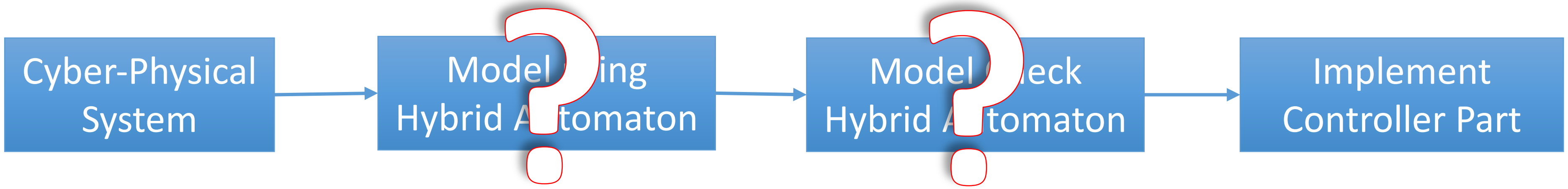


- Ordinary Differential Equations

  "Detailed studies of the real world impel us, albeit reluctantly, to take account of the fact that the rate of change of physical systems depend not only on their present state, but also on their past history."

  Richard, B., Cooke, K.L.: Differential-difference equations. Technical report. P iii, 1963

Formal Verification of Cyber-Physical Systems



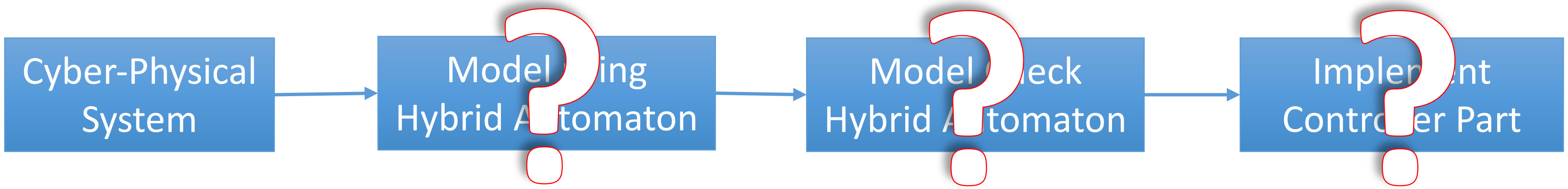
- Ordinary Differential Equations

  “Detailed studies of the real world impel us, albeit reluctantly, to take account of the fact that the rate of change of physical systems depend not only on their present state, but also on their past history.”

  Richard, B., Cooke, K.L.: Differential-difference equations. Technical report. P iii, 1963

- Almost Nothing is Decidable

Formal Verification of Cyber-Physical Systems



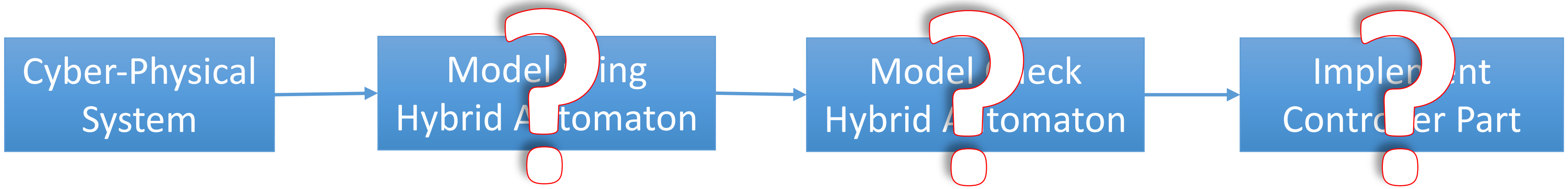
- Ordinary Differential Equations

  “Detailed studies of the real world impel us, albeit reluctantly, to take account of the fact that the rate of change of physical systems depend not only on their present state, but also on their past history.”

  Richard, B., Cooke, K.L.: Differential-difference equations. Technical report. P iii, 1963

- Almost Nothing is Decidable
- Almost Nothing is Implementable

Formal Verification of Cyber-Physical Systems



- Ordinary Differential Equations

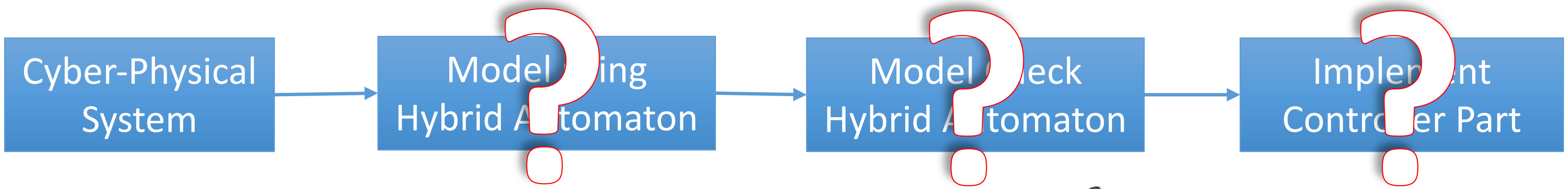
  "Detailed studies of the real world impel us, albeit reluctantly, to take account of the fact that the rate of change of physical systems depend not only on their present state, but also on their past history."

  Richard, B., Cooke, K.L.: Differential-difference equations. Technical report. P iii, 1963

- Almost Nothing is Decidable
- Almost Nothing is Implementable

May be the modeling and/or correctness definition is not good

Formal Verification of Cyber-Physical Systems



- Ordinary Differential Equations

  “Detailed studies of the real world impel us, albeit reluctantly, to take account of the fact that the rate of change of physical systems depend not only on their present state, but also on their past history.”

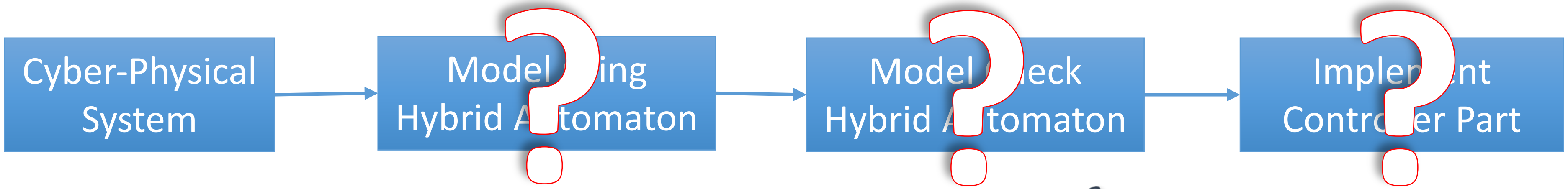
  Richard, B., Cooke, K.L.: Differential-difference equations. Technical report. P iii, 1963

Abstraction Helps

- Almost Nothing is Decidable
- Almost Nothing is Implementable

May be the <u>modeling</u> and/or <u>correctness definition</u> is not good

Formal Verification of Cyber-Physical Systems



- Ordinary Differential Equations

  "Detailed studies of the real world impel us, albeit reluctantly, to take account of the fact that the rate of change of physical systems depend not only on their present state, but also on their past history."

  Richard, B., Cooke, K.L.: Differential-difference equations. Technical report, [illegible], 1963

- Almost Nothing is Decidable
- Almost Nothing is Implementable

Abstraction Helps

Robustness Helps

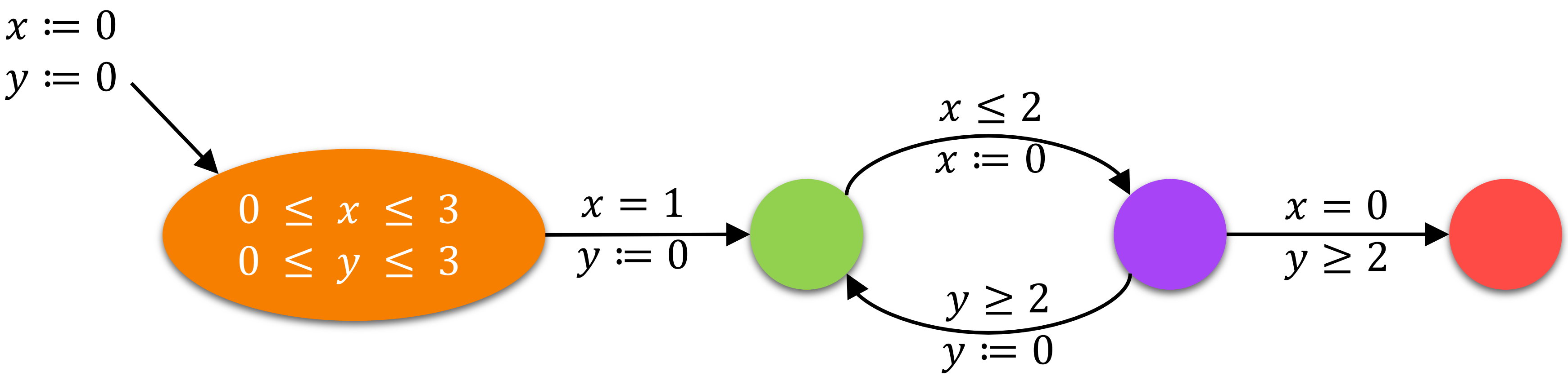
May be the modeling and/or correctness definition is not good

