Lecture 21: Abstractions
Counterexample-guided abstraction refinement

Huan Zhang
huan@huan-zhang.com

Slides adapted from Prof. Sayan Mitra's slides in Fall 2021

Homework and final presentations

HW 3 released last week, due **4/27**

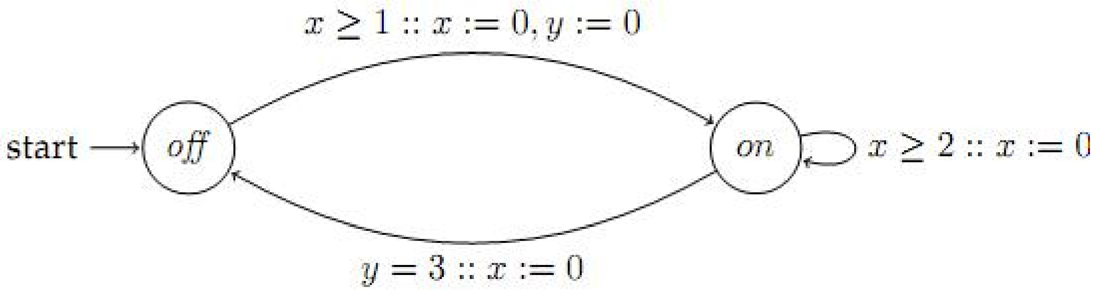
HW 4 will be released this week, due **5/6**

Final project presentation slides due **4/30, 8 am** (hard deadline, since presentations will start at 11 am)

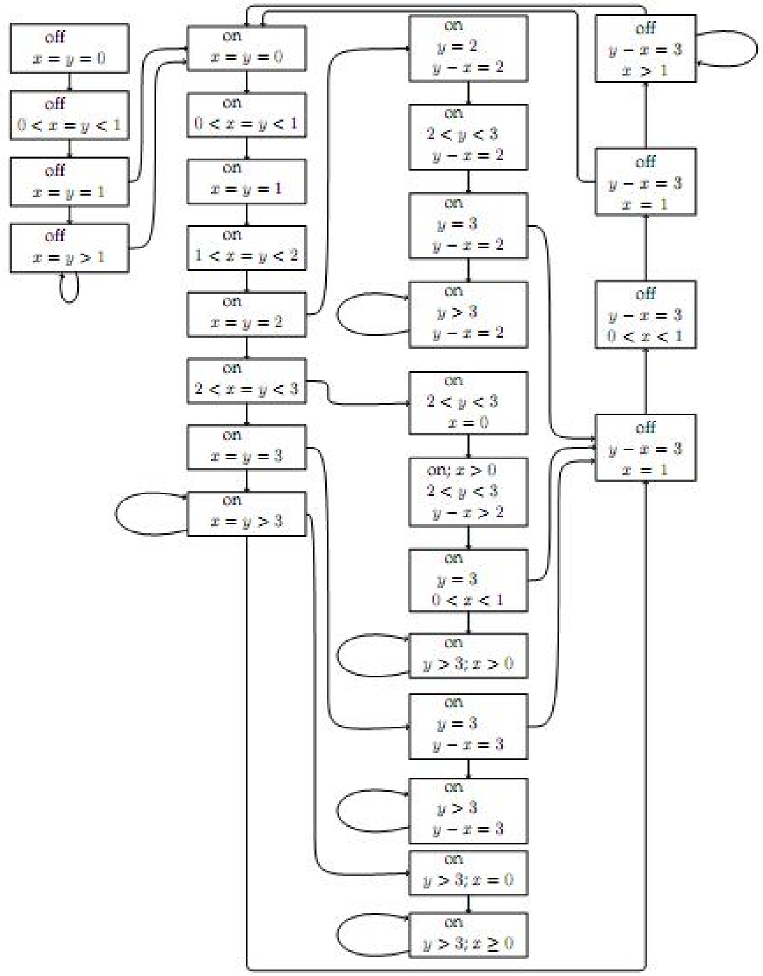
Final project presentations: Last week of instruction (schedule TBA)

Final project report due: **5/11** (hard deadline due to final grades uploading requirement)

Review: reachability of Integral Time Automaton



Integral Time Automaton



Region Automaton

Abstractions and Simulations

Consider models that have the same external interface (input/output variables and actions)

We would like to *approximate* one (hybrid) automaton H_1 with another one H_2

- We can over-approximate the reachable states of H_1 with those of H_2
- This would ensure that invariants of H_2 *carry over* to H_1

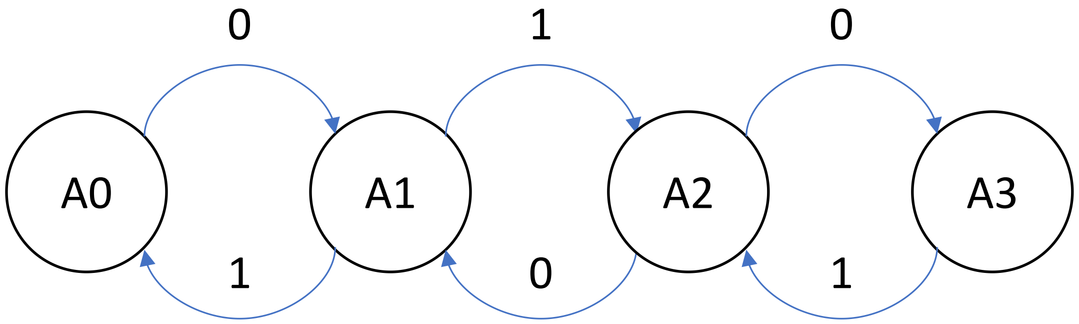
We would like to go beyond invariants, and want to have more general requirements (e.g., CTL) carry over

H_2 should be ***simpler*** (smaller description, fewer states, transitions, linear dynamics, etc.) and preserve **some** properties of H_1 (and not others)

Verifying some requirements of $\boldsymbol{H_2}$ can then carry over requirements to $\boldsymbol{H_1}$

