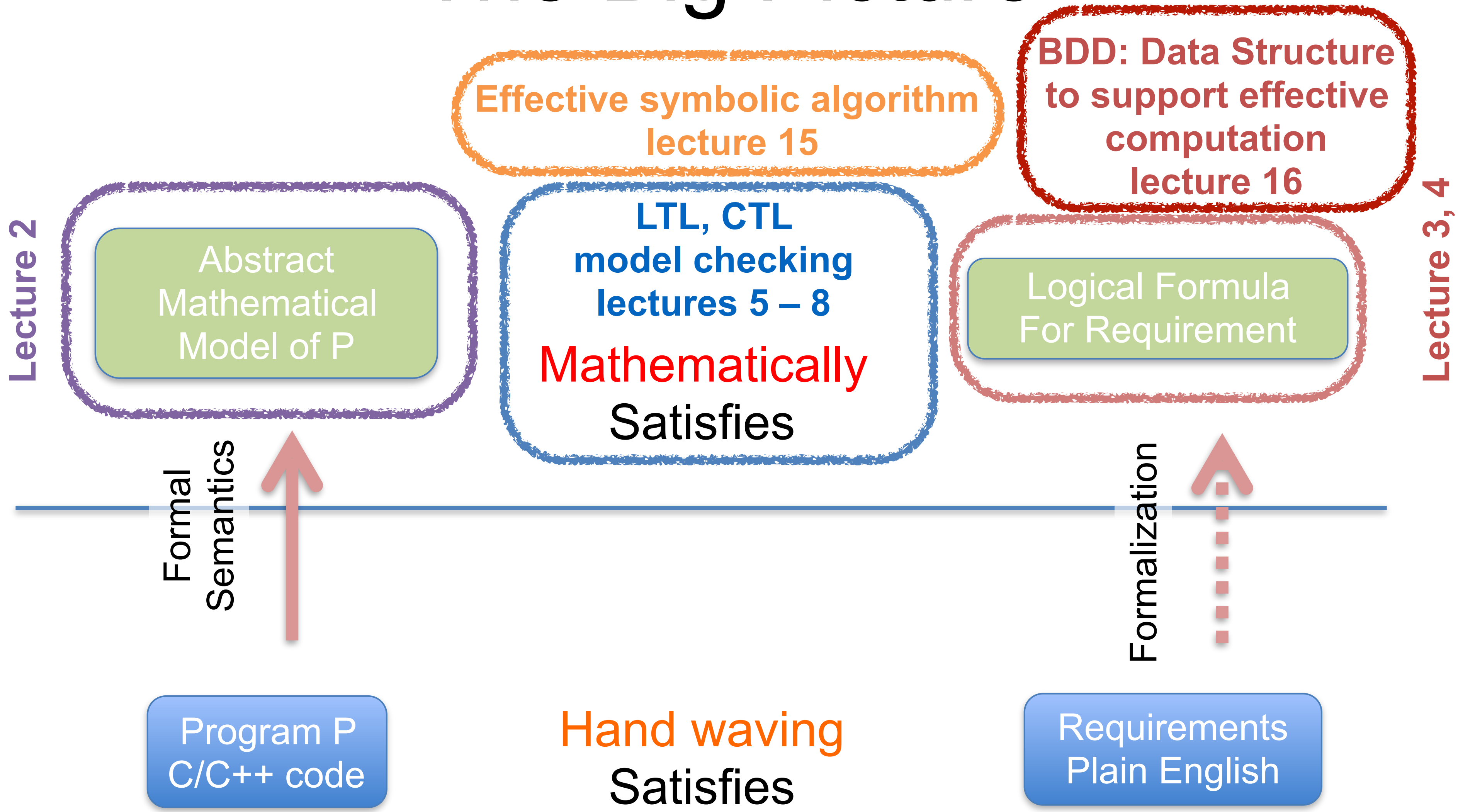
Algorithmic Verification COMP 3153

Lecture 17

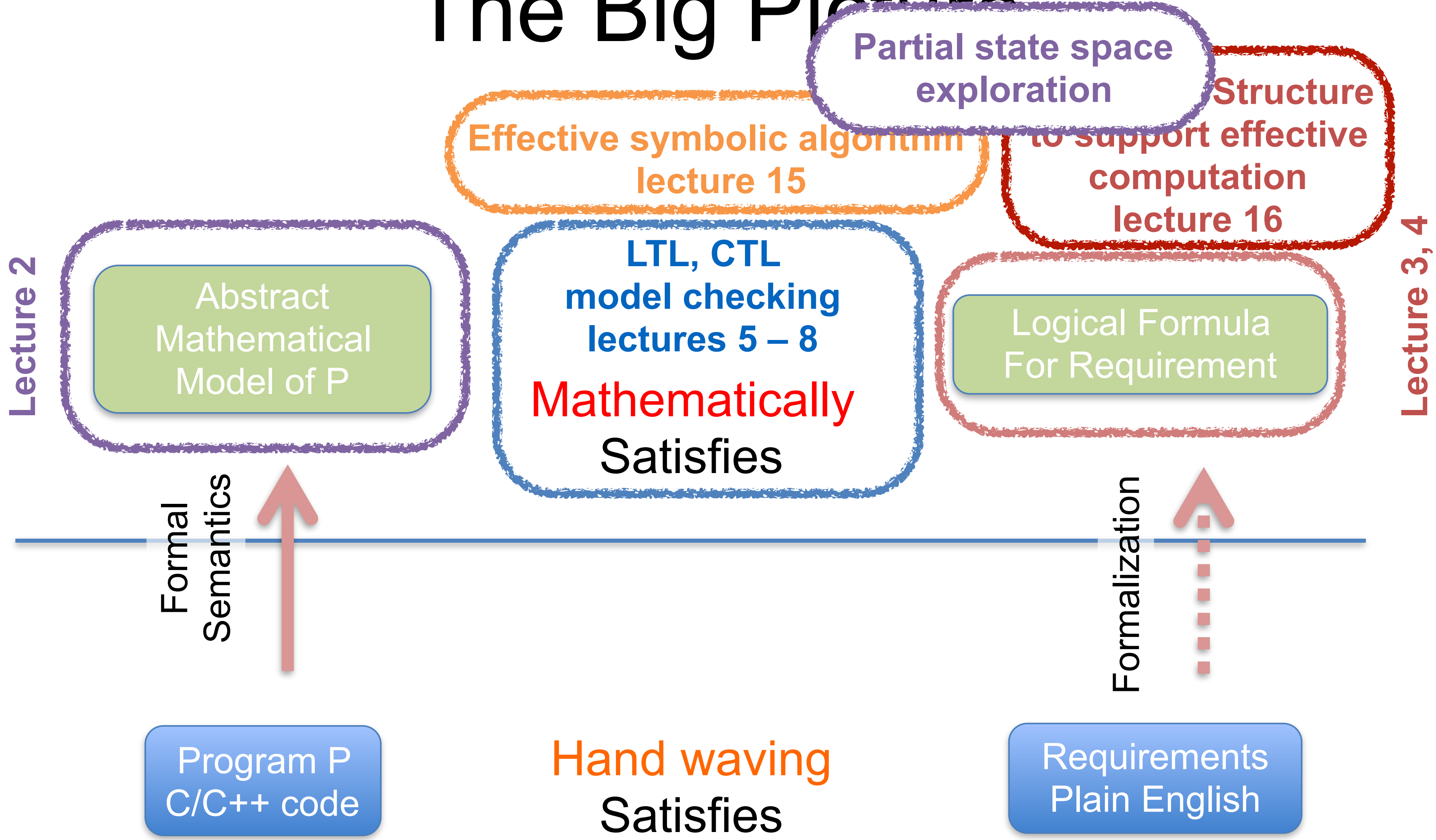
Symbolic Model Checking – 2

(last update: April 16, 2018)

The Big Picture



The Big Picture



Content for this lecture

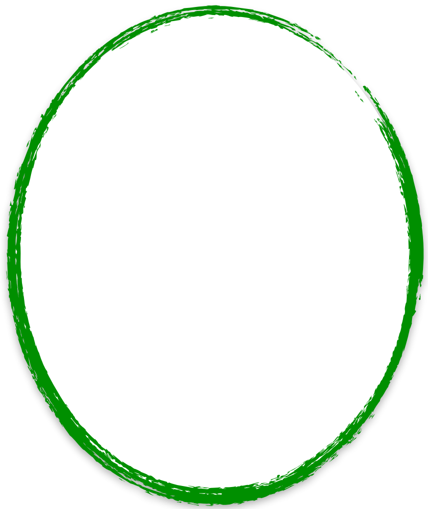
1. SAT problem
2. Model Checking with SAT?
3. Bounded LTL Semantics
4. Bounded Semantics to SAT

SAT Problem

SAT Problem

s vector of ***n*** boolean variables $\boldsymbol{s[1], s[2], \cdots, s[n]}$

Assignment ν : $\{1, 2, \cdots, n\} \rightarrow \{0, 1\}$

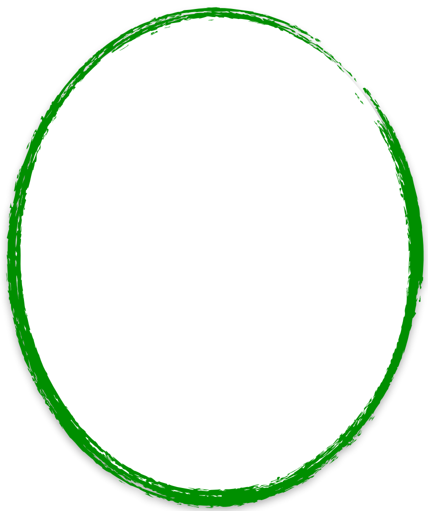


SAT Problem

$\boldsymbol{s}$ vector of $\boldsymbol{n}$ boolean variables $\boldsymbol{s}[1], \boldsymbol{s}[2], \cdots, \boldsymbol{s}[\boldsymbol{n}]$