Robustness Helps

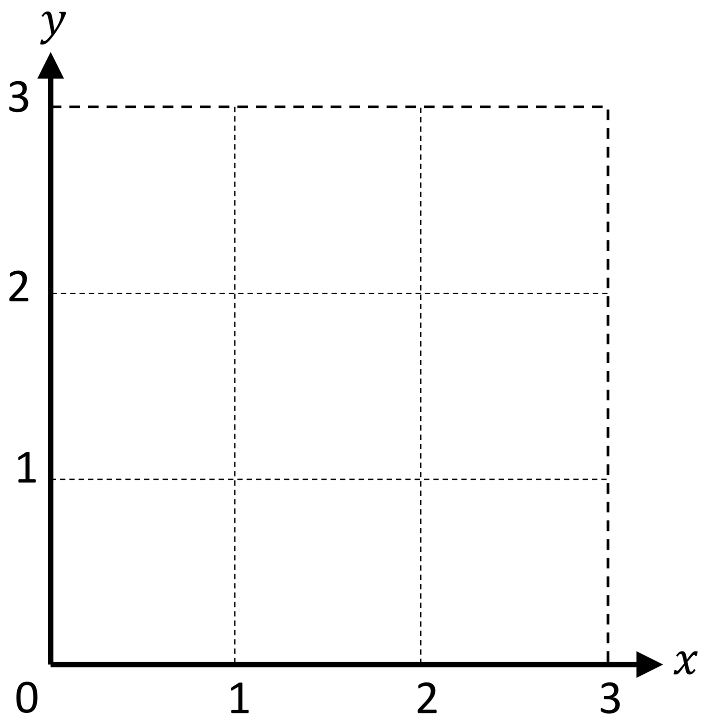
Robust Model Checking of Timed Automata

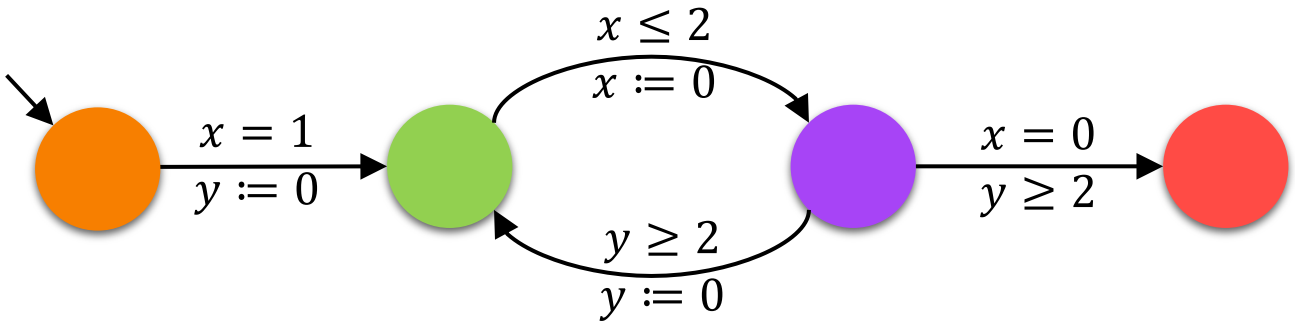
HSCC 2017

Timed Automata

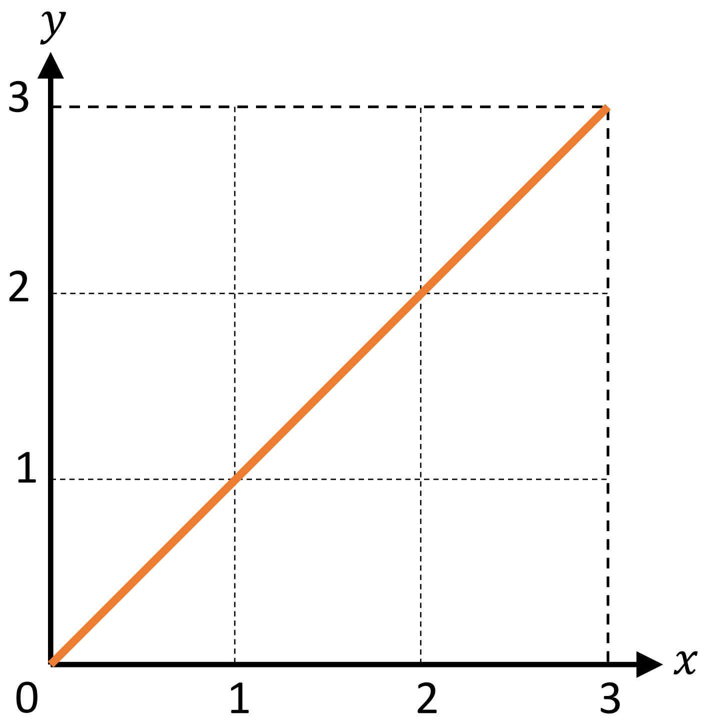


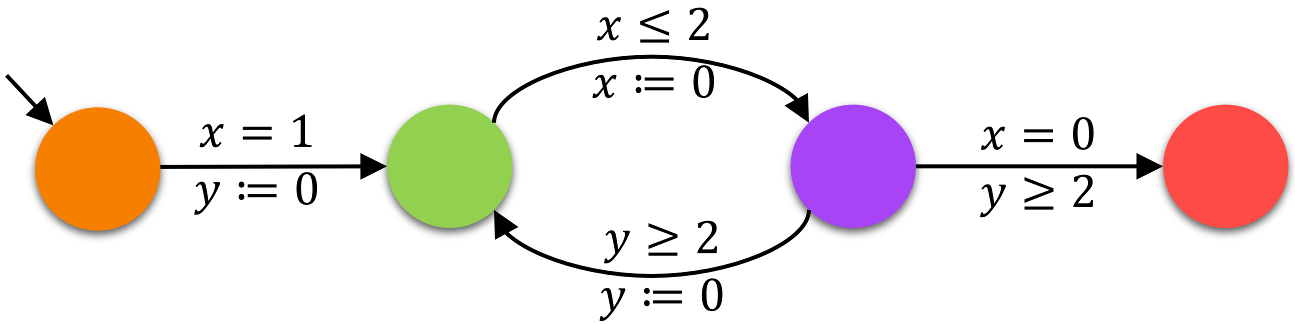
Timed Automata



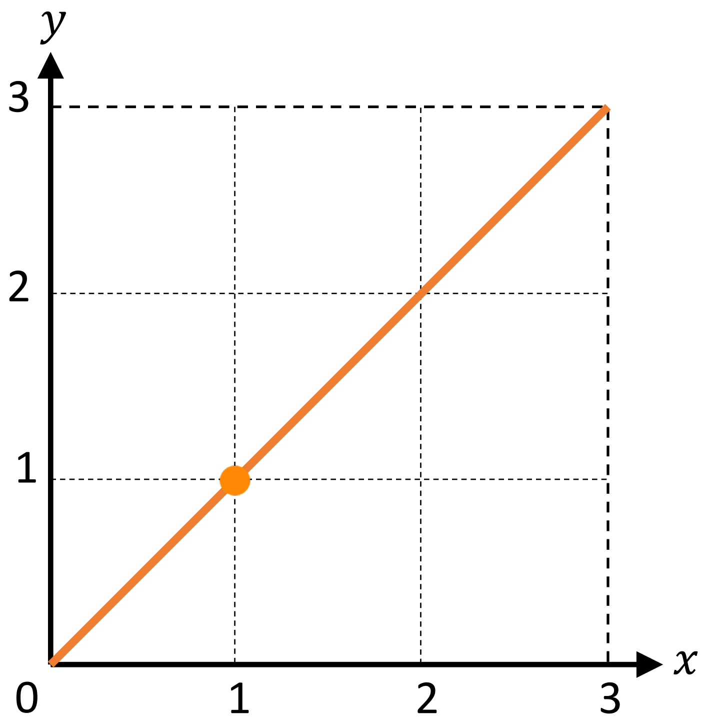


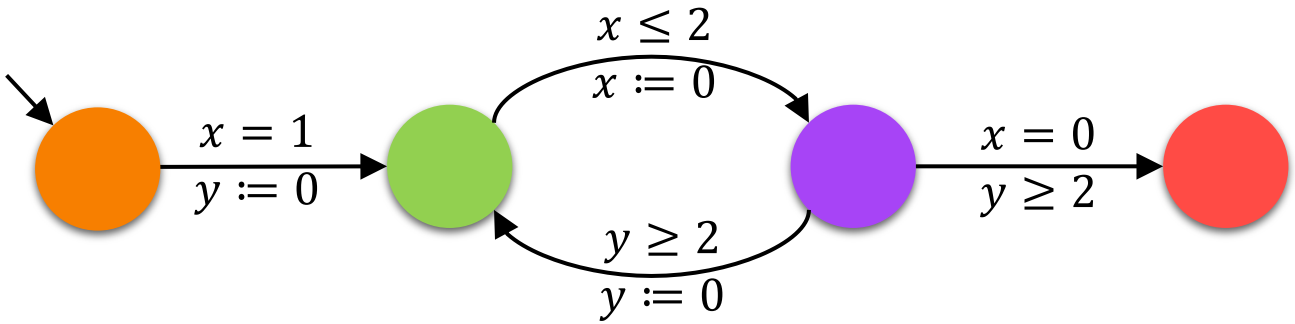
Timed Automata



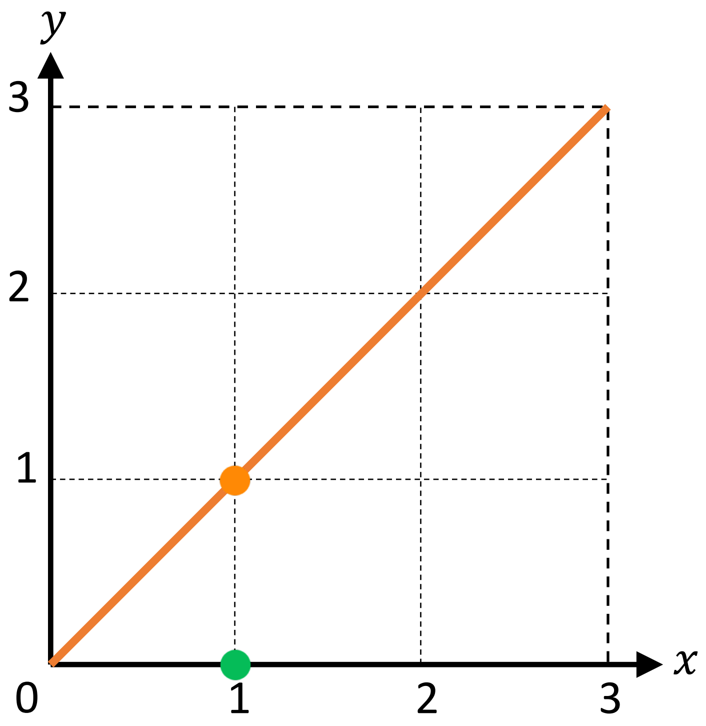


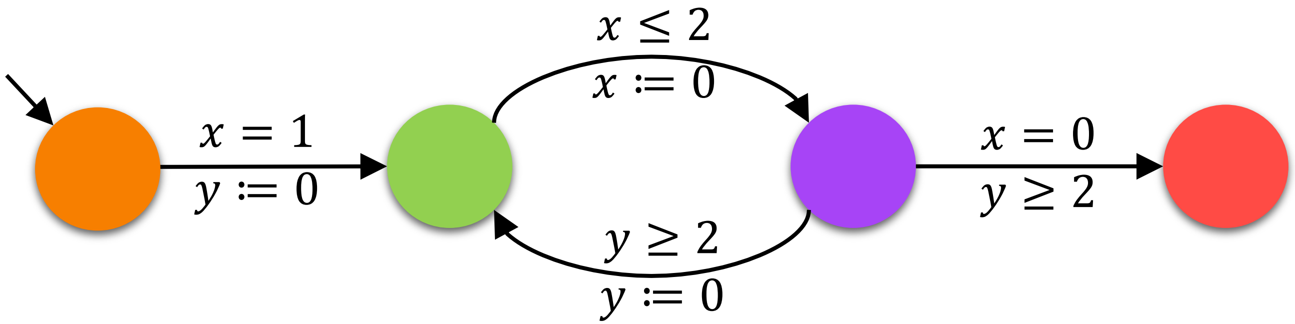
Timed Automata



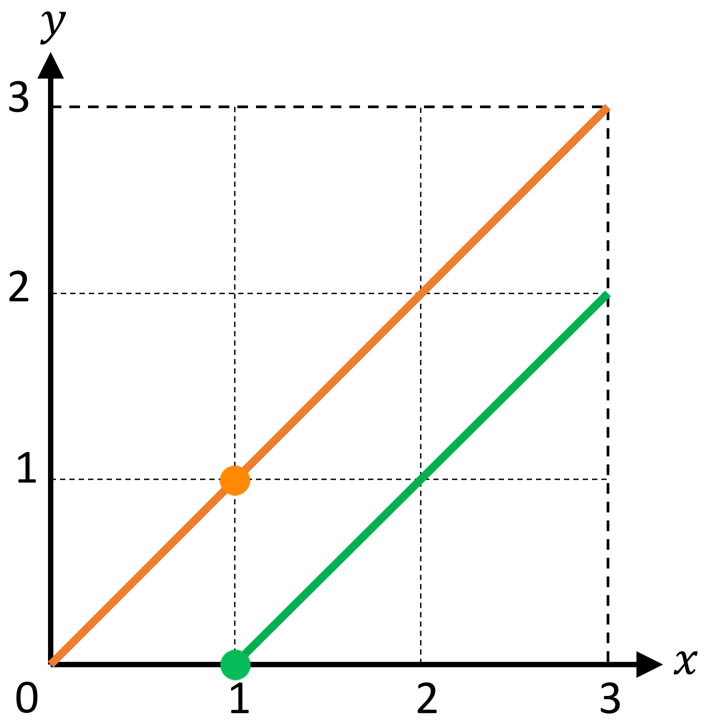


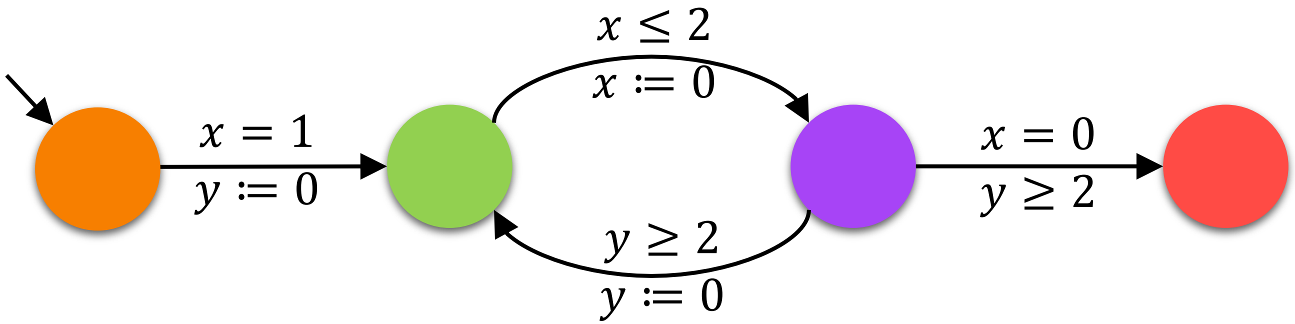
Timed Automata



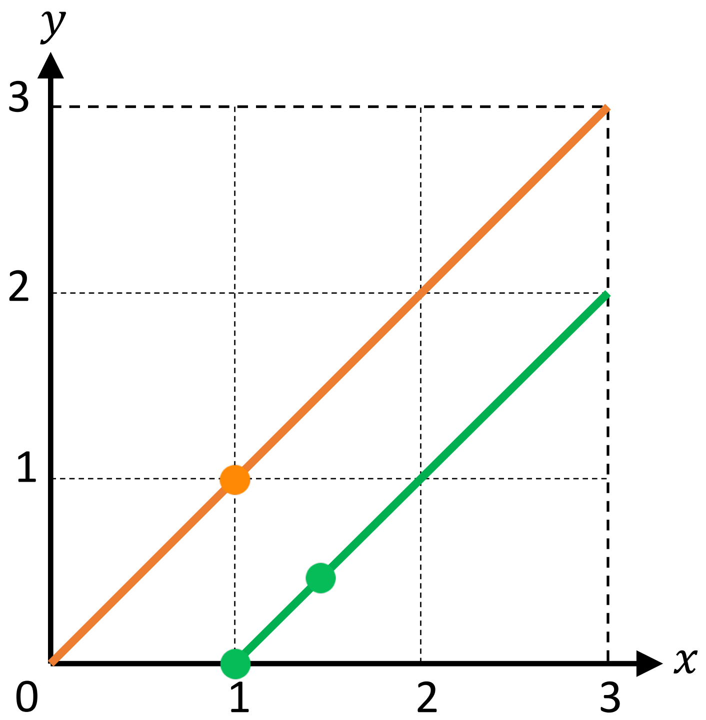


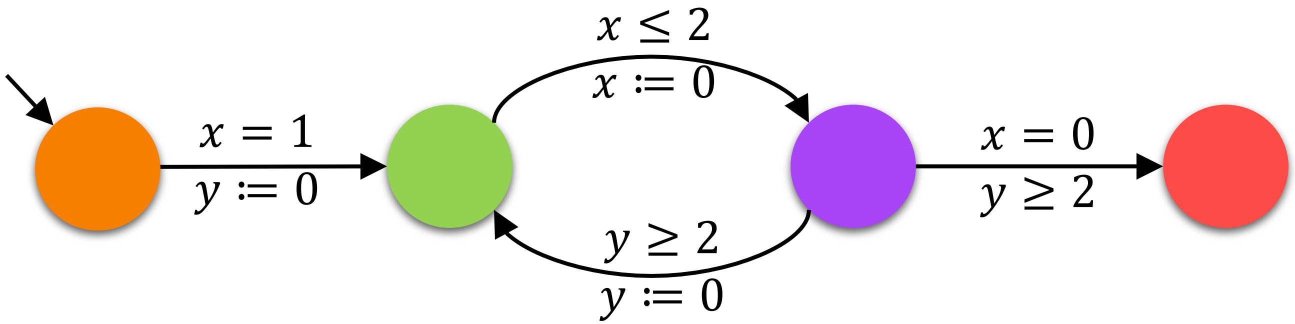
Timed Automata



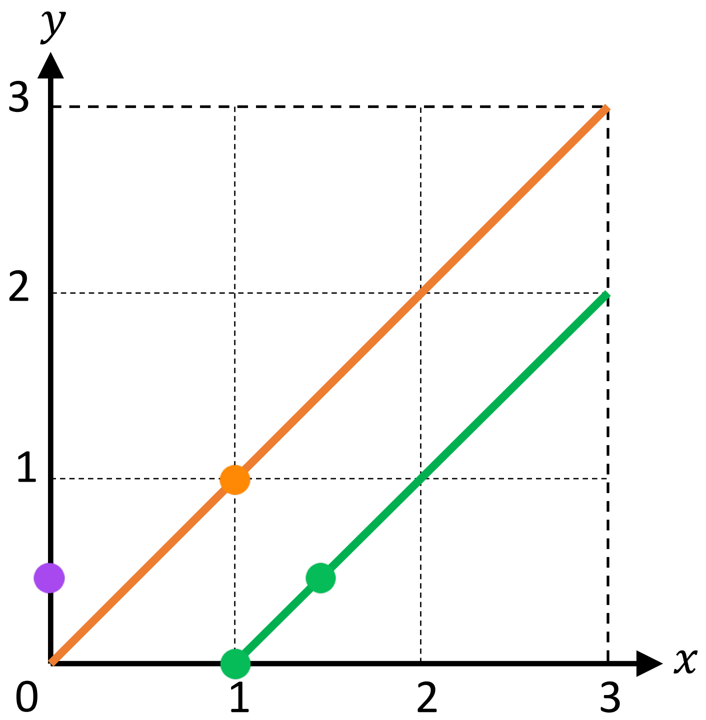


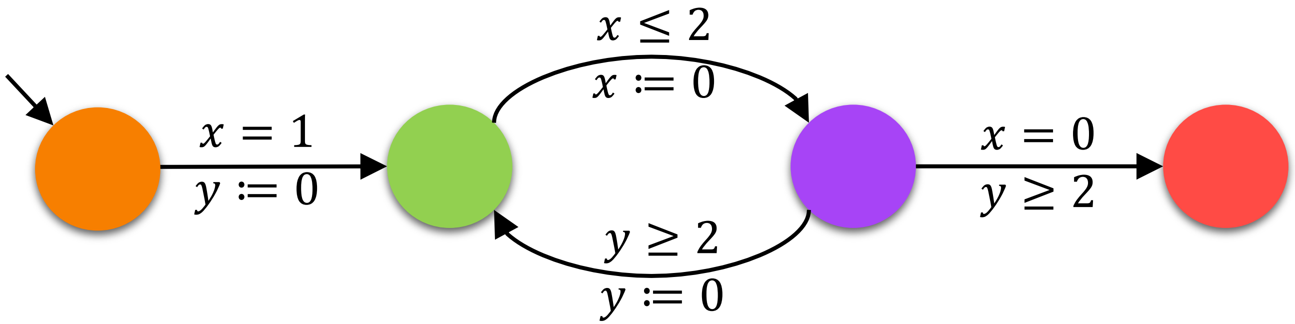
Timed Automata



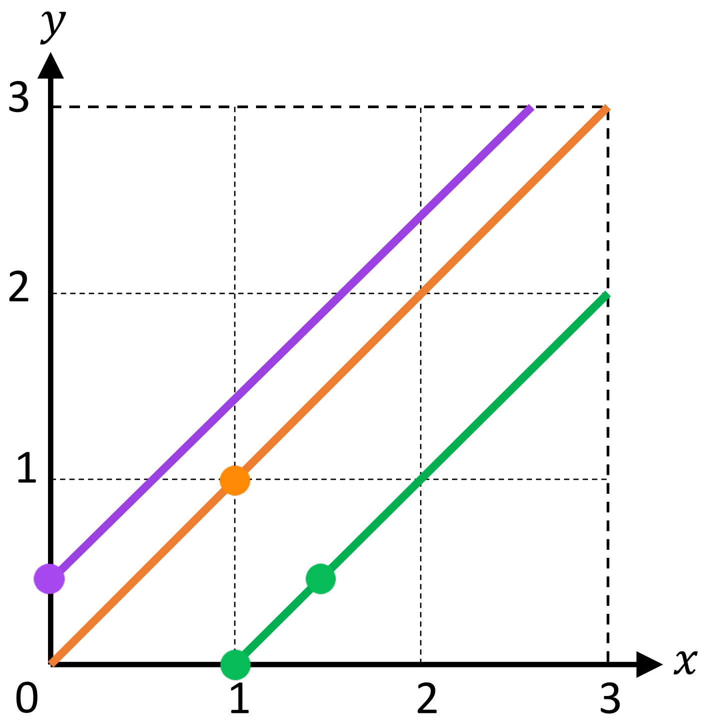


