

# BMC and Induction

*from refutation to verification*

Leonardo de Moura

(joint work with Harald Rueß and Maria Sorea)

`demoura@csl.sri.com`

Computer Science Laboratory

SRI International

Menlo Park, CA

# Overview

---

- ▶ Verifying Invariants
- ▶ Model Checking & Bounded Model Checking
- ▶ Decision Procedures
- ▶  $k$ -induction
- ▶ Compressing Paths
- ▶ Invariant Strengthening
- ▶ Conclusion & Future work

# Verifying Invariants

---

- ▶ Transition system  $M = (I, T)$
- ▶ **Invariants** characterize properties that are **true of all reachable states** in a system
- ▶ Invariants are supersets of the reachable states

# Verifying Invariants

---

- ▶ Transition system  $M = (I, T)$
- ▶ **Invariants** characterize properties that are **true of all reachable states** in a system
- ▶ Invariants are supersets of the reachable states

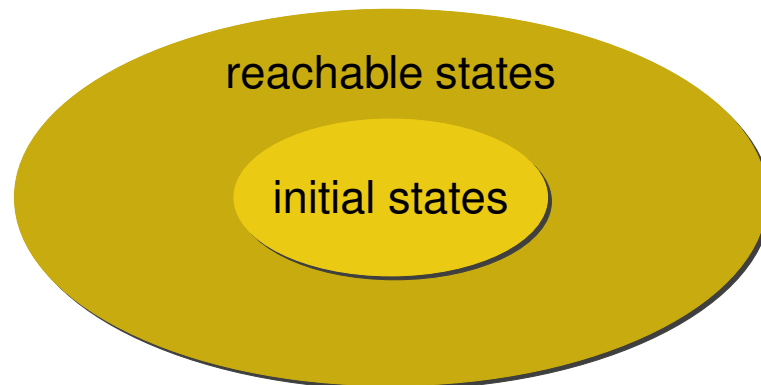


initial states

# Verifying Invariants

---

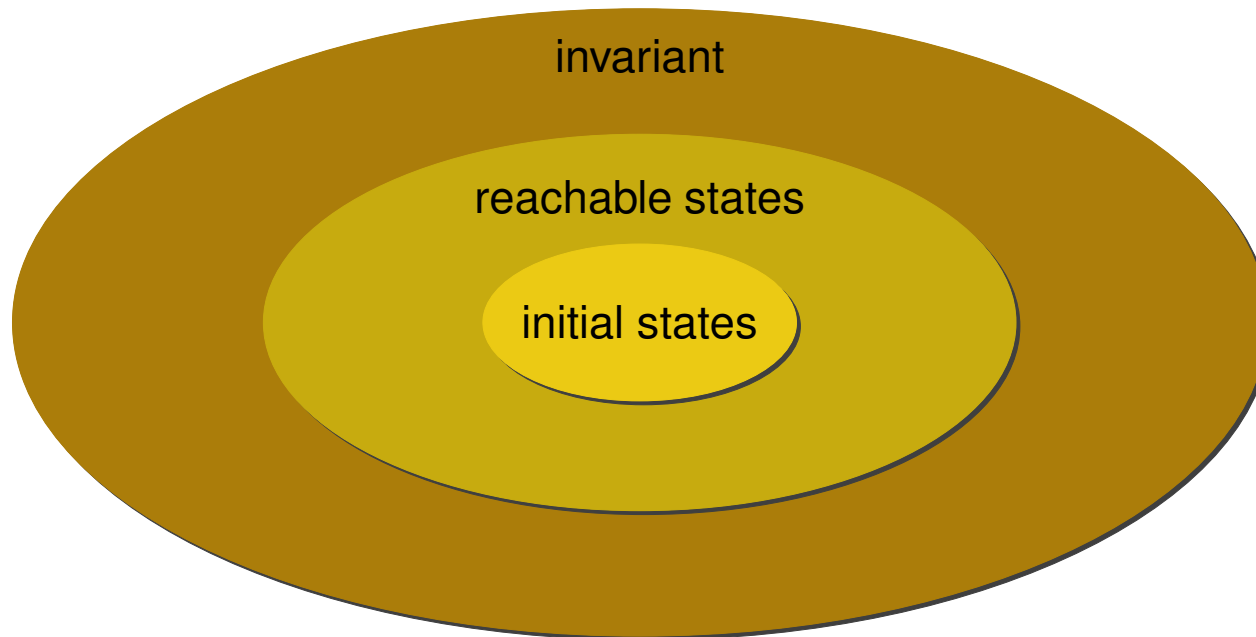
- ▶ Transition system  $M = (I, T)$
- ▶ **Invariants** characterize properties that are **true of all reachable states** in a system
- ▶ Invariants are supersets of the reachable states



# Verifying Invariants

---

- ▶ Transition system  $M = (I, T)$
- ▶ **Invariants** characterize properties that are **true of all reachable states** in a system
- ▶ Invariants are supersets of the reachable states



# Model Checkers

---

- ▶ Model checkers can **verify/refute** an invariant
- ▶ Only works when the number of **reachable states** is **finite** (and relatively small)
- ▶ Or when some kind of **abstraction** is used
- ▶ Examples:
  - ▶ **Explicit state**
  - ▶ **Symbolic** (based on BDDs)
  - ▶ **Bounded** (based on SAT)

# Bounded Model Checking (BMC)

---

- ▶ Is there a counterexample of length  $k$ ?
- ▶ BMC for finite systems by reduction to SAT [Clarke et al, 99]
- ▶ There is a counterexample (in the first  $k$  steps) for the invariant  $\varphi$  if the following formula is satisfiable:

$$I(s_1) \wedge T(s_1, s_2) \wedge \dots \wedge T(s_{k-1}, s_k) \wedge \neg\varphi(s_1) \vee \dots \vee \neg\varphi(s_k)$$

- ▶ There is no counterexample up to depth  $k$  for the invariant  $\varphi$  if the following formula is valid:

$$I(s_1) \wedge T(s_1, s_2) \wedge \dots \wedge T(s_{k-1}, s_k) \rightarrow \varphi(s_1) \wedge \dots \wedge \varphi(s_k)$$

# Bounded Model Checking (cont.)

---

- ▶ Complete method when the diameter of the system is known
  - ▶ It is usually too expensive to compute
  - ▶ Recurrence diameter can be exponentially longer than the diameter
- ▶ Example:
  - ▶ Real time scheduler (Priority Ceiling Protocol)
  - ▶ Property: deadlock free
  - ▶ Instance: 6 jobs and 3 semaphores
    - ▶ diameter: 491
    - ▶ number of reachable states:  $> 5$  billion
    - ▶ 310 BDD variables in the transition relation

# Infinite Bounded Model Checking

---

- ▶ Infinite domains such as: real numbers and data types
- ▶ Uses a combination of constraint solvers and decision procedures [dMRS02]
- ▶ ICS is a quantifier-free, first-order theory with uninterpreted function symbols and a rich combination of datatype theories including arithmetic, tuples, arrays, and bit-vectors
- ▶ Examples:
  - ▶  $x \geq 3 \wedge y = x + 1 \wedge (y \leq 3 \vee y > 3)$
  - ▶  $f(x) = x \wedge (f(f(x)) \neq f(x) \vee f(x) = g(x))$
- ▶ BMC for time automata [Sorea02]
- ▶ Diameter of the system can be infinite

# Verifying Invariants: Induction

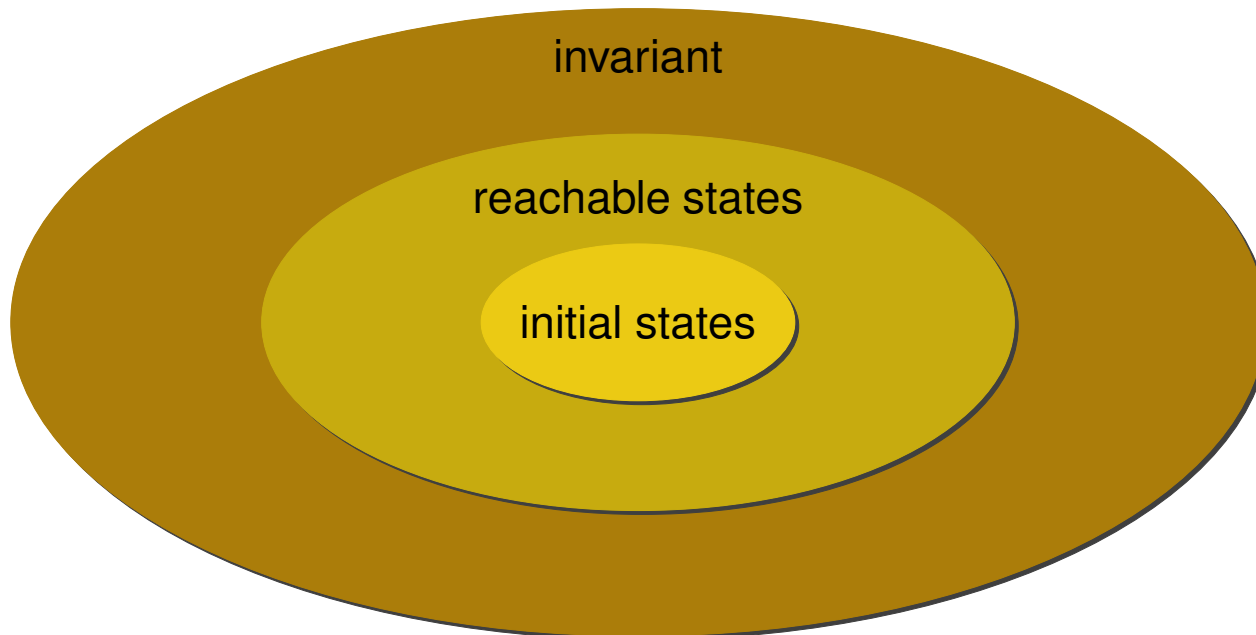
---

- ▶ The invariant  $\varphi$  is inductive if:
  - ▶  $I(s) \rightarrow \varphi(s)$  (base step)
  - ▶  $\varphi(s) \wedge T(s, s') \rightarrow \varphi(s')$  (inductive step)
- ▶ Property of the real time scheduler is proved in 0.09 secs
- ▶ However invariant are not usually inductive
  - ▶ The inductive step is violated

# Verifying Invariants: Induction

---

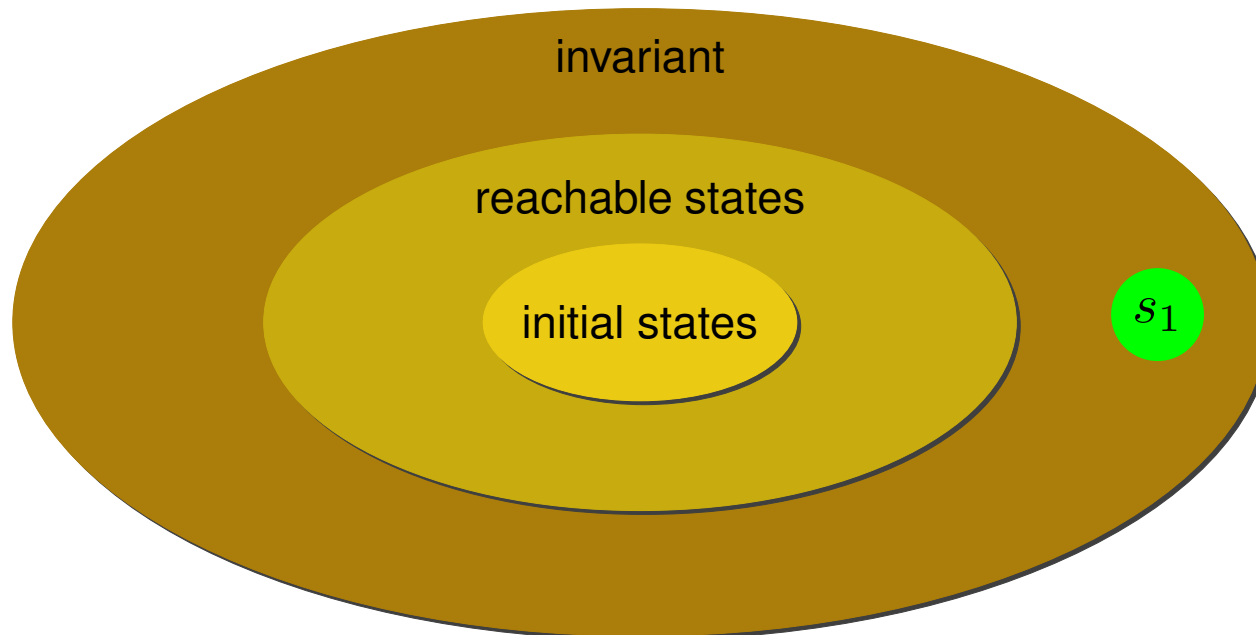
- ▶ The invariant  $\varphi$  is inductive if:
  - ▶  $I(s) \rightarrow \varphi(s)$  (base step)
  - ▶  $\varphi(s) \wedge T(s, s') \rightarrow \varphi(s')$  (inductive step)
- ▶ Property of the real time scheduler is proved in 0.09 secs
- ▶ However invariant are not usually inductive
  - ▶ The inductive step is violated