Assignment $\nu : \{1, 2, \cdots, n\} \to \{0, 1\}$

Set of all assignments: $\mathbb{B}^n$



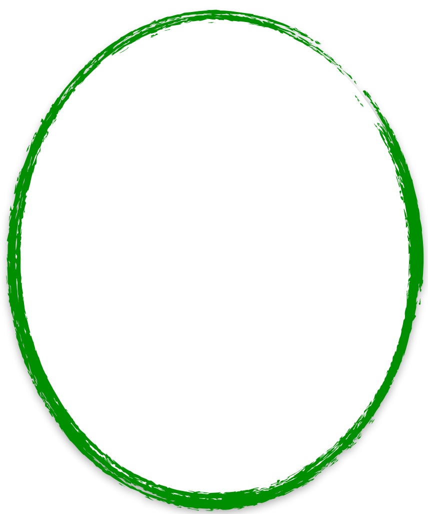
SAT Problem

s vector of ***n*** boolean variables ***s*[1], *s*[2], $\cdots$, *s*[*n*]**

Assignment ν : $\{1, 2, \cdots, n\} \to \{0, 1\}$

Set of all assignments: $\mathbb{B}^n$

$\mathbb{B}^n$

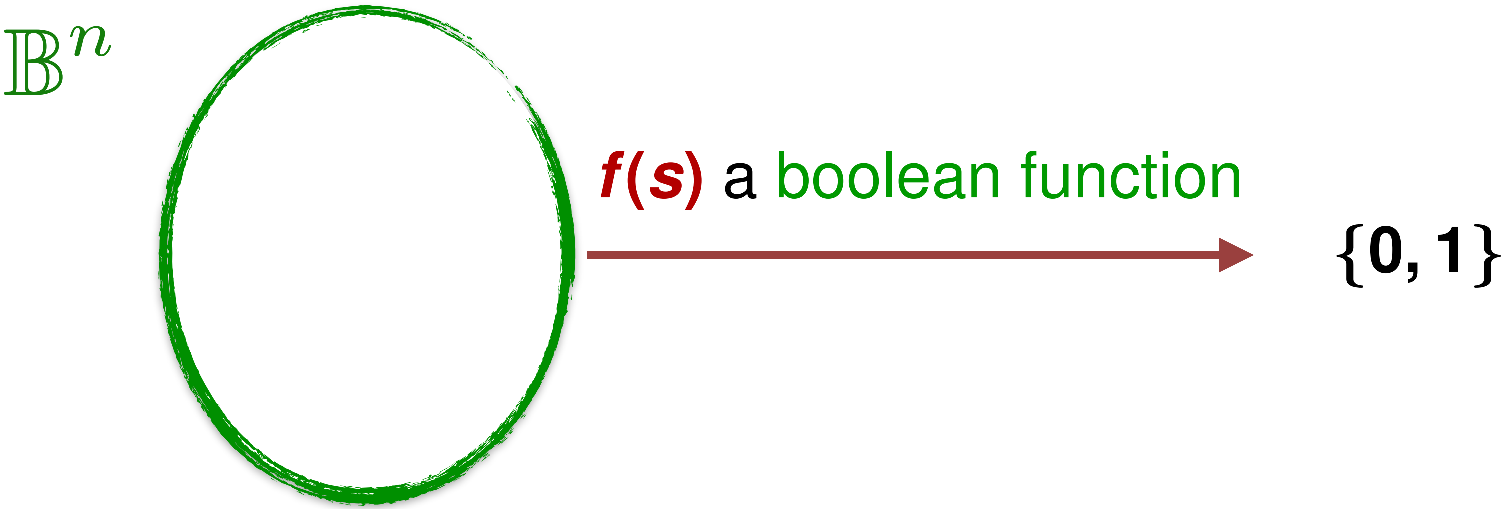


SAT Problem

s vector of ***n*** boolean variables $\boldsymbol{s}[1], \boldsymbol{s}[2], \cdots, \boldsymbol{s}[\boldsymbol{n}]$

Assignment $\nu : \{1, 2, \cdots, n\} \to \{0, 1\}$

Set of all assignments: $\mathbb{B}^n$

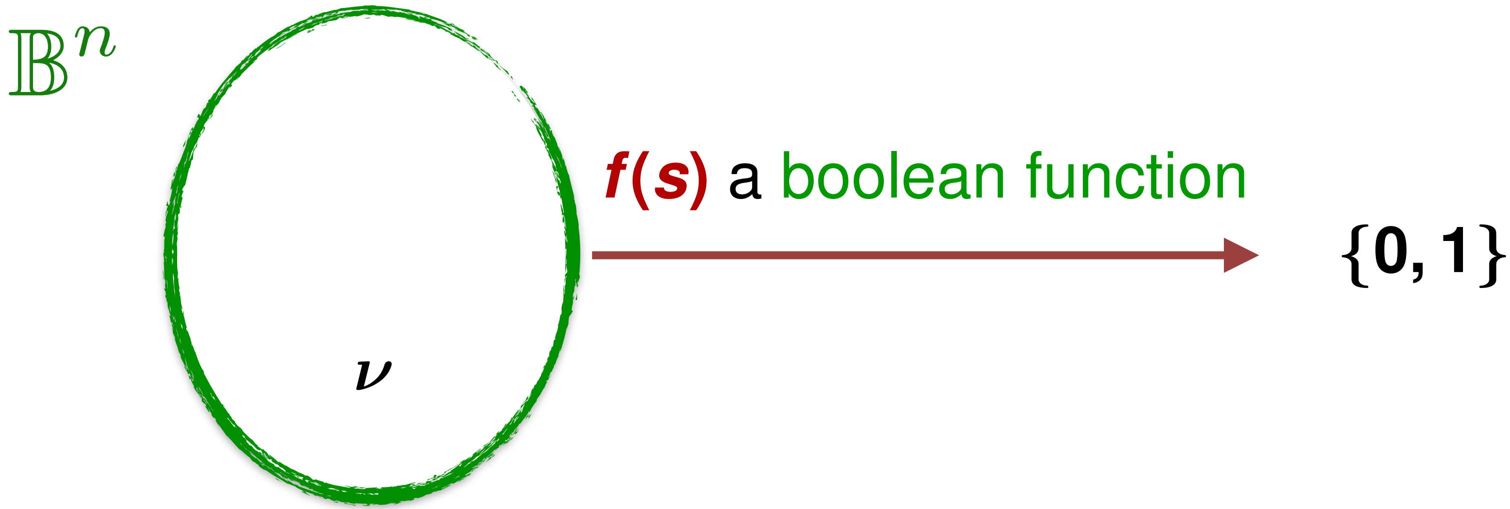


SAT Problem

s vector of ***n*** boolean variables ***s*[1], *s*[2], $\cdots$, *s*[*n*]**

Assignment ν : $\{1, 2, \cdots, n\} \to \{0, 1\}$

Set of all assignments: $\mathbb{B}^n$

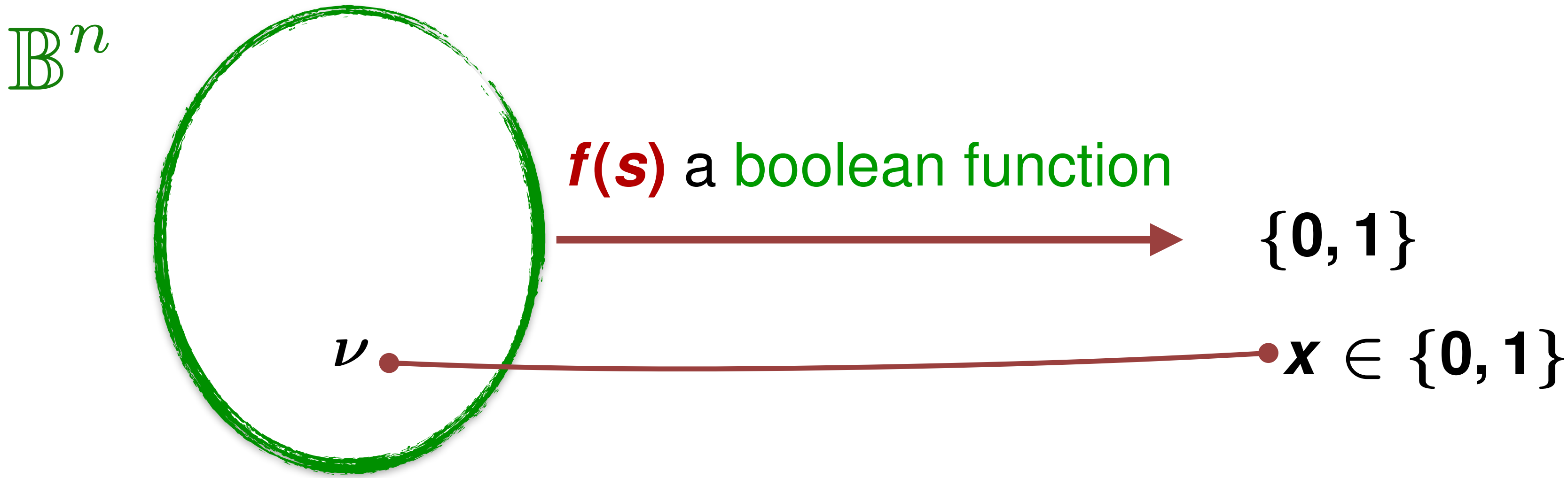


SAT Problem

s vector of ***n*** boolean variables $\boldsymbol{s}[1], \boldsymbol{s}[2], \cdots, \boldsymbol{s}[n]$

Assignment $\nu : \{1, 2, \cdots, n\} \rightarrow \{0, 1\}$

Set of all assignments: $\mathbb{B}^n$

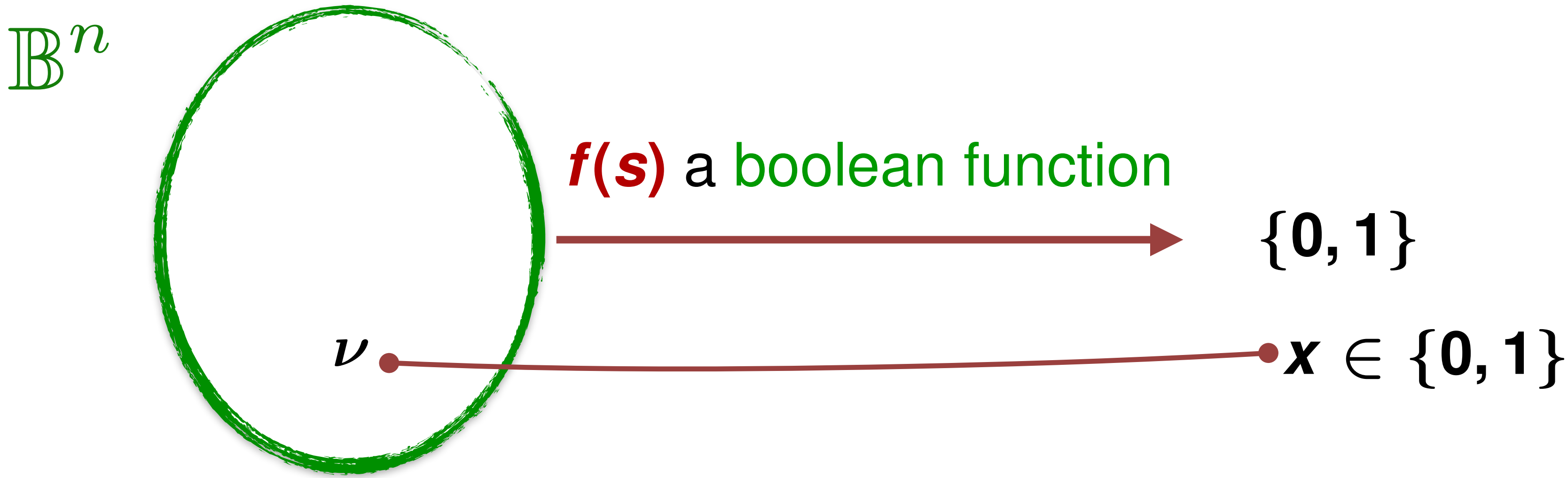


SAT Problem

s vector of ***n*** boolean variables $\boldsymbol{s}[1], \boldsymbol{s}[2], \cdots, \boldsymbol{s}[n]$

Assignment $\nu : \{1, 2, \cdots, n\} \rightarrow \{0, 1\}$

Set of all assignments: $\mathbb{B}^n$



SAT Problem: $\exists \nu \in \mathbb{B}^n$ such that $f(\nu) = 1$?