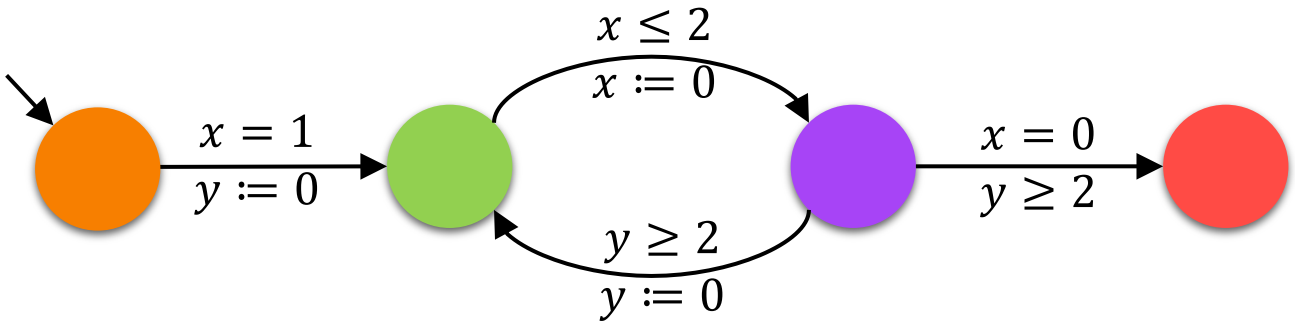
Timed Automata



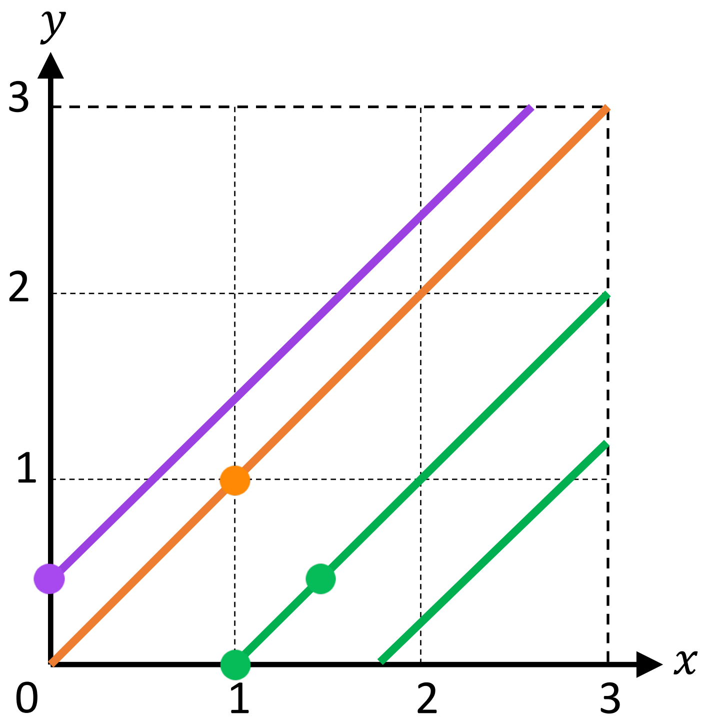


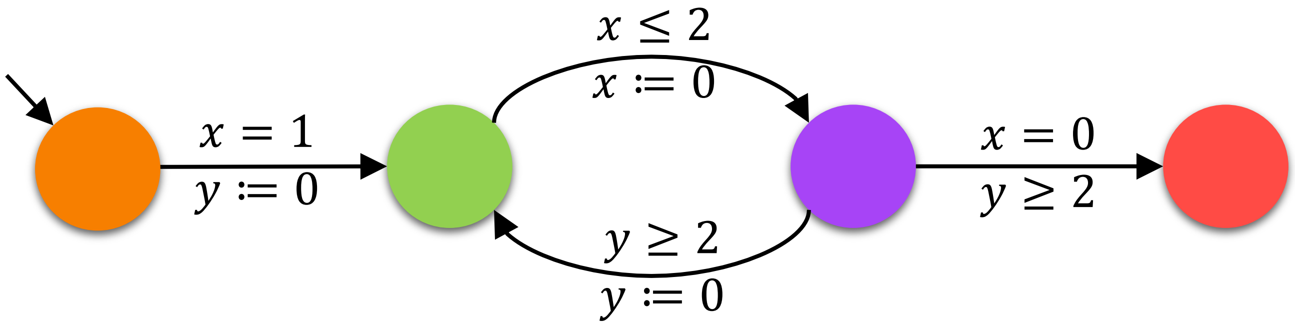
Timed Automata



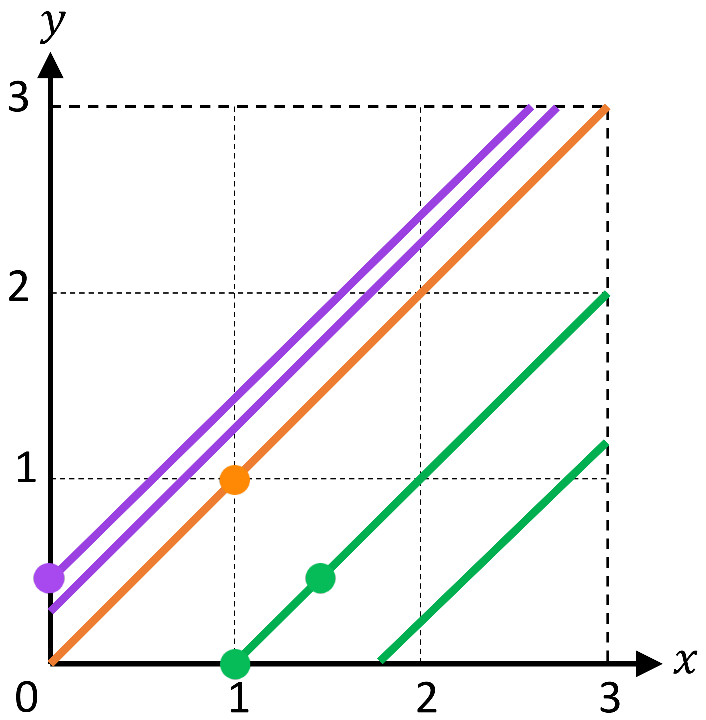


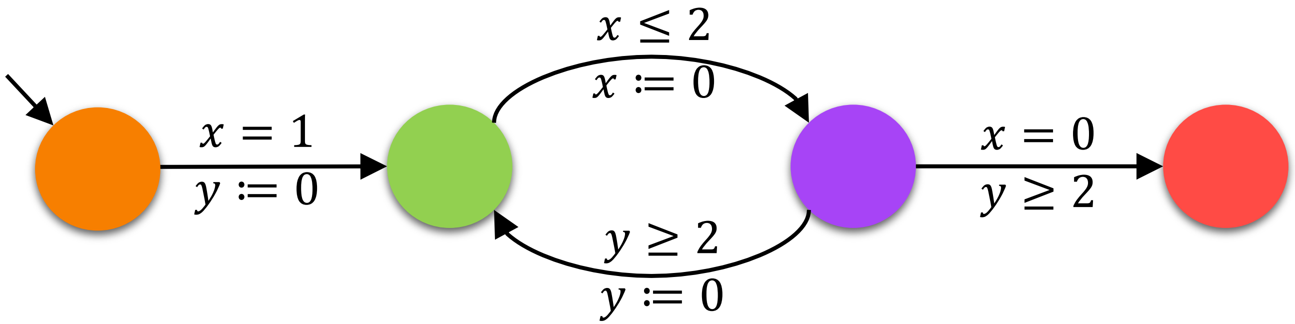
Timed Automata



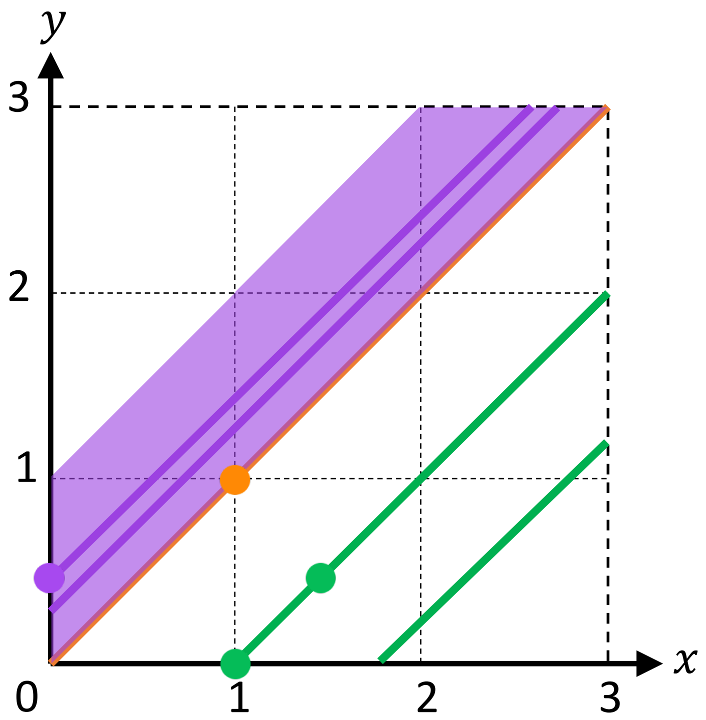


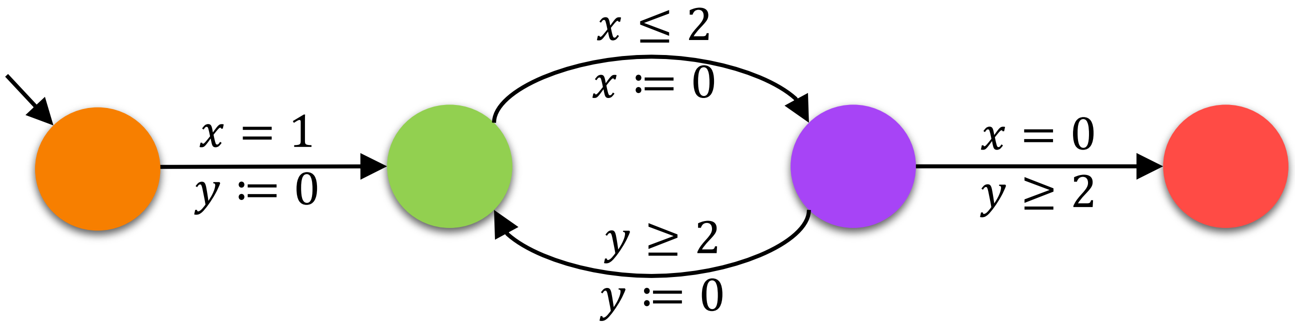
Timed Automata



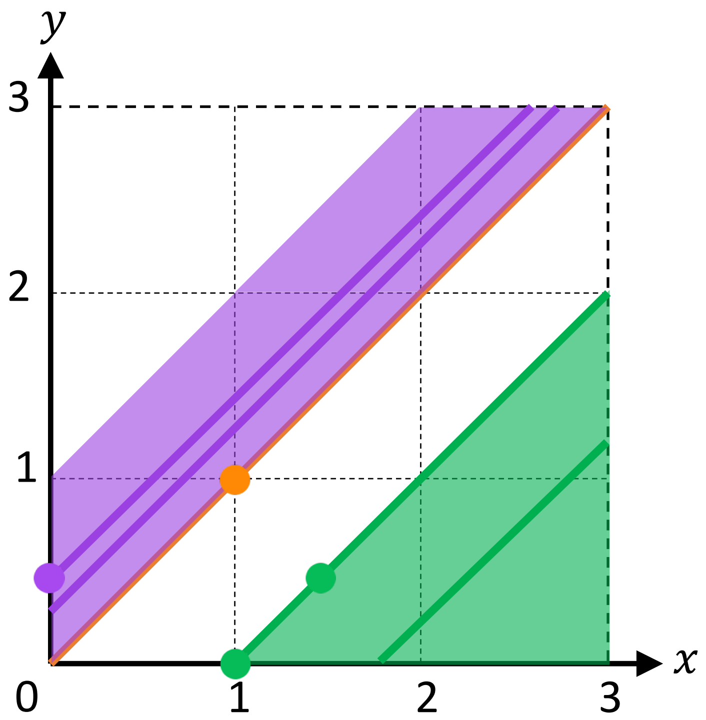


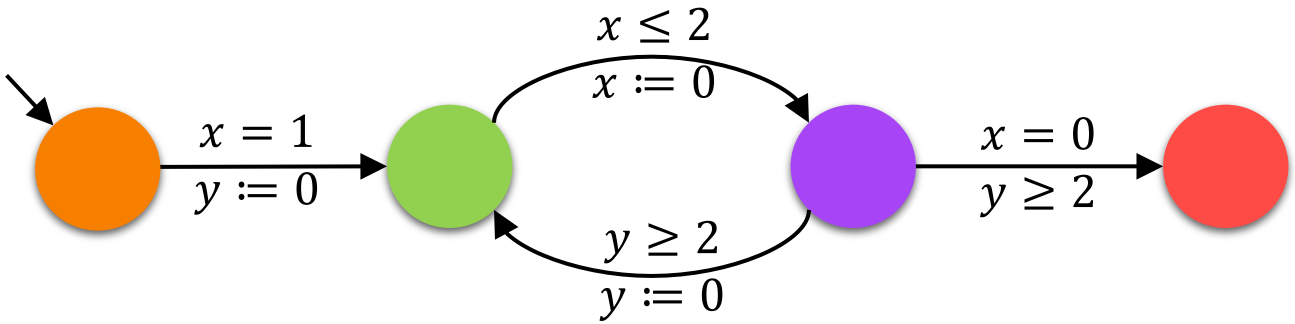
Timed Automata



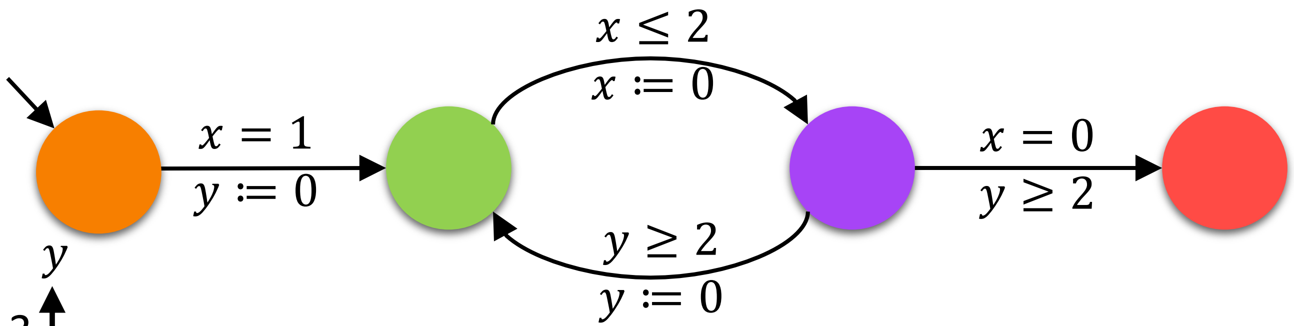


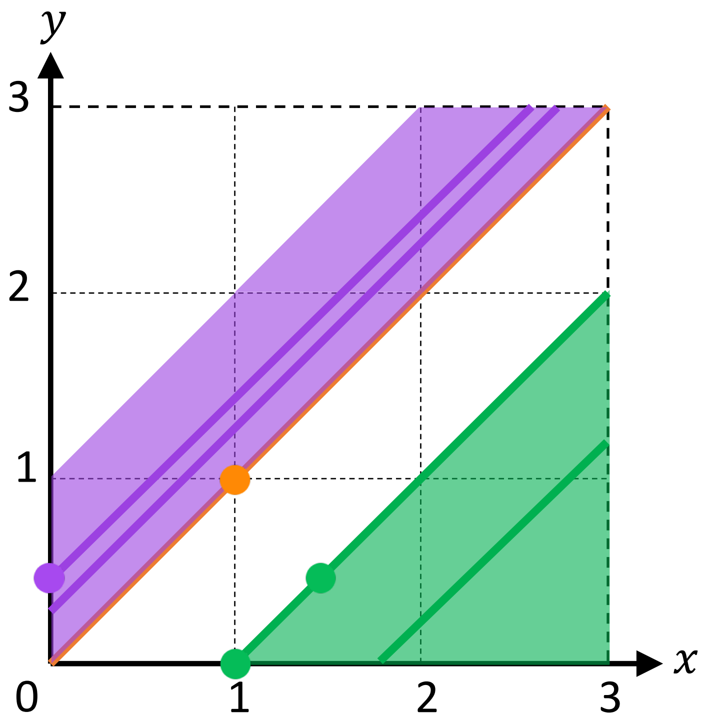
Timed Automata

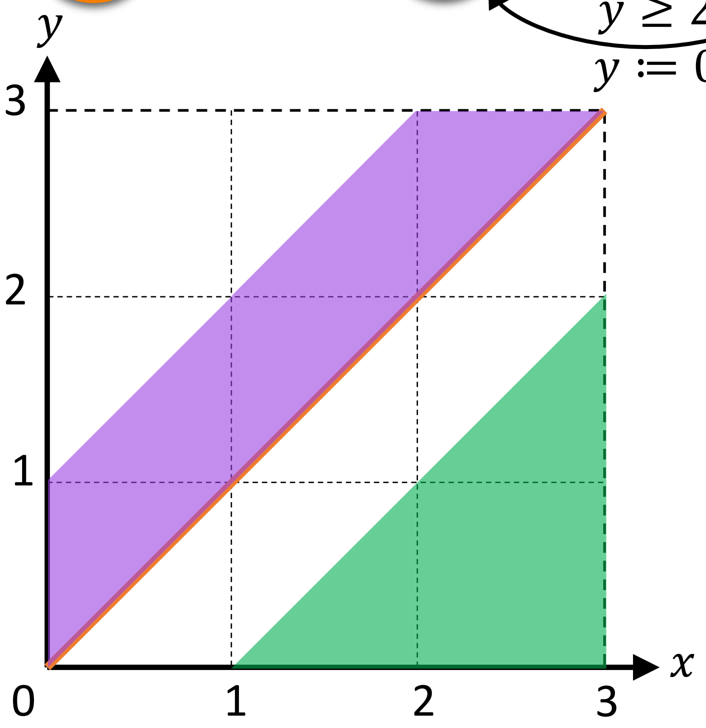




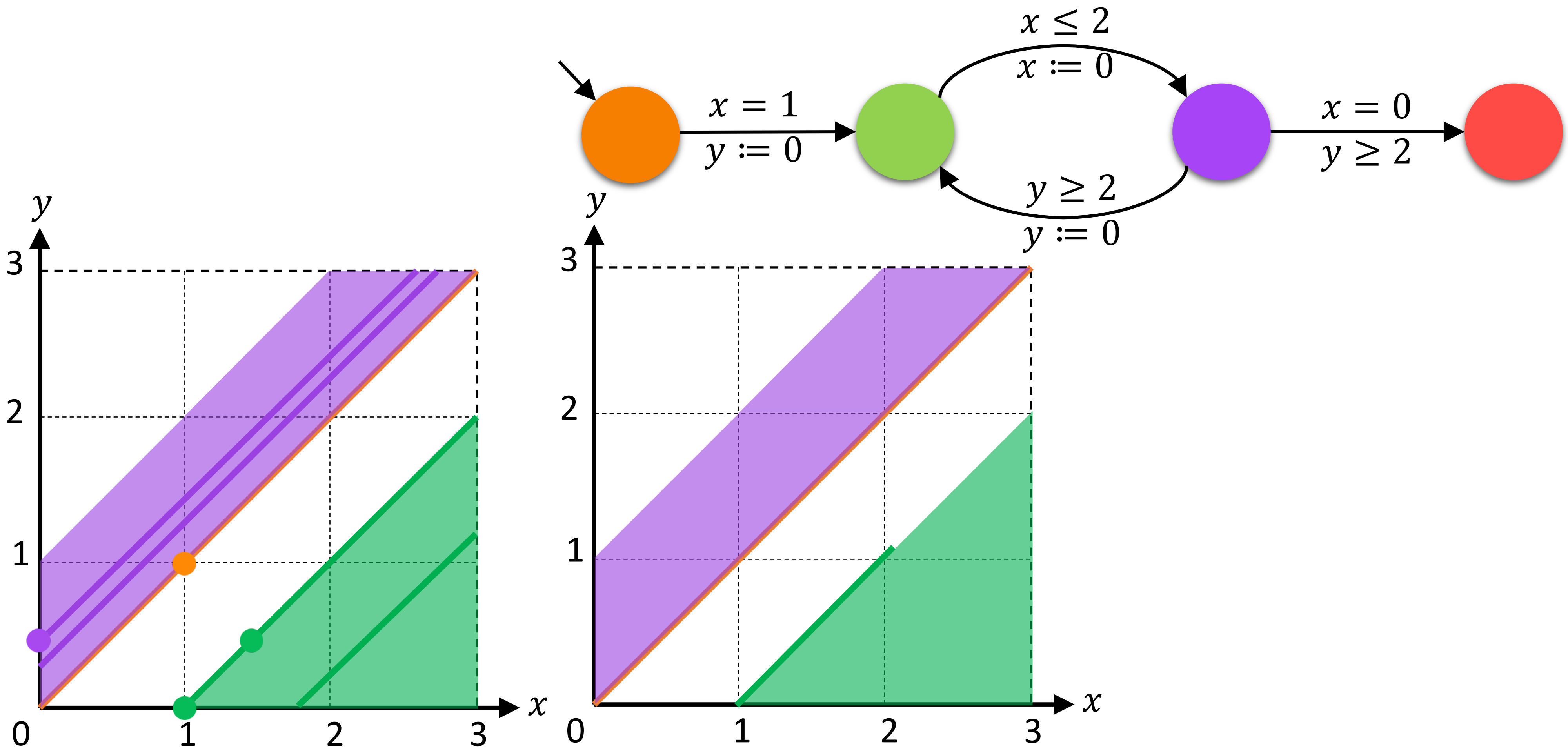
Timed Automata



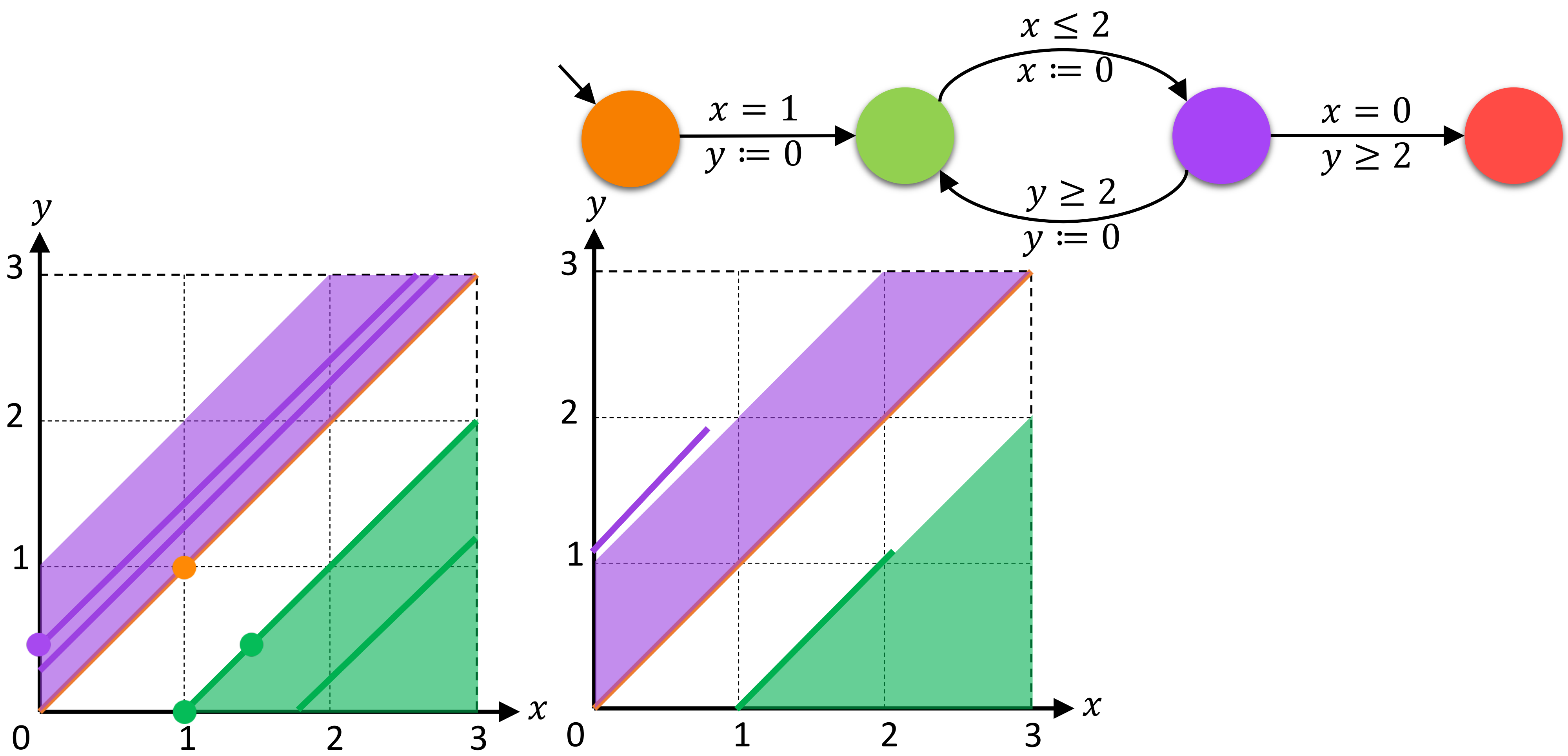




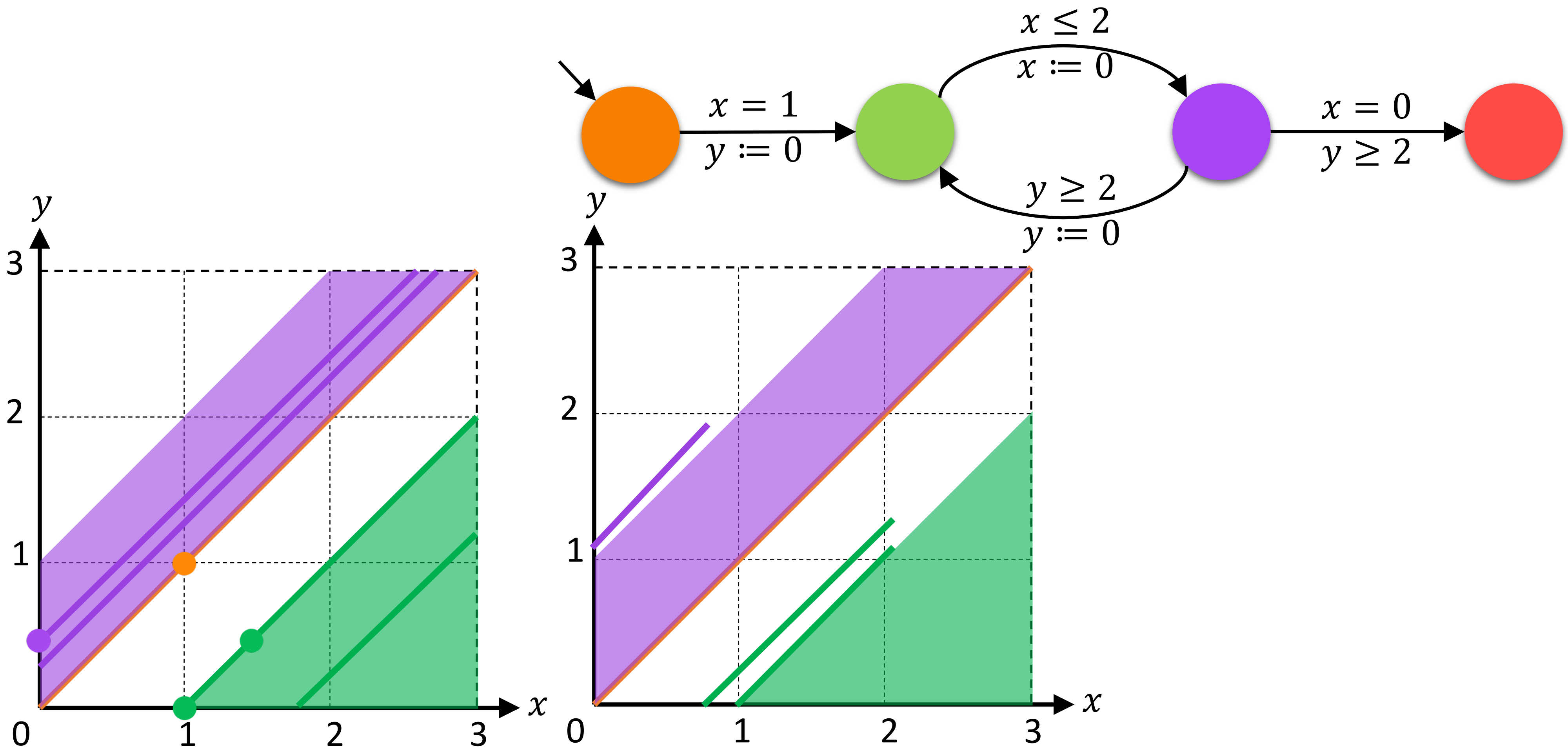
Timed Automata



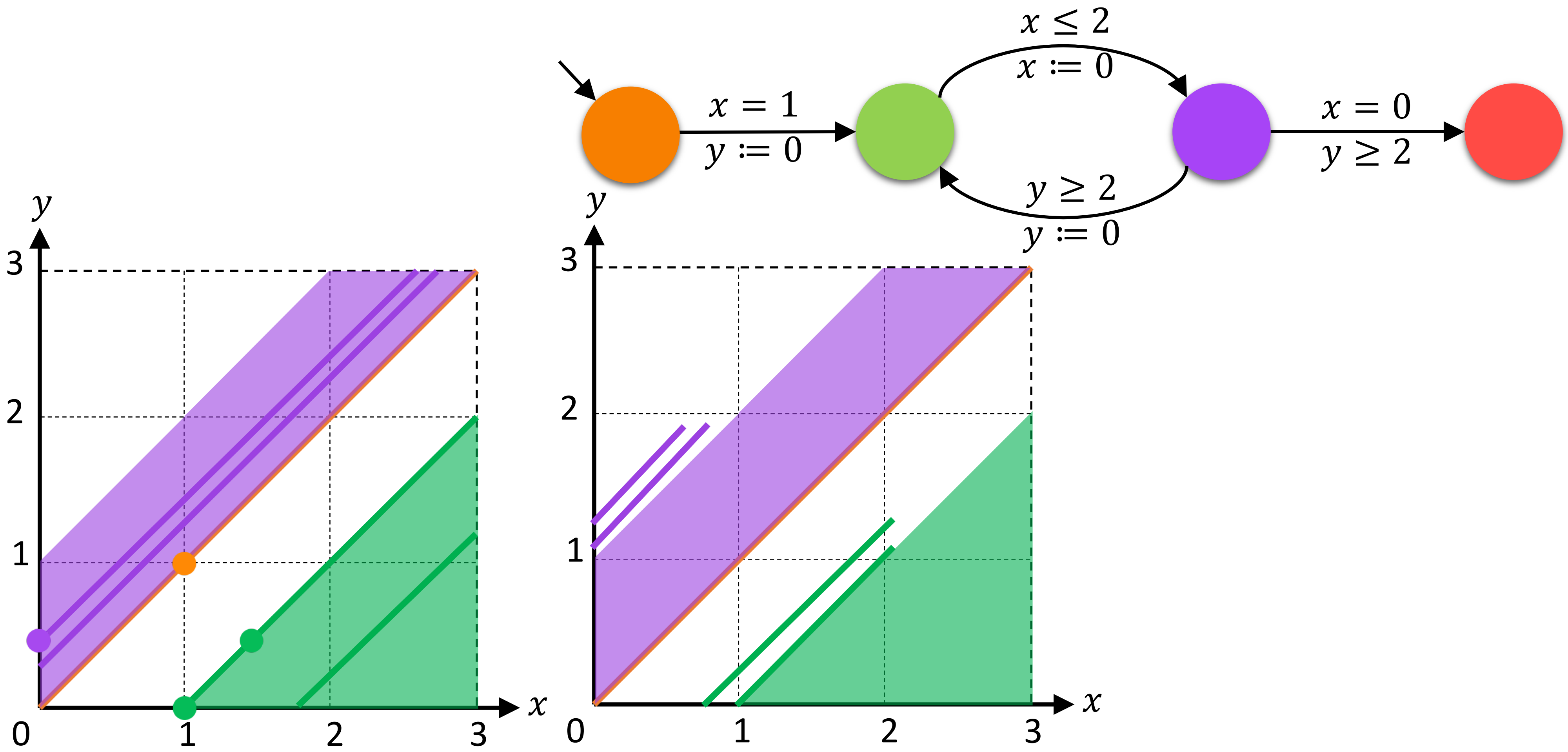