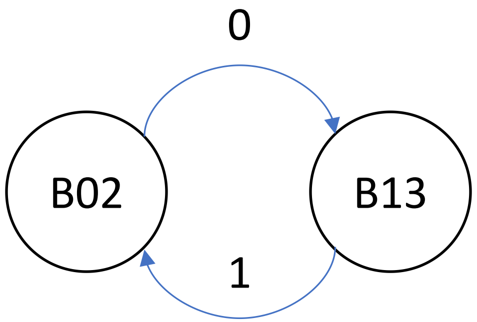
Finite state examples

A



$\text{Traces}_A = (01)^*$

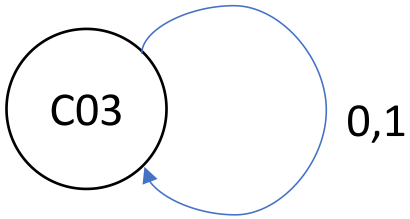
B



$\text{Traces}_B = 01^*$

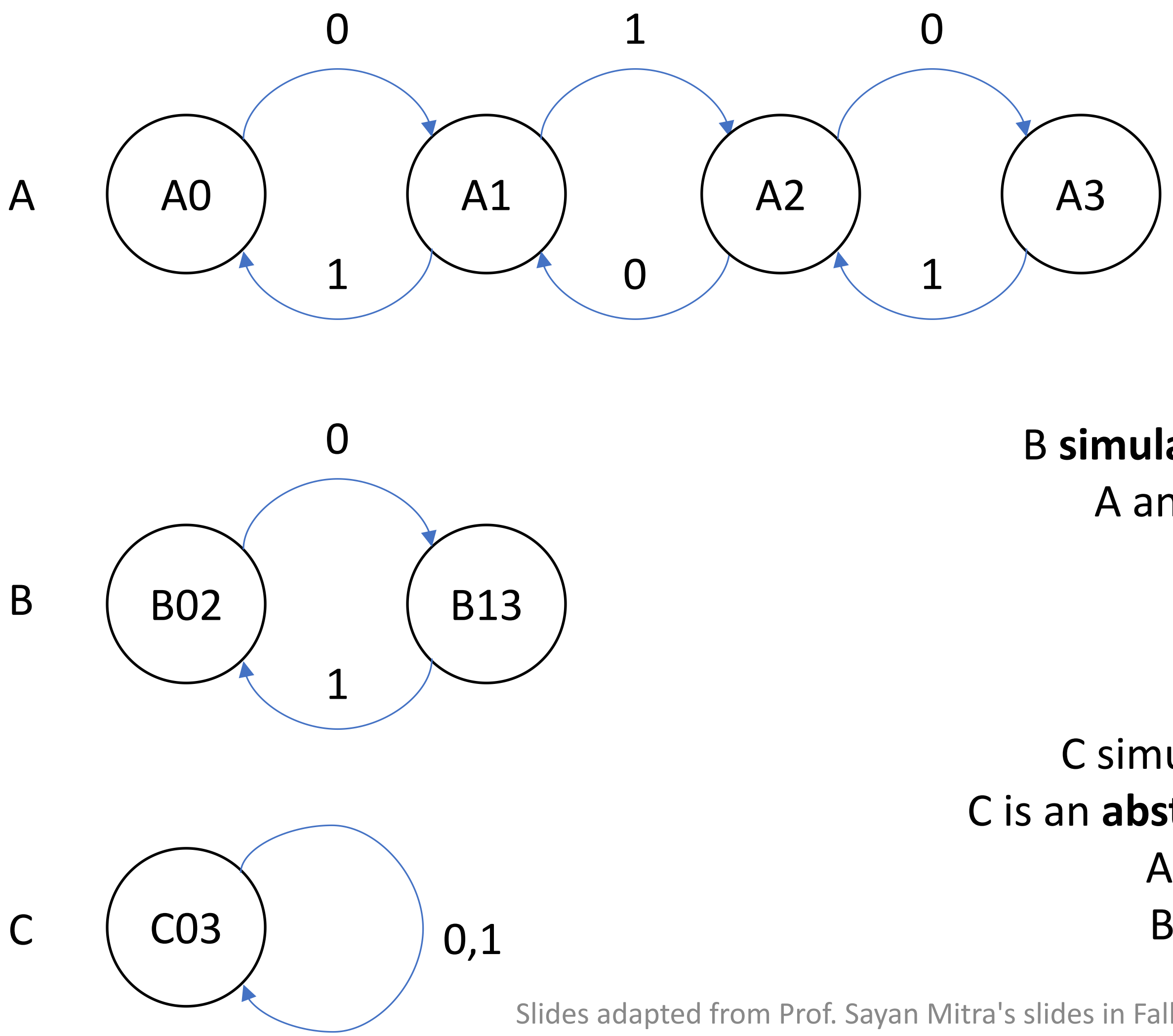
Trace := sequence of actions for some execution
Traces := set of all trace

C



$\text{Traces}_C = \{0,1\}^*$

Finite state examples



B **simulates** A and vice versa.
A and B are **bisimilar**.

C simulates both A and B.
C is an **abstraction** of both A and B.
A **implements** C.
B implements C.

Definitions

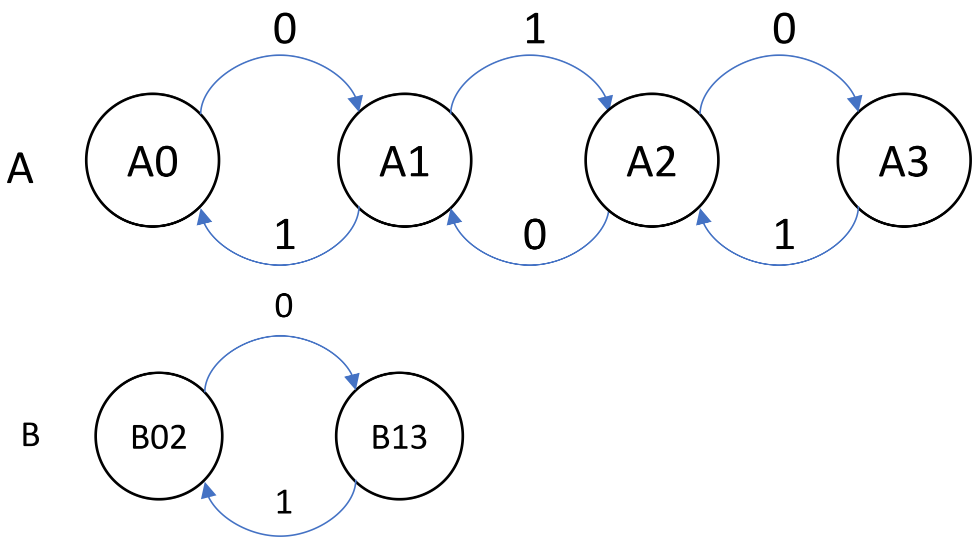
Let $\mathcal{A}$, $\mathcal{B}$, and $\mathcal{C}$ be **comparable** (identical I/O varaibles and actions) HA. If R_1 is a forward simulation from $\mathcal{A}$ to $\mathcal{B}$ and R_2 is a forward simulation from $\mathcal{B}$ to $\mathcal{C}$, then $R_1 \circ R_2$ is a forward simulation from $\mathcal{A}$ to $\mathcal{C}$

If $\mathcal{A}_1$ and $\mathcal{A}_2$ are comparable and $Traces_{\mathcal{A}_1} \subseteq Traces_{\mathcal{A}_2}$, we say $\mathcal{A}_1$ implements $\mathcal{A}_2$, and $\mathcal{A}_2$ is an abstraction of $\mathcal{A}_1$

The **implementation relation** is a preorder of the set of all (comparable) hybrid automata

(A preorder is a reflexive and transitive relation)

How to prove B simulates A?