Complexity of SAT Problem

SAT is NP-complete

Complexity of SAT Problem

SAT is NP-complete

Cook-Levin Theorem, 1971

Complexity of SAT Problem

SAT is NP-complete

Cook-Levin Theorem, 1971

3SAT

- **3** variables
- Formulas in Conjunctive Normal Form

Complexity of SAT Problem

SAT is NP-complete

Cook-Levin Theorem, 1971

3SAT

- **3** variables
- Formulas in Conjunctive Normal Form

3SAT is NP-complete

Karp Theorem, 1971

SAT Solvers in Practise

SAT Solvers in Practise

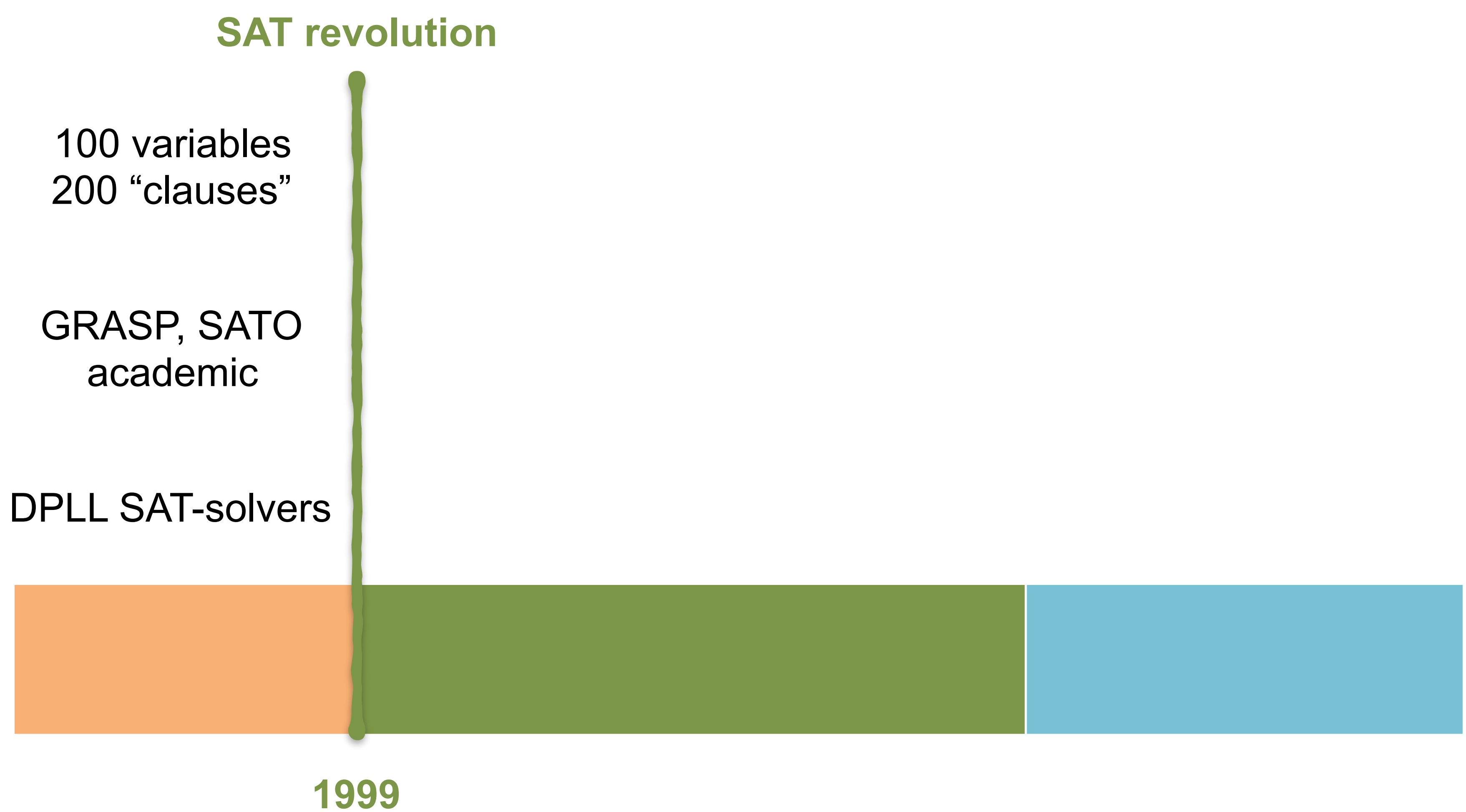
100 variables
200 "clauses"