Timed Automata



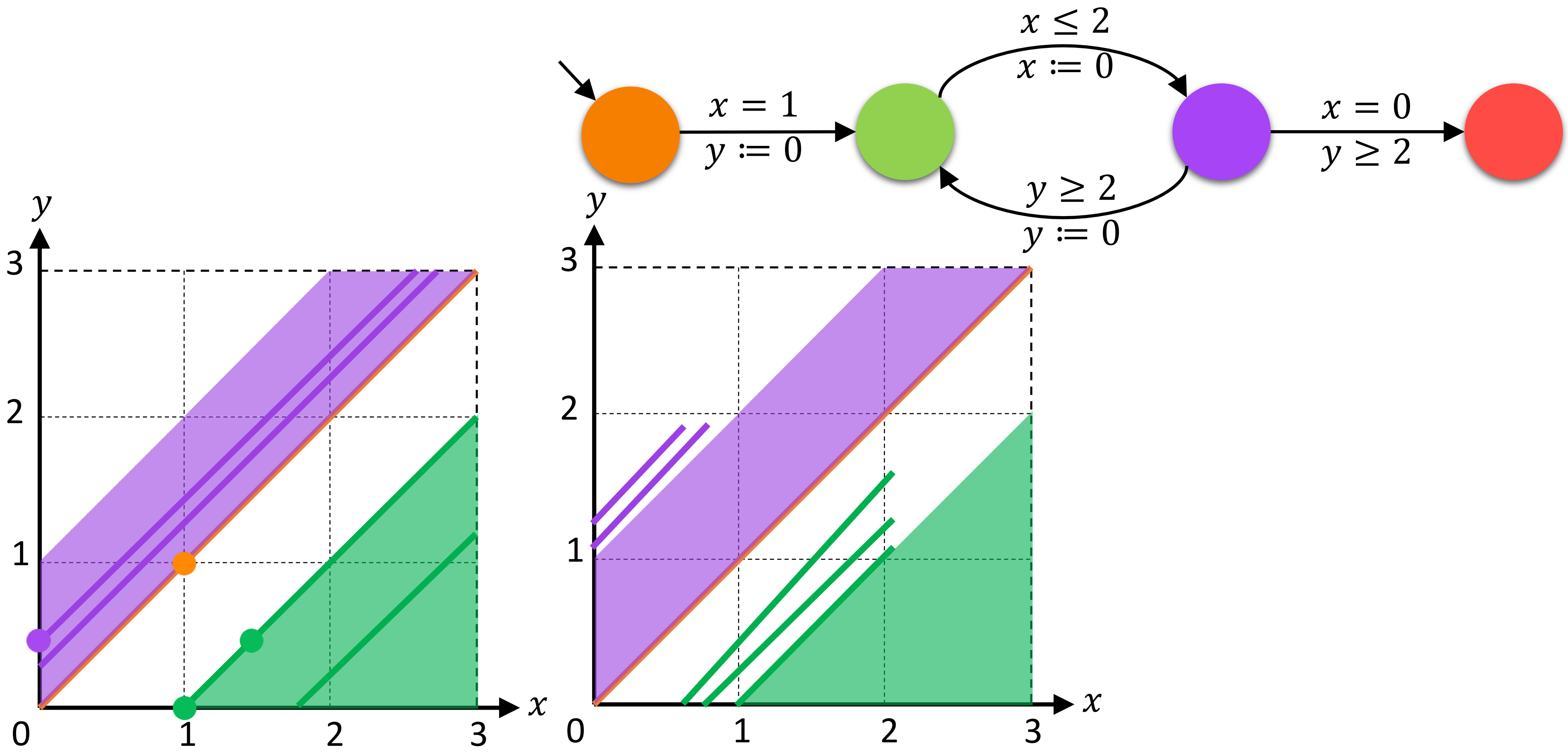
Timed Automata



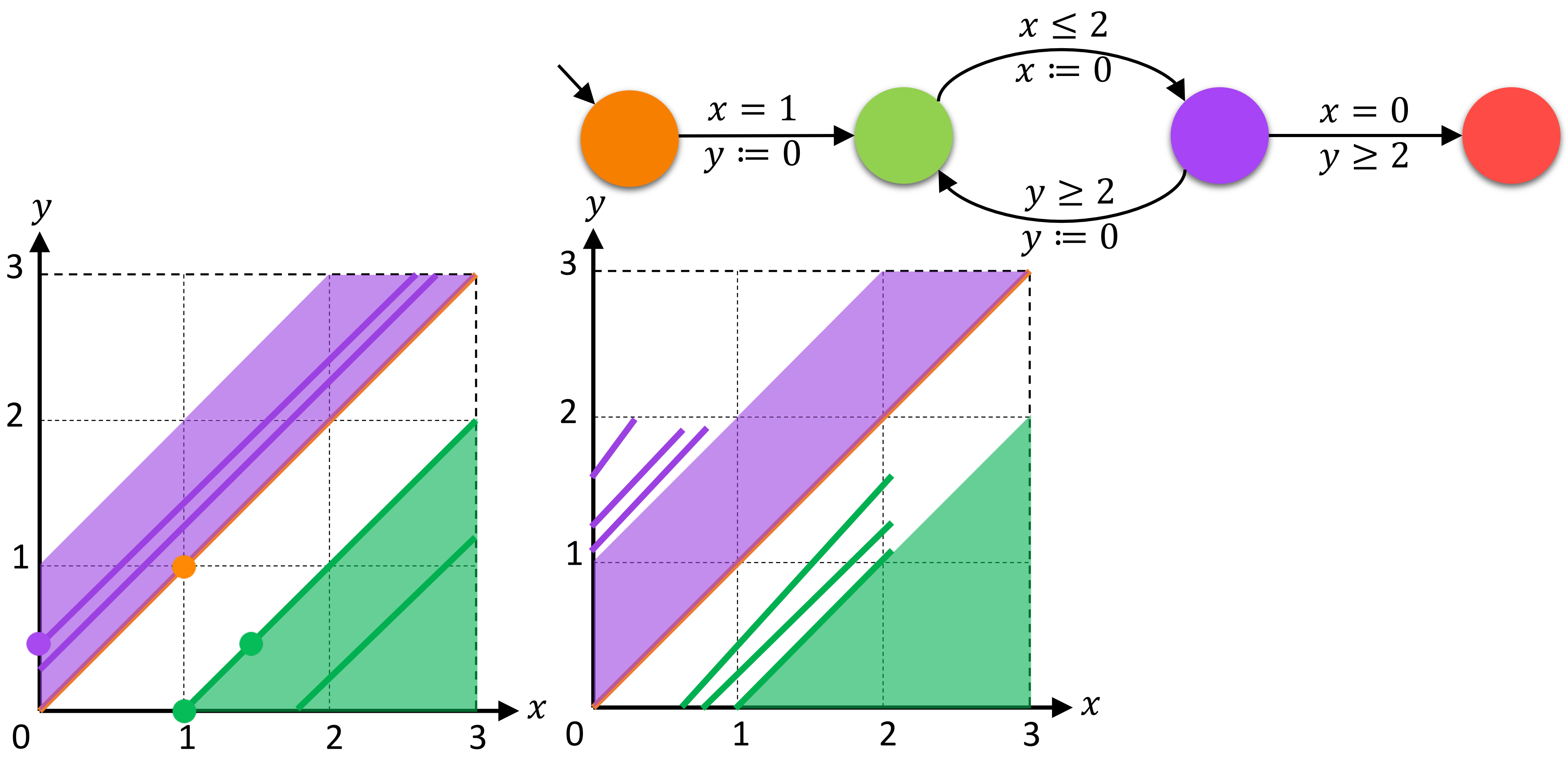
Timed Automata



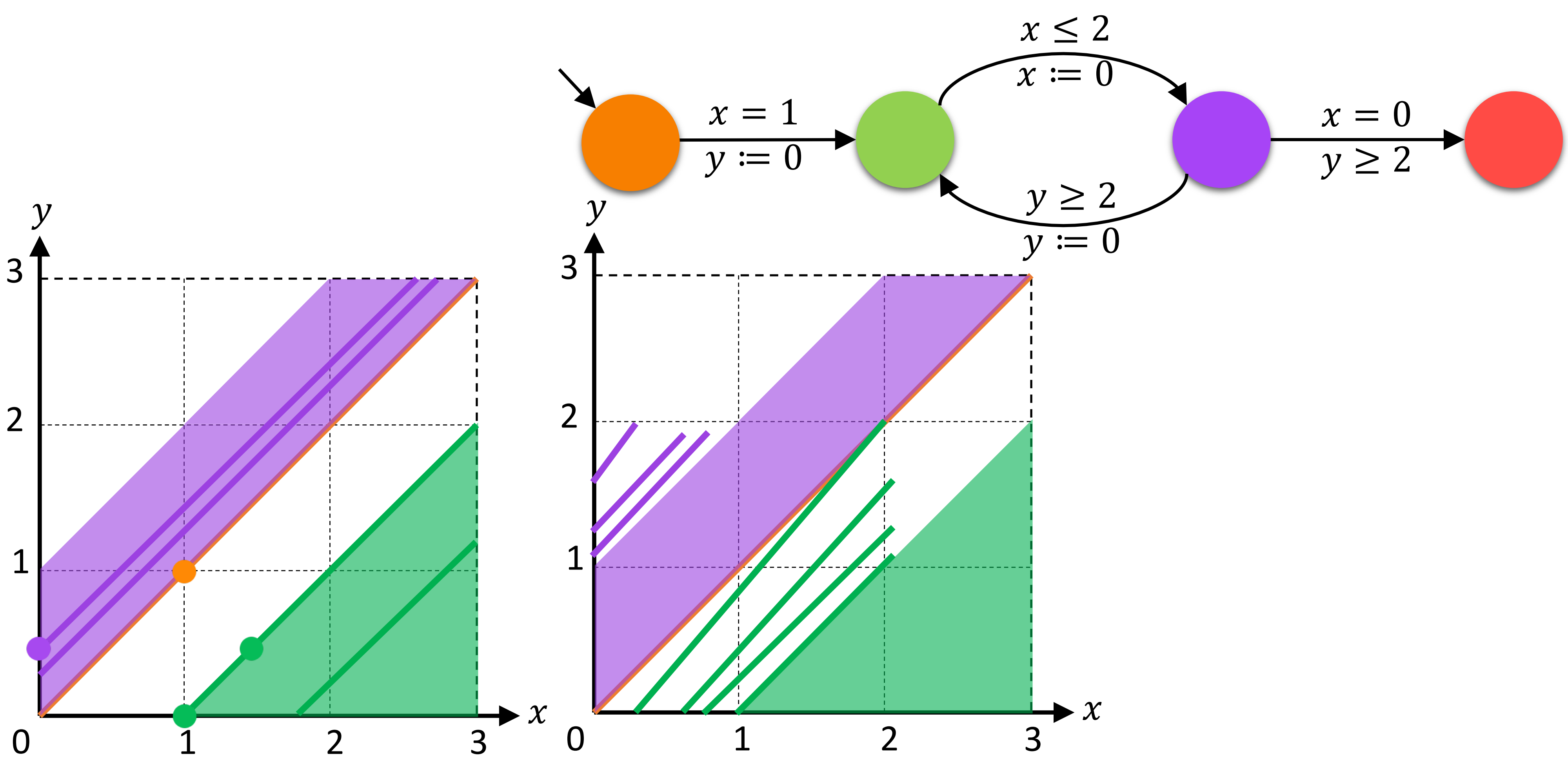
Timed Automata



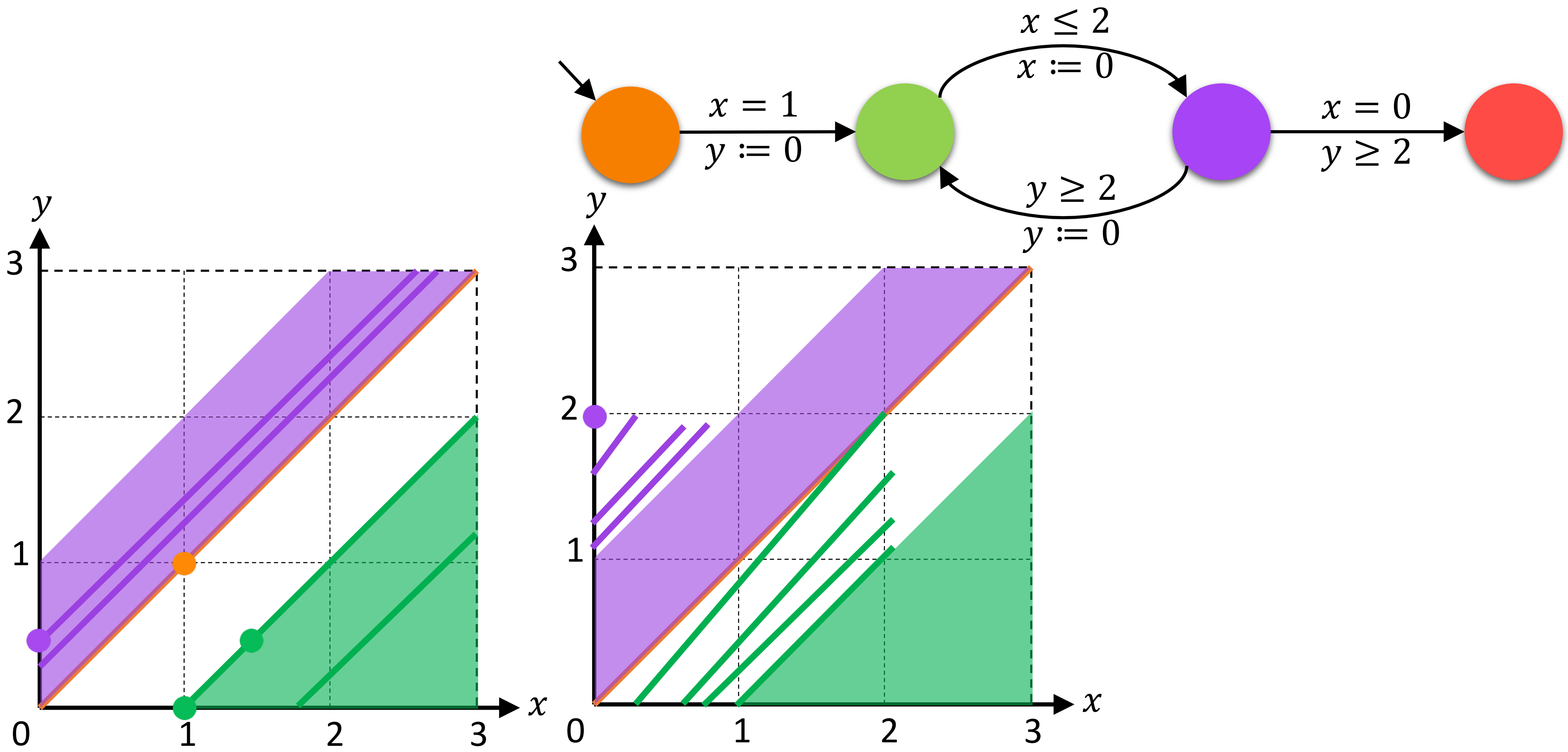
Timed Automata



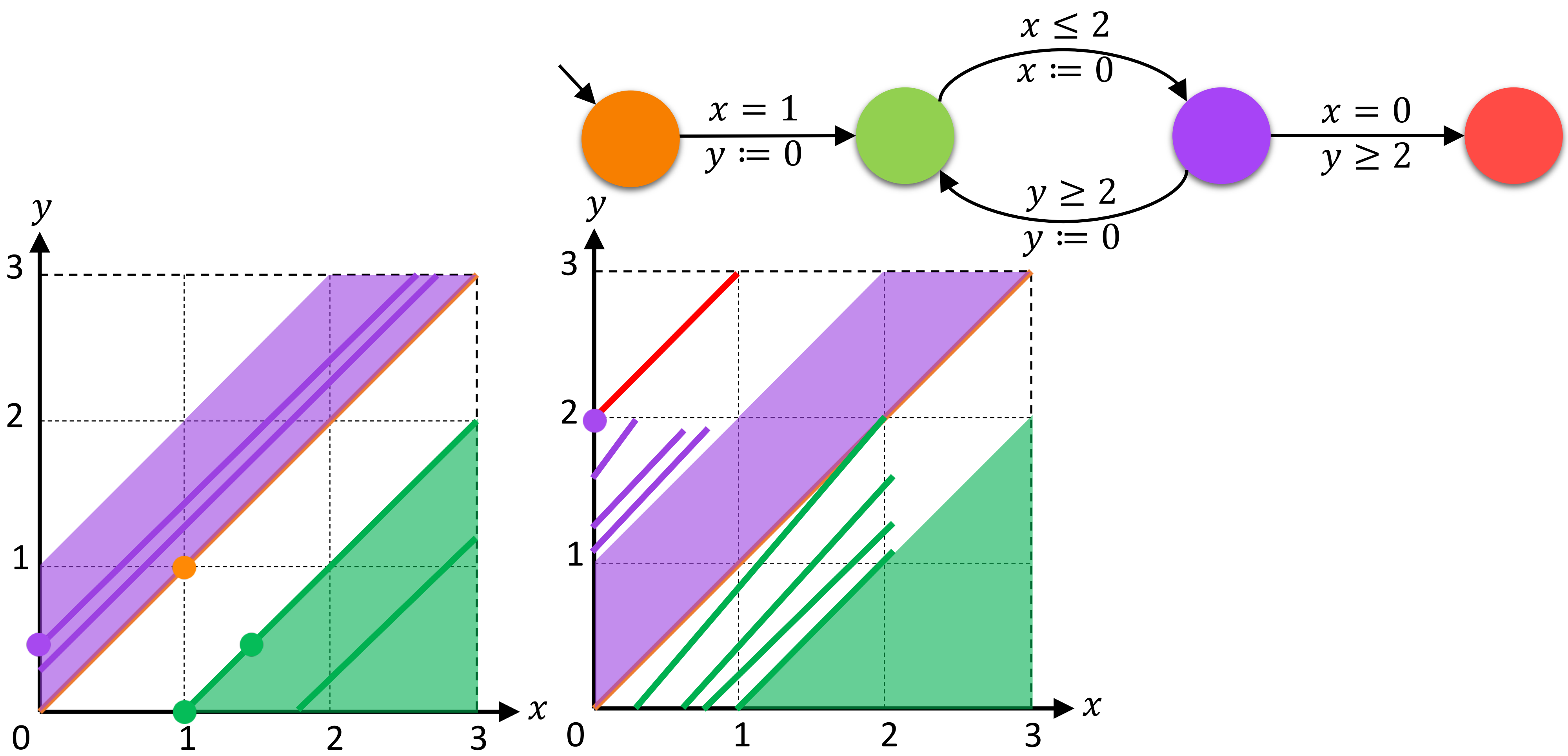
Timed Automata



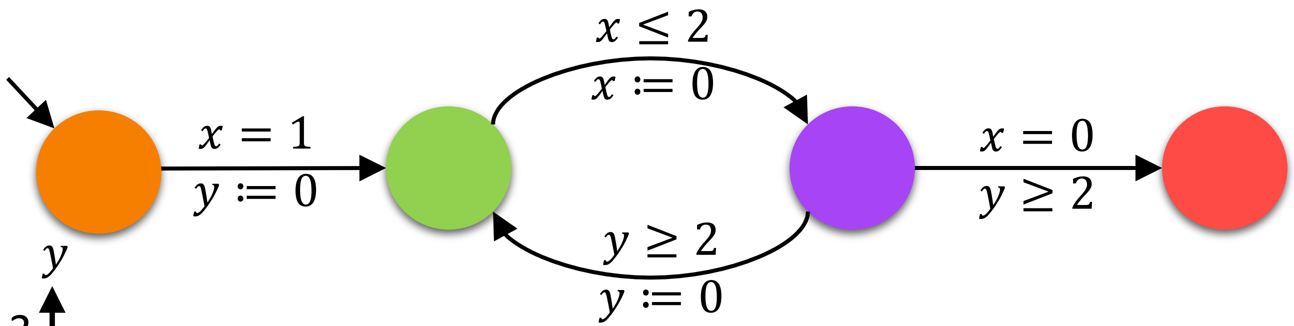
Timed Automata

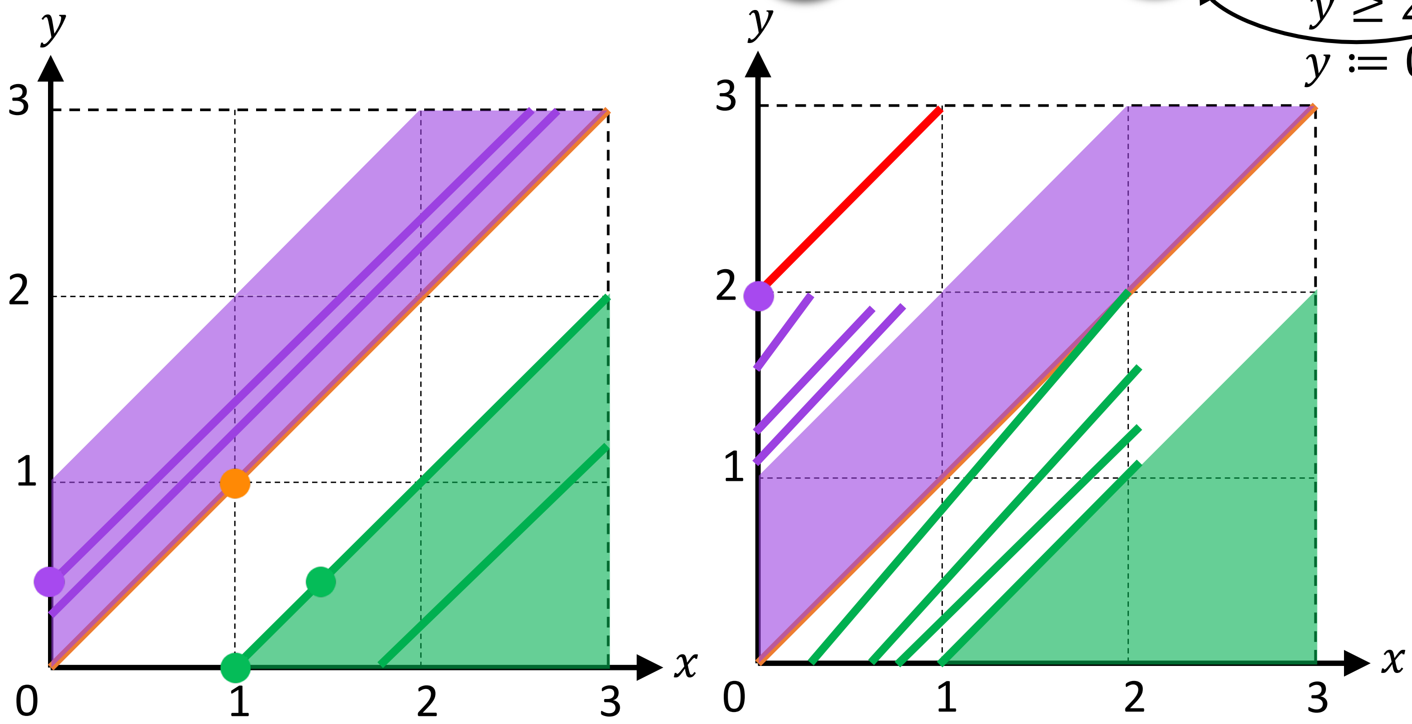


Timed Automata



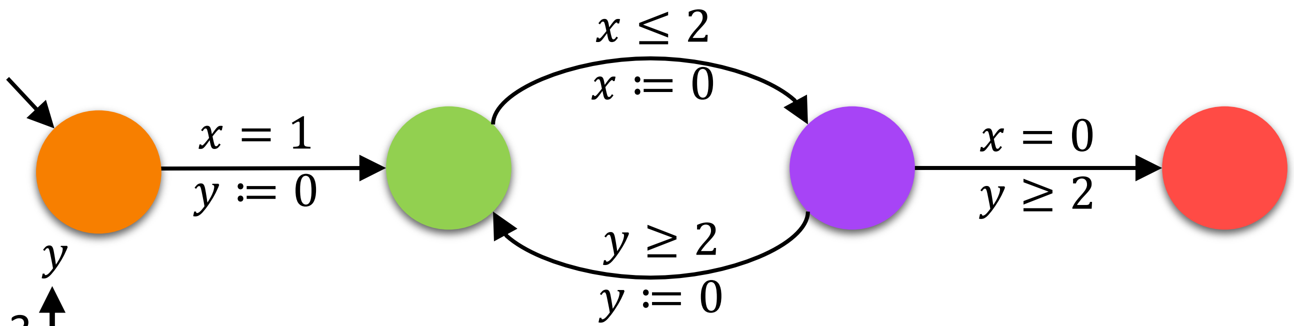
Timed Automata

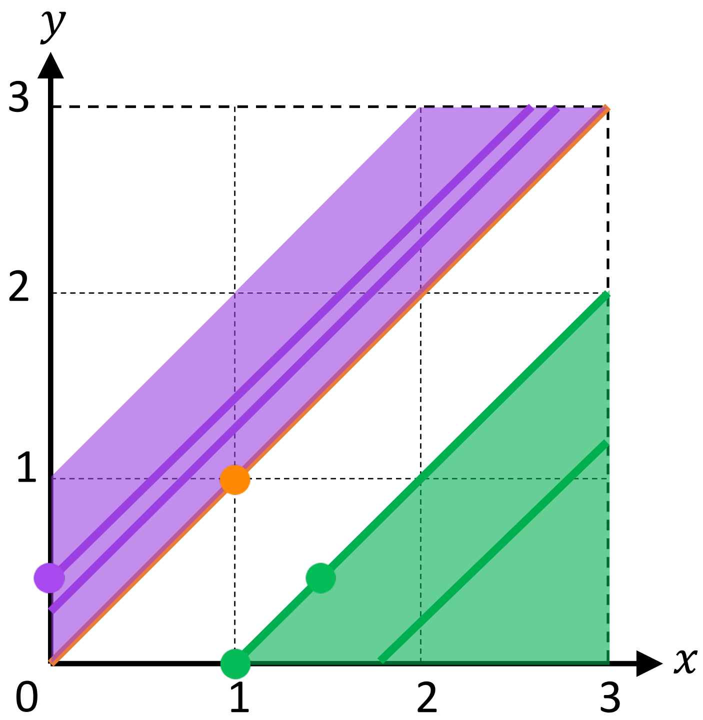


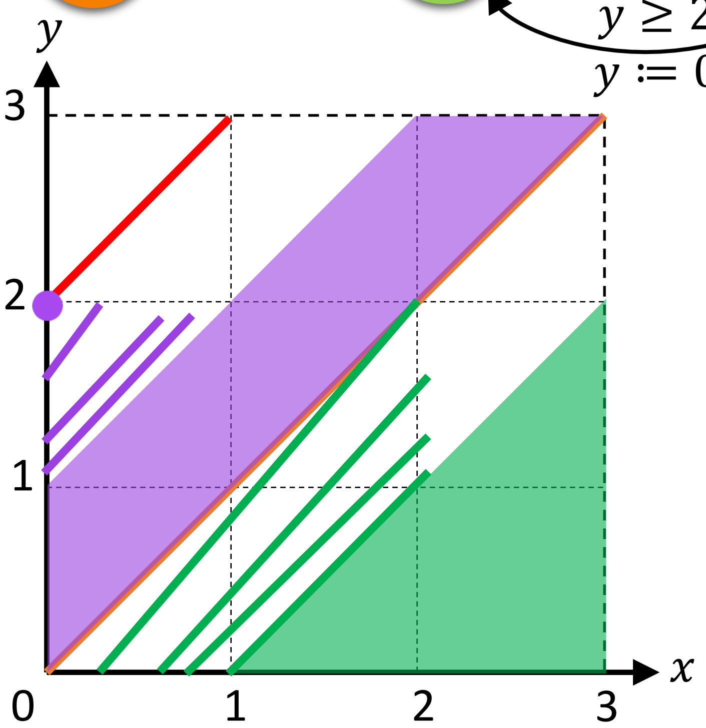


- Trajectory: an infinite sequence of continuous and discrete transition

Timed Automata



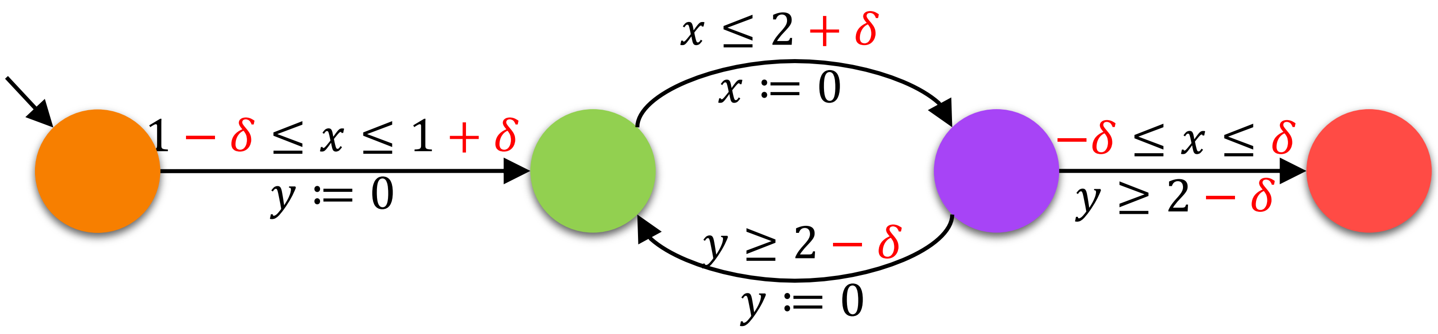




- Trajectory: an infinite sequence of continuous and discrete transition