# Verifying Invariants: Induction

---

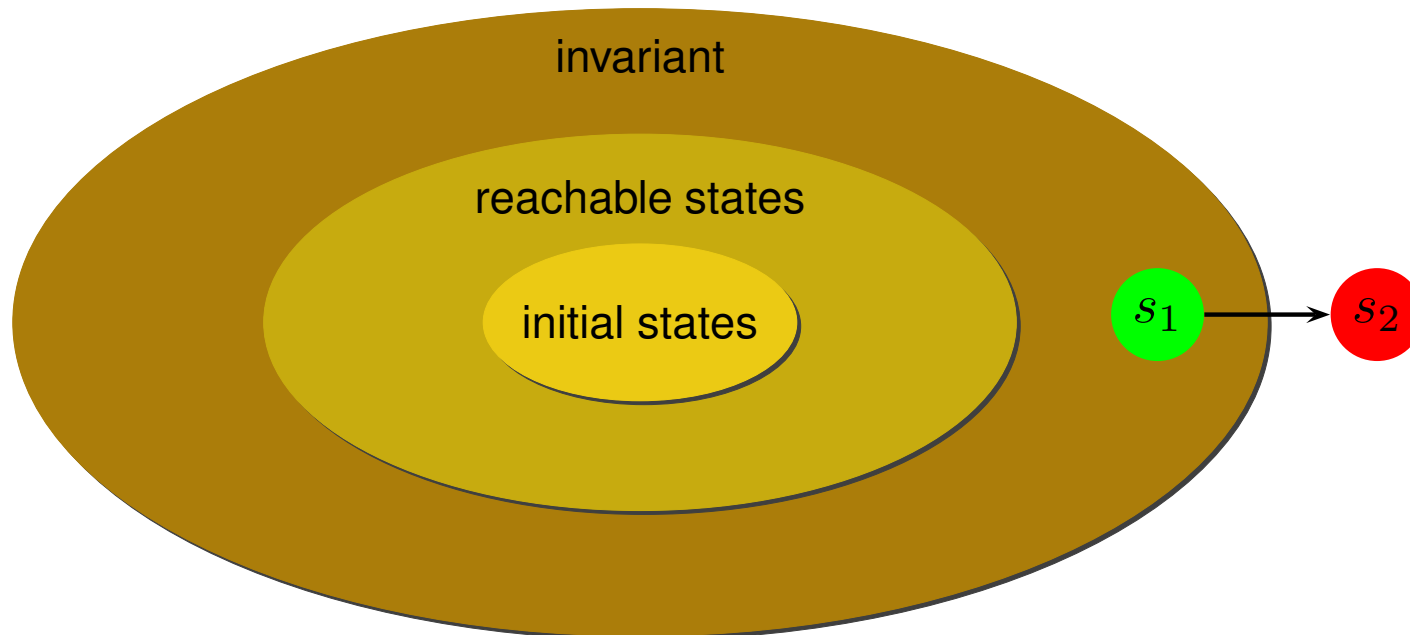
- ▶ The invariant  $\varphi$  is inductive if:
  - ▶  $I(s) \rightarrow \varphi(s)$  (base step)
  - ▶  $\varphi(s) \wedge T(s, s') \rightarrow \varphi(s')$  (inductive step)
- ▶ Property of the real time scheduler is proved in 0.09 secs
- ▶ However invariant are not usually inductive
  - ▶ The inductive step is violated



# Verifying Invariants: Induction

---

- ▶ The invariant  $\varphi$  is inductive if:
  - ▶  $I(s) \rightarrow \varphi(s)$  (base step)
  - ▶  $\varphi(s) \wedge T(s, s') \rightarrow \varphi(s')$  (inductive step)
- ▶ Property of the real time scheduler is proved in 0.09 secs
- ▶ However invariant are not usually inductive
  - ▶ The inductive step is violated



# *k-Induction*

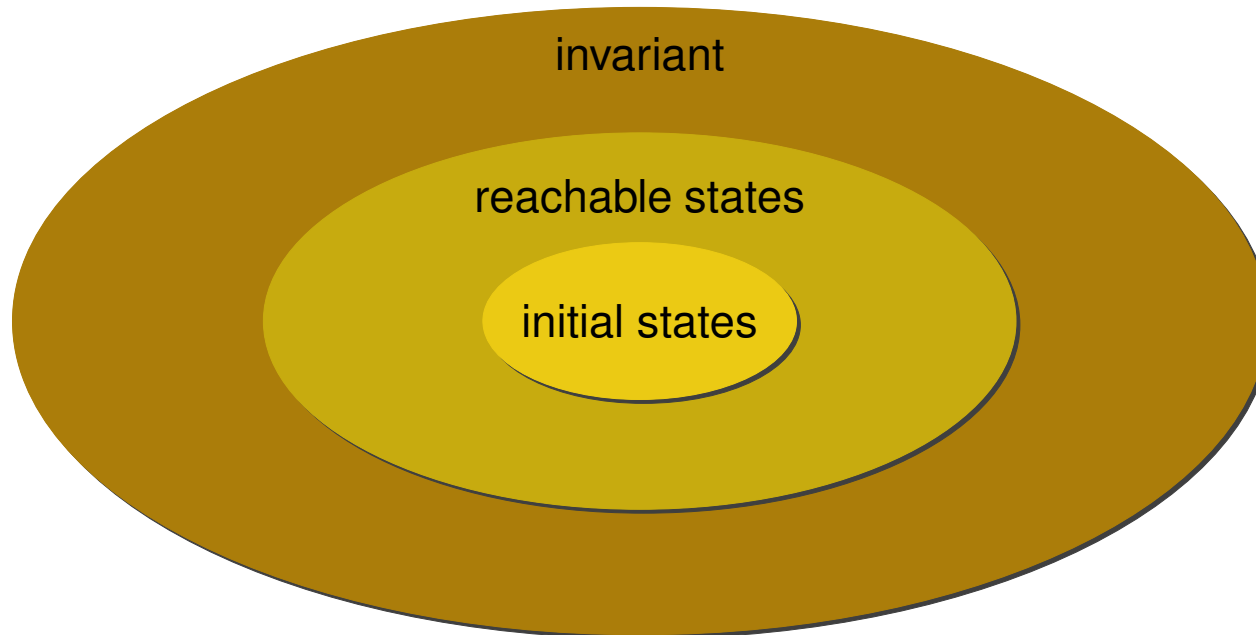
---

- ▶ The invariant  $\varphi$  is *k-inductive* [Sheeran00] if:
  - ▶  $I(s_1) \wedge T(s_1, s_2) \wedge \dots \wedge T(s_{k-1}, s_k) \rightarrow \varphi(s_1) \wedge \dots \wedge \varphi(s_k)$
  - ▶  $\varphi(s_1) \wedge \dots \wedge \varphi(s_k) \wedge T(s_1, s_2) \wedge \dots \wedge T(s_k, s_{k+1}) \rightarrow \varphi(s_{k+1})$
- ▶ It is **harder** to violate the inductive step
- ▶ The **base case** is BMC
- ▶ If  $\varphi$  is *k<sub>1</sub>-inductive* then it is also *k<sub>2</sub>-inductive* for  $k_2 \geq k_1$

# *k-Induction*

---

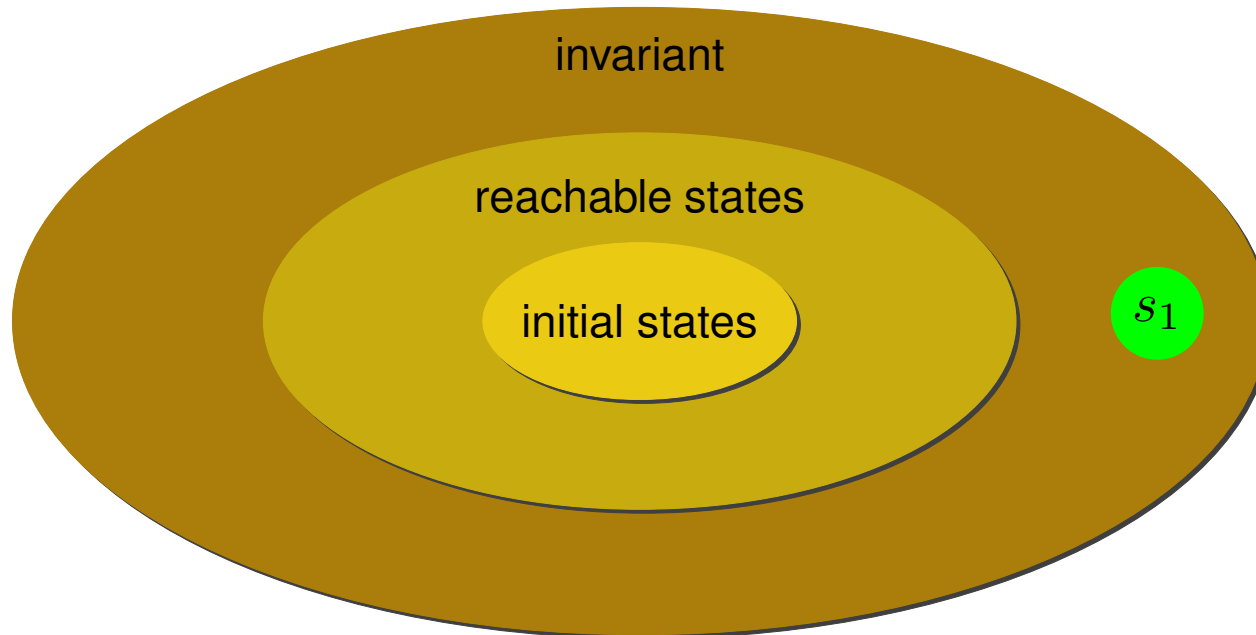
- ▶ The invariant  $\varphi$  is *k-inductive* [Sheeran00] if:
  - ▶  $I(s_1) \wedge T(s_1, s_2) \wedge \dots \wedge T(s_{k-1}, s_k) \rightarrow \varphi(s_1) \wedge \dots \wedge \varphi(s_k)$
  - ▶  $\varphi(s_1) \wedge \dots \wedge \varphi(s_k) \wedge T(s_1, s_2) \wedge \dots \wedge T(s_k, s_{k+1}) \rightarrow \varphi(s_{k+1})$
- ▶ It is harder to violate the inductive step
- ▶ The base case is BMC
- ▶ If  $\varphi$  is *k<sub>1</sub>-inductive* then it is also *k<sub>2</sub>-inductive* for  $k_2 \geq k_1$



# *k-Induction*

---

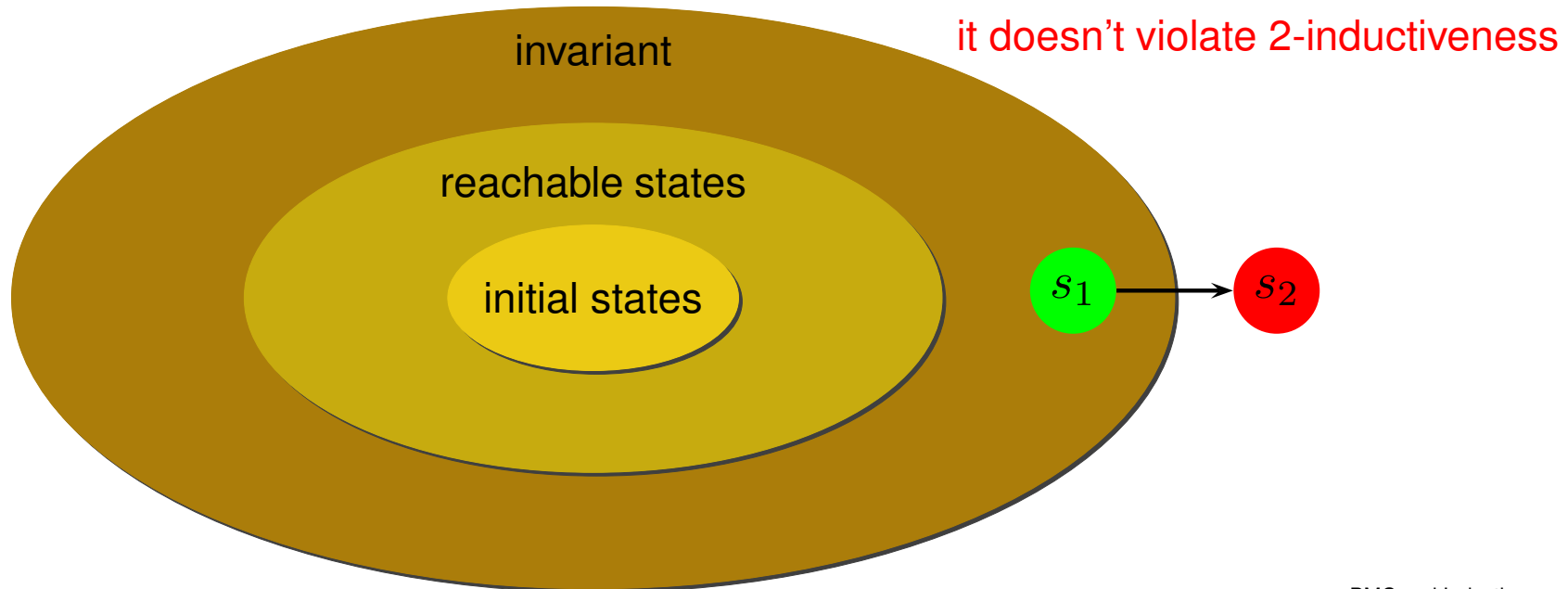
- ▶ The invariant  $\varphi$  is *k-inductive* [Sheeran00] if:
  - ▶  $I(s_1) \wedge T(s_1, s_2) \wedge \dots \wedge T(s_{k-1}, s_k) \rightarrow \varphi(s_1) \wedge \dots \wedge \varphi(s_k)$
  - ▶  $\varphi(s_1) \wedge \dots \wedge \varphi(s_k) \wedge T(s_1, s_2) \wedge \dots \wedge T(s_k, s_{k+1}) \rightarrow \varphi(s_{k+1})$
- ▶ It is harder to violate the inductive step
- ▶ The base case is BMC
- ▶ If  $\varphi$  is *k<sub>1</sub>-inductive* then it is also *k<sub>2</sub>-inductive* for  $k_2 \geq k_1$