GRASP, SATO
academic

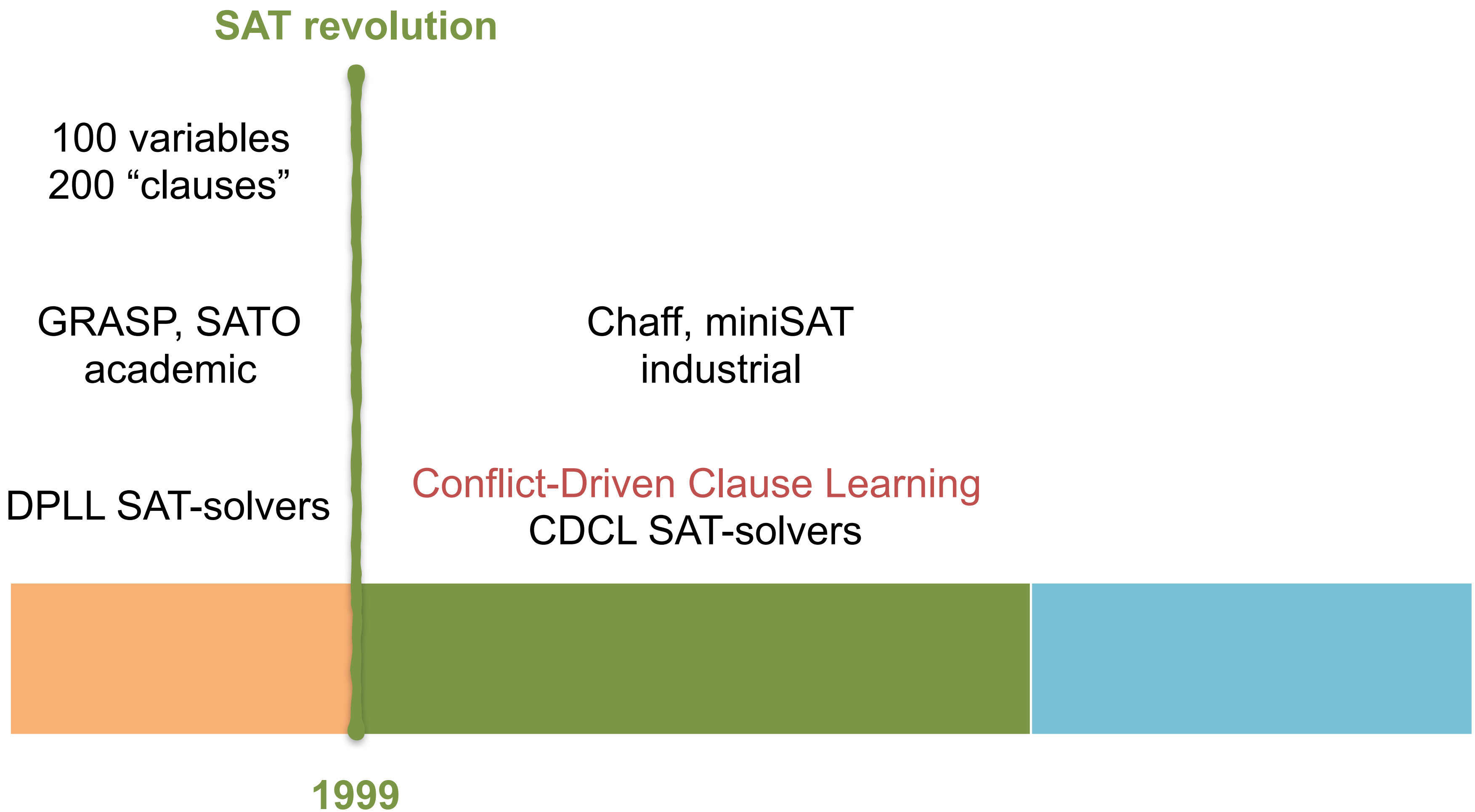
DPLL SAT-solvers

1999

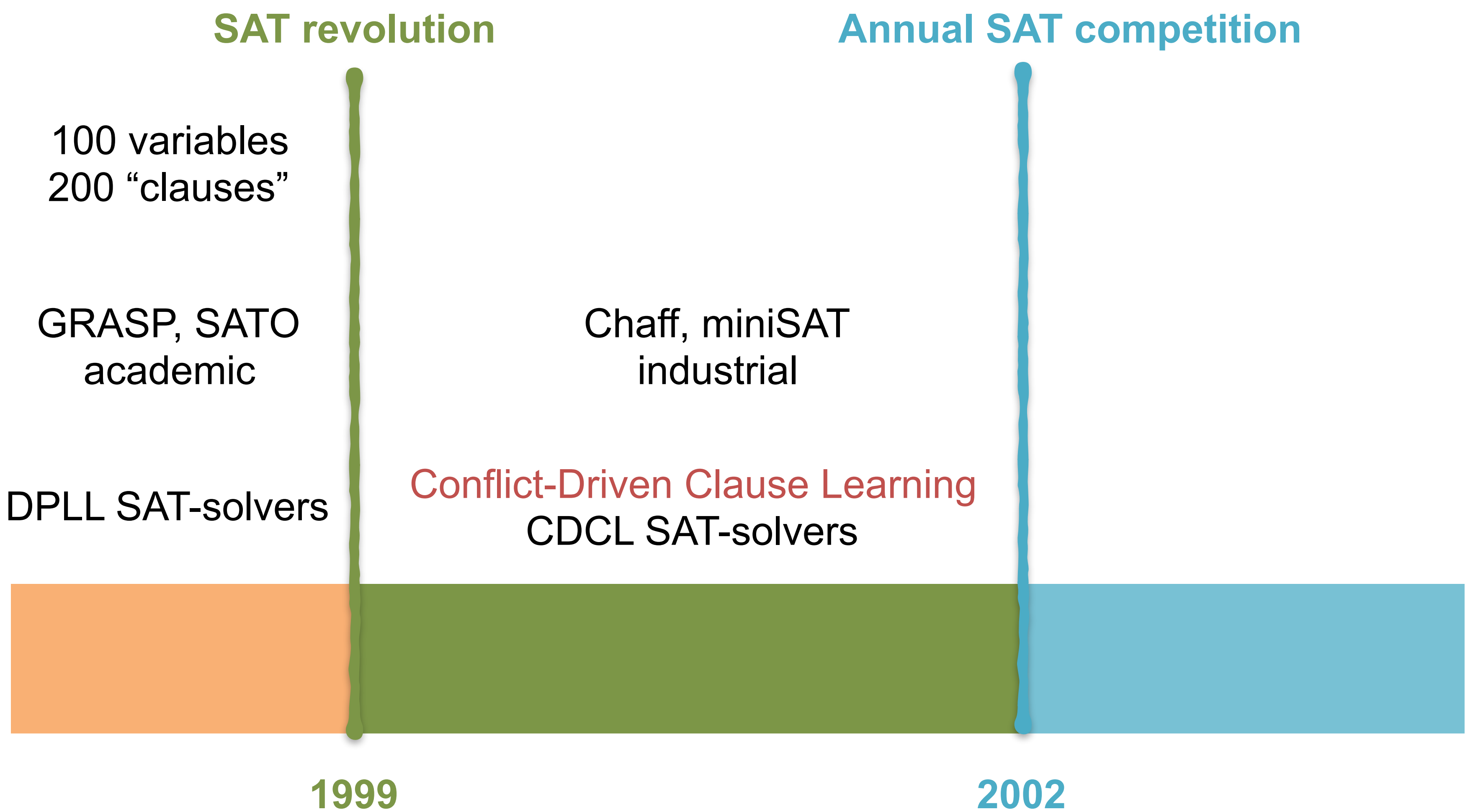
SAT Solvers in Practise



SAT Solvers in Practise



SAT Solvers in Practise



SAT Solvers in Practise

SAT revolution — **1999**

Annual SAT competition — **2002**

100 variables
200 “clauses”

GRASP, SATO
academic

DPLL SAT-solvers

Chaff, miniSAT
industrial

Conflict-Driven Clause Learning
CDCL SAT-solvers

1M variables
5M “clauses”

50+ SAT-solvers
+ SMT-solvers

Model Checking with SAT?

Motivating Example

3-bit Shift Register

Right Shift

1	**1**	**0**

3-bit Shift Register

Right Shift

0	**1**	**1**	**0**

3-bit Shift Register

Right Shift

***x*[2]**	***x*[1]**	***x*[0]**	
0	**1**	**1**	**0**

Store register state in vector ***x***

3-bit Shift Register

Right Shift

***x*[2]**	***x*[1]**	***x*[0]**	
0	**1**	**1**	**0**

Store register state in vector ***x***

Next state ***x′***

3-bit Shift Register

Right Shift

$x[2]$	$x[1]$	$x[0]$	
0	**1**	**1**	**0**

Store register state in vector $\boldsymbol{x}$

Next state $\boldsymbol{x}'$

Transition relation

$$x'[0] = x[1] \land x'[1] = x[2] \land x'[2] = 0$$

3-bit Shift Register

Right Shift

$x[2]$	$x[1]$	$x[0]$	
0	1	1	0

Store register state in vector x

Next state x'

Transition relation

$$x'[0] = x[1] \wedge x'[1] = x[2] \wedge x'[2] = \overset{1}{\cancel{0}}$$

Error in design

3-bit Shift Register

Right Shift

$x[2]$	$x[1]$	$x[0]$	
0	**1**	**1**	**0**

Store register state in vector x

Property: F(x = 0)

Next state x'

Transition relation

$$x'[0] = x[1] \wedge x'[1] = x[2] \wedge x'[2] = \overset{1}{\cancel{0}}$$

Error in design

Strategy

- try to find error/bug
- prove that $\neg\mathrm{F}(\boldsymbol{x} = \mathbf{0})$ holds
  $\mathrm{G}(\boldsymbol{x} \neq \mathbf{0})$ for some path
- look for path with $\leq \boldsymbol{k}$ different states

Paths with less than 3 States

One transition to encode:

$$\boldsymbol{T(x, x')} \stackrel{def}{=} \boldsymbol{x'[0] = x[1] \wedge x'[1] = x[2] \wedge x'[2] = 1}$$

Paths with less than 3 States

One transition to encode:

$$T(x, x') \stackrel{def}{=} x'[0] = x[1] \wedge x'[1] = x[2] \wedge x'[2] = 1$$

3 vectors: $x_0,\ x_1,\ x_2$

Paths with less than 3 States

One transition to encode:

$$T(x, x') \overset{def}{=} x'[0] = x[1] \wedge x'[1] = x[2] \wedge x'[2] = 1$$

3 vectors: x_0, x_1, x_2

Paths of length $\leq$ **2**

$$T(x_0, x_1) \wedge T(x_1, x_2)$$

Paths with less than 3 States

One transition to encode:

$$T(x, x') \stackrel{def}{=} x'[0] = x[1] \wedge x'[1] = x[2] \wedge x'[2] = 1$$

3 vectors: x_0, x_1, x_2

Paths of length $\leq$ **2**

$$T(x_0, x_1) \wedge T(x_1, x_2)$$

$x_0[0]$
$x_0[1]$
$x_0[2]$

Paths with less than 3 States

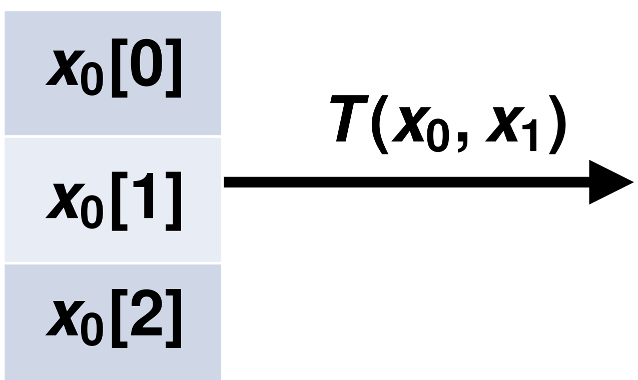
One transition to encode:

$$T(x, x') \stackrel{def}{=} x'[0] = x[1] \wedge x'[1] = x[2] \wedge x'[2] = 1$$

3 vectors: x_0, x_1, x_2

Paths of length $\leq$ **2**

$$T(x_0, x_1) \wedge T(x_1, x_2)$$



Paths with less than 3 States

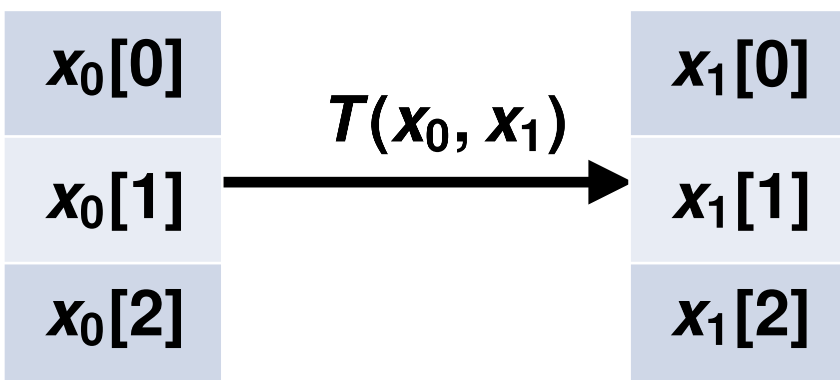
One transition to encode:

$$T(x, x') \overset{def}{=} x'[0] = x[1] \land x'[1] = x[2] \land x'[2] = 1$$

3 vectors: $x_0,\ x_1,\ x_2$

Paths of length $\leq$ **2**

$$T(x_0, x_1) \land T(x_1, x_2)$$



Paths with less than 3 States

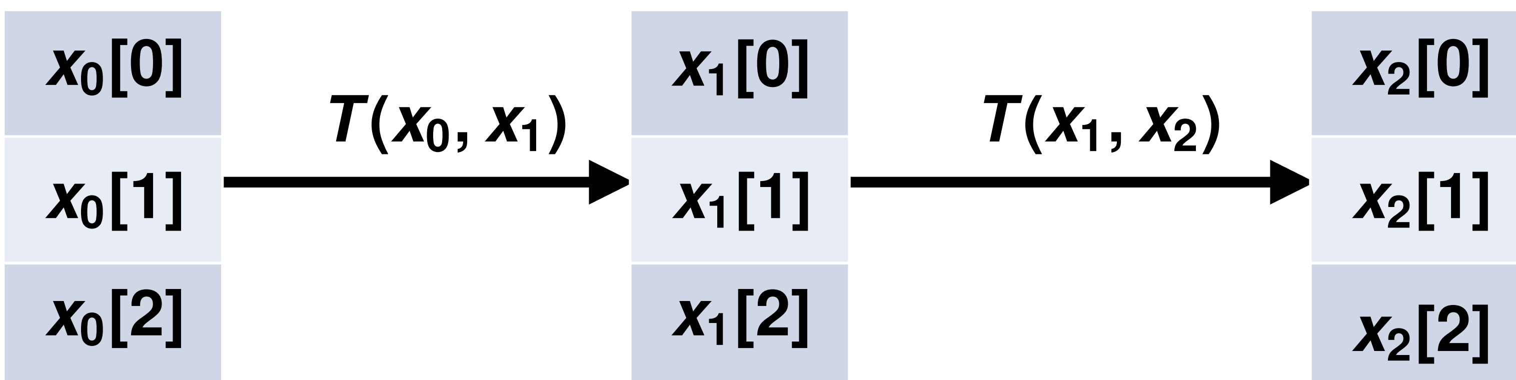
One transition to encode:

$$T(x, x') \stackrel{def}{=} x'[0] = x[1] \wedge x'[1] = x[2] \wedge x'[2] = 1$$

3 vectors: $x_0,\ x_1,\ x_2$

Paths of length $\leq$ **2**

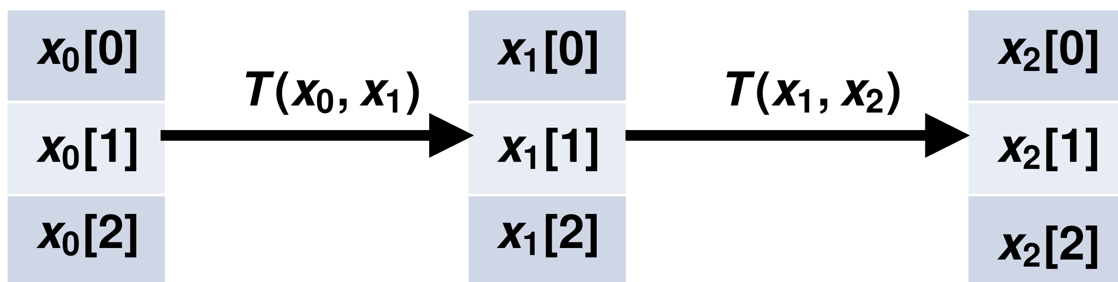
$$T(x_0, x_1) \wedge T(x_1, x_2)$$



Infinite Paths

$T(x, x') \stackrel{def}{=} x'[0] = x[1] \wedge x'[1] = x[2] \wedge x'[2] = 1$

Goal: find a witness path for $\mathbf{G}(x \neq 0)$

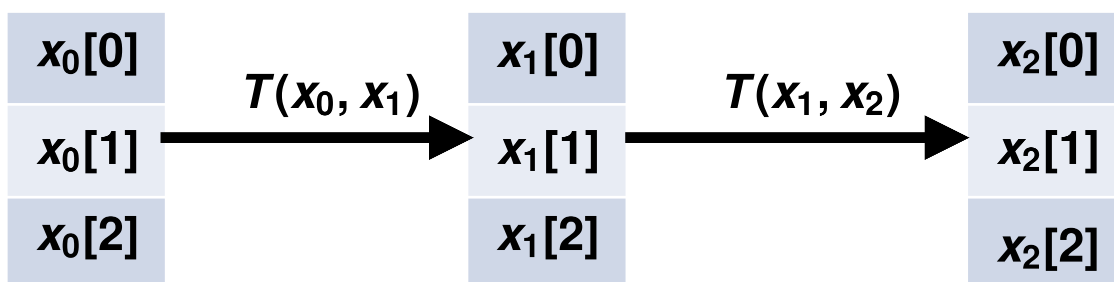


Infinite Paths

$$T(x, x') \stackrel{def}{=} x'[0] = x[1] \wedge x'[1] = x[2] \wedge x'[2] = 1$$

Goal: find a witness path for $\mathbf{G}(x \neq 0)$

Witness path should be infinite

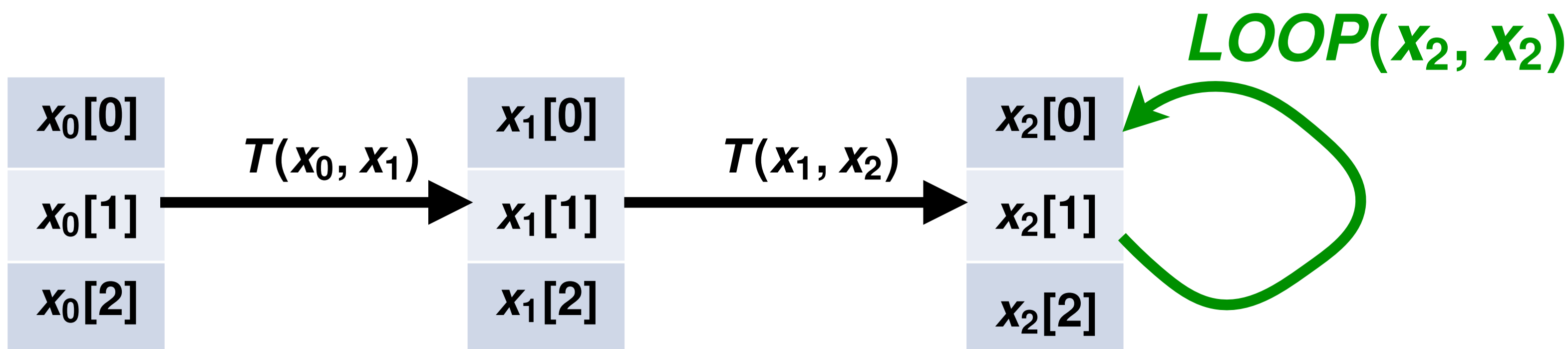


Infinite Paths

$T(x, x') \stackrel{def}{=} x'[0] = x[1] \land x'[1] = x[2] \land x'[2] = 1$

Goal: find a witness path for $\mathbf{G}(x \neq 0)$

Witness path should be infinite

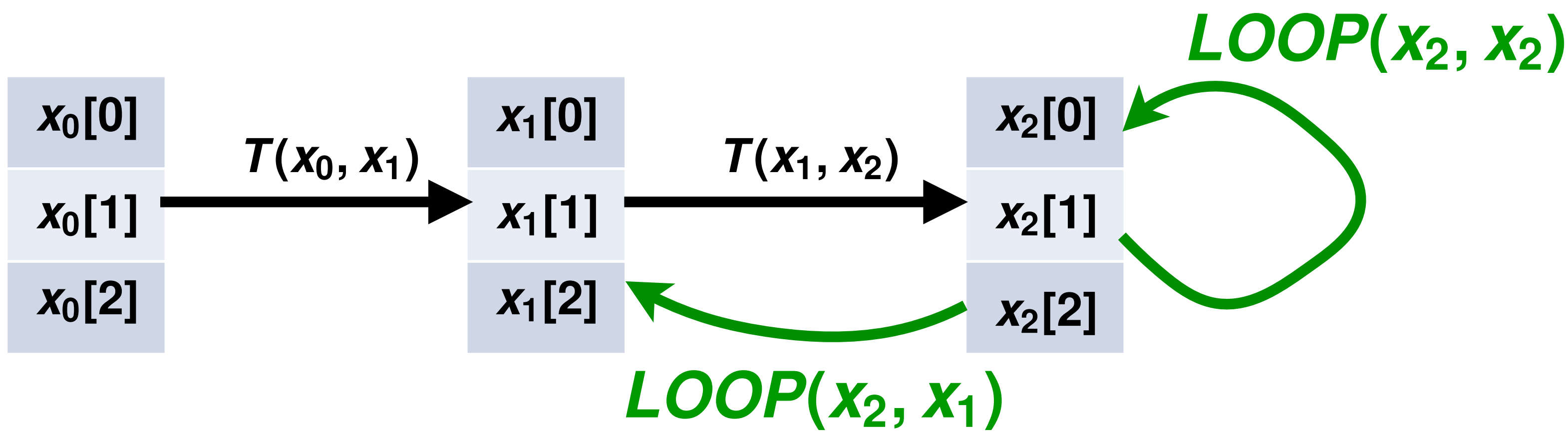


Infinite Paths

$$T(x, x') \stackrel{def}{=} x'[0] = x[1] \wedge x'[1] = x[2] \wedge x'[2] = 1$$

Goal: find a witness path for $\mathbf{G}(x \neq 0)$

Witness path should be infinite

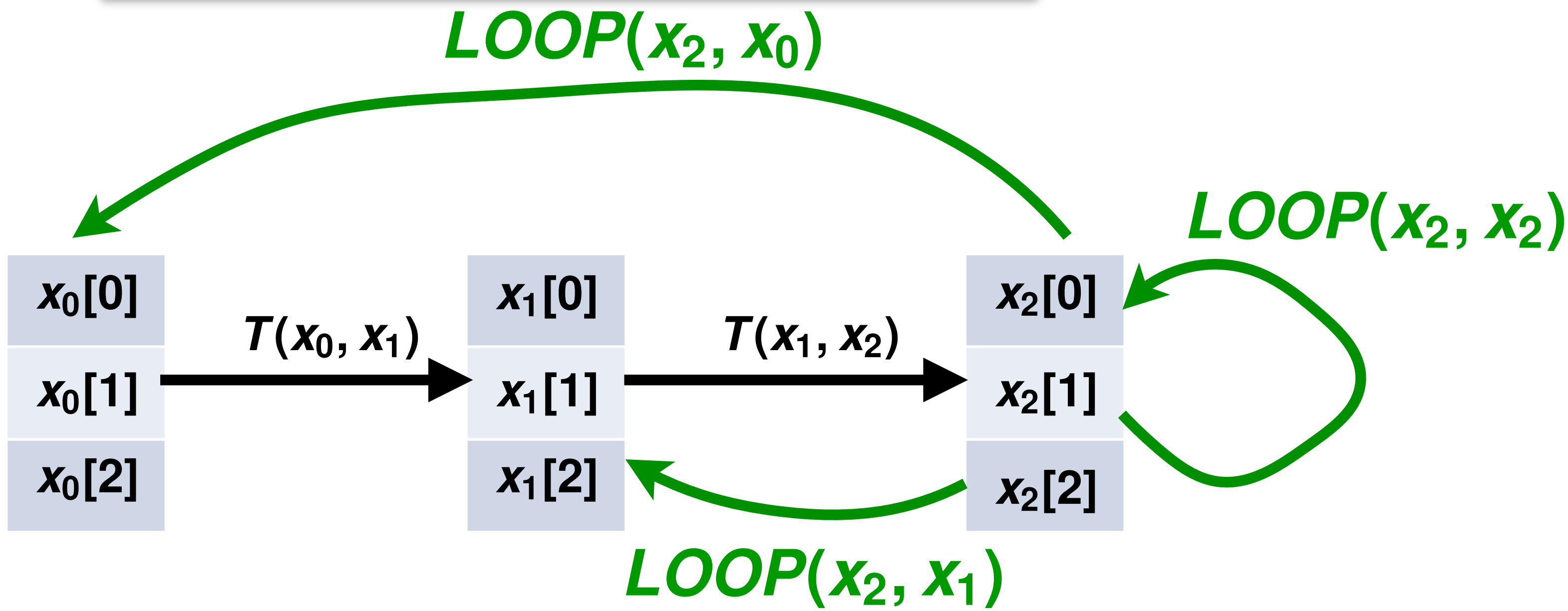


Infinite Paths

$T(x, x') \stackrel{def}{=} x'[0] = x[1] \wedge x'[1] = x[2] \wedge x'[2] = 1$

Goal: find a witness path for $\mathbf{G}(x \neq 0)$

Witness path should be infinite

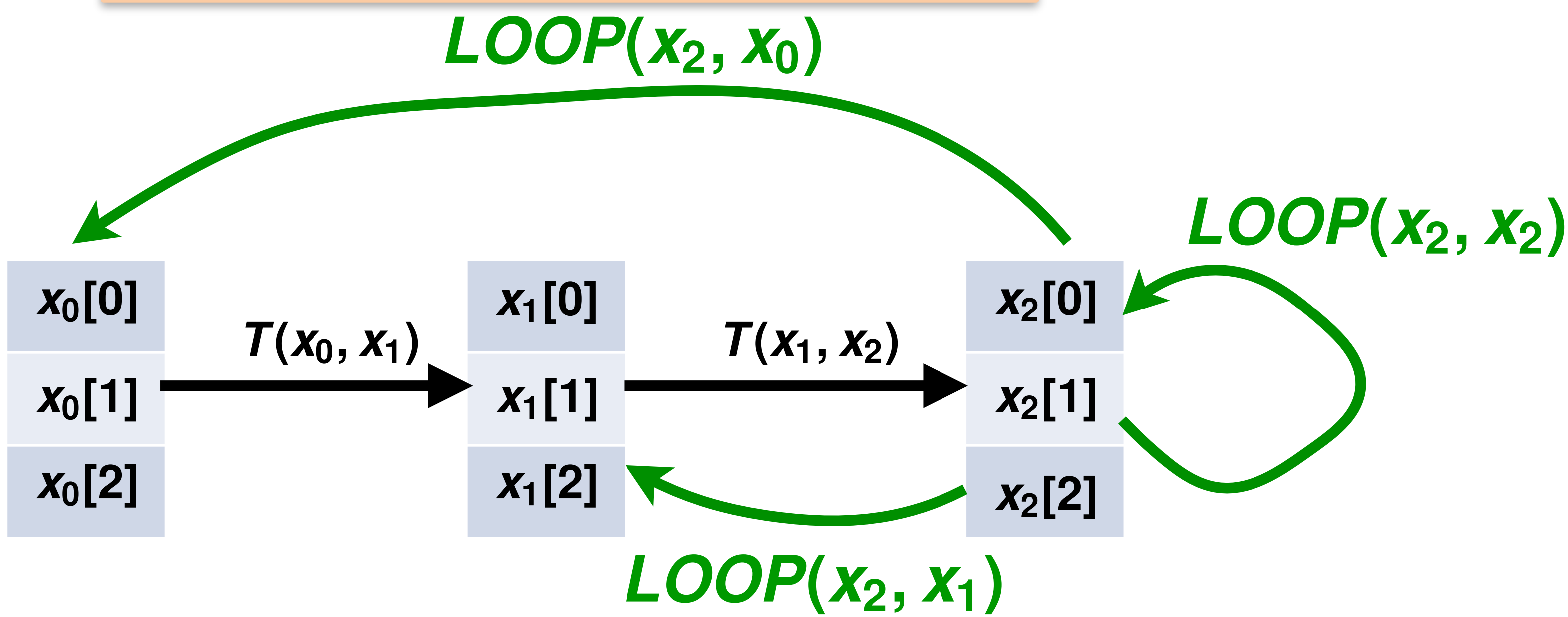


Infinite Paths

$T(x, x') \stackrel{def}{=} x'[0] = x[1] \wedge x'[1] = x[2] \wedge x'[2] = 1$

Goal: find a witness path for $\mathbf{G}(x \neq 0)$

Witness path should be infinite



$LOOP(x_2, x_i) = x_i[0] = x_2[1] \wedge x_i[1] = x_2[2] \wedge x_i[2] = 1$

Encoding of 3-State Infinite Paths

$$T(x, x') \stackrel{def}{=} x'[0] = x[1] \wedge x'[1] = x[2] \wedge x'[2] = 1$$

Encoding of 3-State Infinite Paths

$\boldsymbol{T(x, x') \stackrel{def}{=} x'[0] = x[1] \wedge x'[1] = x[2] \wedge x'[2] = 1}$

First two transition: $\boldsymbol{T(x_0, x_1) \wedge T(x_1, x_2)}$

Encoding of 3-State Infinite Paths

$$T(x, x') \stackrel{def}{=} x'[0] = x[1] \wedge x'[1] = x[2] \wedge x'[2] = 1$$

First two transition: $T(x_0, x_1) \wedge T(x_1, x_2)$

Loop transitions:

$$LOOP(x_2, x_i) = x_i[0] = x_2[1] \wedge x_i[1] = x_2[2] \wedge x_i[2] = 1$$

Encoding of 3-State Infinite Paths

$T(x, x') \stackrel{def}{=} x'[0] = x[1] \wedge x'[1] = x[2] \wedge x'[2] = 1$

First two transition: $T(x_0, x_1) \wedge T(x_1, x_2)$

Loop transitions:

$$LOOP(x_2, x_i) = x_i[0] = x_2[1] \wedge x_i[1] = x_2[2] \wedge x_i[2] = 1$$

Initial States: $I(x_0) = True$

Encoding of 3-State Infinite Paths

$T(x, x') \stackrel{def}{=} x'[0] = x[1] \wedge x'[1] = x[2] \wedge x'[2] = 1$

First two transition: $T(x_0, x_1) \wedge T(x_1, x_2)$

Loop transitions:

$$LOOP(x_2, x_i) = x_i[0] = x_2[1] \wedge x_i[1] = x_2[2] \wedge x_i[2] = 1$$

Initial States: $I(x_0) = True$

Paths with less than **3** states:

$$I(x_0) \wedge T(x_0, x_1) \wedge T(x_1, x_2) \wedge \left(\bigvee_{i=0}^{2} LOOP(x_2, x_i)\right)$$

Encoding of 3-State Infinite Paths