Show there exists a **simulation relation** from states of A to states of B.
Say, $R = ((A0, B02), (A2, B02), (A1, B13), (A3, B13))$

Show that for every transition $Ai \rightarrow_A Ai'$ and $(Ai, Bj) \in R$ there exists Bj' such that

1. $Bj \rightarrow_B Bj'$
2. $(Ai', Bj') \in R$ (also written as $Ai' \; R \; Bj'$)
3. $Trace(Bj \rightarrow_B Bj') = Trace(Ai \rightarrow_A Ai')$

Forward simulation relation

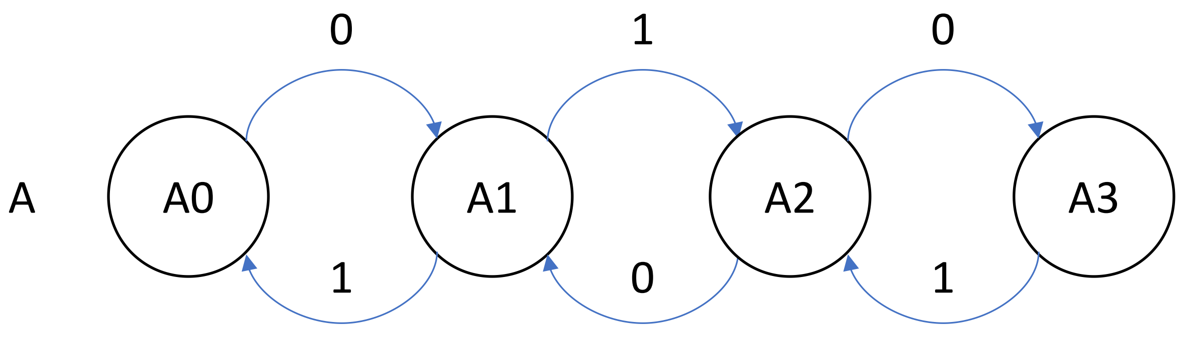
Consider a pair of automata $\mathcal{A}_1 = \langle Q_1,\ \Theta_1, A_1, D_1 \rangle$ and $\mathcal{A}_2 = \langle Q_2,\ \Theta_2, A_2, D_2 \rangle$.
Recall *trace* of an execution preserves the visible part of an execution

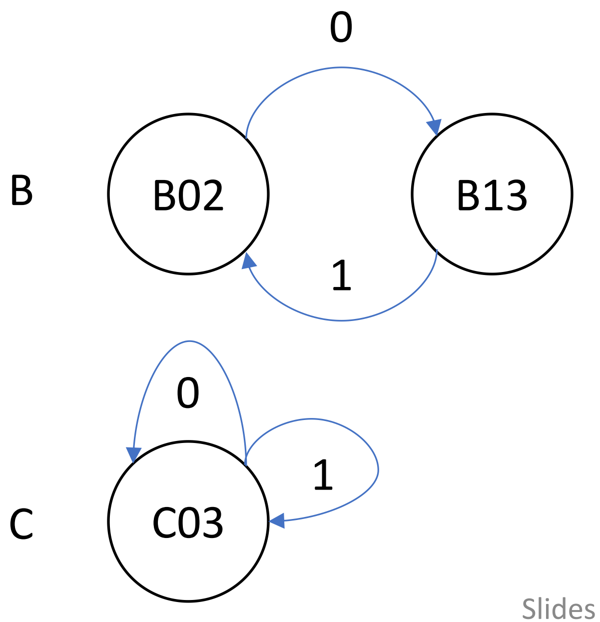
Definition. A **relation** $R \subseteq Q_1 \times Q_2$ is a forward simulation relation from $\mathcal{A}_1$ to $\mathcal{A}_2$ if

1. For every $q_1 \in \Theta_1$ there exists a $q_2 \in \Theta_2$ such that $q_1 R q_2$
2. For every transition $q_1 \rightarrow_1^{a_1} q_1'$ and $q_1 R q_2$ there exists q_2', a_2 such that
   - $q_2 \rightarrow_2^{a_2} q_2'$
   - $q_1'\ R\ q_2'$
   - $Trace(q_1, a_1, q_1') = Trace(q_2, a_2, q_2{}')$

Theorem. If there exists a forward simulation from $\mathcal{A}_1$ to $\mathcal{A}_2$ then $Traces_{A1} \subseteq Traces_{A2}$.

Finite state examples





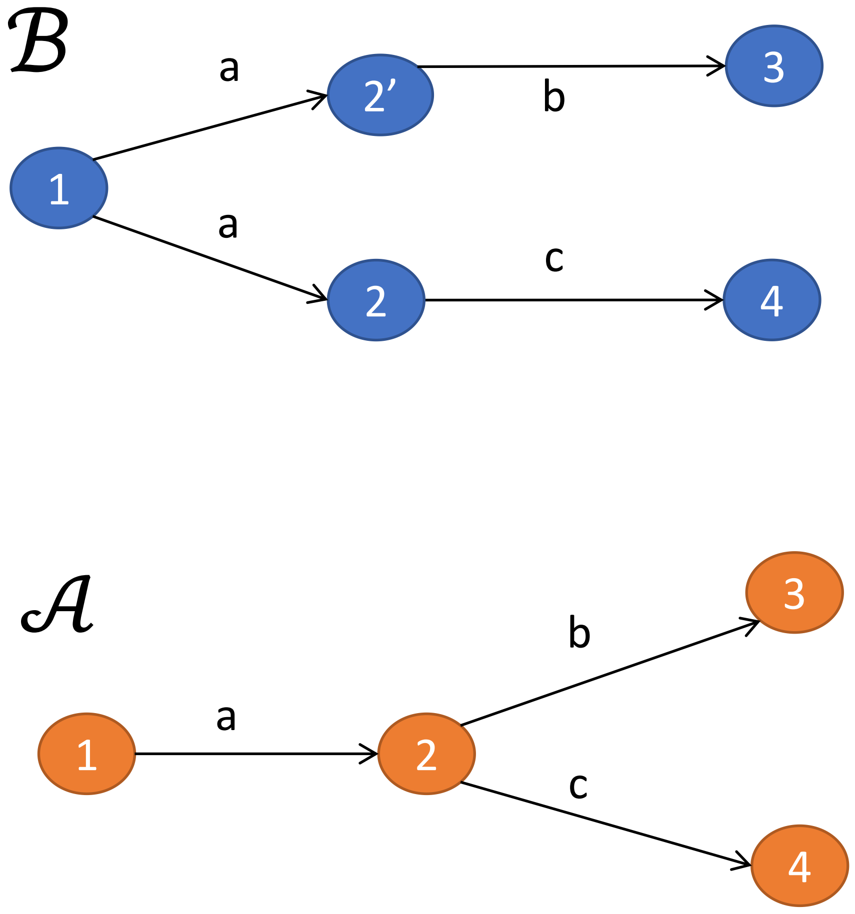
Check that A also simulates B
and that C simulates both A and B.

Therefore, $Traces_A = Traces_B \subseteq Traces_C$?

Does A simulate C?