- Execution: a trajectory that starts from the initial state
  - The set of executions $[\![\mathcal{T}]\!]$

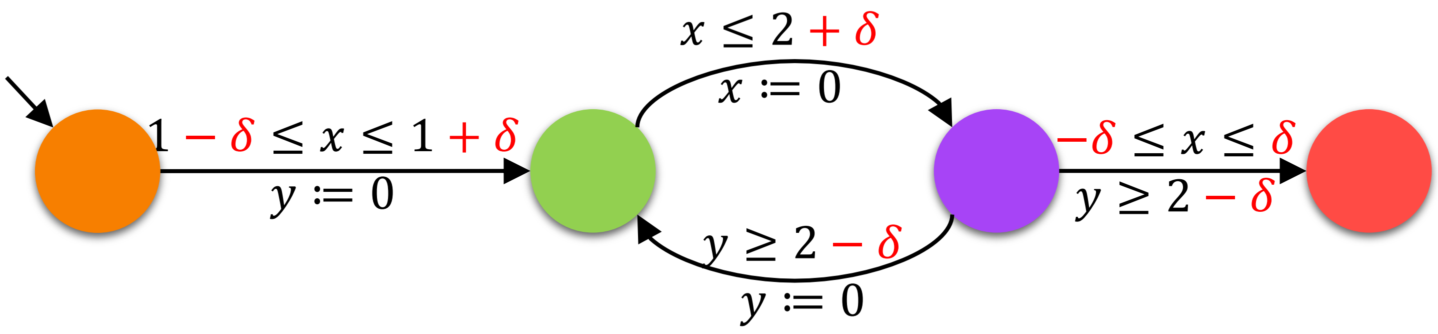
Types of Perturbation

- Only guards are perturbed by δ
  - $[\![\mathcal{T}_\delta]\!]$

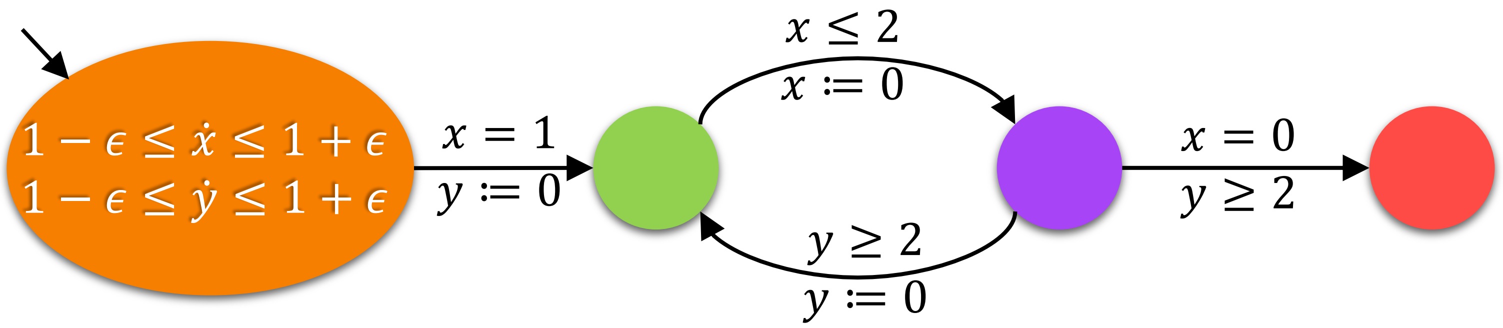


Types of Perturbation

- Only guards are perturbed by δ
  - $[\![\mathcal{T}_\delta]\!]$

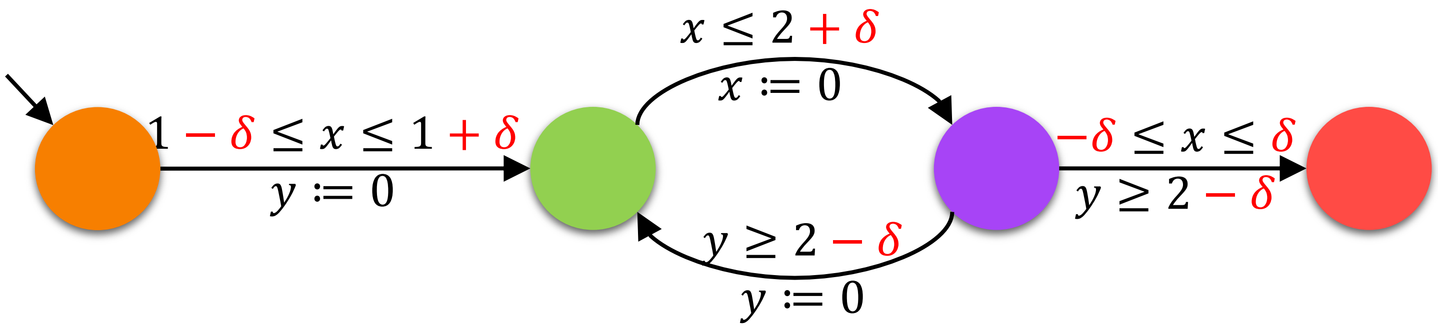


- Only clocks are drifted by ϵ
  - $[\![\mathcal{T}^\epsilon]\!]$

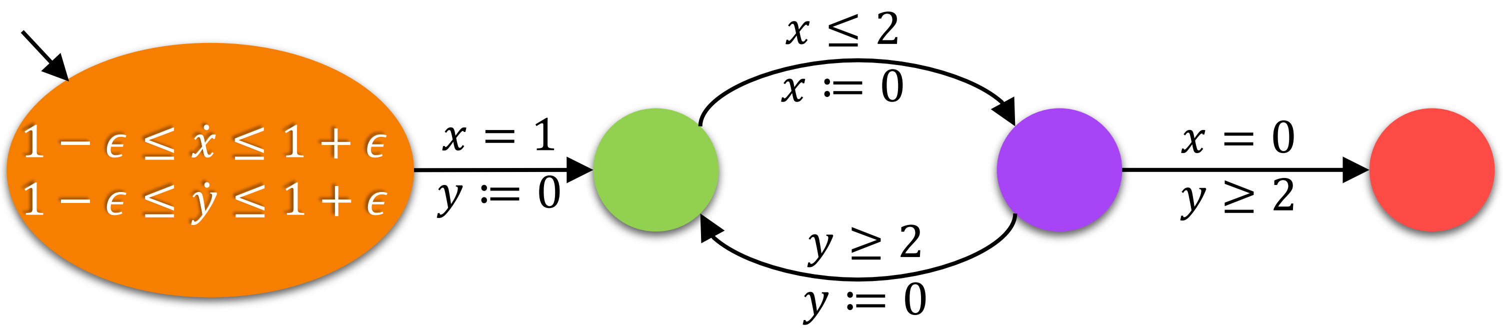


Types of Perturbation

- Only guards are perturbed by δ
  - $[\![\mathcal{T}_\delta]\!]$

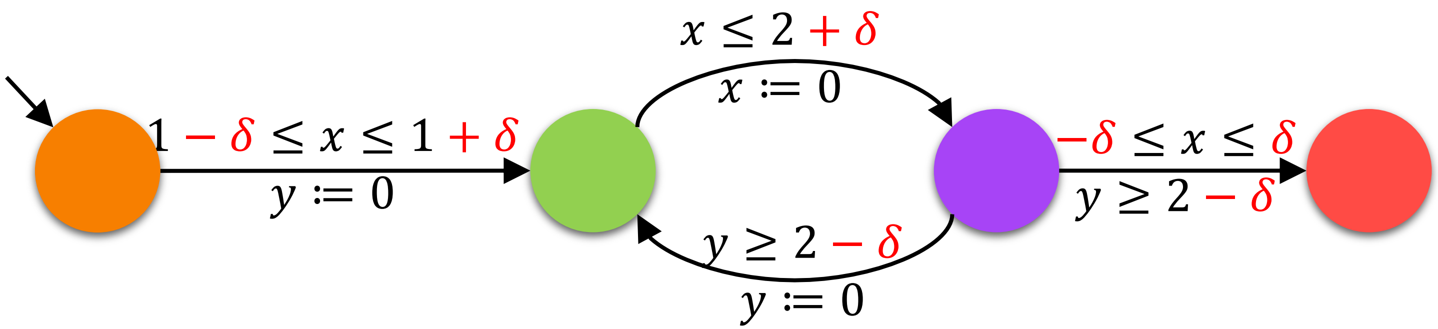


- Only clocks are drifted by ϵ
  - $[\![\mathcal{T}^\epsilon]\!]$
- Guards are perturbed by δ
  Clocks are perturbed by ϵ
  - $[\![\mathcal{T}_\delta^\epsilon]\!]$