# *k-Induction*

---

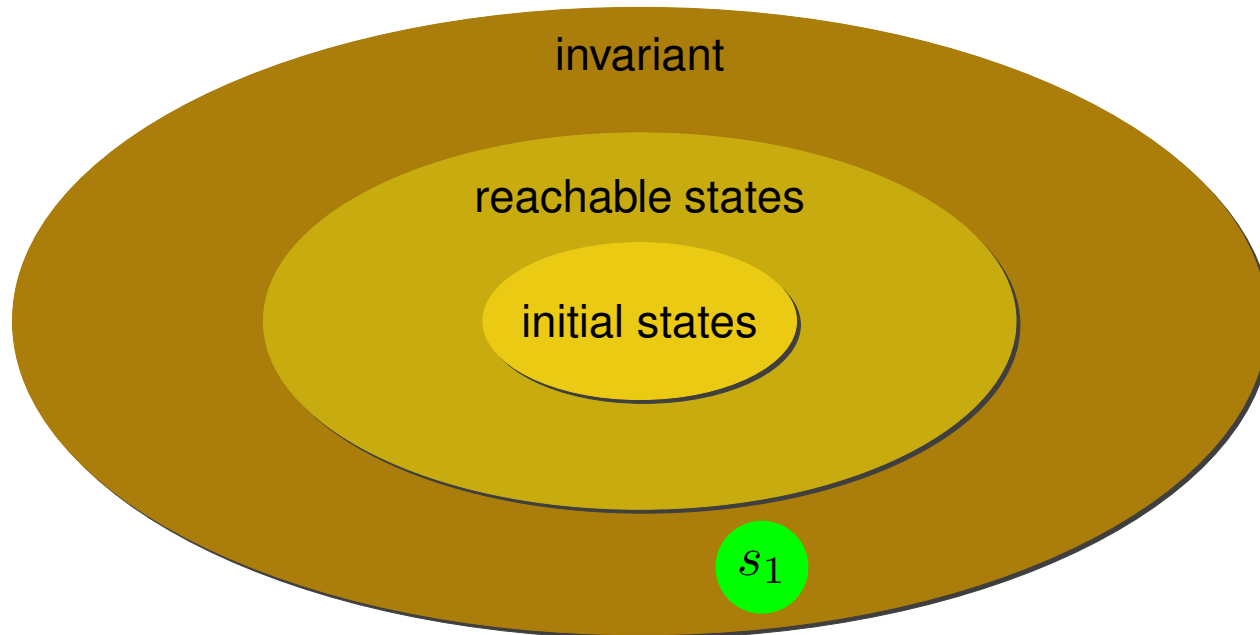
- ▶ The invariant  $\varphi$  is *k*-inductive [Sheeran00] if:
  - ▶  $I(s_1) \wedge T(s_1, s_2) \wedge \dots \wedge T(s_{k-1}, s_k) \rightarrow \varphi(s_1) \wedge \dots \wedge \varphi(s_k)$
  - ▶  $\varphi(s_1) \wedge \dots \wedge \varphi(s_k) \wedge T(s_1, s_2) \wedge \dots \wedge T(s_k, s_{k+1}) \rightarrow \varphi(s_{k+1})$
- ▶ It is harder to violate the inductive step
- ▶ The base case is BMC
- ▶ If  $\varphi$  is *k*<sub>1</sub>-inductive then it is also *k*<sub>2</sub>-inductive for  $k_2 \geq k_1$



# *k-Induction*

---

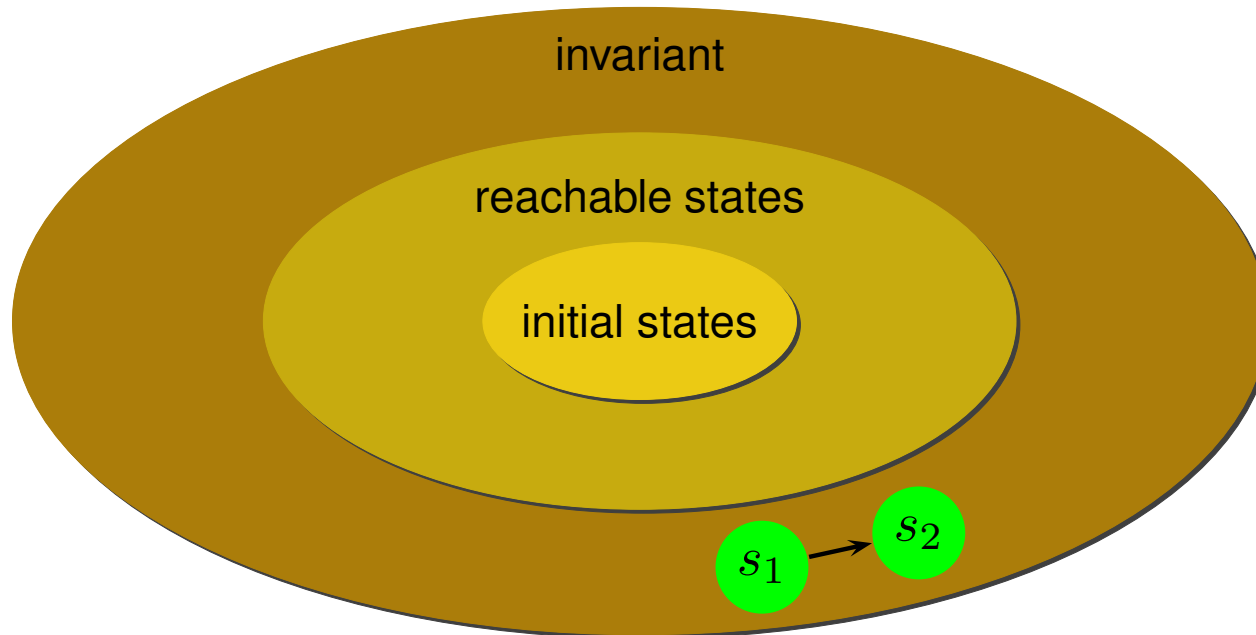
- ▶ The invariant  $\varphi$  is *k-inductive* [Sheeran00] if:
  - ▶  $I(s_1) \wedge T(s_1, s_2) \wedge \dots \wedge T(s_{k-1}, s_k) \rightarrow \varphi(s_1) \wedge \dots \wedge \varphi(s_k)$
  - ▶  $\varphi(s_1) \wedge \dots \wedge \varphi(s_k) \wedge T(s_1, s_2) \wedge \dots \wedge T(s_k, s_{k+1}) \rightarrow \varphi(s_{k+1})$
- ▶ It is harder to violate the inductive step
- ▶ The base case is BMC
- ▶ If  $\varphi$  is  $k_1$ -inductive then it is also  $k_2$ -inductive for  $k_2 \geq k_1$



# *k-Induction*

---

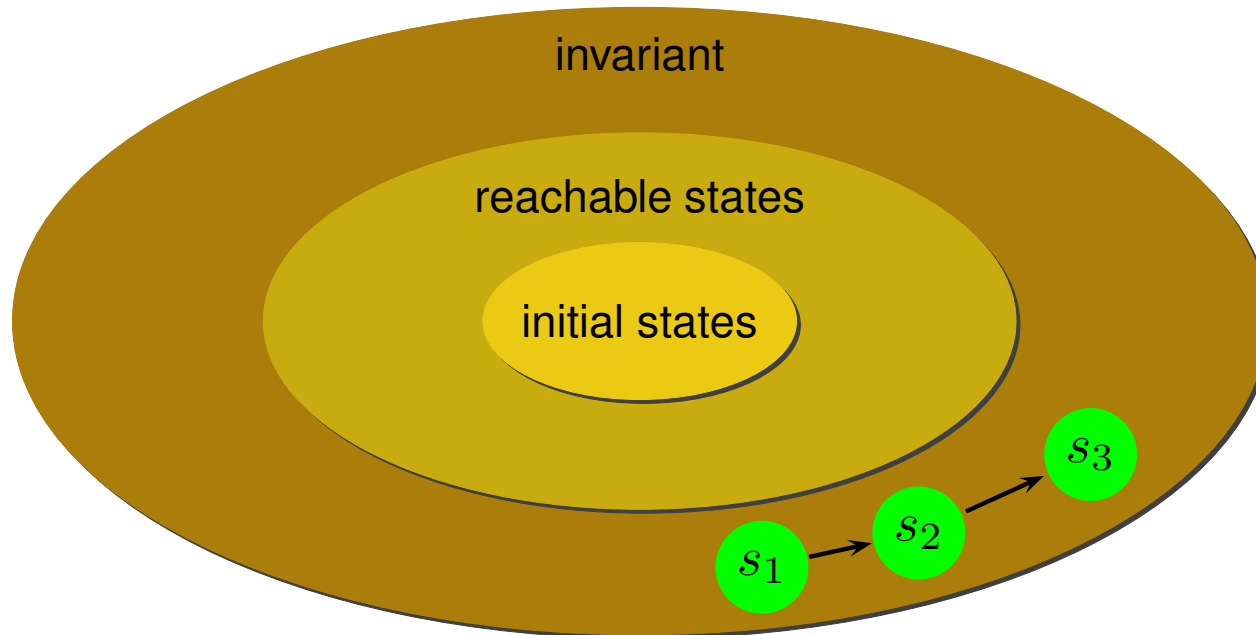
- ▶ The invariant  $\varphi$  is *k-inductive* [Sheeran00] if:
  - ▶  $I(s_1) \wedge T(s_1, s_2) \wedge \dots \wedge T(s_{k-1}, s_k) \rightarrow \varphi(s_1) \wedge \dots \wedge \varphi(s_k)$
  - ▶  $\varphi(s_1) \wedge \dots \wedge \varphi(s_k) \wedge T(s_1, s_2) \wedge \dots \wedge T(s_k, s_{k+1}) \rightarrow \varphi(s_{k+1})$
- ▶ It is harder to violate the inductive step
- ▶ The base case is BMC
- ▶ If  $\varphi$  is *k<sub>1</sub>-inductive* then it is also *k<sub>2</sub>-inductive* for  $k_2 \geq k_1$