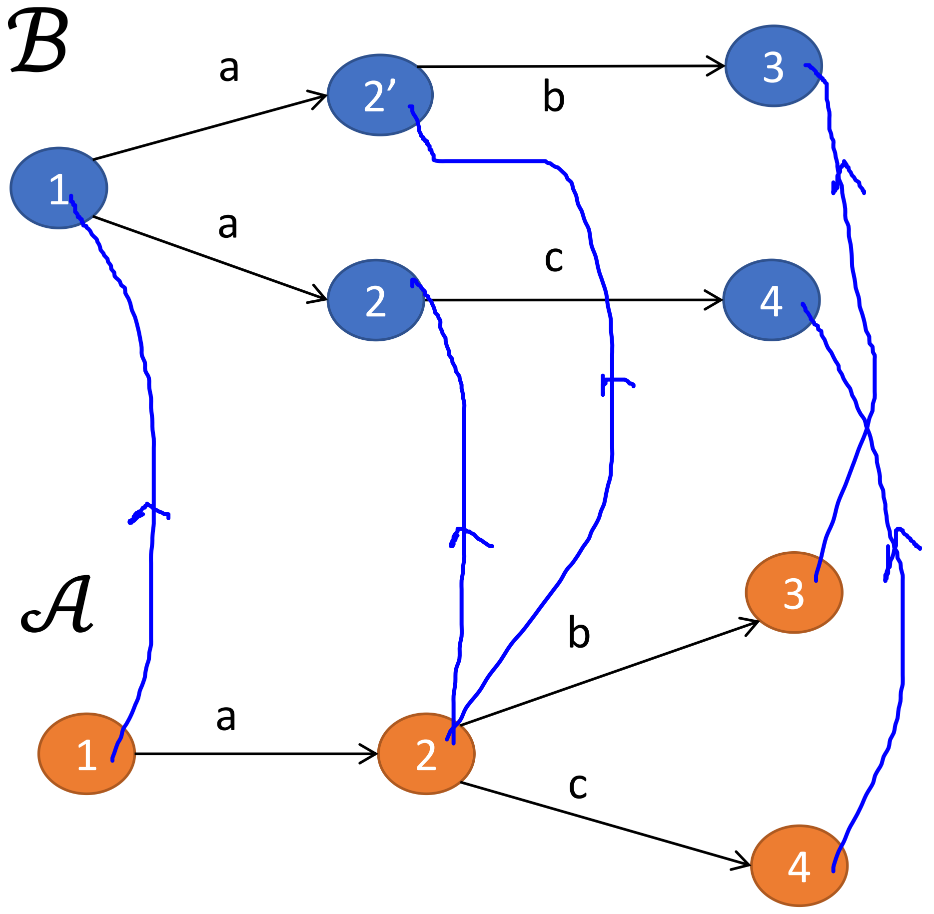
A Simulation Example

- Is there a forward simulation from $\mathcal{A}$ to $\mathcal{B}$?



A Simulation Example

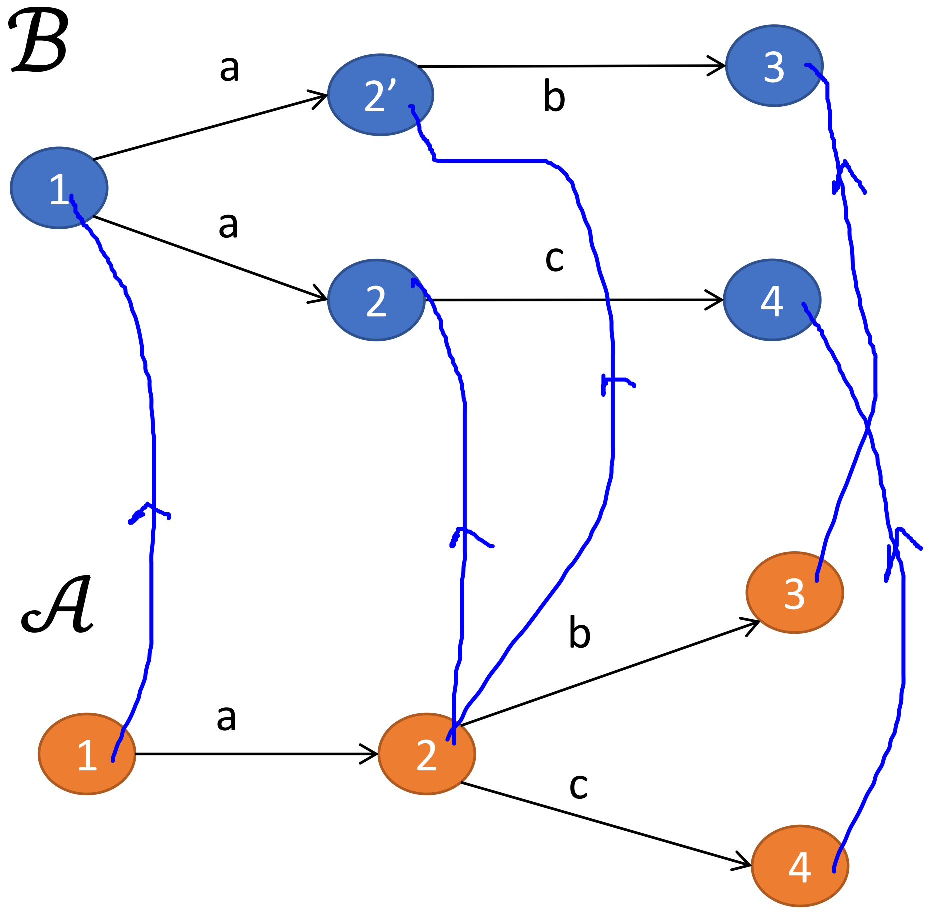
- Is there a forward simulation from $\mathcal{A}$ to $\mathcal{B}$?
- Consider the forward simulation relation



A Simulation Example

- Is there a forward simulation from $\mathcal{A}$ to $\mathcal{B}$?
- Consider the forward simulation relation

$\mathcal{A}$: $2 \rightarrow_c 4$ cannot be simulated by $\mathcal{B}$ from 2' although (2,2') are related.



Simulations for hybrid systems

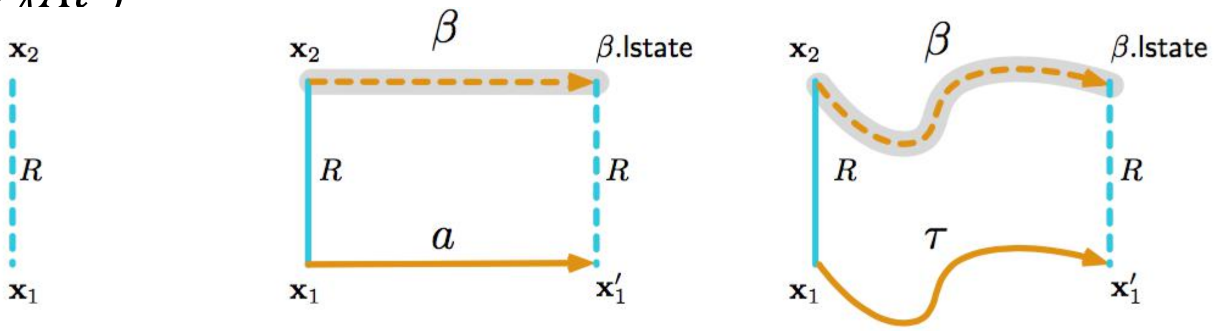
Forward simulation relation from $\mathcal{A}_1$ **to** $\mathcal{A}_2$ is a relation R $\subseteq val(X_1) \times val(X_2)$ such that

1. For every $\mathbf{x_1} \in \Theta_1$ there exists $\mathbf{x_2} \in \Theta_2$ such that $\mathbf{x_1}$ R $\mathbf{x_2}$
2. For every $\mathbf{x_1} \rightarrow_{\boldsymbol{a_1}} \mathbf{x_1}' \in \mathcal{D}$ and $\mathbf{x_2}$ such that $\mathbf{x_1}$ R $\mathbf{x_2}$, there exists $\mathbf{x_2}'$ such that
   - $\mathbf{x_2} \rightarrow_{\boldsymbol{a_1}} \mathbf{x_2}'$ and
   - $\mathbf{x_1}'$ R $\mathbf{x_2}'$
3. For every $\boldsymbol{\tau_1} \in \mathcal{T}_1$ and $\mathbf{x_2}$ such that $\tau_1.fstate$ R $\mathbf{x_2}$, there exists $\tau_2 \in \mathcal{T}_2$ that
   - $\mathbf{x_2} = \tau_2.fstate$ and
   - $\mathbf{x_1}'$ R $\tau_2.lstate$
   - $\tau_2.\mathrm{dom} = \tau_1.dom$

Theorem. If there exists a forward simulation relation from hybrid automaton $\mathcal{A}_1$ to $\mathcal{A}_2$ then for every execution of $\mathcal{A}_1$ there exists a corresponding execution of $\mathcal{A}_2$.