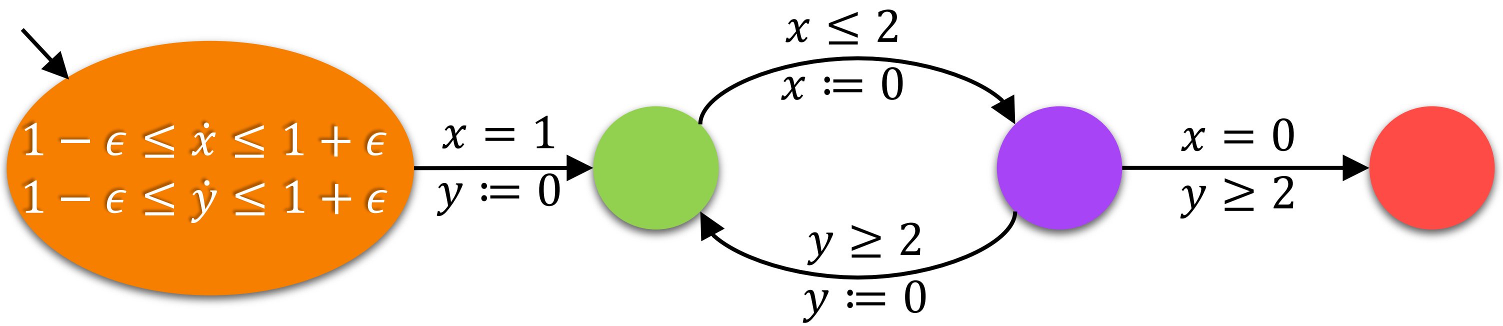


Types of Perturbation

- Only guards are perturbed by δ
  - $[\![\mathcal{T}_\delta]\!]$



- Only clocks are drifted by ϵ
  - $[\![\mathcal{T}^\epsilon]\!]$



- Guards are perturbed by δ
  Clocks are perturbed by ϵ
  - $[\![\mathcal{T}_\delta^\epsilon]\!]$
- Only positive guards are perturbed by δ
  - $[\![\mathcal{T}_{+\delta}]\!]$

ω-Regular Properties

- We only consider Repeated Reachability $\Box\Diamond E$
  - Only to simplify presentation

$$\exists \epsilon : \mathbb{R}_+ \bullet \forall \tau : [\![\mathcal{T}^{\epsilon}]\!] \bullet \tau \vDash \Box\Diamond E$$

$$\exists \delta : \mathbb{R}_+ \bullet \forall \tau : [\![\mathcal{T}_{\delta}]\!] \bullet \tau \vDash \Box\Diamond E$$

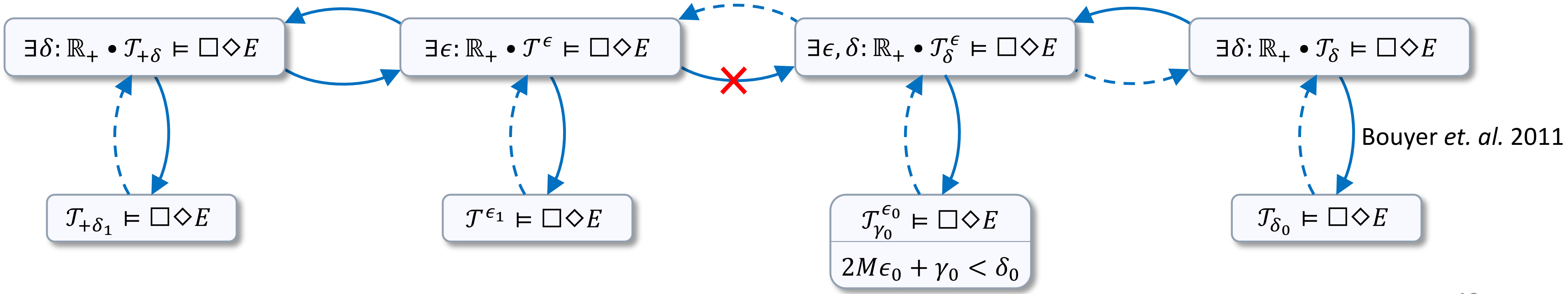
$$\exists \epsilon, \delta : \mathbb{R}_+ \bullet \forall \tau : [\![\mathcal{T}_{\delta}^{\epsilon}]\!] \bullet \tau \vDash \Box\Diamond E$$

$$\exists \delta : \mathbb{R}_+ \bullet \forall \tau : [\![\mathcal{T}_{+\delta}]\!] \bullet \tau \vDash \Box\Diamond E$$

  - Proofs directly apply to Büchi Condition

ω-Regular Model Checking Results

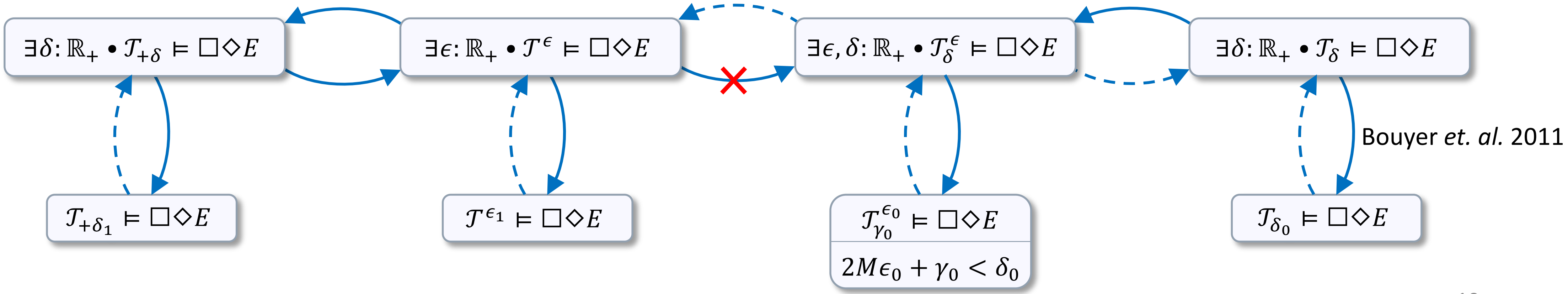
- $\delta_0 \coloneqq \frac{1}{2}\left(5(W+1)|X|^3\left(2|Q|(|X|!)4^{|X|}+4\right)^2\right)^{-1}$
  - Only Exponentially Small
  - Adding one location makes δ_0 at most 12 times smaller
  - Independent of Number of Edges
- M is the maximum constant in $\mathcal{T}$
- $\delta_1 \coloneqq \frac{\delta_0}{24}$
- $\epsilon_1 \coloneqq \frac{\delta_1}{2\mathrm{M}}$



ω-Regular Model Checking Results

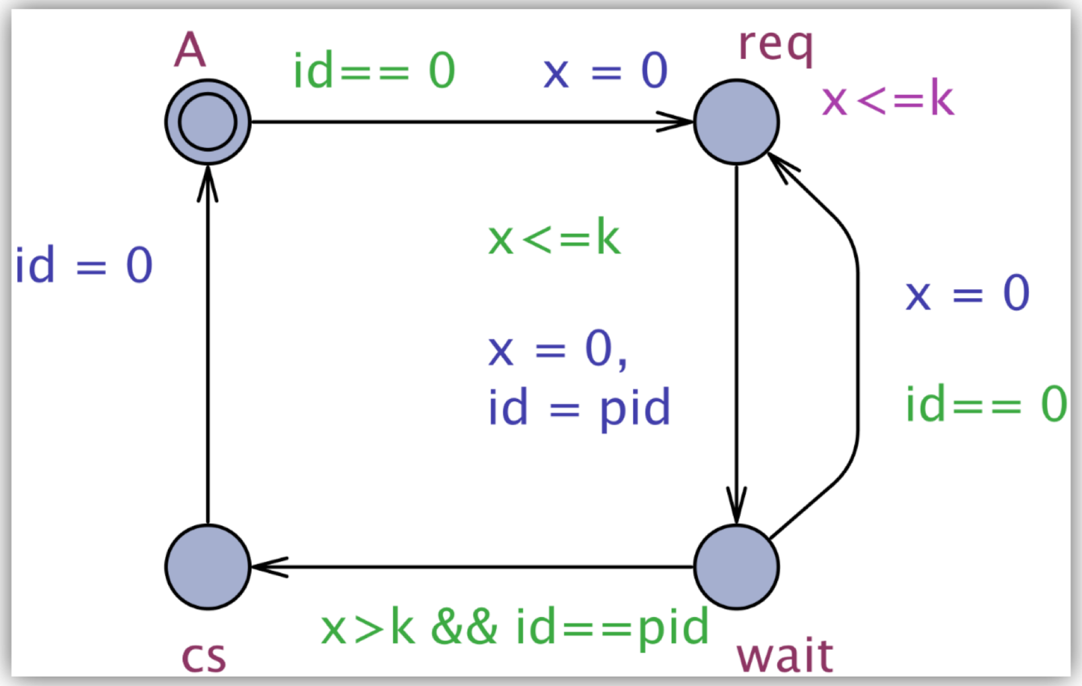
- $\delta_0 \coloneqq \frac{1}{2}\left(5(W+1)|X|^3\left(2|Q|(|X|!)4^{|X|}+4\right)^2\right)^{-1}$
  - Only Exponentially Small
  - Adding one location makes δ_0 at most 12 times smaller
  - Independent of Number of Edges
- M is the maximum constant in $\mathcal{T}$
- $\delta_1 \coloneqq \frac{\delta_0}{24}$
- $\epsilon_1 \coloneqq \frac{\delta_1}{2M}$

- All Problems are PSPACE-complete



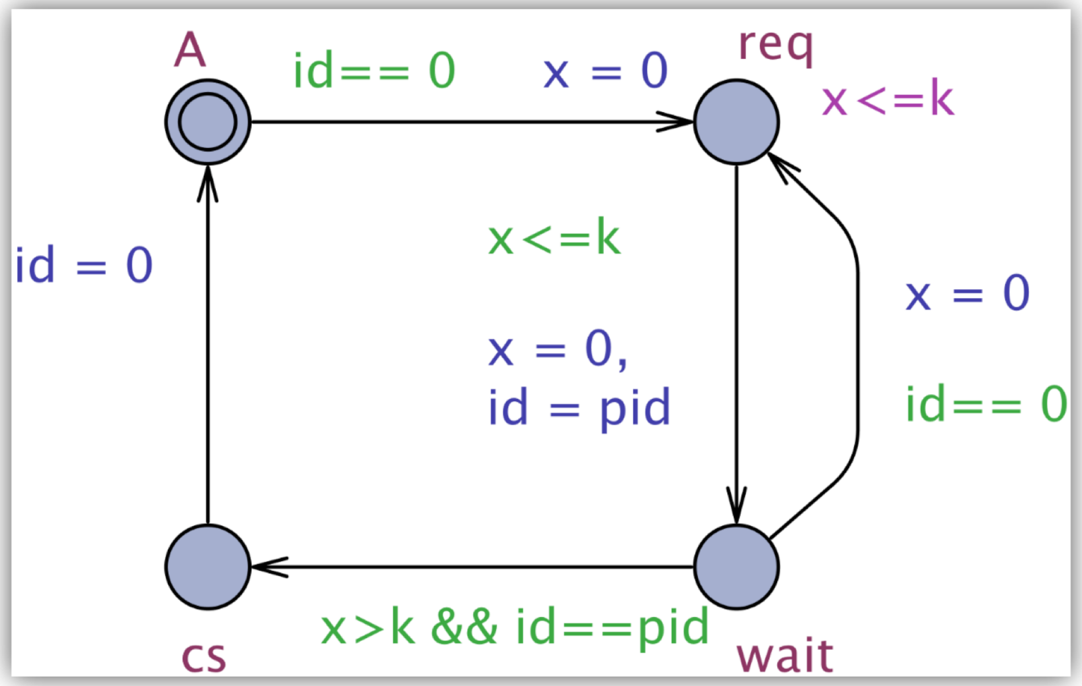
Experimental Results

- Fischer Mutual Exclusion Protocol
  - No two processes go to CS at the same time
  - No deadlock
  - Every request will eventually be answered



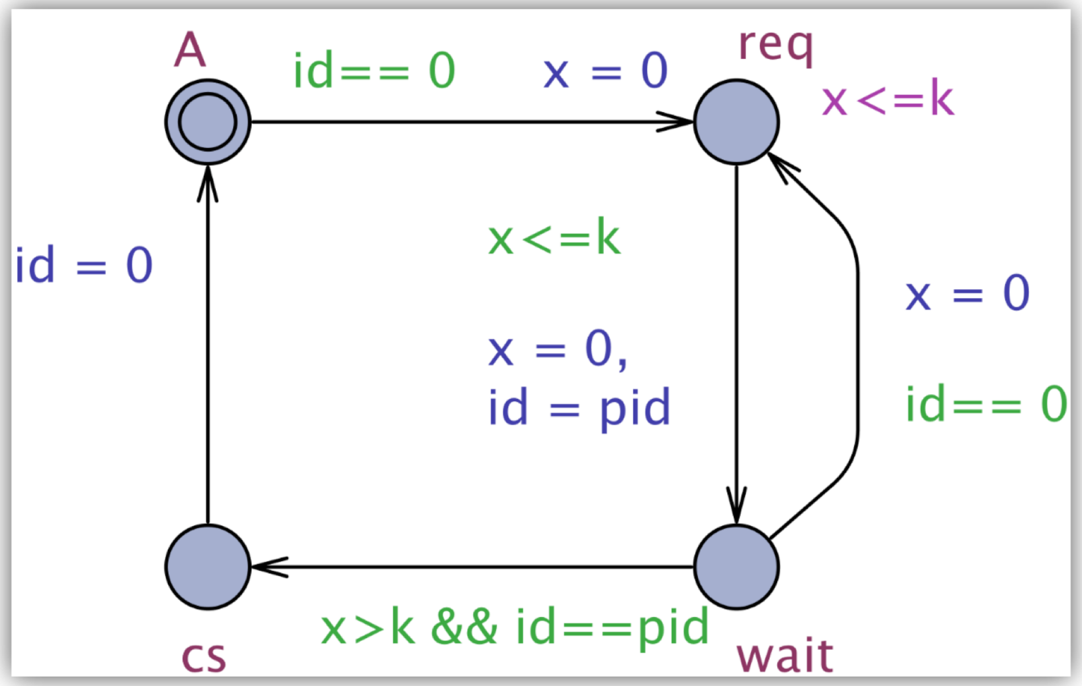
Experimental Results

- Fischer Mutual Exclusion Protocol
  - No two processes go to CS at the same time
  - No deadlock
  - Every request will eventually be answered
- We tested it for 6 processes
  - 4096 Locations
    - 4032 Backward Reachable
  - 30336 Edges



Experimental Results

- Fischer Mutual Exclusion Protocol
  - No two processes go to CS at the same time
  - No deadlock
  - Every request will eventually be answered
- We tested it for 6 processes
  - 4096 Locations
    - 4032 Backward Reachable
  - 30336 Edges
- $\mathcal{T}_{0.01}$ satisfies all these properties
  - Less than 2 seconds
- We conclude $\mathcal{T}_{\delta}^{\epsilon}$ does the same
  - For $\epsilon \coloneqq \frac{0.01}{12}$ and $\delta \coloneqq \frac{0.01}{2}$

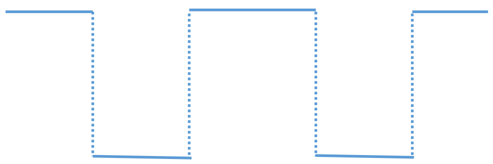


What Next?

- Robust Satisfiability/Model Checking of Metric Temporal Logic (MTL)
  - Or its subclasses
- Robust Monitoring of Signal Temporal Logic
  - The current robust semantics might fail even for valid formulas!

What Next?

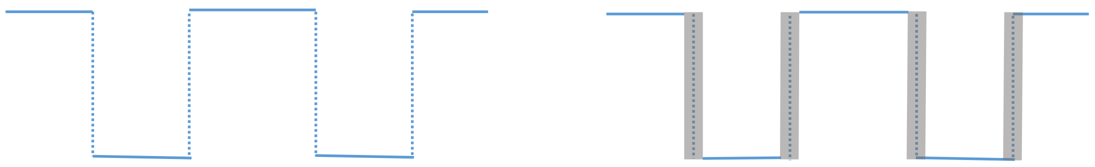
- Robust Satisfiability/Model Checking of Metric Temporal Logic (MTL)
  - Or its subclasses
- Robust Monitoring of Signal Temporal Logic
  - The current robust semantics might fail even for valid formulas!



What Next?

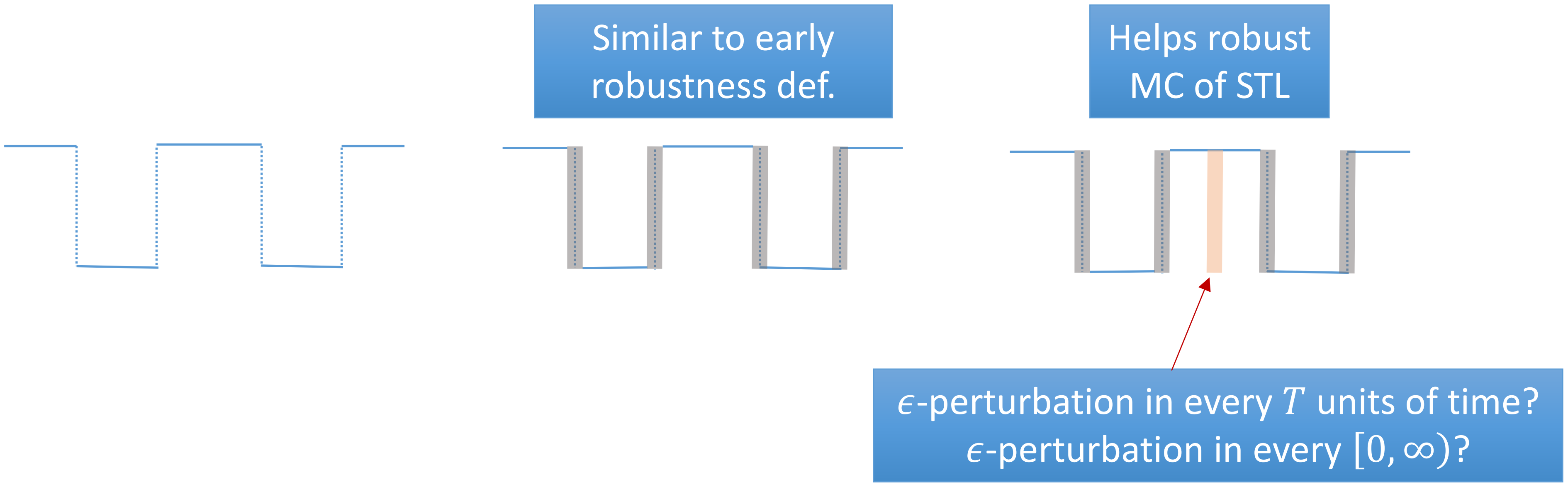
- Robust Satisfiability/Model Checking of Metric Temporal Logic (MTL)
  - Or its subclasses
- Robust Monitoring of Signal Temporal Logic
  - The current robust semantics might fail even for valid formulas!

Similar to early robustness def.



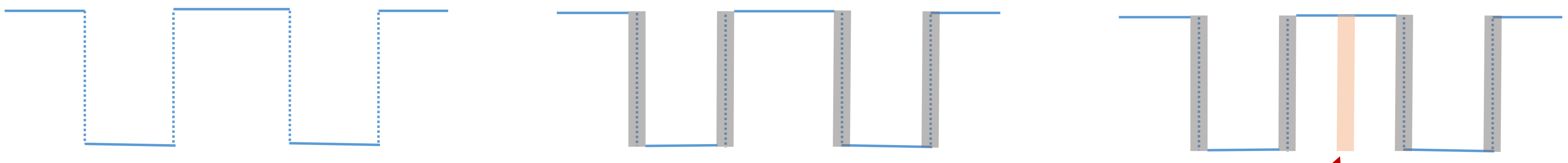
What Next?

- Robust Satisfiability/Model Checking of Metric Temporal Logic (MTL)
  - Or its subclasses
- Robust Monitoring of Signal Temporal Logic
  - The current robust semantics might fail even for valid formulas!



What Next?

- Robust Satisfiability/Model Checking of Metric Temporal Logic (MTL)
  - Or its subclasses
- Robust Monitoring of Signal Temporal Logic
  - The current robust semantics might fail even for valid formulas!