# *k-Induction*

---

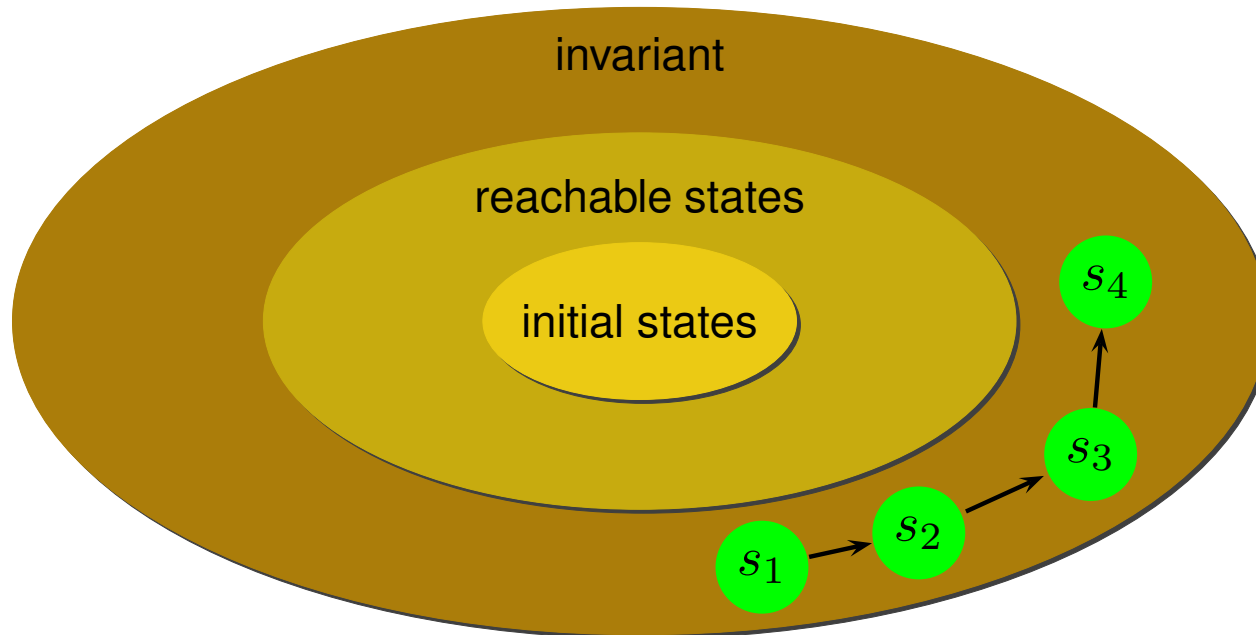
- ▶ The invariant  $\varphi$  is *k-inductive* [Sheeran00] if:
  - ▶  $I(s_1) \wedge T(s_1, s_2) \wedge \dots \wedge T(s_{k-1}, s_k) \rightarrow \varphi(s_1) \wedge \dots \wedge \varphi(s_k)$
  - ▶  $\varphi(s_1) \wedge \dots \wedge \varphi(s_k) \wedge T(s_1, s_2) \wedge \dots \wedge T(s_k, s_{k+1}) \rightarrow \varphi(s_{k+1})$
- ▶ It is harder to violate the inductive step
- ▶ The base case is BMC
- ▶ If  $\varphi$  is *k<sub>1</sub>-inductive* then it is also *k<sub>2</sub>-inductive* for  $k_2 \geq k_1$



# *k-Induction*

---

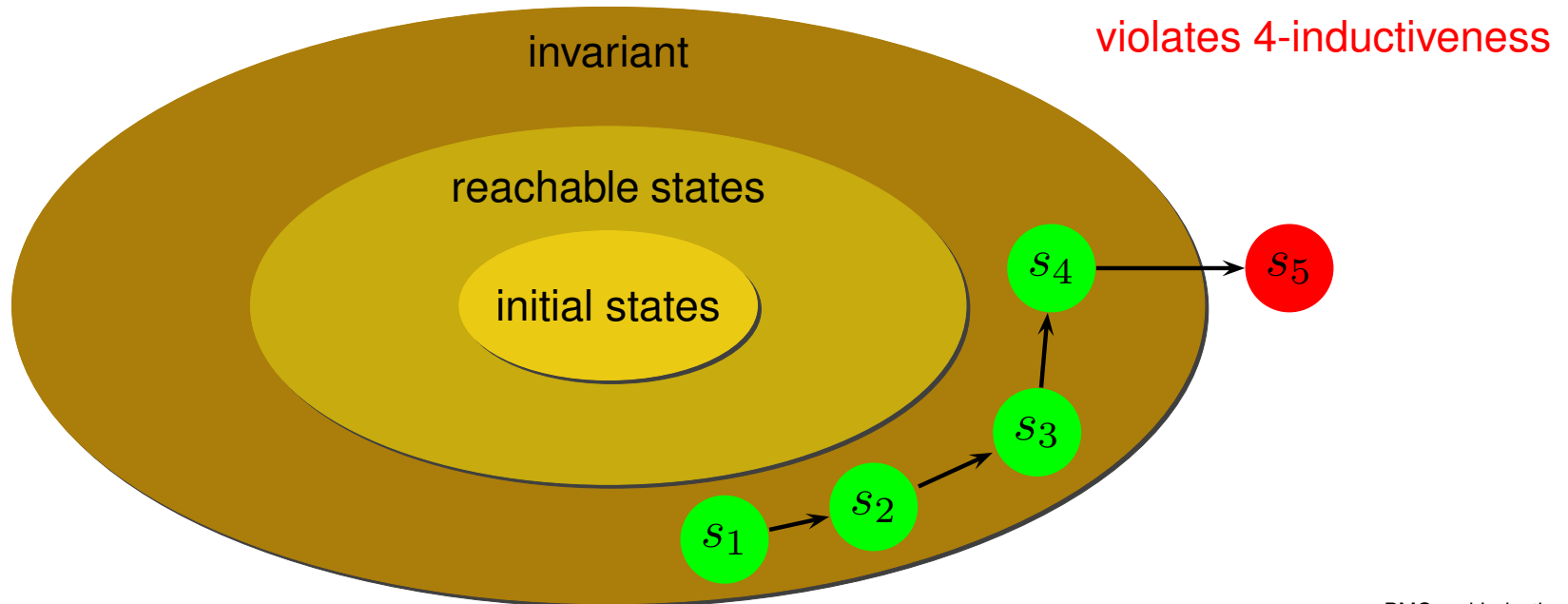
- ▶ The invariant  $\varphi$  is *k-inductive* [Sheeran00] if:
  - ▶  $I(s_1) \wedge T(s_1, s_2) \wedge \dots \wedge T(s_{k-1}, s_k) \rightarrow \varphi(s_1) \wedge \dots \wedge \varphi(s_k)$
  - ▶  $\varphi(s_1) \wedge \dots \wedge \varphi(s_k) \wedge T(s_1, s_2) \wedge \dots \wedge T(s_k, s_{k+1}) \rightarrow \varphi(s_{k+1})$
- ▶ It is harder to violate the inductive step
- ▶ The base case is BMC
- ▶ If  $\varphi$  is *k<sub>1</sub>-inductive* then it is also *k<sub>2</sub>-inductive* for  $k_2 \geq k_1$



# *k-Induction*

---

- ▶ The invariant  $\varphi$  is *k-inductive* [Sheeran00] if:
  - ▶  $I(s_1) \wedge T(s_1, s_2) \wedge \dots \wedge T(s_{k-1}, s_k) \rightarrow \varphi(s_1) \wedge \dots \wedge \varphi(s_k)$
  - ▶  $\varphi(s_1) \wedge \dots \wedge \varphi(s_k) \wedge T(s_1, s_2) \wedge \dots \wedge T(s_k, s_{k+1}) \rightarrow \varphi(s_{k+1})$
- ▶ It is harder to violate the inductive step
- ▶ The base case is BMC
- ▶ If  $\varphi$  is *k<sub>1</sub>-inductive* then it is also *k<sub>2</sub>-inductive* for  $k_2 \geq k_1$



## *k-Induction (cont.)*

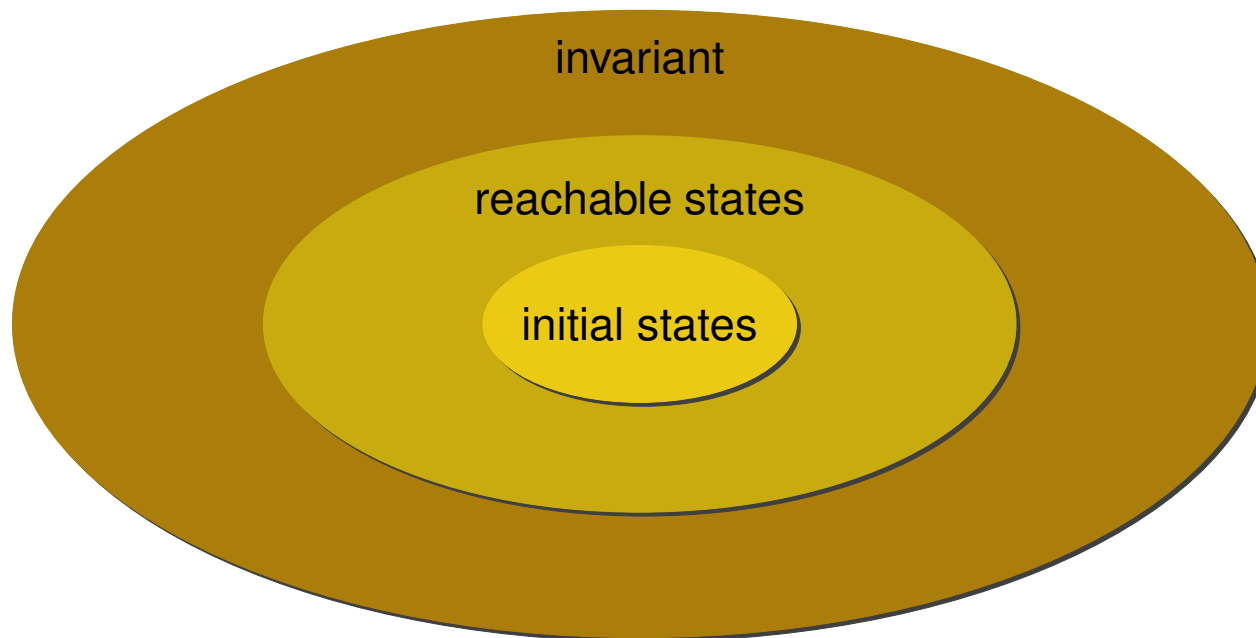
---

- ▶ Can be used to verify **finite** and **infinite** systems
- ▶ Not **complete** even for **finite** systems
- ▶ **Self loops** in **unreachable** states
  - ▶ Suppose the state  $s_1$  contains a self loop:

## *k-Induction (cont.)*

---

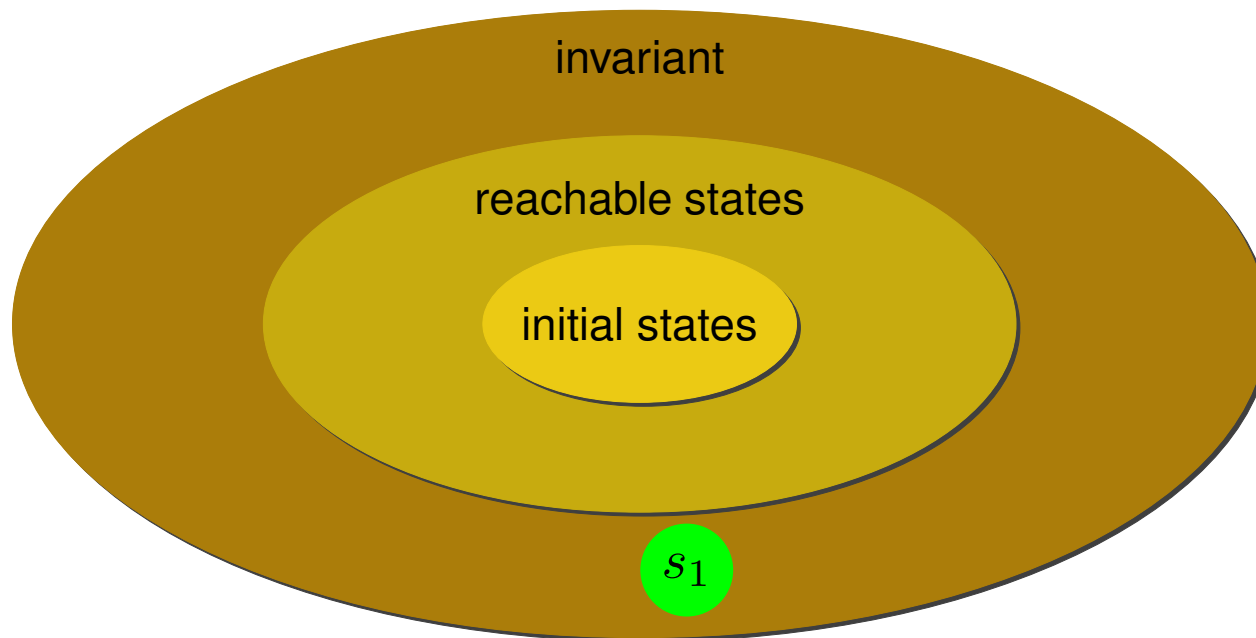
- ▶ Can be used to verify **finite** and **infinite** systems
- ▶ Not **complete** even for **finite** systems
- ▶ Self loops in **unreachable** states
  - ▶ Suppose the state  $s_1$  contains a self loop:



## *k-Induction (cont.)*

---

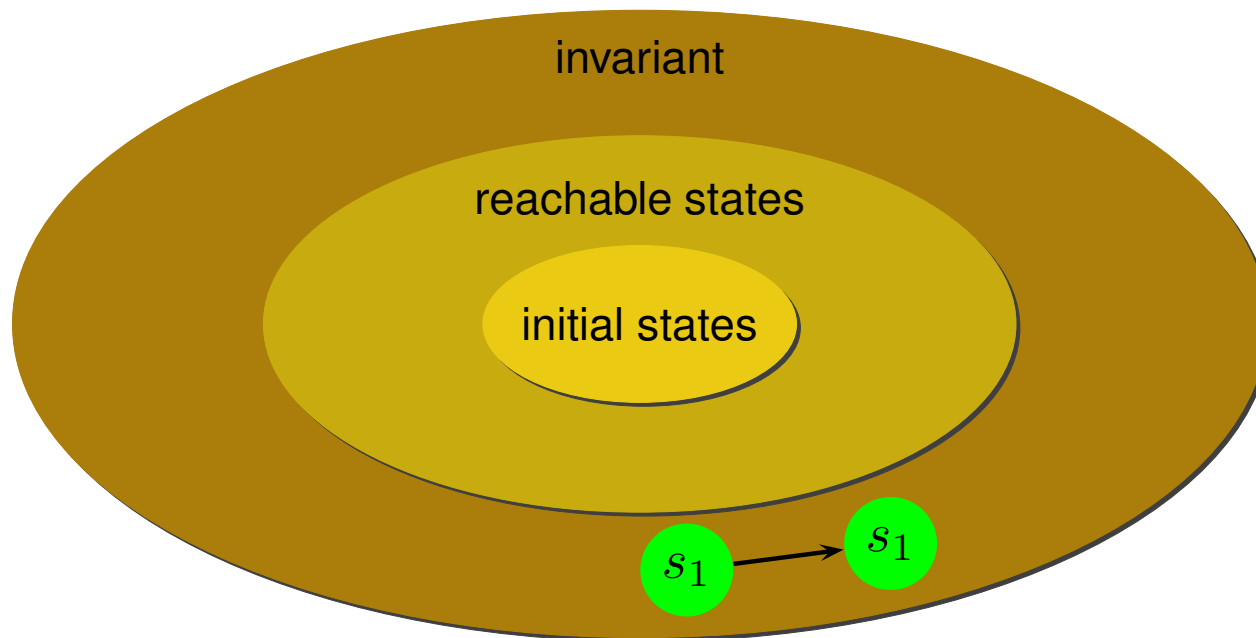
- ▶ Can be used to verify **finite** and **infinite** systems
- ▶ Not **complete** even for **finite** systems
- ▶ Self loops in **unreachable** states
  - ▶ Suppose the state  $s_1$  contains a self loop:



## *k-Induction (cont.)*

---

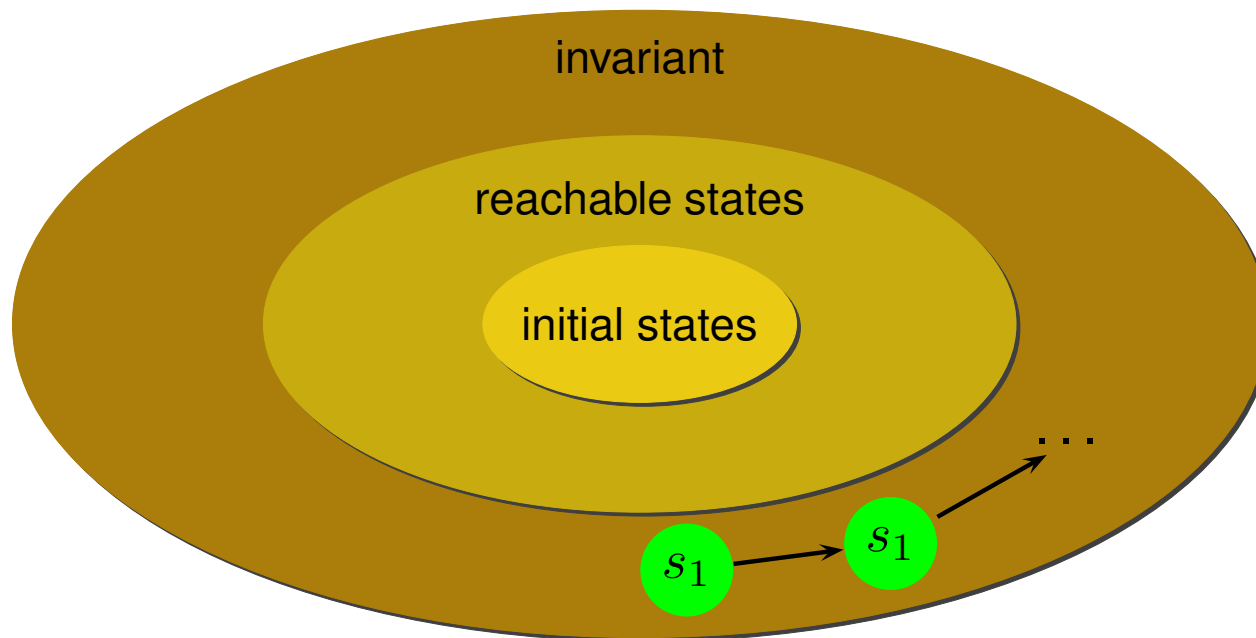
- ▶ Can be used to verify **finite** and **infinite** systems
- ▶ Not **complete** even for **finite** systems
- ▶ Self loops in **unreachable** states
  - ▶ Suppose the state  $s_1$  contains a self loop:



## *k-Induction (cont.)*

---

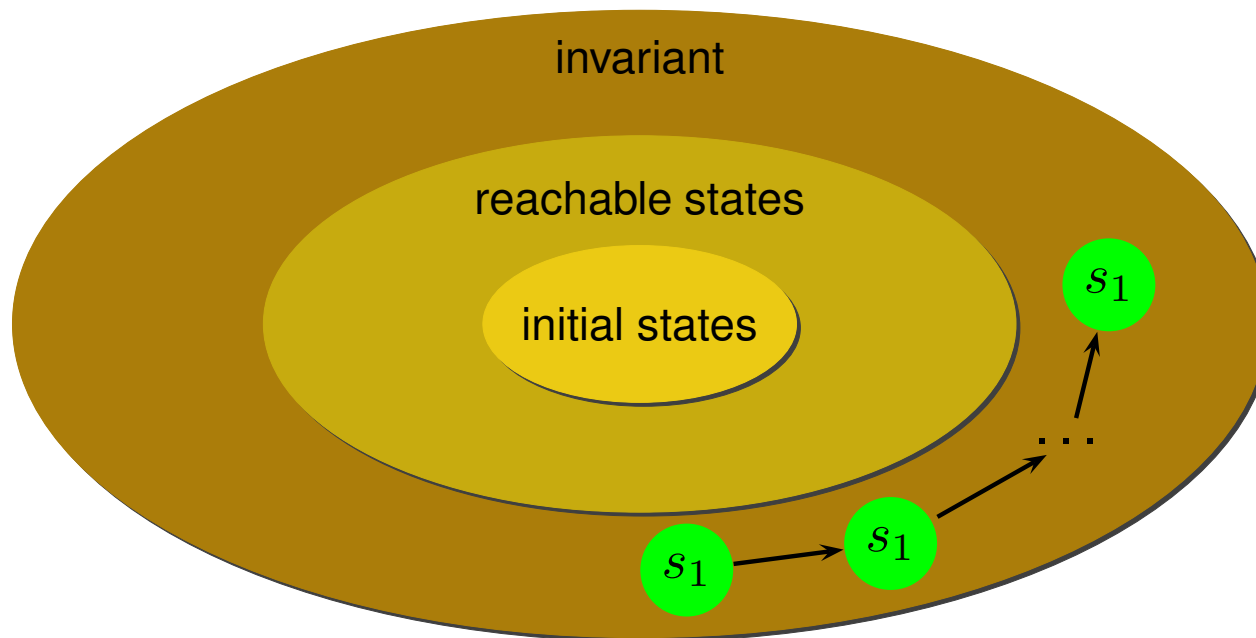
- ▶ Can be used to verify **finite** and **infinite** systems
- ▶ Not **complete** even for **finite** systems
- ▶ Self loops in **unreachable** states
  - ▶ Suppose the state  $s_1$  contains a self loop:



## *k-Induction (cont.)*

---

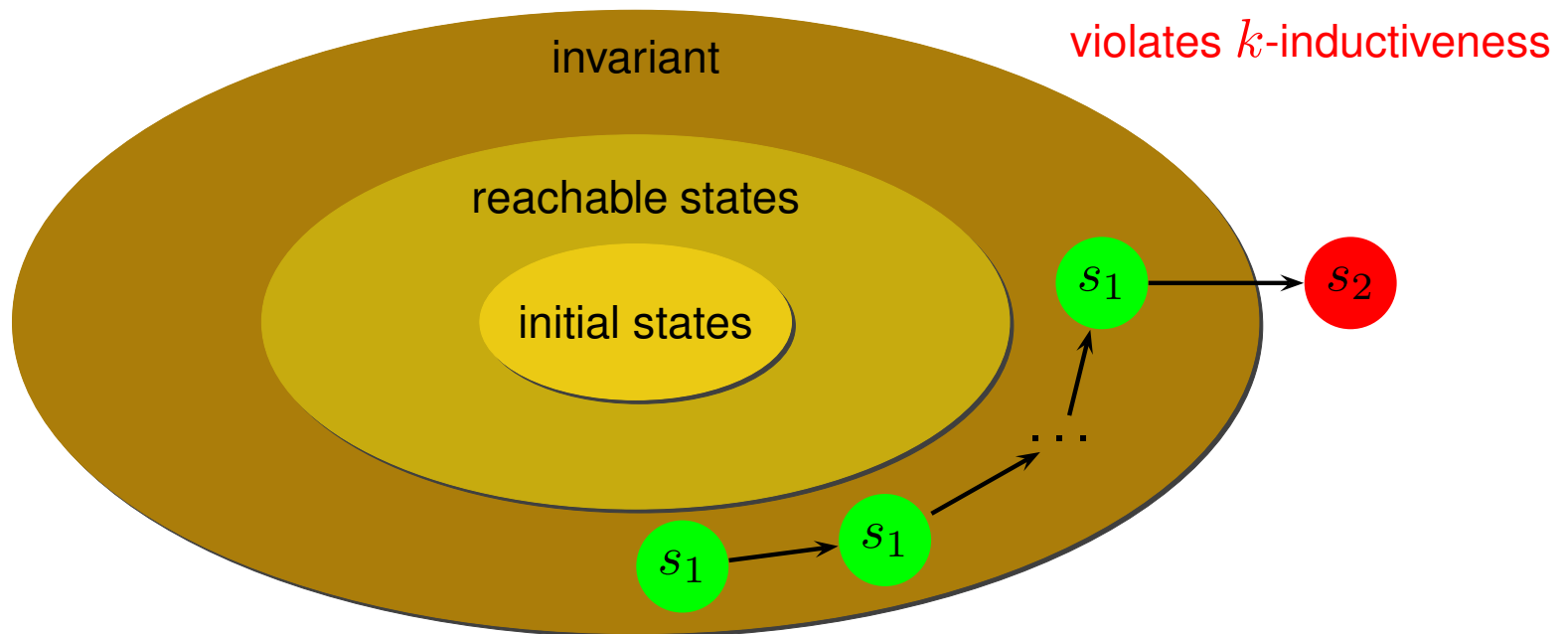
- ▶ Can be used to verify **finite** and **infinite** systems
- ▶ Not **complete** even for finite systems
- ▶ Self loops in **unreachable** states
  - ▶ Suppose the state  $s_1$  contains a self loop:



## *k-Induction (cont.)*

---

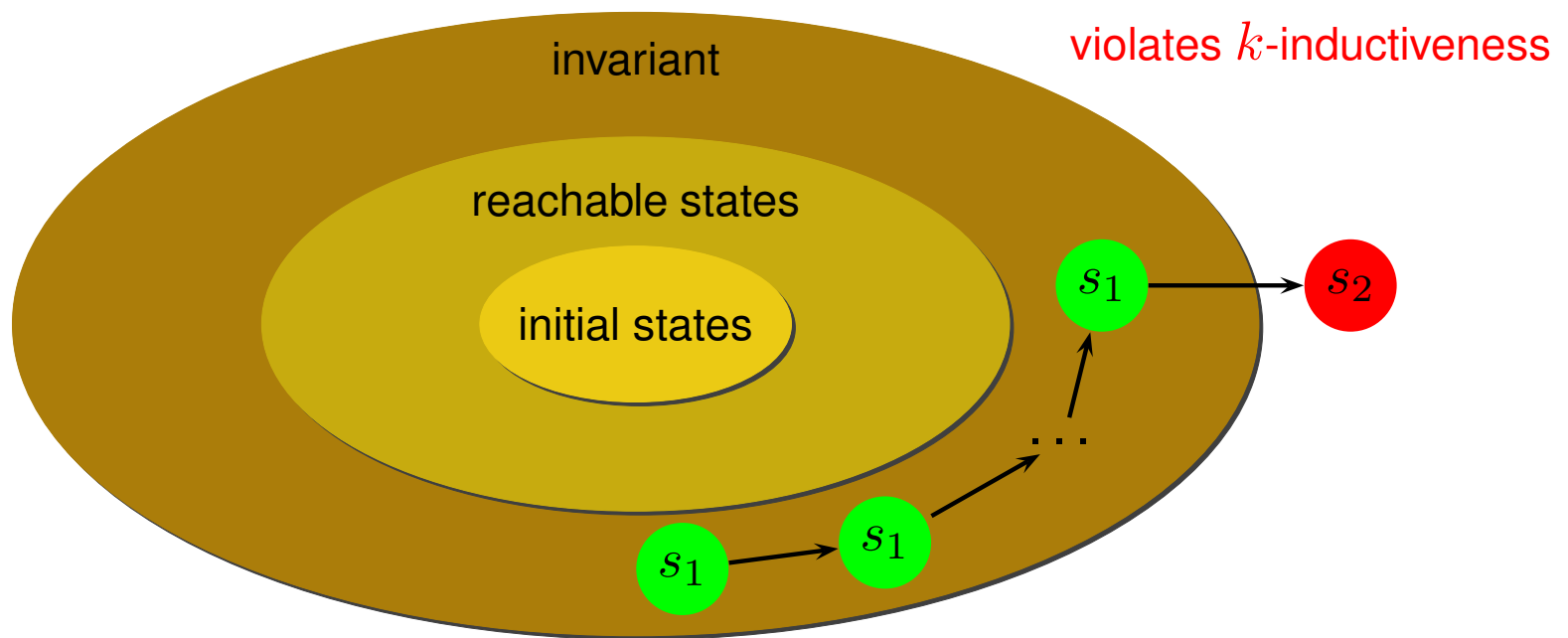
- ▶ Can be used to verify **finite** and **infinite** systems
- ▶ Not **complete** even for finite systems
- ▶ Self loops in unreachable states
  - ▶ Suppose the state  $s_1$  contains a self loop:



## *k-Induction (cont.)*

---

- ▶ Can be used to verify **finite** and **infinite** systems
- ▶ Not **complete** even for finite systems
- ▶ Self loops in **unreachable** states
  - ▶ Suppose the state  $s_1$  contains a self loop:



- ▶ Solution: consider only **acyclic paths** (complete for finite systems)

# Compressing Paths in $k$ -induction

---

- ▶ Consider only some paths (“compressed paths”)
- ▶ Example: **acyclic paths**
- ▶ It is **harder** to violate the **inductive step**
- ▶ Optimization: **decrease** the  $k$  necessary to verify an invariant
- ▶ Based on **direct/reverse simulations**
- ▶ Several **tricks/improvements** found in the literature can be described as “**compressed paths**”
- ▶ Can also be used in the base case (BMC)

# Direct simulations

---

- ▶ The relation  $\preceq$  is a **direct simulation** with respect to  $M = (I, T)$ , and invariant  $\varphi$  if:

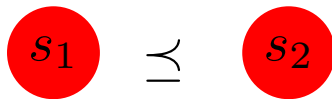
$$s_1 \preceq s_2 \rightarrow \begin{cases} \text{if } \neg\varphi(s_1) \text{ then } \neg\varphi(s_2) \\ \text{else } \forall s'_1 . T(s_1, s'_1) \rightarrow \exists s'_2 . s'_1 \preceq s'_2 \wedge T(s_2, s'_2) \end{cases}$$

# Direct simulations

---

- ▶ The relation  $\preceq$  is a **direct simulation** with respect to  $M = (I, T)$ , and invariant  $\varphi$  if:

$$s_1 \preceq s_2 \rightarrow \begin{cases} \text{if } \neg\varphi(s_1) \text{ then } \neg\varphi(s_2) \\ \text{else } \forall s'_1 . T(s_1, s'_1) \rightarrow \exists s'_2 . s'_1 \preceq s'_2 \wedge T(s_2, s'_2) \end{cases}$$



# Direct simulations

---

- ▶ The relation  $\preceq$  is a **direct simulation** with respect to  $M = (I, T)$ , and invariant  $\varphi$  if:

$$s_1 \preceq s_2 \rightarrow \begin{cases} \text{if } \neg\varphi(s_1) \text{ then } \neg\varphi(s_2) \\ \text{else } \forall s'_1 . T(s_1, s'_1) \rightarrow \exists s'_2 . s'_1 \preceq s'_2 \wedge T(s_2, s'_2) \end{cases}$$



# Direct simulations

---

- ▶ The relation  $\preceq$  is a **direct simulation** with respect to  $M = (I, T)$ , and invariant  $\varphi$  if:

$$s_1 \preceq s_2 \rightarrow \begin{cases} \text{if } \neg\varphi(s_1) \text{ then } \neg\varphi(s_2) \\ \text{else } \forall s'_1 . T(s_1, s'_1) \rightarrow \exists s'_2 . s'_1 \preceq s'_2 \wedge T(s_2, s'_2) \end{cases}$$

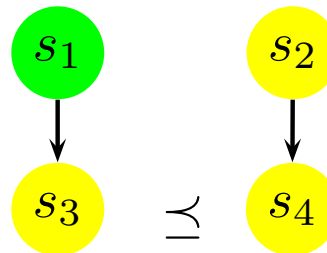
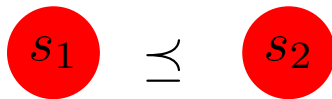


# Direct simulations

---

- ▶ The relation  $\preceq$  is a **direct simulation** with respect to  $M = (I, T)$ , and invariant  $\varphi$  if:

$$s_1 \preceq s_2 \rightarrow \begin{cases} \text{if } \neg\varphi(s_1) \text{ then } \neg\varphi(s_2) \\ \text{else } \forall s'_1 . T(s_1, s'_1) \rightarrow \exists s'_2 . s'_1 \preceq s'_2 \wedge T(s_2, s'_2) \end{cases}$$

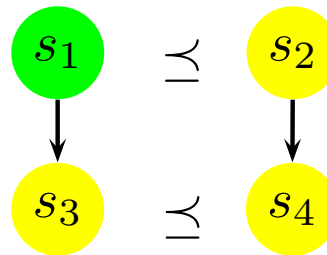


# Direct simulations

---

- ▶ The relation  $\preceq$  is a **direct simulation** with respect to  $M = (I, T)$ , and invariant  $\varphi$  if:

$$s_1 \preceq s_2 \rightarrow \begin{cases} \text{if } \neg\varphi(s_1) \text{ then } \neg\varphi(s_2) \\ \text{else } \forall s'_1 . T(s_1, s'_1) \rightarrow \exists s'_2 . s'_1 \preceq s'_2 \wedge T(s_2, s'_2) \end{cases}$$

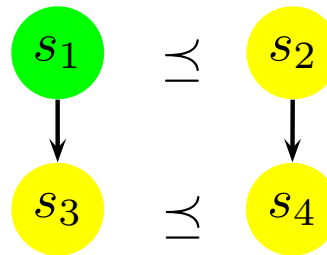


# Direct simulations

---

- ▶ The relation  $\preceq$  is a **direct simulation** with respect to  $M = (I, T)$ , and invariant  $\varphi$  if:

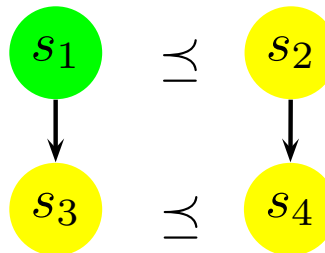
$$s_1 \preceq s_2 \rightarrow \begin{cases} \text{if } \neg\varphi(s_1) \text{ then } \neg\varphi(s_2) \\ \text{else } \forall s'_1 . T(s_1, s'_1) \rightarrow \exists s'_2 . s'_1 \preceq s'_2 \wedge T(s_2, s'_2) \end{cases}$$



# Direct simulations

- ▶ The relation  $\preceq$  is a **direct simulation** with respect to  $M = (I, T)$ , and invariant  $\varphi$  if:

$$s_1 \preceq s_2 \rightarrow \begin{cases} \text{if } \neg\varphi(s_1) \text{ then } \neg\varphi(s_2) \\ \text{else } \forall s'_1 . T(s_1, s'_1) \rightarrow \exists s'_2 . s'_1 \preceq s'_2 \wedge T(s_2, s'_2) \end{cases}$$



- ▶ Example: **Equality**

## Direct simulations (cont.)

---

▶  $s_1 \preceq s_2$

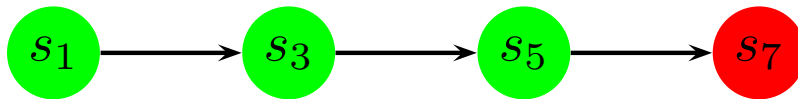
if there is a **path** to a **bad state** starting at  $s_1$ , then there is also a (possibly **shorter**) **path** to a **bad state** starting at  $s_2$

## Direct simulations (cont.)

---

▶  $s_1 \preceq s_2$

if there is a **path** to a **bad state** starting at  $s_1$ , then there is also a (possibly **shorter**) **path** to a **bad state** starting at  $s_2$

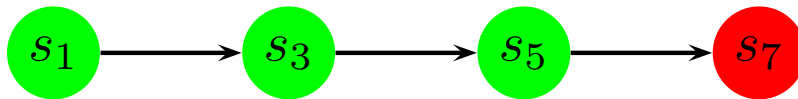


## Direct simulations (cont.)

---

▶  $s_1 \preceq s_2$

if there is a **path** to a **bad state** starting at  $s_1$ , then there is also a (possibly **shorter**) **path** to a **bad state** starting at  $s_2$



|∧

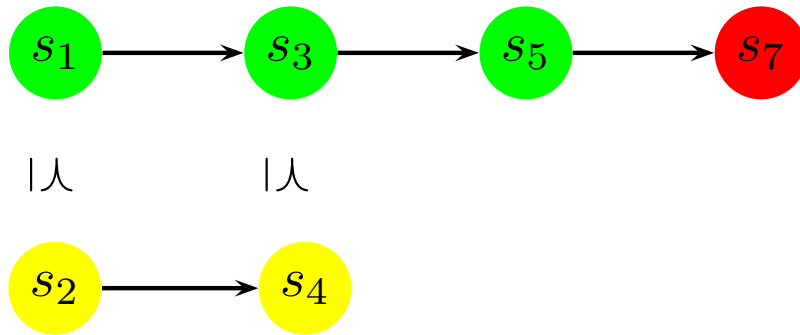


## Direct simulations (cont.)

---

▶  $s_1 \preceq s_2$

if there is a **path** to a **bad state** starting at  $s_1$ , then there is also a (possibly **shorter**) **path** to a **bad state** starting at  $s_2$

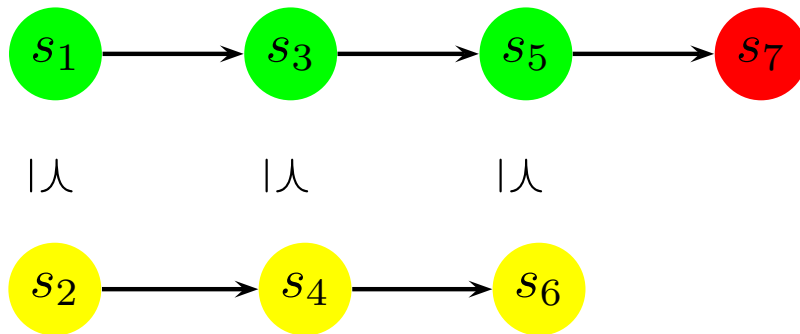


## Direct simulations (cont.)

---

▶  $s_1 \preceq s_2$

if there is a **path** to a **bad state** starting at  $s_1$ , then there is also a (possibly **shorter**) **path** to a **bad state** starting at  $s_2$

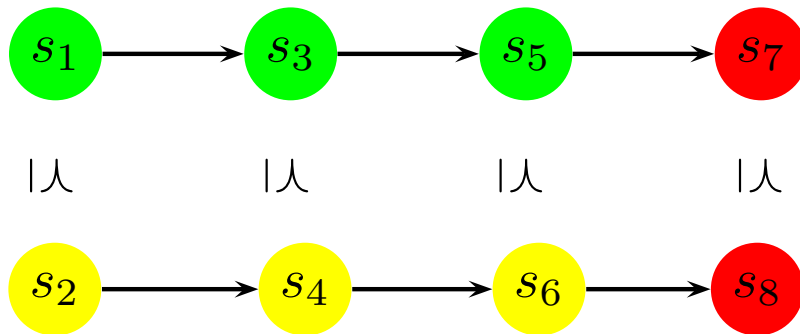


## Direct simulations (cont.)

---

▶  $s_1 \preceq s_2$

if there is a **path** to a **bad state** starting at  $s_1$ , then there is also a (possibly **shorter**) **path** to a **bad state** starting at  $s_2$



## Direct simulations (cont.)

---

- ▶ A path  $\pi(s_1, \dots, s_n)$  is compressed with respect to  $\preceq$  if:

$$s_i \not\preceq s_j \text{ for } 1 \leq j < i \leq n$$

- ▶ Consider only compressed paths in  $k$ -induction

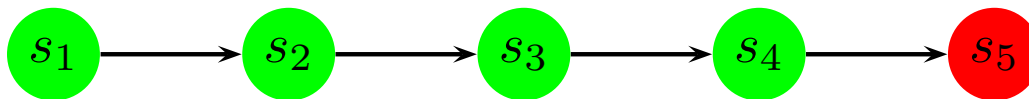
## Direct simulations (cont.)