Simulation relations for hybrid automata

- Recall condition 3 in definition of simulation relation: $Trace(Bj \rightarrow_B Bj') = Trace(Ai \rightarrow_A Ai')$



- Hybrid automata have transitions and trajectories
- Different types of simulation depending on different notions for "Trace"
  - Match for all variable values, action names, and time duration of trajectories (abstraction)
  - Match variables but not time (time abstract simulation)
  - Match a subset (external) of variables and actions (trace inclusion)
  - Match single action/trajectory of A with a sequence of actions and trajectories of B

Timer simulates Ball (w.r.t. timing of bounce actions)

Automaton Ball(c,v_0,g)
variables:
x: Reals := 0
v: Reals := v_0
actions: bounce
transitions:
bounce
pre $x = 0 \wedge v < 0$
eff v := -cv
trajectories:
evolve d(x) = v; d(v) = -g
invariant $x \geq 0$

Automaton Timer(c, v_0, g)
variables: analog
timer: Reals := $2v_0/g$,
n:Naturals=0;
actions: bounce
transitions:
bounce
pre *timer = 0*
eff n:=n+1; timer := $\frac{2v_0}{gc^n}$
trajectories:
evolve d(timer) = -1
invariant timer ≥ 0

Some nice properties of Forward Simulation

Let $\mathcal{A}$, $\mathcal{B}$, and $\mathcal{C}$ be **comparable** (identical I/O varaibles and actions) HA.

If R_1 is a forward simulation from $\mathcal{A}$ to $\mathcal{B}$ and R_2 is a forward simulation from $\mathcal{B}$ to $\mathcal{C}$, then $R_1 \circ R_2$ is a forward simulation from $\mathcal{A}$ to $\mathcal{C}$

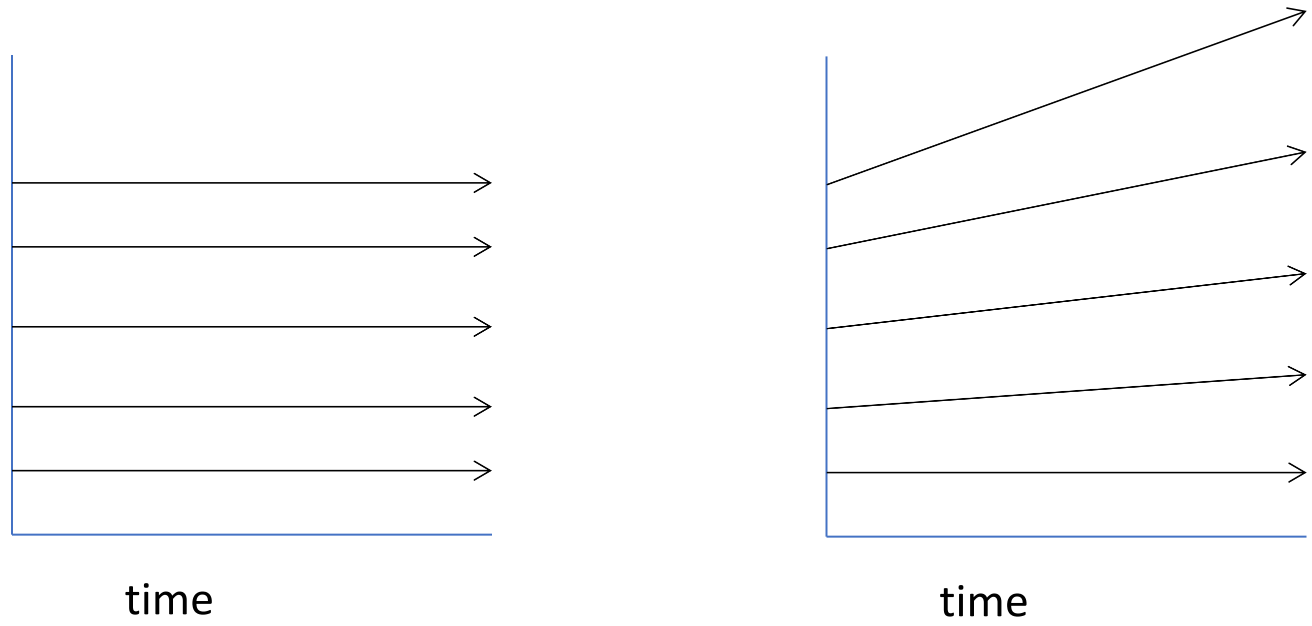
If R is a forward simulation from $\mathcal{A}$ to $\mathcal{B}$ and R^{-1} is a forward simulation from $\mathcal{B}$ to $\mathcal{A}$ then R is called a **bisimulation** and $\mathcal{B}$ are $\mathcal{A}$ **bisimilar**

Bisimilarity is an **equivalence relation**

(reflexive, transitive, and symmetric)

Remark on Simulations and Stability

Stability not preserved by ordinary simulations and bisimulations
[Prabhakar, et. al 15]



Stability Preserving Simulations and Bisimulations for Hybrid Systems, Prabhakar, Dullerud, Viswanathan IEEE Trans. Automatic Control 2015

Backward Simulations

Backward simulation relation from $\mathcal{A}_1$ **to** $\mathcal{A}_2$ is a relation R $\subseteq$ $Q_1 \times Q_2$ such that

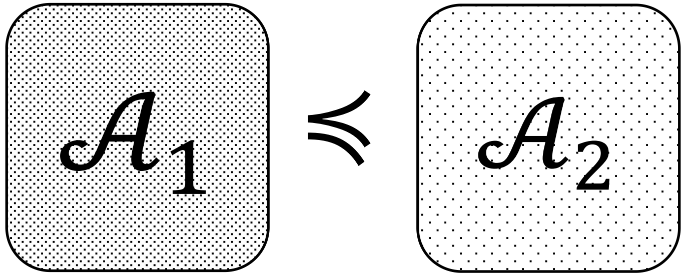
1. If $\mathbf{x_1} \in \Theta_1$ and $\mathbf{x_1}$ R $\mathbf{x_2}$ then $\mathbf{x_2} \in \Theta_2$ such that
2. If $\mathbf{x'_1}$ R $\mathbf{x'_2}$ and x_1—**a**→ x_1' then there exists an execution fragment $\boldsymbol{\beta}$
   - $\mathbf{x_2}$–$\boldsymbol{\beta}$→ $\mathbf{x_2'}$ and
   - $\mathbf{x_1}$ R $\mathbf{x_2}$
   - **Trace(**$\boldsymbol{\beta}$) = **a**
3. For every $\boldsymbol{\tau} \in \mathcal{T}$ and $\mathbf{x_2} \in Q_2$ such that $\mathbf{x_1'}$ R $\mathbf{x_2'}$, there exists $\mathbf{x_2}$ such that
   - $\mathbf{x_2}$–$\boldsymbol{\beta}$→ $\mathbf{x_2'}$ and
   - $\mathbf{x_1}$ R $\mathbf{x_2}$
   - **Trace(**$\boldsymbol{\beta}$) = $\boldsymbol{\tau}$