$T(x, x') \stackrel{def}{=} x'[0] = x[1] \wedge x'[1] = x[2] \wedge x'[2] = 1$

First two transition: $T(x_0, x_1) \wedge T(x_1, x_2)$

Loop transitions:

$$LOOP(x_2, x_i) = x_i[0] = x_2[1] \wedge x_i[1] = x_2[2] \wedge x_i[2] = 1$$

Initial States: $I(x_0) = True$

Paths with less than **3** states:

$$I(x_0) \wedge T(x_0, x_1) \wedge T(x_1, x_2) \wedge \left(\bigvee_{i=0}^{2} LOOP(x_2, x_i)\right)$$

Exercise 1

$$\varphi \stackrel{def}{=} I(x_0) \wedge T(x_0, x_1) \wedge T(x_1, x_2) \wedge \left(\bigvee_{i=0}^{2} LOOP(x_2, x_i)\right)$$

- Is the following path ϱ allowed by φ?

  $\varrho = x_0\ x_1\ x_2\ x_1\ x_2\ x_0\ x_1\ x_2 \cdots$

- What are the infinite paths specified by φ?

Encoding of G($\boldsymbol{x} \neq \mathbf{0}$)

$$x \neq 0 \iff \boxed{x[0] = 1 \lor x[1] = 1 \lor x[2] = 1}$$

Encoding of G($\boldsymbol{x \neq 0}$)

$$x \neq 0 \iff \underbrace{\boxed{x[0] = 1 \lor x[1] = 1 \lor x[2] = 1}}_{\psi(x)}$$

Encoding of G($\boldsymbol{x} \neq \mathbf{0}$)

$$\boldsymbol{x} \neq \mathbf{0} \iff \underbrace{\boxed{\boldsymbol{x}[0] = 1 \vee \boldsymbol{x}[1] = 1 \vee \boldsymbol{x}[2] = 1}}_{\psi(\boldsymbol{x})}$$

$$\boxed{\mathrm{G}(\boldsymbol{x} \neq \mathbf{0}) \stackrel{def}{=} \bigwedge_{i=0}^{2} \psi(\boldsymbol{x}_i)}$$

Encoding of $G(x \neq 0)$

$$x \neq 0 \iff \underbrace{\boxed{x[0] = 1 \lor x[1] = 1 \lor x[2] = 1}}_{\psi(x)}$$

$$\boxed{G(x \neq 0) \stackrel{def}{=} \bigwedge_{i=0}^{2} \psi(x_i)}$$

Paths satisfying $G(x \neq 0)$

$$\boxed{I(x_0) \land T(x_0, x_1) \land T(x_1, x_2) \land \left(\bigvee_{i=0}^{2} LOOP(x_2, x_i)\right) \land \bigwedge_{i=0}^{2} \psi(x_i)}$$

Reduction to SAT

Reduction to SAT

There exists an infinite path with less than **3** states that satisfy G$(\boldsymbol{x} \neq \mathbf{0})$

Reduction to SAT

There **exists** an **infinite path** with less than **3** states that satisfy $G(\boldsymbol{x} \neq \mathbf{0})$

$$\Longleftrightarrow$$

Reduction to SAT

There exists an infinite path with less than **3** states that satisfy G($\boldsymbol{x} \neq \mathbf{0}$)

$$\Longleftrightarrow$$

$\exists \boldsymbol{x}_0, \boldsymbol{x}_1, \boldsymbol{x}_2$ such that

$$\boldsymbol{I}(\boldsymbol{x}_0) \wedge \boldsymbol{T}(\boldsymbol{x}_0, \boldsymbol{x}_1) \wedge \boldsymbol{T}(\boldsymbol{x}_1, \boldsymbol{x}_2) \wedge \left(\bigvee_{i=0}^{2} \boldsymbol{LOOP}(\boldsymbol{x}_2, \boldsymbol{x}_i)\right) \wedge \bigwedge_{i=0}^{2} \psi(\boldsymbol{x}_i)$$

is SATisfiable

Reduction to SAT

There **exists** an **infinite path** with less than **3** states that satisfy $G(x \neq 0)$

$$\Longleftrightarrow$$

$\exists x_0, x_1, x_2$ such that

$$\varphi \quad I(x_0) \wedge T(x_0, x_1) \wedge T(x_1, x_2) \wedge \left(\bigvee_{i=0}^{2} LOOP(x_2, x_i)\right) \wedge \bigwedge_{i=0}^{2} \psi(x_i)$$

is SATisfiable

Reduction to SAT

There exists an infinite path with less than **3** states that satisfy $\mathrm{G}(\boldsymbol{x} \neq \mathbf{0})$

$\Longleftrightarrow$

Run SAT-solver on φ: SAT!