$\square_{\geq 0}(p \rightarrow \Diamond_{(0+\epsilon,3-\epsilon]} q)$

Robustness Helps

Statistical Verification of Hybrid Automata

HSCC 2015, 2017

ADHS 2015, 2018

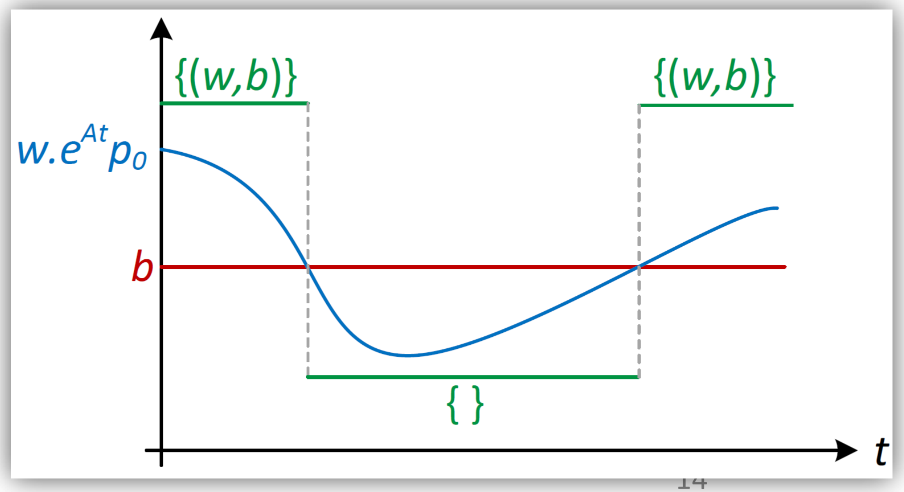
CDC 2016

Temporal Properties about CTMC

- System is expressed using a Continuous Time Markov Chains
  - Rate matrix A is given
  - Initial probability distribution p_0 is also given
    - Probability distribution at time t is given by $e^{At}p_0$

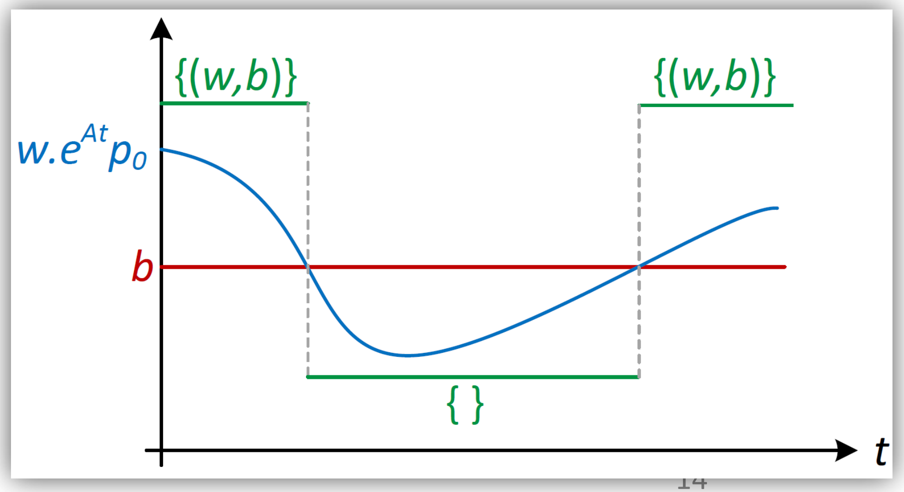
Temporal Properties about CTMC

- System is expressed using a Continuous Time Markov Chains
  - Rate matrix A is given
  - Initial probability distribution p_0 is also given
    - Probability distribution at time t is given by $e^{At}p_0$
- Properties are expressed using Signal Temporal Logic (STL)
  - Atomic propositions are in the form of $w \cdot e^{At}p_0 \geq b$



Temporal Properties about CTMC

- System is expressed using a Continuous Time Markov Chains
  - Rate matrix A is given
  - Initial probability distribution p_0 is also given
    - Probability distribution at time t is given by $e^{At}p_0$
- Properties are expressed using Signal Temporal Logic (STL)
  - Atomic propositions are in the form of $w \cdot e^{At}p_0 \geq b$
- Deterministic behavior
  - Non-probabilistic
  - Unique signal



Temporal Properties about CTMC

- Very similar problem has been solved algebraically in 2001
  - Model Checking Continuous Time Markov Chains by Adnan Aziz *et. al.*

Temporal Properties about CTMC

- Very similar problem has been solved algebraically in 2001
  - Model Checking Continuous Time Markov Chains by Adnan Aziz *et. al.*
- So problem is decidable
  - They use algebraic numbers
  - What is complexity of checking $\ln \frac{a}{b} \geq c$ when $a, b, c \colon \mathbb{N}_+$?

Temporal Properties about CTMC

- Very similar problem has been solved algebraically in 2001
  - Model Checking Continuous Time Markov Chains by Adnan Aziz *et. al.*
- So problem is decidable
  - They use algebraic numbers
  - What is complexity of checking $\ln\frac{a}{b} \geq c$ when $a, b, c: \mathbb{N}_+$?
- To improve performance, we wanted to use statistical techniques
  - Simulate the system enough number of times
  - Provide some error guarantee

What can be guaranteed?

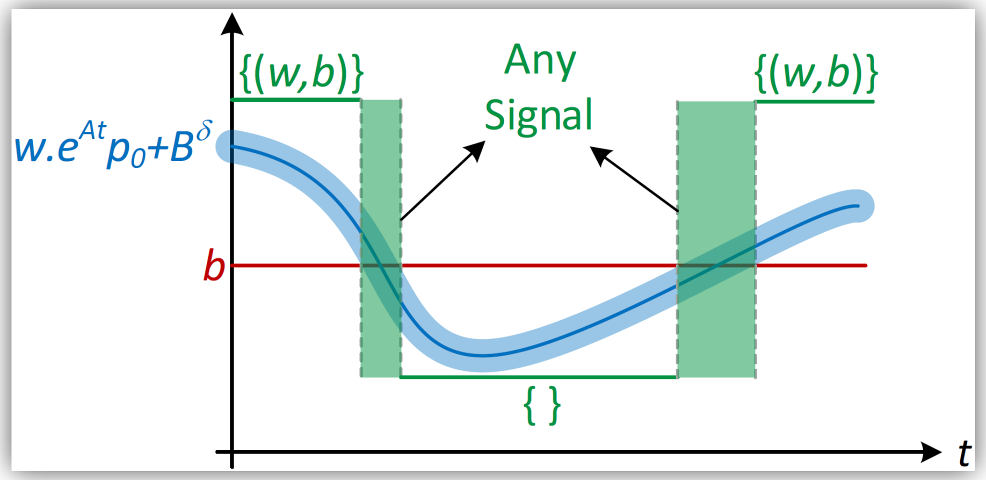
- Probability of returning wrong YES/NO is bounded

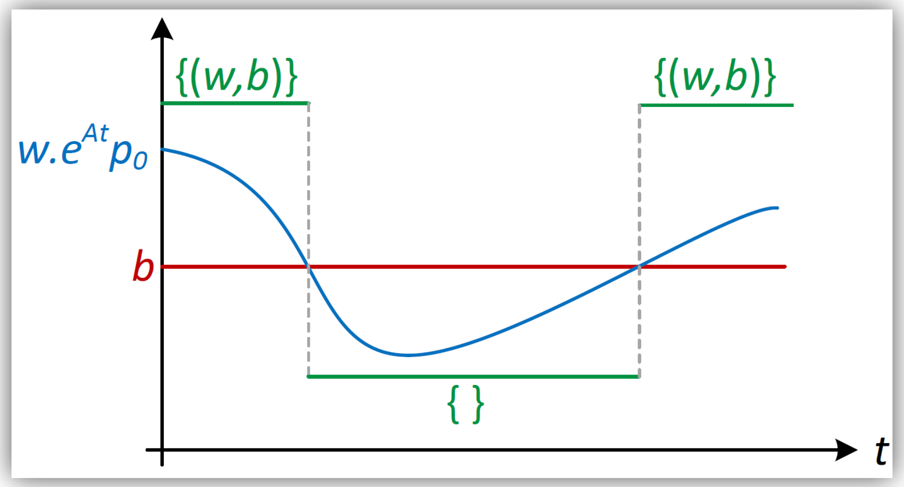
$$\mathbb{P}[res = \texttt{no} \mid C \models \phi] \leq \alpha$$

$$\mathbb{P}[res = \texttt{yes} \mid C \not\models \phi] \leq \alpha$$

- Probability of returning UNKNOWN is also bounded

$$\mathbb{P}[res = \texttt{unknown}] \leq \alpha + \beta$$





What is Next?

- When and how we can do this?
  - Verify deterministic (non-probabilistic) system using statistical techniques?
  - Much better performance
- What kind of robustness we need?

Abstraction Helps

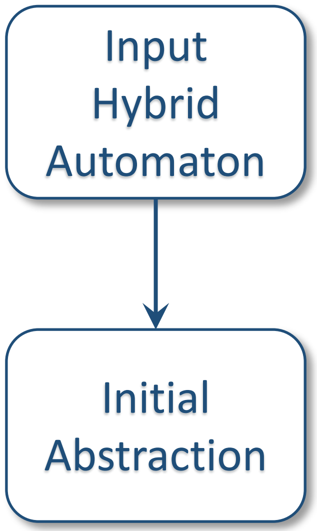
Reachability in Hybrid Automata

TACAS 2016-2017

CONCUR 2018

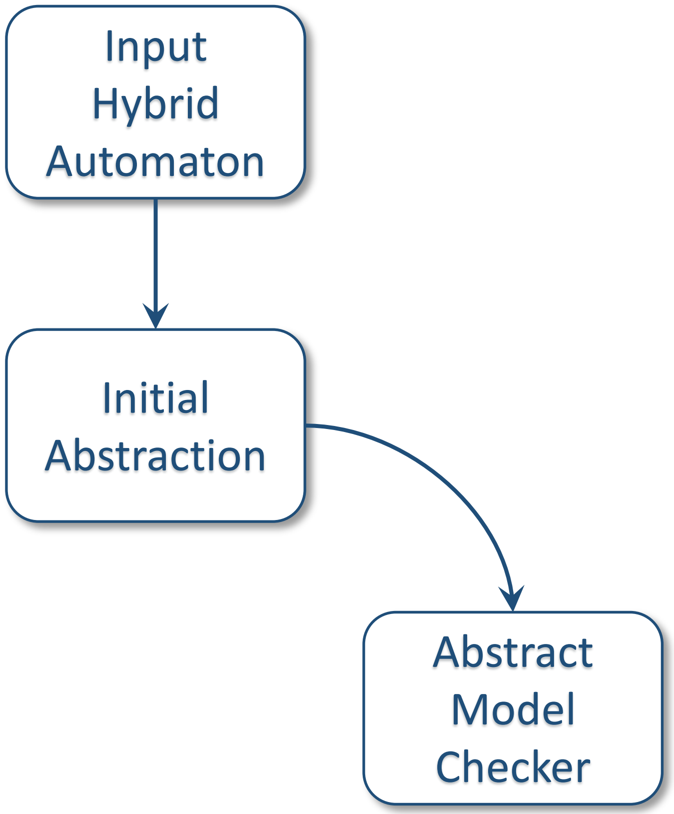
CEGAR Loop Edmund Clarke, 2000

- Simpler Differential Inclusions
- Abstraction
  - Finite vs. Infinite
  - Merging Locations Location
  - Removing Variables
  - Must over-approximate



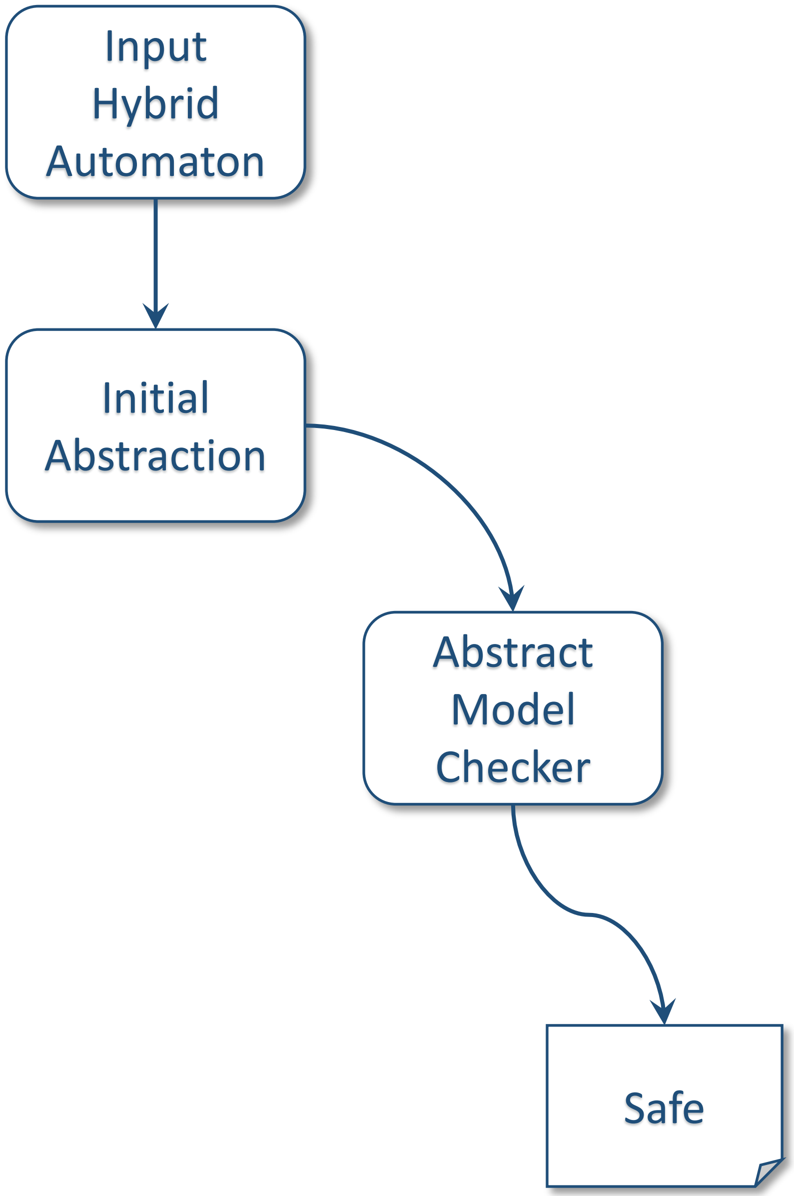
CEGAR Loop Edmund Clarke, 2000

- Simpler Differential Inclusions
- Abstraction
  - Finite vs. Infinite
  - Merging Locations Location
  - Removing Variables
  - Must over-approximate



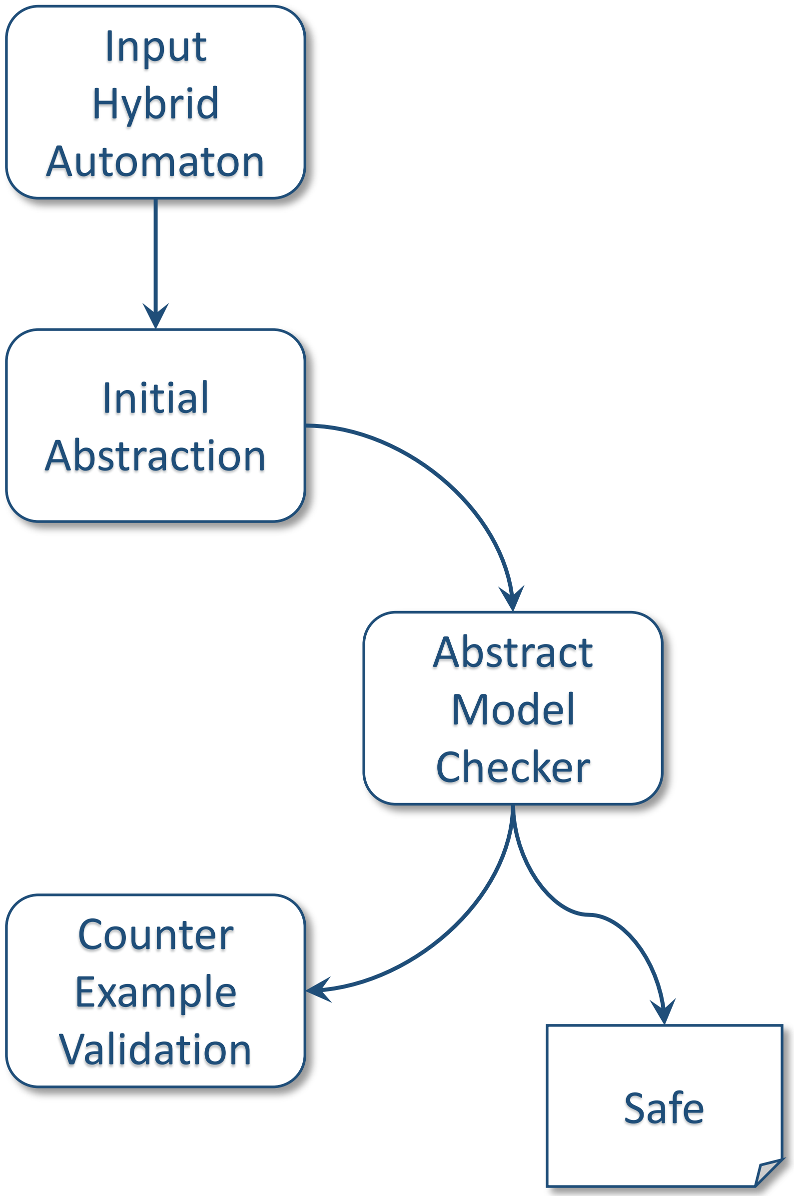
CEGAR Loop Edmund Clarke, 2000

- Simpler Differential Inclusions
- Abstraction
  - Finite vs. Infinite
  - Merging Locations Location
  - Removing Variables
  - Must over-approximate



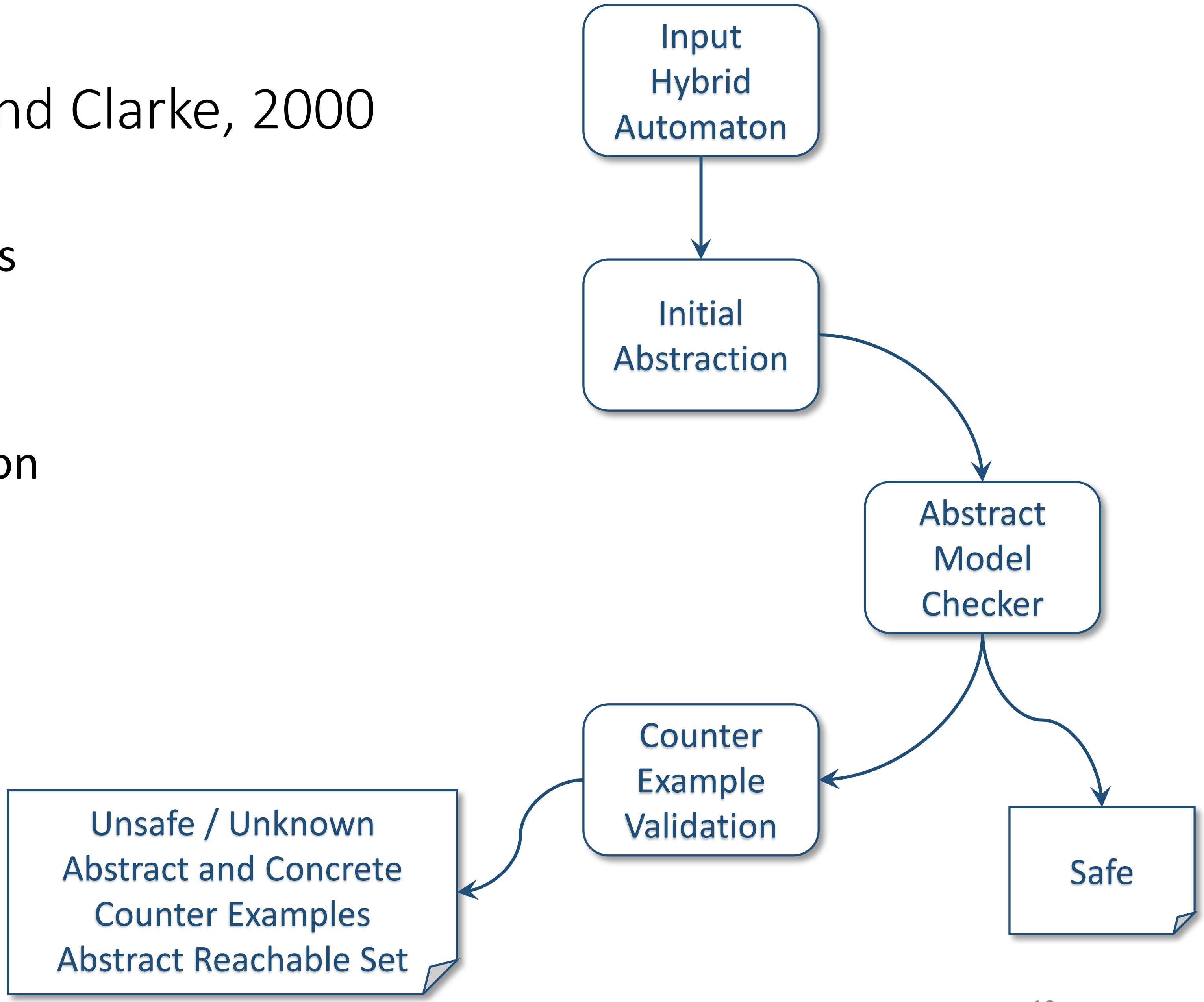
CEGAR Loop Edmund Clarke, 2000

- Simpler Differential Inclusions
- Abstraction
  - Finite vs. Infinite
  - Merging Locations Location
  - Removing Variables
  - Must over-approximate