Theorem. If there exists a backward simulation relation from $\mathcal{A}_1$ to $\mathcal{A}_2$ then $\text{ClosedTraces}_1 \subseteq \text{ClosedTraces}_2$

"Closed" means: Finite execution with final trajectory with closed domain $\tau_0\ a_1\ \tau_1 a_2 \tau_2 \ldots \tau_k$ and $\tau_k.dom = [0, T]$

Abstraction recap

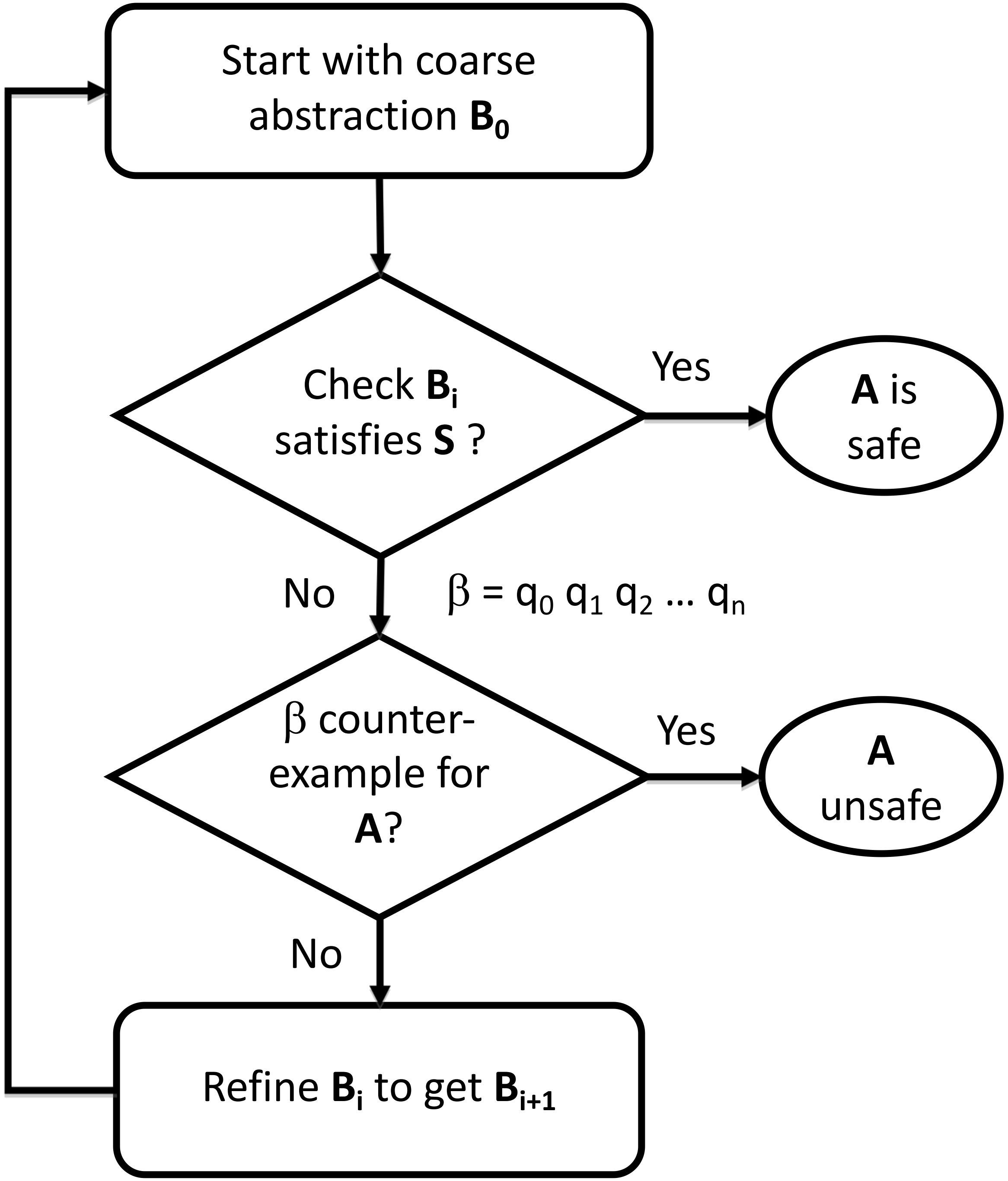
- Defined what it means for $\mathcal{A}_2$ to be abstraction of$\mathcal{A}_1$
- $Traces_{\mathcal{A}_1} \subseteq Traces_{\mathcal{A}_2}$
- $\mathcal{A}_1 \preccurlyeq_T \mathcal{A}_2$
- If $\mathcal{A}_1 \preccurlyeq_T \mathcal{A}_2$ and $\mathcal{A}_2 \preccurlyeq_T \mathcal{A}_1$ then $\mathcal{A}_1 \preccurlyeq_T \mathcal{A}_3$
- Transitive, $\preccurlyeq_T$ defines a preordering on compatible automata
- We saw methods for proving $\mathcal{A}_1 \preccurlyeq_T \mathcal{A}_2$
  - *Forward simulation and backward simulation*
- $\preccurlyeq_T$ defines a preorder



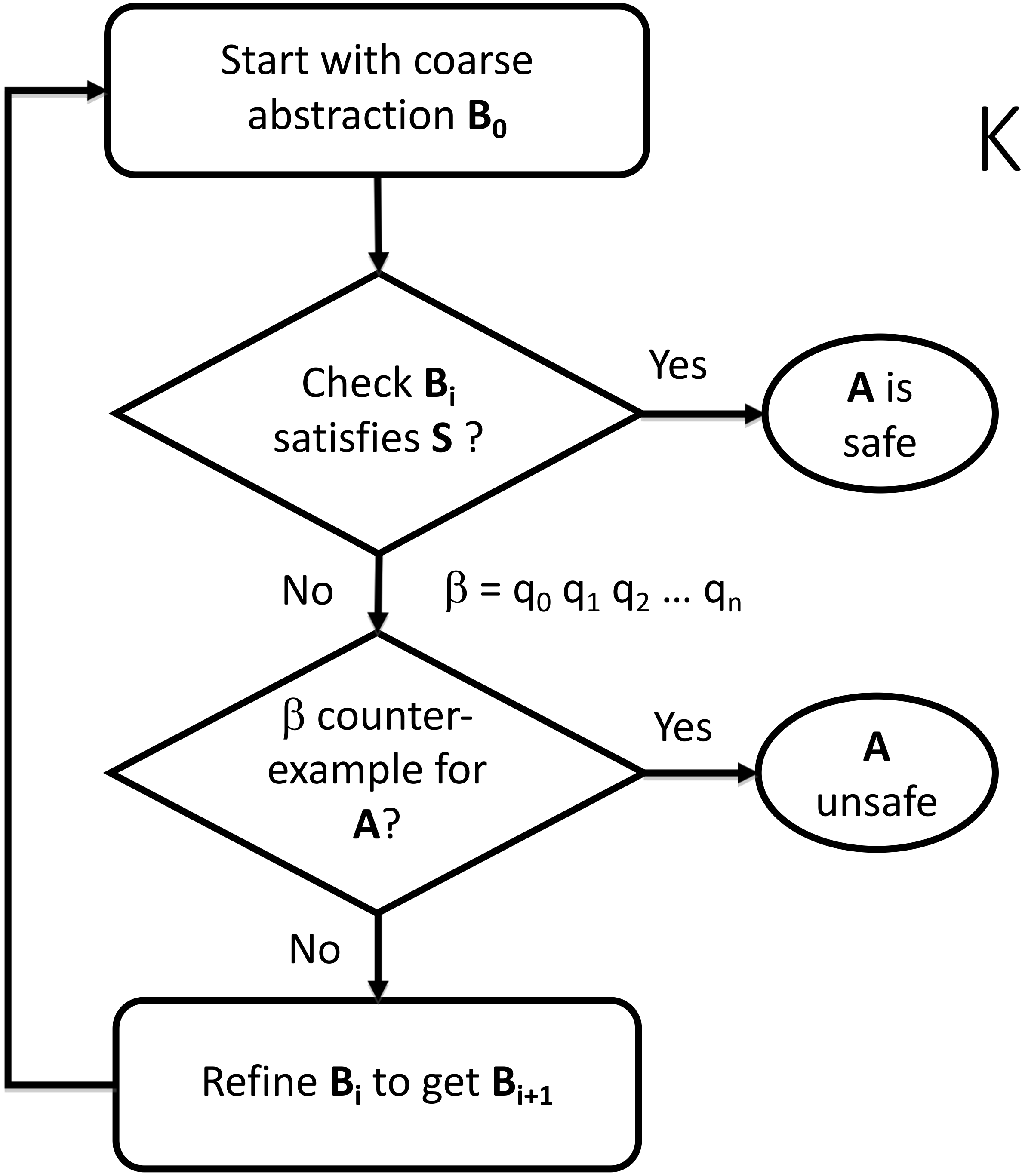
Counter-example guided abstraction-refinement