$\exists \boldsymbol{x}_0, \boldsymbol{x}_1, \boldsymbol{x}_2$ such that

$$\varphi \quad \boldsymbol{I}(\boldsymbol{x}_0) \wedge \boldsymbol{T}(\boldsymbol{x}_0, \boldsymbol{x}_1) \wedge \boldsymbol{T}(\boldsymbol{x}_1, \boldsymbol{x}_2) \wedge \left(\bigvee_{i=0}^{2} \boldsymbol{LOOP}(\boldsymbol{x}_2, \boldsymbol{x}_i)\right) \wedge \bigwedge_{i=0}^{2} \psi(\boldsymbol{x}_i)$$

is SATisfiable

Bounded LTL Semantics

Existential Model Checking Problem

LTL model checking problem

Given $\boldsymbol{A}$, labeled automaton, $\boldsymbol{\Phi}$ LTL formula

$$\boldsymbol{A} \models \boldsymbol{\Phi} \iff \forall \rho \in \boldsymbol{Runs(A)},\ \rho \models \boldsymbol{\Phi}$$

Existential Model Checking Problem

LTL model checking problem

Given $\boldsymbol{A}$, labeled automaton, $\boldsymbol{\Phi}$ LTL formula

$$\boldsymbol{A} \models \boldsymbol{\Phi} \iff \forall \rho \in \boldsymbol{Runs(A)}, \rho \models \boldsymbol{\Phi}$$

LTL Universal Validity

$$\boldsymbol{A} \models \mathrm{A}\boldsymbol{\Phi}$$

$$\iff$$

$$\forall \rho \in \boldsymbol{Runs(A)}, \rho \models \boldsymbol{\Phi}$$

Existential Model Checking Problem

LTL model checking problem

Given $\boldsymbol{A}$, labeled automaton, $\boldsymbol{\Phi}$ LTL formula

$$\boldsymbol{A} \models \boldsymbol{\Phi} \iff \forall \rho \in \boldsymbol{Runs}(\boldsymbol{A}),\ \rho \models \boldsymbol{\Phi}$$

LTL Universal Validity

$$\boldsymbol{A} \models \mathrm{A}\boldsymbol{\Phi}$$
$$\iff$$
$$\forall \rho \in \boldsymbol{Runs}(\boldsymbol{A}),\ \rho \models \boldsymbol{\Phi}$$

LTL Existential Validity

$$\boldsymbol{A} \models \mathrm{E}\boldsymbol{\Phi}$$
$$\iff$$
$$\exists \rho \in \boldsymbol{Runs}(\boldsymbol{A}),\ \rho \models \boldsymbol{\Phi}$$

Existential Model Checking Problem

LTL model checking problem

Given $\boldsymbol{A}$, labeled automaton, $\boldsymbol{\Phi}$ LTL formula

$$\boldsymbol{A} \models \boldsymbol{\Phi} \iff \forall \rho \in \boldsymbol{Runs}(\boldsymbol{A}),\ \rho \models \boldsymbol{\Phi}$$

LTL Universal Validity

$$\boldsymbol{A} \models \mathrm{A}\boldsymbol{\Phi}$$
$$\iff$$
$$\forall \rho \in \boldsymbol{Runs}(\boldsymbol{A}),\ \rho \models \boldsymbol{\Phi}$$

LTL Existential Validity

$$\boldsymbol{A} \models \mathrm{E}\boldsymbol{\Phi}$$
$$\iff$$
$$\exists \rho \in \boldsymbol{Runs}(\boldsymbol{A}),\ \rho \models \boldsymbol{\Phi}$$

$$\boldsymbol{A} \models \mathrm{A}\boldsymbol{\Phi} \iff \boldsymbol{A} \not\models \mathrm{E}\neg\boldsymbol{\Phi}$$

LTL Formulas – Semantics

Given a run $\rho = s_0 \cdots s_n \cdots$

Notation: $\rho[i] = s_i \cdots s_n \cdots, \; i \geq 0$

P propositional formula, φ_1, φ_2 LTL formulas

- $\rho \models P \iff s_0 \models P$
- Next: $\rho \models X\varphi_1 \iff \rho[1] \models \varphi_1$
- Until: $\rho \models \varphi_1 \text{ UNTIL } \varphi_2 \iff \begin{cases} \exists k \geq 0, \rho[k] \models \varphi_2 \\ \forall 0 \leq j < k, \rho[j] \models \varphi_1 \end{cases}$
- Eventually: $\rho \models \mathbf{F}\,\varphi_1 \iff \exists k \geq 0, \rho[k] \models \varphi_1$
- Globally: $\rho \models \mathbf{G}\,\varphi_1 \iff \forall k \geq 0, \rho[k] \models \varphi_1$

Negation Normal Form

Negation Normal Form = $\neg$ only allowed on atomic propositions

Negation Normal Form

Negation Normal Form = $\neg$ only allowed on atomic propositions

Rewrite rules

$$\neg(\varphi_1 \wedge \varphi_2) \equiv (\neg\varphi_1) \vee (\neg\varphi_2)$$

$$\neg \mathrm{F}\varphi \equiv \mathrm{G}\neg\varphi$$

$$\neg \mathrm{X}\varphi \equiv \mathrm{X}\neg\varphi$$

$$\neg \mathrm{G}\varphi \equiv \mathrm{F}\neg\varphi$$

Negation Normal Form

Negation Normal Form = $\neg$ only allowed on atomic propositions

Rewrite rules

$$\neg(\varphi_1 \wedge \varphi_2) \equiv (\neg\varphi_1) \vee (\neg\varphi_2) \qquad \neg F\varphi \equiv G\neg\varphi$$

$$\neg X\varphi \equiv X\neg\varphi \qquad \neg G\varphi \equiv F\neg\varphi$$

From now on: Every LTL formula in NNF

- $\neg$ only atomic propositions
- $\vee$ allowed (semantics easy to write)