---

- ▶ A path  $\pi(s_1, \dots, s_n)$  is compressed with respect to  $\preceq$  if:

$$s_i \not\preceq s_j \text{ for } 1 \leq j < i \leq n$$

- ▶ Consider only compressed paths in  $k$ -induction



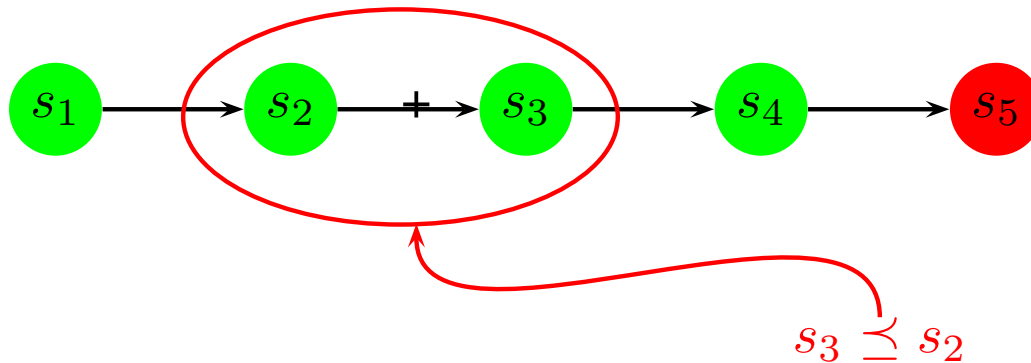
## Direct simulations (cont.)

---

- ▶ A path  $\pi(s_1, \dots, s_n)$  is compressed with respect to  $\preceq$  if:

$$s_i \not\preceq s_j \text{ for } 1 \leq j < i \leq n$$

- ▶ Consider only compressed paths in  $k$ -induction



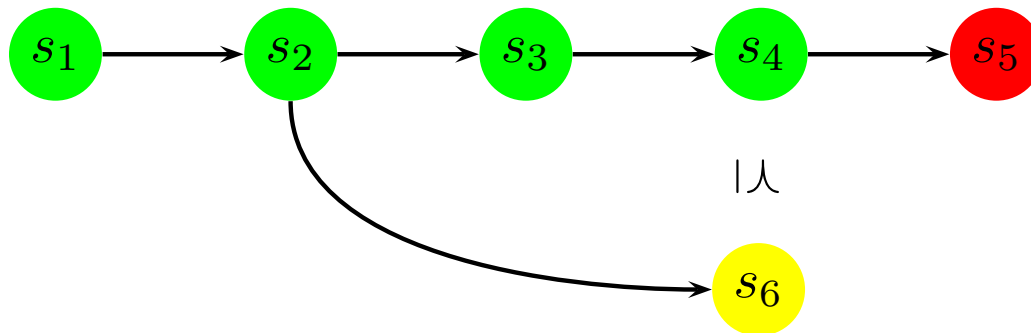
## Direct simulations (cont.)

---

- ▶ A path  $\pi(s_1, \dots, s_n)$  is compressed with respect to  $\preceq$  if:

$$s_i \not\preceq s_j \text{ for } 1 \leq j < i \leq n$$

- ▶ Consider only compressed paths in  $k$ -induction



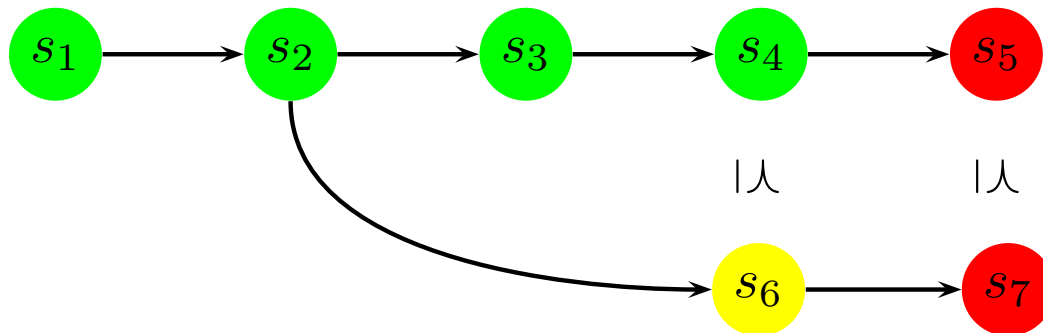
## Direct simulations (cont.)

---

- ▶ A path  $\pi(s_1, \dots, s_n)$  is compressed with respect to  $\preceq$  if:

$$s_i \not\preceq s_j \text{ for } 1 \leq j < i \leq n$$

- ▶ Consider only compressed paths in  $k$ -induction



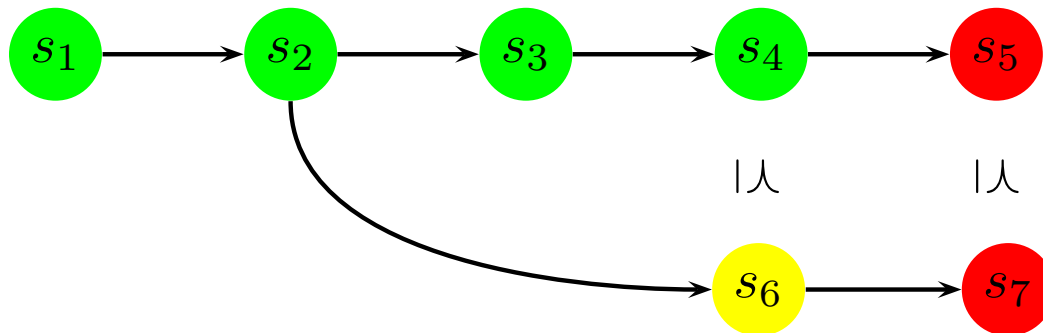
## Direct simulations (cont.)

---

- ▶ A path  $\pi(s_1, \dots, s_n)$  is compressed with respect to  $\preceq$  if:

$$s_i \not\preceq s_j \text{ for } 1 \leq j < i \leq n$$

- ▶ Consider only compressed paths in  $k$ -induction



- ▶ A compressed path with respect to  $=$  is an acyclic path

# Reverse simulations

---

- ▶ The relation  $\preceq$  is a reverse simulation with respect to  $M = (I, T)$  if:

$$s_1 \preceq s_2 \rightarrow \begin{cases} \text{if } I(s_1) \text{ then } I(s_2) \\ \text{else } \forall s'_1 . T(s'_1, s_1) \rightarrow \exists s'_2 . s'_1 \preceq s'_2 \wedge T(s'_2, s_2) \end{cases}$$

- ▶  $s_1 \preceq s_2$

if there is a path from an initial state to  $s_1$ , then there is also a (possibly shorter) path from an initial state to  $s_2$

- ▶ A path  $\pi(s_1, \dots, s_n)$  is compressed with respect to  $\preceq$  if:

$$s_i \not\preceq s_j \text{ for } 1 \leq i < j \leq n$$

# Reverse simulations: examples

---

- ▶ Equality
- ▶ At most one initial state

$$s_1 \preceq_I s_2 \leftrightarrow I(s_1) \wedge I(s_2)$$

- ▶ Equality modulo non-input variables
  - ▶ input variables are not constrained by  $I$
  - ▶ the next value of input variables is not constrained by  $T$

# Compressing Paths in $k$ -induction (cont.)

---

- ▶ The union of two direct/reverse simulations  $\preceq_1$  and  $\preceq_2$  is a direct/reverse simulation
- ▶ Consider only compressed paths with respect to direct/reverse simulations in  $k$ -induction
- ▶  $k$ -induction with respect to  $\preceq_d$  and  $\preceq_r$ 
  - ▶  $I(s_1) \wedge \pi_{d,r}(s_1, \dots, s_k) \rightarrow \varphi(s_1) \wedge \dots \wedge \varphi(s_k)$
  - ▶  $\varphi(s_1) \wedge \dots \wedge \varphi(s_k) \wedge \pi_{d,r}(s_1, \dots, s_{k+1}) \rightarrow \varphi(s_{k+1})$

$$\pi_{d,r}(s_1, \dots, s_k) := \begin{cases} T(s_1, s_2) \wedge \dots \wedge T(s_{k-1}, s_k) \\ \wedge \bigwedge_{1 \leq j < i \leq k} s_i \not\preceq_d s_j \\ \wedge \bigwedge_{1 \leq i < j \leq k} s_i \not\preceq_r s_j \end{cases}$$

# Timed automata

---

- ▶ Decidability of the model-checking problem for timed automata rests on the fact that the space of clock valuations is partitioned into finitely many clock regions
- ▶ Two clock valuations  $v_1$  and  $v_2$  that belong to the same clock region are region equivalent
- ▶ Equality modulo region equivalence  $\sim_{TA}$  is a direct simulation
- ▶ Path compression with respect to  $\sim_{TA}$ 
  - ▶  $k$ -induction is complete for timed automata
  - ▶ There are finitely many clock regions (upper bound for the length of compressed paths)

# *Invariant Strengthening*

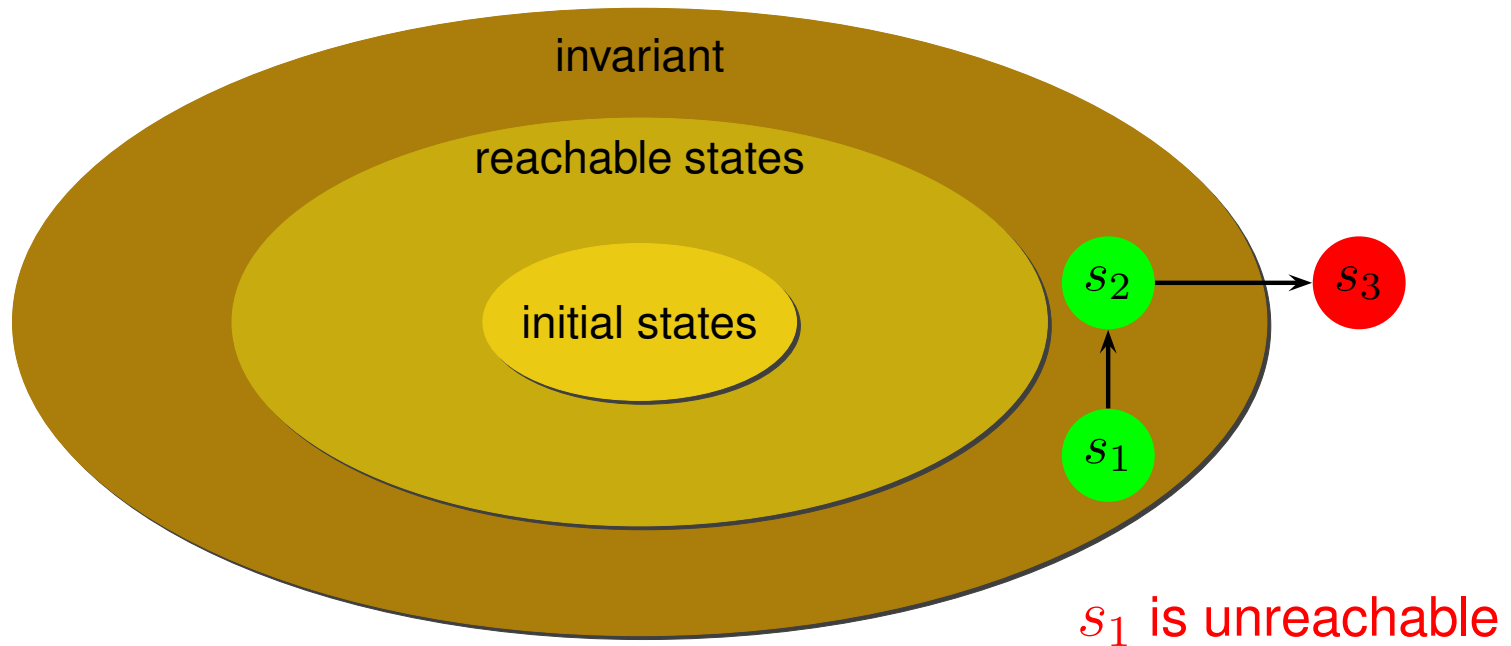
---

- ▶ Violation (counterexample) for the inductive step of  $k$ -inductiveness: 2 cases