Counterexample guided abstraction refinement (CEGAR)

- A general algorithmic framework for automatically constructing and verifying property-specific abstractions [Clarke:2000]
- CEGAR has been applied to discrete automata, software, and hybrid systems [Holzman 00,Ball 01, Alur 2006,Clarke 2003, Fehnker2005, Prabhakar 15, Roohi 17]
- We will discuss the basic idea of the CEGAR and the key design choices, and their implications.

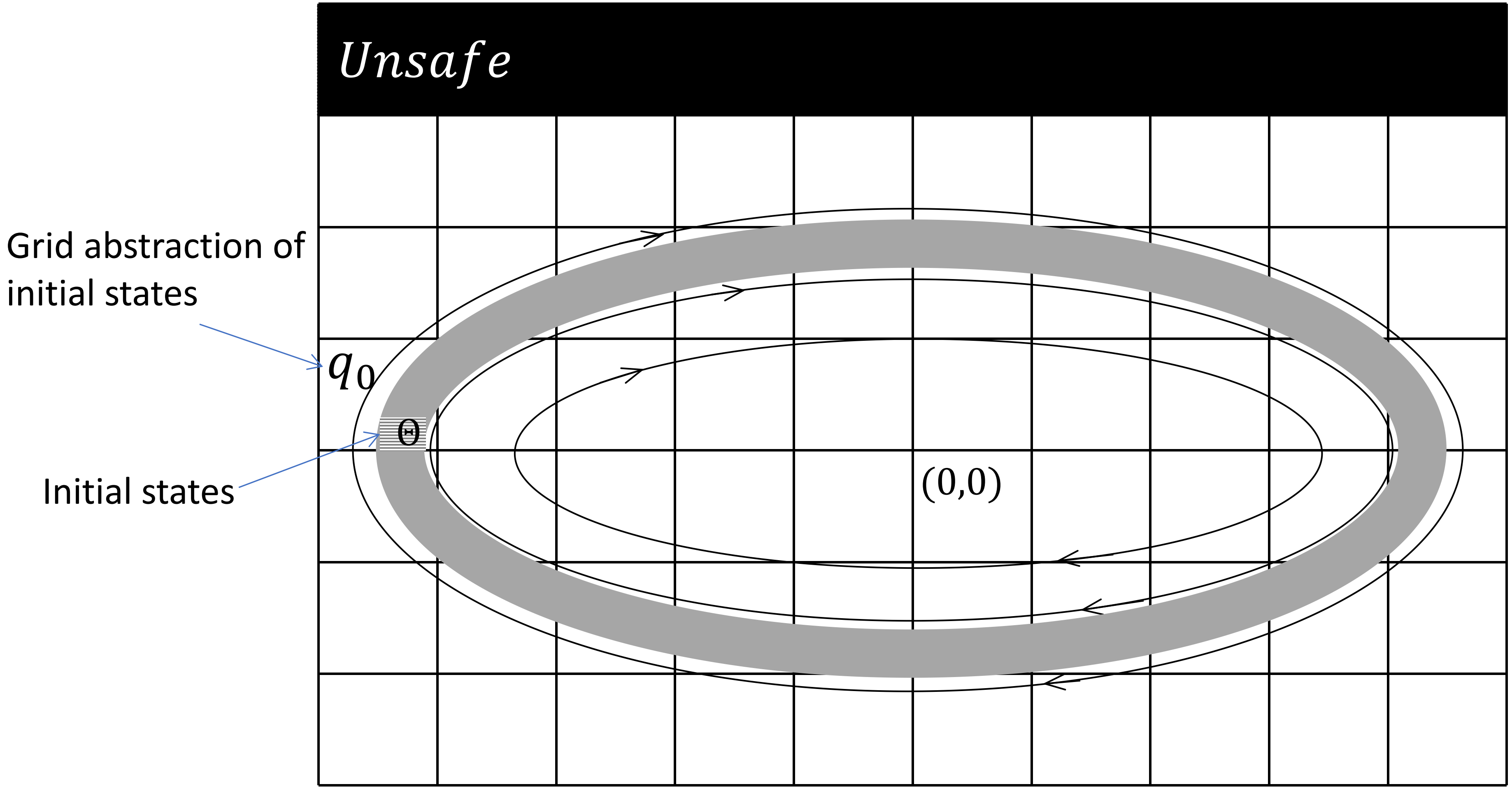


Idea of CEGAR



Key design choices

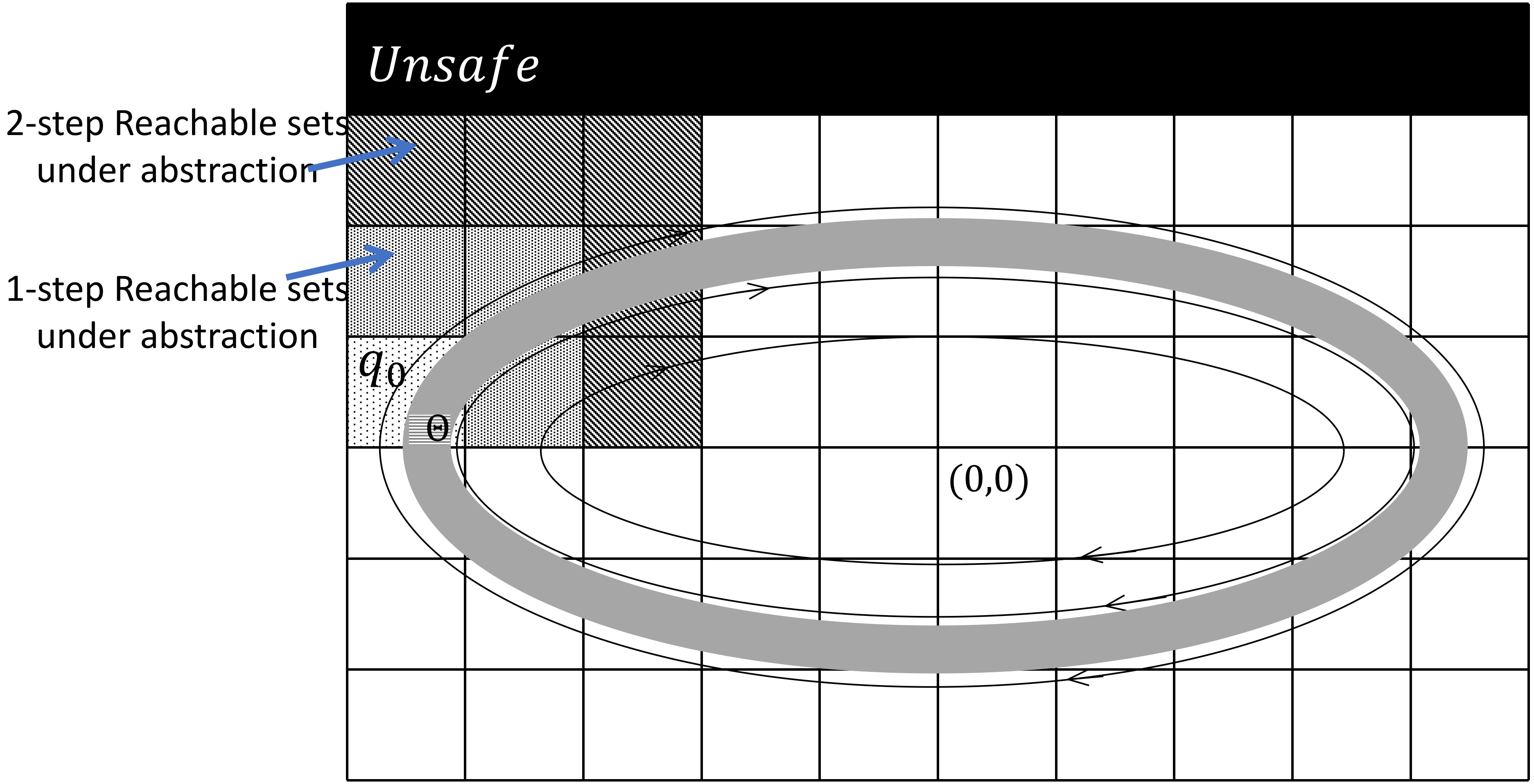
- Space of the abstract automata (finite, timed, linear)
- Model checker for abstract automaton (decidable?)
- Counter-example validation procedure
- Refinement strategy



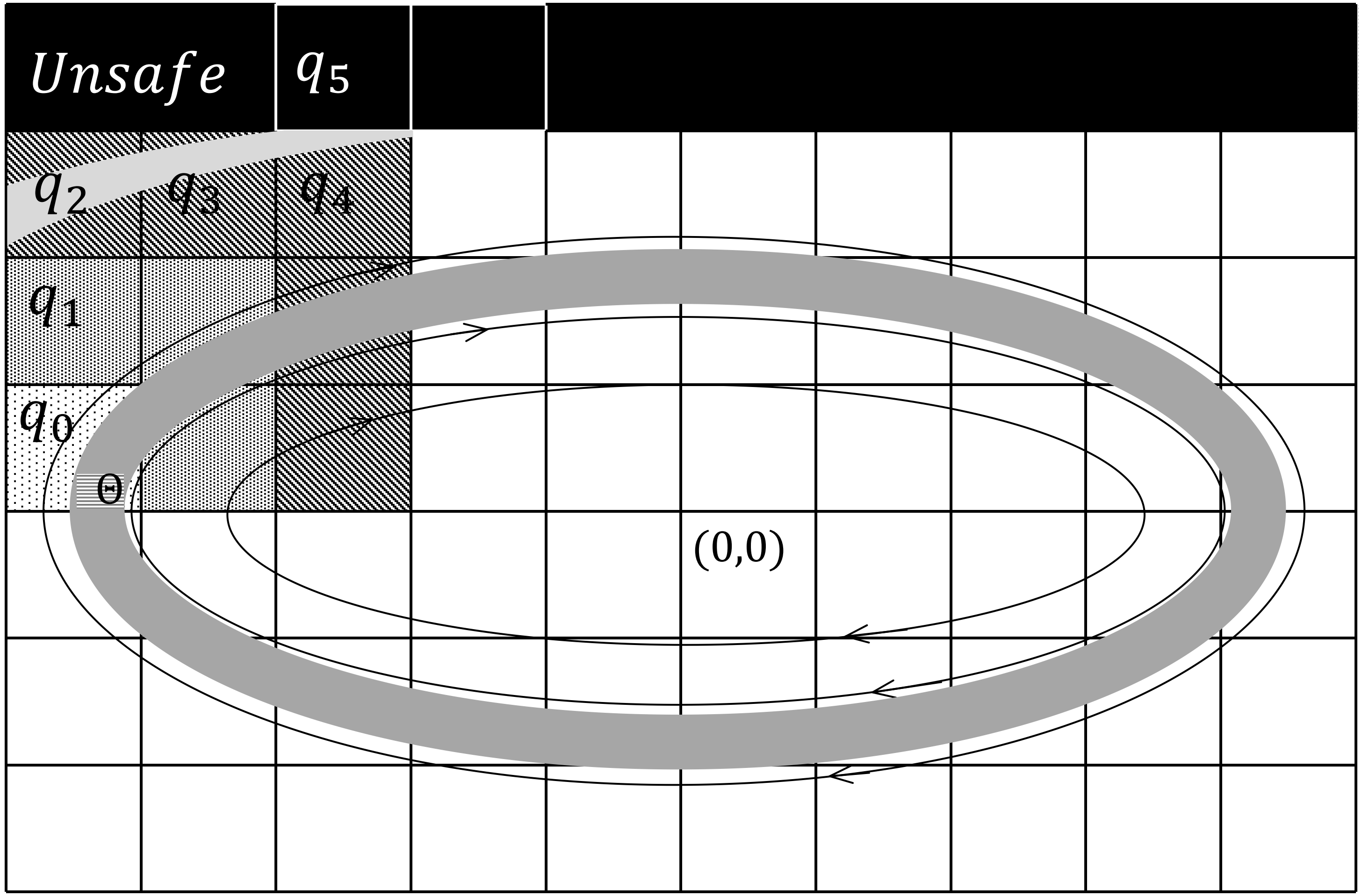
Example: dynamical systems with elliptical orbits

Abstraction: maps a box in state space to a discrete state q_i

Verification goal: will we reach unsafe regions on the top?


Unsafe
2-step Reachable sets under abstraction
1-step Reachable sets under abstraction
q_0
Θ
(0,0)

Counterexample with abstraction: q0 q1 q2 q3 q4 q5 (unsafe)



Is it a real counterexample?

Check using backward-reachability

$$S_5 = R^{-1}(q_5)$$

$$S_4 = Pre_A(S_5) \cap R^{-1}(q_4) \neq \emptyset$$

$$S_3 = Pre_A(S_4) \cap R^{-1}(q_3) \neq \emptyset$$

$$S_2 = Pre_A(S_3) \cap R^{-1}(q_2) \neq \emptyset$$

$$S_1 = Pre_A(S_2) \cap R^{-1}(q_1) = \emptyset$$

Impossible from q2 to q1!

$R^{-1}(q_i)$ is the box region in original dynamical system state space, corresponding to the discrete state q_i