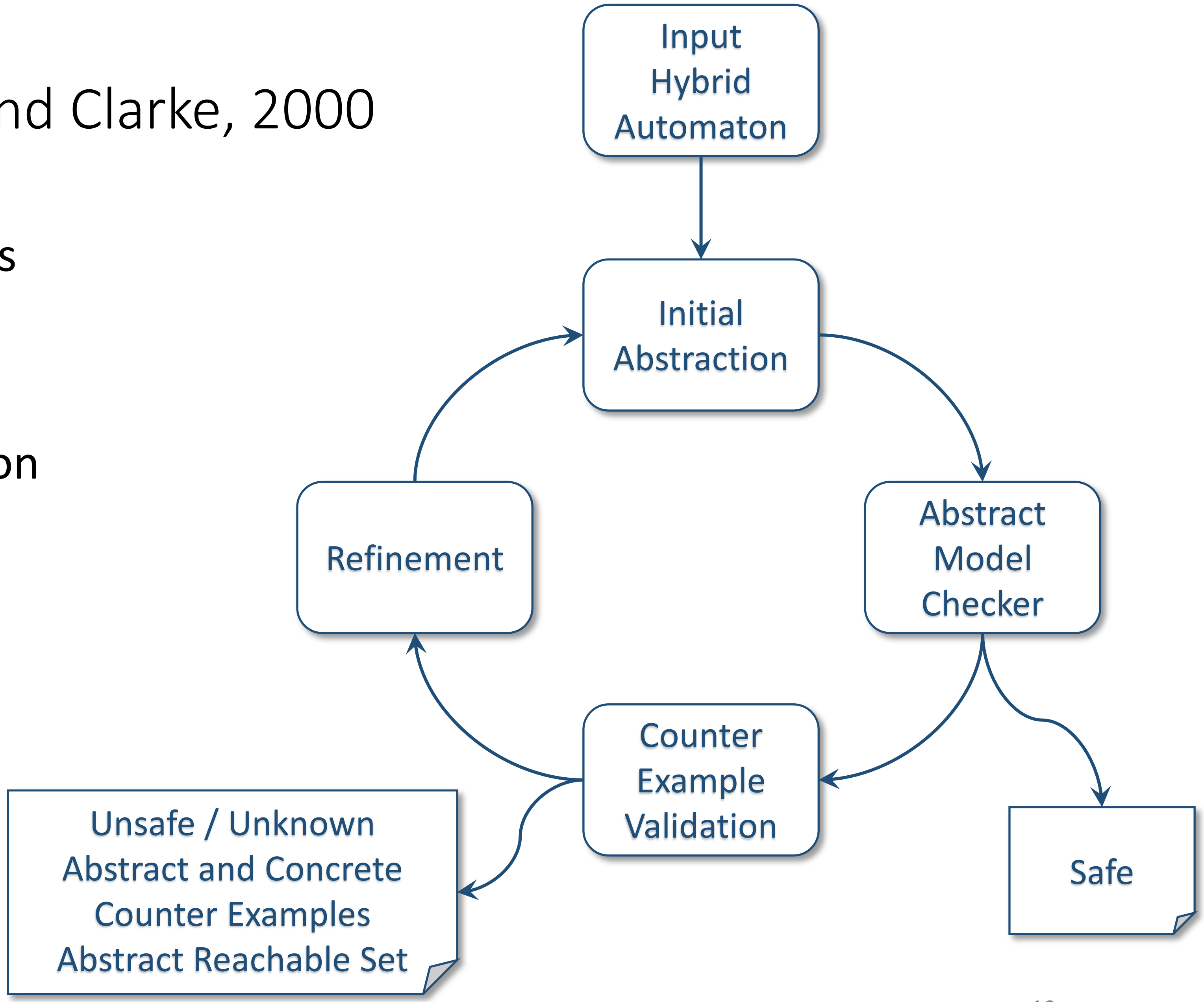
CEGAR Loop Edmund Clarke, 2000

- Simpler Differential Inclusions
- Abstraction
  - Finite vs. Infinite
  - Merging Locations Location
  - Removing Variables
  - Must over-approximate



CEGAR Loop Edmund Clarke, 2000

- Simpler Differential Inclusions
- Abstraction
  - Finite vs. Infinite
  - Merging Locations Location
  - Removing Variables
  - Must over-approximate
- What should be refined?



Experimental Results (affine dynamics)

- Constraints and continuous dynamics are specified using polyhedra

			HARE						SpaceEx			PHAVer			SpaceEx AGAR			
Model	Dim.	Size	Time		Iters.		Safe		Time	FP.	Safe	Time	FP.	Safe	Merged Locs.	Time	FP.	Safe
			old	new	old	new	old	new										
Tank 16	3	3 / 6	< 1	< 1	1	1	✓	✓	3	✗	✗	1414	✗	✓	2	1133	✗	✓
Tank 17	3	3 / 6	< 1	< 1	1	1	✓	✓	5	✗	✓	1309	✗	✓	2	1041	✗	✓
Satellite 03	4	64 / 198	91	< 1	1	1	✗	✓	< 1	✗	✗	1804	✗	✗	28	> 600	---	---
Satellite 04	4	100 / 307	< 1	< 1	1	1	✓	✓	< 1	✗	✓	< 1	✓	✓	91	49	✓	✓
Satellite 11	4	576 / 1735	1	< 1	1	1	✓	✓	< 1	✗	✓	< 1	✓	✓	449	> 600	---	---
Satellite 15	4	1296 / 3895	2	< 1	1	1	✓	✓	< 1	✗	✓	< 1	✓	✓	264	> 600	---	---
Heater 03	3	4 / 6	> 600	54	---	1	---	✓	84	✗	✓	< 1	✓	✗	---	---	---	---
Heater 05	3	4 / 6	< 1	58	1	38	✗	✓	61	✗	✓	< 1	✓	✗	---	---	---	---
Heater 09	3	4 / 6	< 1	80	1	15	✗	✓	42	✗	✗	< 1	✓	✗	---	---	---	---
Nav 01	4	25 / 80	9	18	11	11	✓	✓	< 1	✓	✓	< 1	✓	✓	21	5	✓	✓
Nav 08	4	16 / 48	7	< 1	13	1	✓	✓	685	✗	✓	< 1	✓	✓	10	< 1	✓	✓
Nav 09	4	16 / 48	7	< 1	10	1	✓	✓	< 1	✗	✗	< 1	✓	✗	4	< 1	✓	✗
Nav 13	4	9 / 18	8	< 1	15	1	✓	✓	< 1	✗	✓	< 1	✓	✓	4	< 1	✓	✓
Nav 19	4	33 / 97	29	< 1	17	1	✓	✓	2	✗	✓	< 1	✓	✓	11	< 1	✓	✓

Experimental Results (non-linear dynamics)

- Constraints are specified using polyhedra
- Continuous dynamics are specified using (non-linear) ODEs
  - Whatever can be supported by dReach

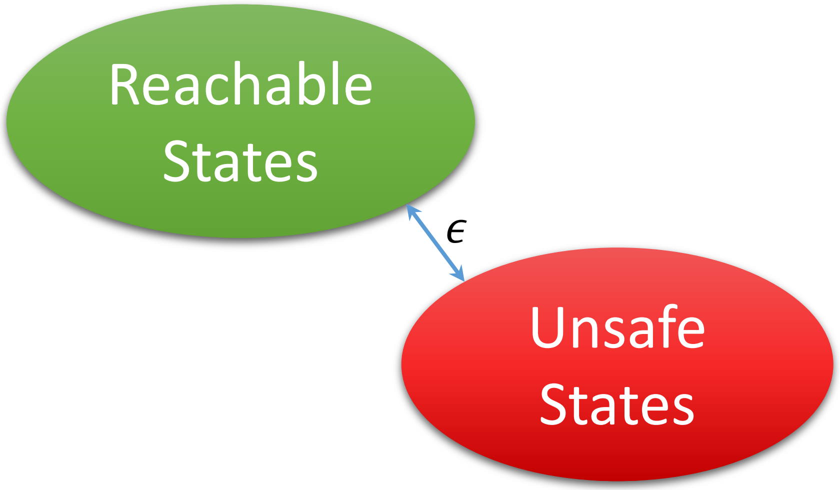
			HARE			C2E2	HSolver	FLOW*
Model	Dim.	Size	Reached Abst. Size	Time Bound	Time	Time	Time	Time
Van der Pol	2	1 / 0	26 / 194	∞	< 1	56	3*	> 600
Jet Engine	2	1 / 0	189 / 1330	∞	55	56	2*	> 600
Cardiac Cell	2	2 / 2	249 / 1783	∞	16	50	< 1*	25
Cardiac Control	3	2 / 2	270 / 3974	∞	153	> 600	> 600*	41
Clock	3	1 / 0	9 / 56	∞	< 1	---	< 1	< 1
Sinusoid	2	1 / 0	32 / 62	10	< 1	1	7	---

What can be guaranteed?

- Assume every constraint uses non-strict inequality

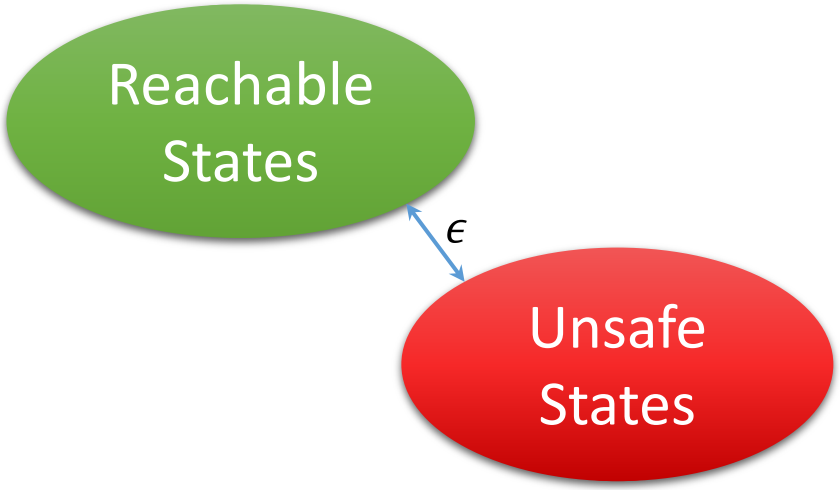
What can be guaranteed?

- Assume every constraint uses non-strict inequality
- Assume there is a positive distance between reachable and unsafe regions
  - System is robustly safe
  - Reachable and unsafe regions are robustly separated
  - Definition based on semantics of the system



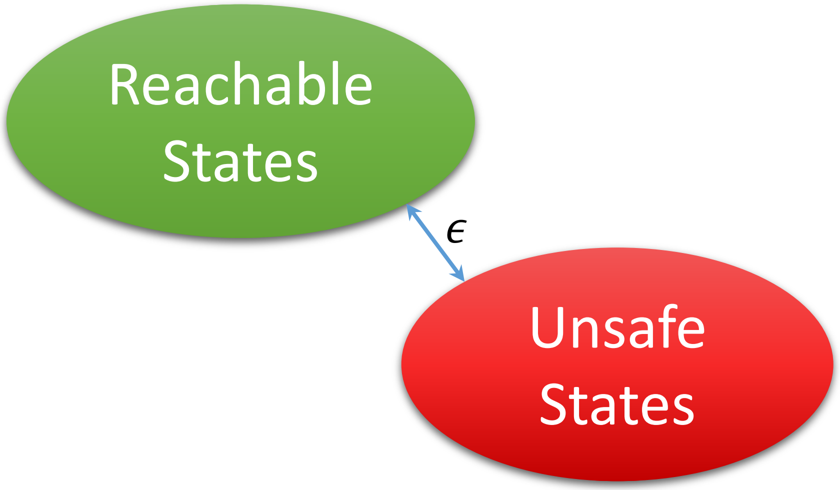
What can be guaranteed?

- Assume every constraint uses non-strict inequality
- Assume there is a positive distance between reachable and unsafe regions
    - System is robustly safe
    - Reachable and unsafe regions are robustly separated
    - Definition based on semantics of the system
- Prove: every spurious counter-example will be eventually eliminated



What can be guaranteed?

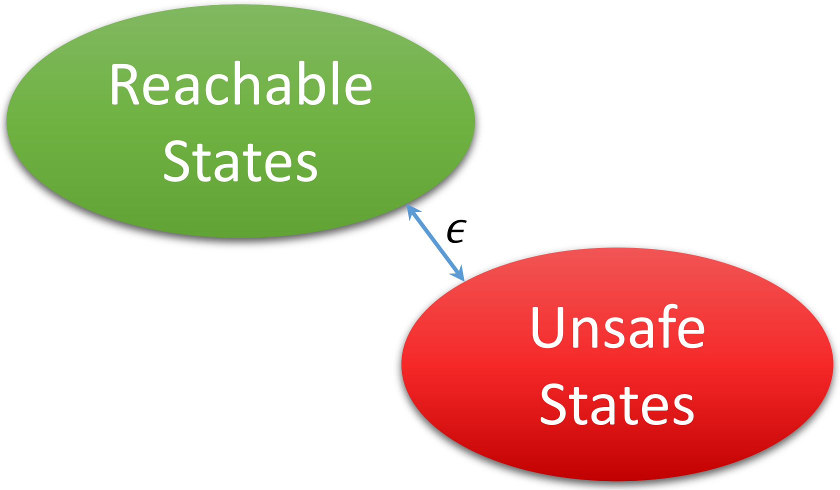
- Assume every constraint uses non-strict inequality
- Assume there is a positive distance between reachable and unsafe regions
  - System is robustly safe
  - Reachable and unsafe regions are robustly separated
  - Definition based on semantics of the system
- Prove: every spurious counter-example will be eventually eliminated



- We use dReach

What can be guaranteed?

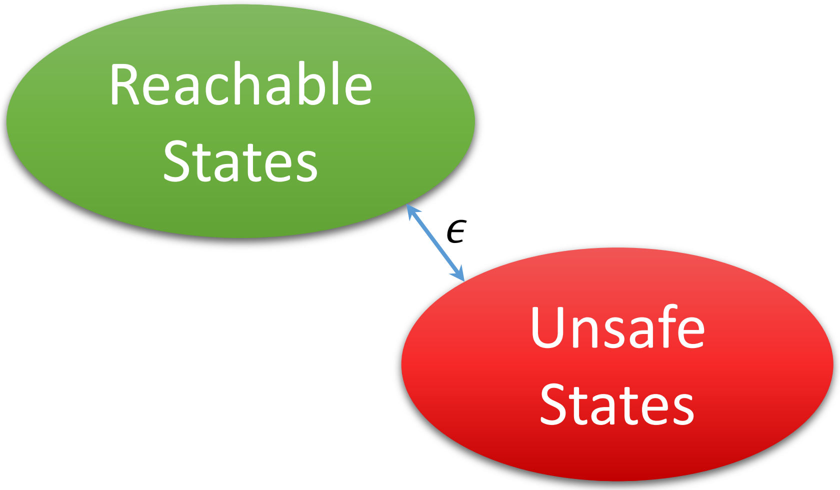
- Assume every constraint uses non-strict inequality
- Assume there is a positive distance between reachable and unsafe regions
  - System is robustly safe
  - Reachable and unsafe regions are robustly separated
  - Definition based on semantics of the system
- Prove: every spurious counter-example will be eventually eliminated



- We use dReach
- dReach uses dReal

What can be guaranteed?

- Assume every constraint uses non-strict inequality
- Assume there is a positive distance between reachable and unsafe regions
  - System is robustly safe
  - Reachable and unsafe regions are robustly separated
  - Definition based on semantics of the system
- Prove: every spurious counter-example will be eventually eliminated



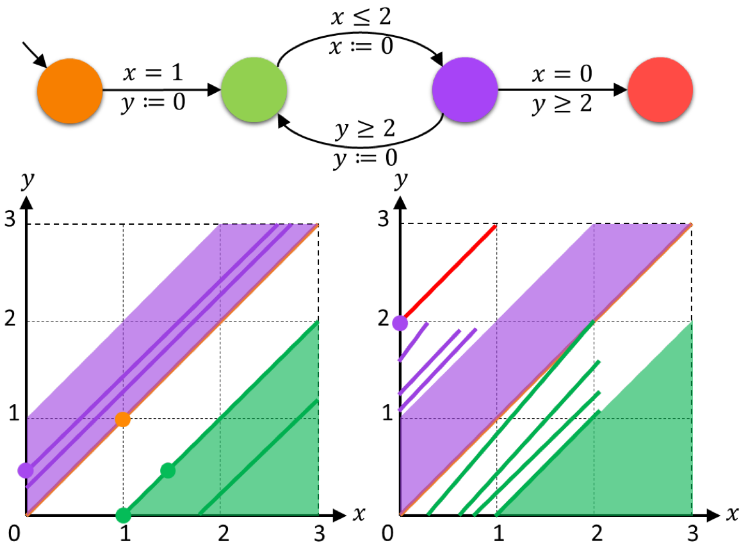
- We use dReach
- dReach uses dReal
- dReal perturbs syntax of formulas
  - UNSAT: the system is safe (spurious counter-example)
  - δ-SAT: the perturbed system is unsafe

What can be guaranteed?

- What is the relation between syntactic and semantic perturbations/robustness?
  - Can they become arbitrary close?
  - Syntactic perturbation is used to deal with computational complexity
  - Sematic perturbation is used to represent robustness

What can be guaranteed?

- What is the relation between syntactic and semantic perturbations/robustness?
  - Can they become arbitrary close?
  - Syntactic perturbation is used to deal with computational complexity
  - Sematic perturbation is used to represent robustness
  - In general NO
    - Unbounded number of transitions
    - Strict inequalities

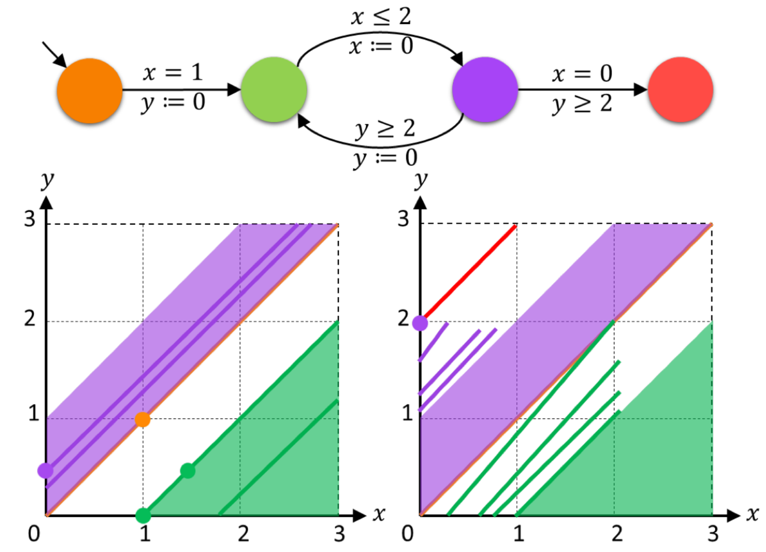


What can be guaranteed?

- What is the relation between syntactic and semantic perturbations/robustness?
  - Can they become arbitrary close?
  - Syntactic perturbation is used to deal with computational complexity
  - Sematic perturbation is used to represent robustness
  - In general NO
    - Unbounded number of transitions
    - Strict inequalities
- We proved bounded ϵ-Simulation is possible

$$\forall \epsilon \in \mathbb{R}_+, k \in \mathbb{N} \bullet \exists \delta \in \mathbb{R}_+ \bullet \mathcal{H}^\delta \preceq_k^\epsilon \mathcal{H} \wedge \mathcal{H} \preceq_k^\epsilon \mathcal{H}^\delta$$

- Bisimulation is impossible



What is Next?

- We proved bounded ϵ-Simulation is possible

$$\forall \epsilon \in \mathbb{R}_+, k \in \mathbb{N} \bullet \exists \delta \in \mathbb{R}_+ \bullet \mathcal{H}^\delta \preceq_k^\epsilon \mathcal{H} \wedge \mathcal{H} \preceq_k^\epsilon \mathcal{H}^\delta$$

- Find δ for the given ϵ
  - Anything more expressive than Timed Automata

Thank You