# Invariant Strengthening

---

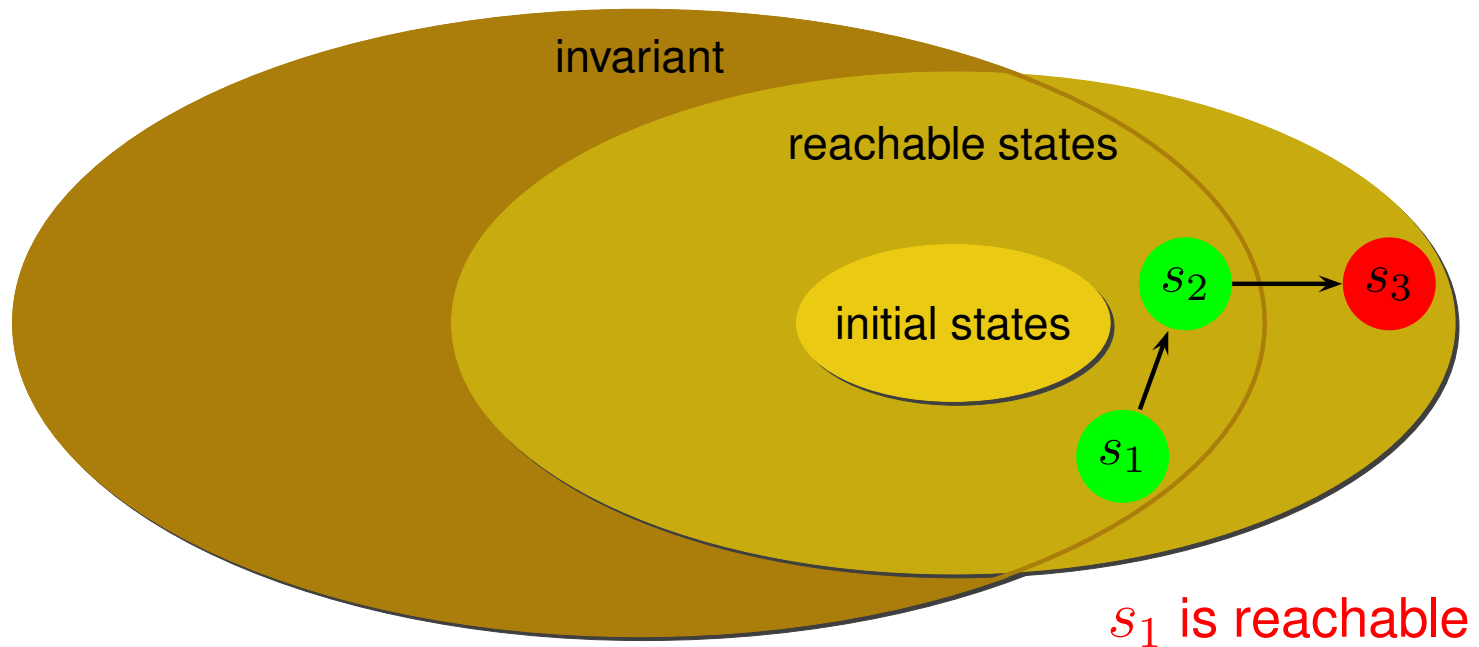
- ▶ Violation (counterexample) for the inductive step of  $k$ -inductiveness: 2 cases



# Invariant Strengthening

---

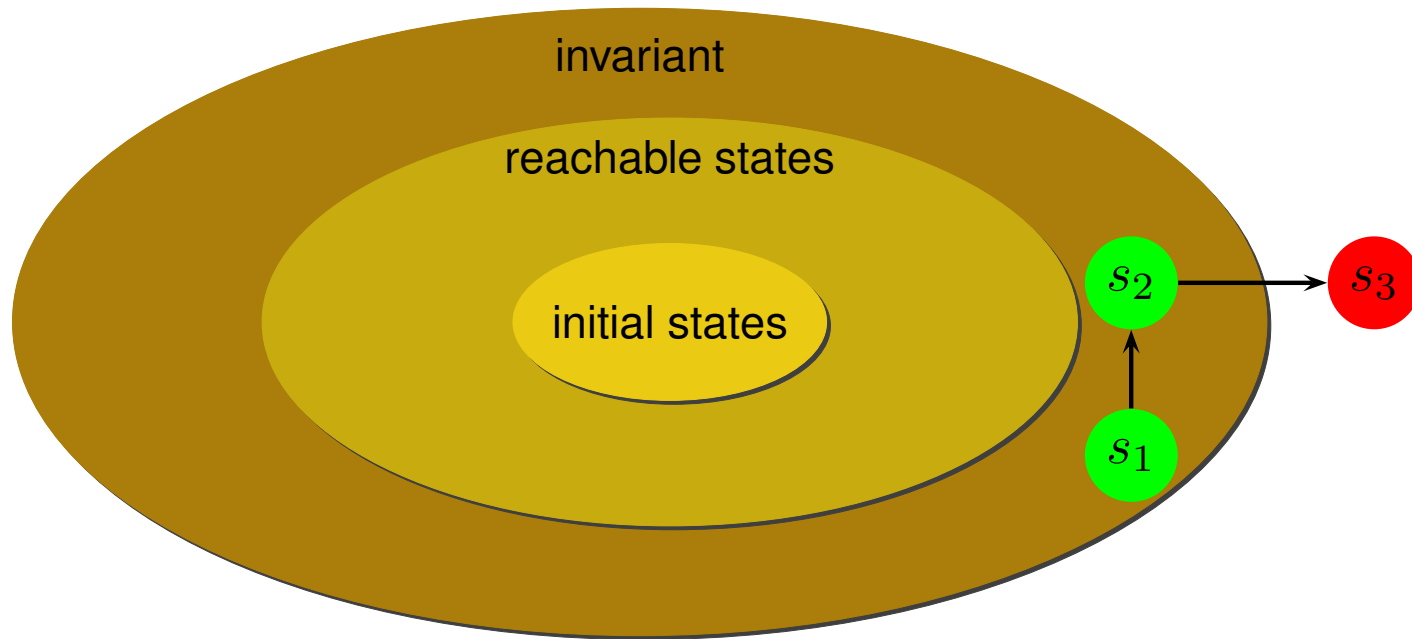
- ▶ Violation (counterexample) for the inductive step of  $k$ -inductiveness: 2 cases



# Invariant Strengthening

---

- ▶ Violation (counterexample) for the inductive step of  $k$ -inductiveness: 2 cases

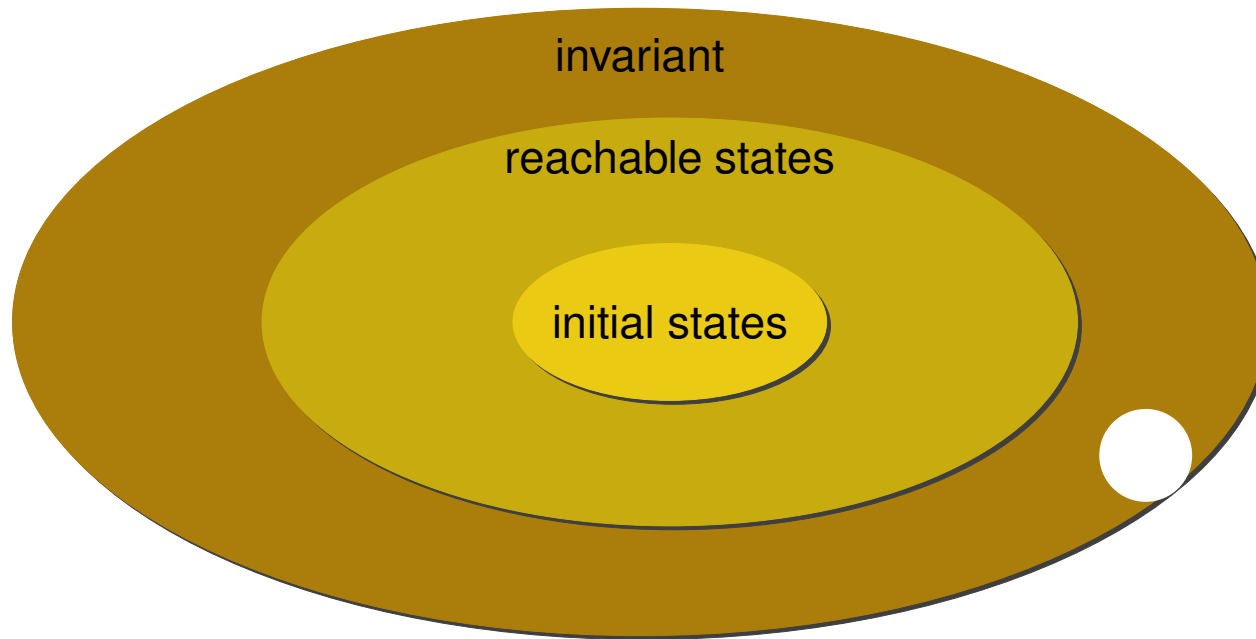


- ▶ Let us **assume** that  $s_1$  is unreachable

# Invariant Strengthening

---

- ▶ Violation (counterexample) for the inductive step of  $k$ -inductiveness: 2 cases



- ▶ Let us **assume** that  $s_1$  is unreachable
- ▶ new invariant:  $\varphi'(s) = \varphi(s) \wedge s \neq s_1$
- ▶ but if  $s_1$  is **reachable**: we will eventually **violate** the **base case**

# Invariant Strengthening (cont.)

---

- ▶ One state at a time: too slow
- ▶ Collect all violations for the inductive step

$$Q(s_1, \dots, s_{k+1}) := \varphi(s_1) \wedge \dots \wedge \varphi(s_k) \wedge \pi(s_1, \dots, s_{k+1}) \wedge \neg\varphi(s_{k+1})$$
$$violations(s_1) := \exists s_2, \dots, s_{k+1}. Q(s_1, s_2, \dots, s_{k+1})$$

- ▶ new invariant:  $\varphi'(s) = \varphi(s) \wedge \neg violations(s)$

# Quantifier Elimination

---

- ▶ Avoid DNF conversion
- ▶ Similar to the approach used by [McMillan02]
- ▶ Example:

$$q := \exists x_1, y_1 .$$

$$\begin{aligned} & [((x_0 = 1 \vee x_0 = 3 \vee y_0 > 1) \wedge x_1 = x_0 - 1 \wedge y_1 = y_0 + 1) \vee \\ & ((x_0 = -1 \vee x_0 = -3) \wedge x_1 = x_0 + 2 \wedge y_1 = y_0 - 1)] \wedge \\ & x_1 < 0 \end{aligned}$$

Extracting relevant atoms from the satisfying assignment

$$c_1 := y_0 > 1 \wedge x_1 = x_0 - 1 \wedge y_1 = y_0 + 1 \wedge x_1 < 0$$

Removing  $x_1$  and  $y_1$  from  $c_1$

$$c'_1 := y_0 > 1 \wedge x_0 - 1 < 0$$

# Quantifier Elimination (cont.)

---

► Example (cont.)

Extracting relevant atoms from the satisfying assignment for  $q \wedge \neg c'_1$

$$c_2 := x_0 = -3 \wedge x_1 = x_0 + 2 \wedge y_1 = y_0 - 1 \wedge x_1 < 0$$

Removing  $x_1$  and  $y_1$  from  $c_2$

$$c'_2 := x_0 = -3$$

$q \wedge \neg c'_1 \wedge \neg c'_2$  is **unsatisfiable**, since there are no further solutions, the quantifier eliminated formula is

$$(y_0 > 1 \wedge x_0 - 1 < 0) \vee x_0 = -3$$

# Experiments

---

System Name	Proved with $k$	Time (secs)	n. Strengthening
Bakery Protocol	3	0.21	1
Simpson Protocol	2	0.16	2
Train Gate Controller	5	0.52	0
Fischer Protocol	4	0.71	0
Water Level Monitor	1	0.08	0
Leaking Gas Burner	6	1.13	0
Multi Rate Fischer	4	0.84	0

# Conclusion & Future work

---

- ▶  $k$ -induction scheme based on the notion of direct/reverse simulations
- ▶ Completeness for time automata
- ▶ Implemented using ICS & SAL
- ▶  $k$ -induction  $\times$  invariant strengthening
- ▶ Integrating  $k$ -induction with other approaches:
  - ▶ Invariant generation
  - ▶ Static analysis
  - ▶ Auxiliary lemmas
  - ▶ Abstraction