Bounded Semantics

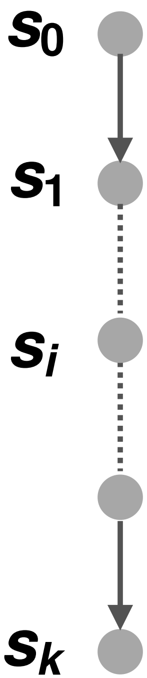
Subset of all paths in $\boldsymbol{A}$

$\leq \boldsymbol{k+1}$ states

Bounded Semantics

Subset of all paths in $\boldsymbol{A}$

$\leq \boldsymbol{k+1}$ states

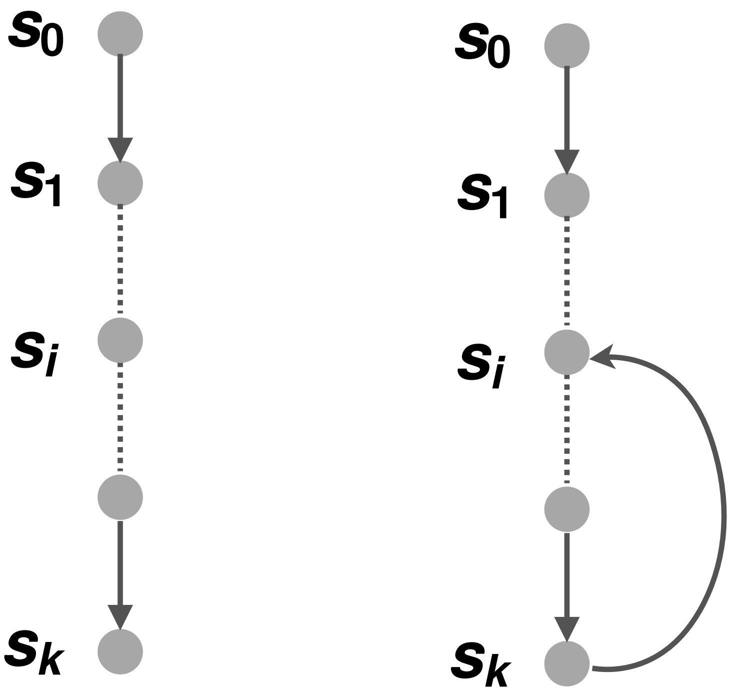


Finite paths

Bounded Semantics

Subset of all paths in $\boldsymbol{A}$

$\leq$ $\boldsymbol{k}$ **+ 1** states



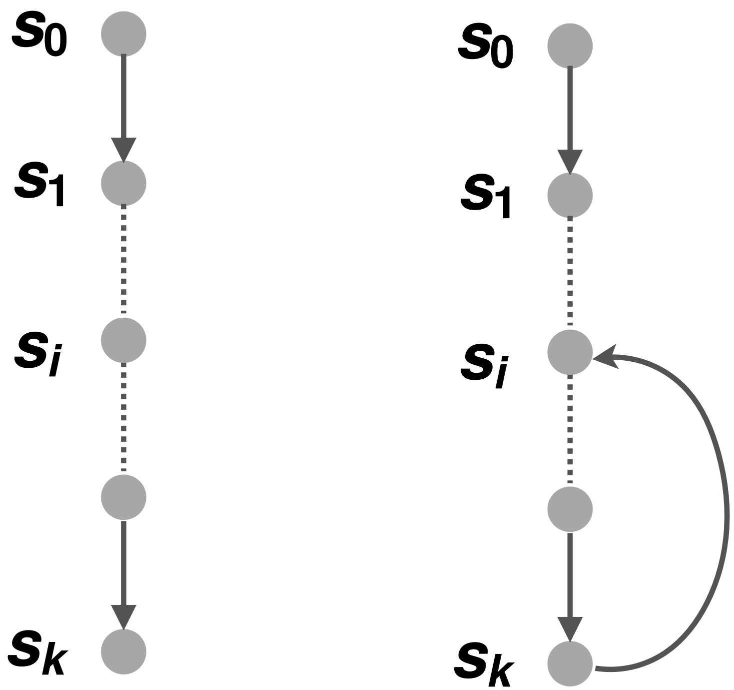
(k, i)-loop

Finite paths Lasso infinite paths

Bounded Semantics

Subset of all paths in $\boldsymbol{A}$

$\leq \boldsymbol{k+1}$ states



(*k*, *i*)-loop

Finite paths   Lasso infinite paths

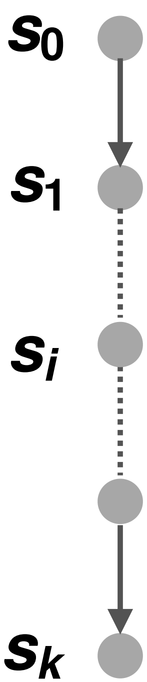
Bounded Semantics of LTL

$$\rho \models_k \varphi \implies \rho \models \varphi$$

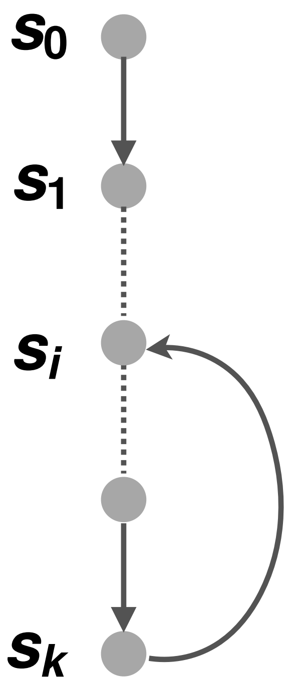
Bounded Semantics

Subset of all paths in $\boldsymbol{A}$

$\leq k + 1$ states



Finite paths



(k, i)-loop

Lasso infinite paths

Bounded Semantics of LTL

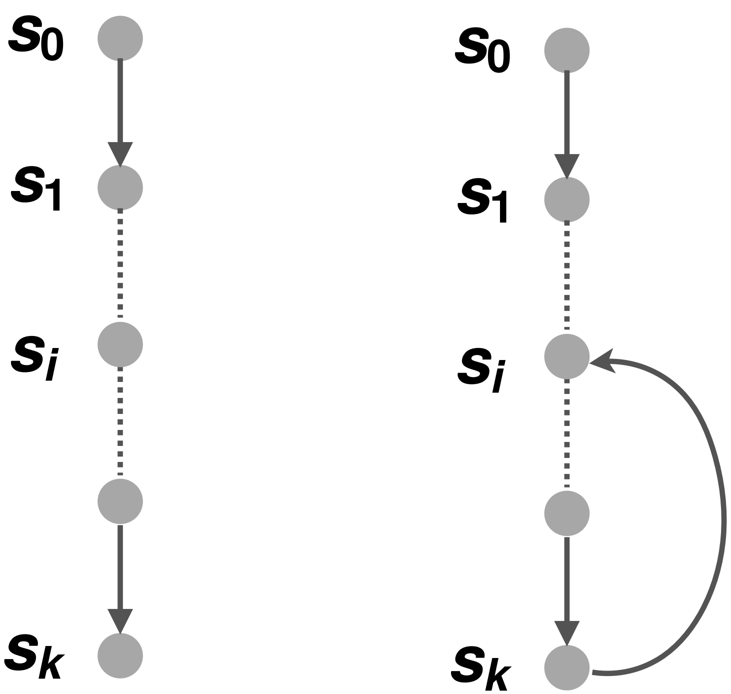
$$\rho \models_k \varphi \implies \rho \models \varphi$$

Soundness

Bounded Semantics

Subset of all paths in A

$\leq k + 1$ states



(k, i)-loop

Finite paths Lasso infinite paths

Bounded Semantics of LTL

$$\rho \models_k \varphi \implies \rho \models \varphi$$

Soundness

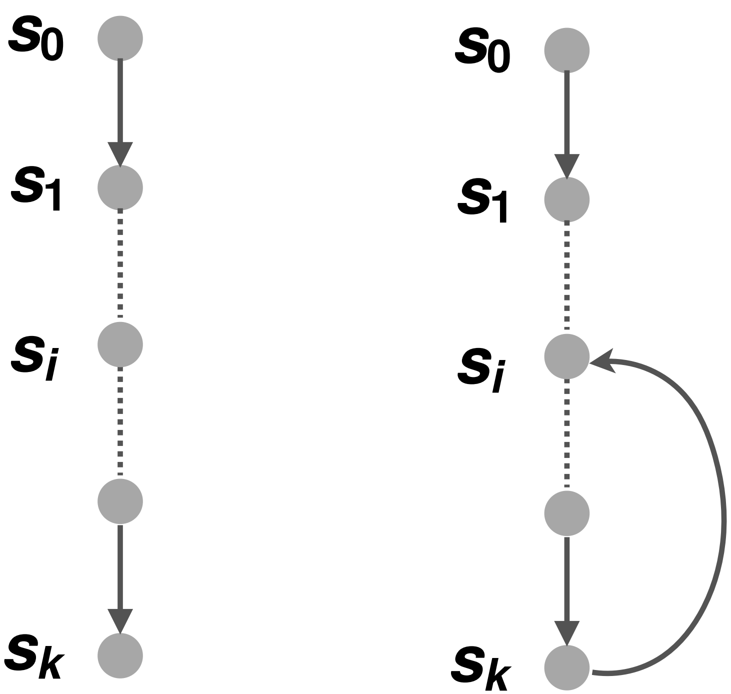
k-loop path ρ

$\exists l$ ρ is a (k, l)-loop

Bounded Semantics

Subset of all paths in ***A***

$\leq$ ***k* + 1** states



(***k*, *i***)-loop

Finite paths Lasso infinite paths

Bounded Semantics of LTL

$$\rho \models_k \varphi \implies \rho \models \varphi$$

Soundness

k-loop path ρ

$\exists$***l*** ρ is a (***k*, *l***)-loop

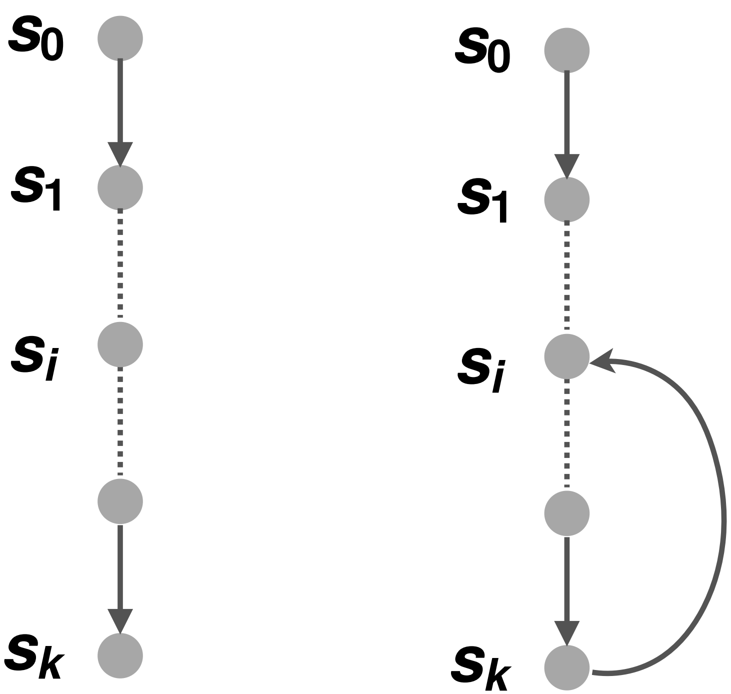
ρ is a ***k***-loop

$$\rho \models_k \varphi \iff \rho \models \varphi$$

Bounded Semantics

Subset of all paths in A

$\leq k + 1$ states



(k, i)-loop

Finite paths   Lasso infinite paths

Bounded Semantics of LTL

$$\rho \models_k \varphi \implies \rho \models \varphi$$

Soundness

k-loop path ρ

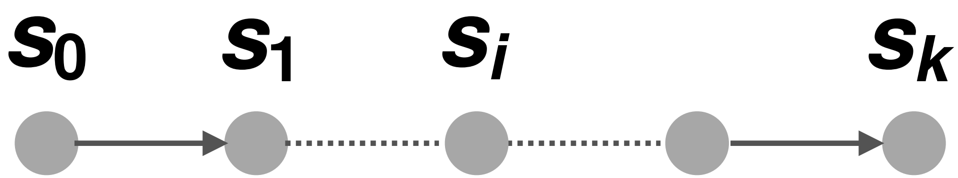
$\exists l\ \rho$ is a (k, l)-loop

ρ is a k-loop

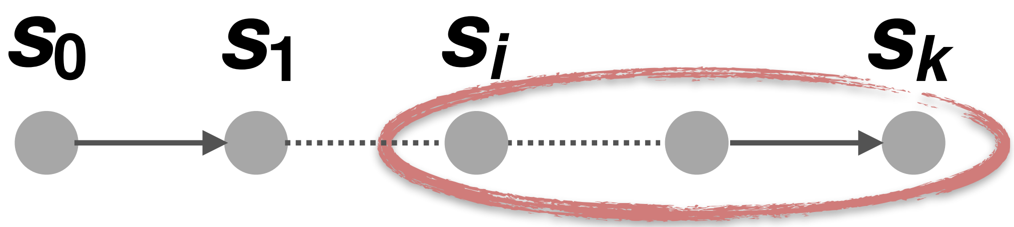
$$\rho \models_k \varphi \iff \rho \models \varphi$$

What about finite paths?

Bounded Length Paths – 1

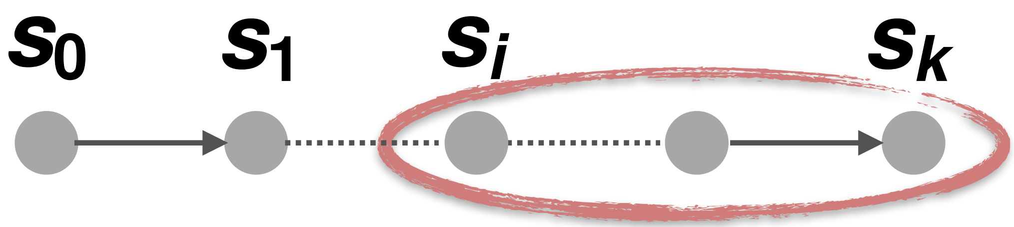


Bounded Length Paths – 1



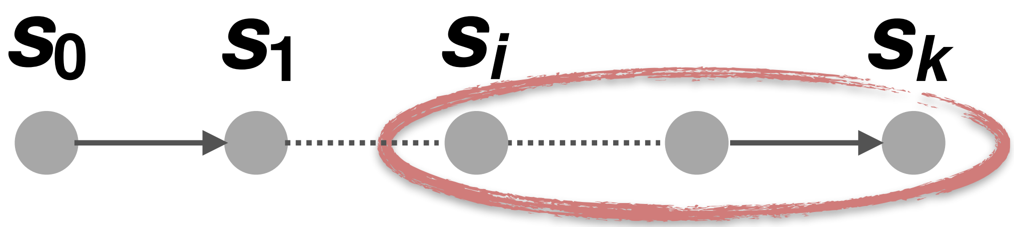
$0 \leq i \leq k, \models^i_k$ satisfaction relation on subpath $s_i \cdots s_k$

Bounded Length Paths – 1



$0 \leq i \leq k$, $\models^i_k$ satisfaction relation on subpath $s_i \cdots s_k$

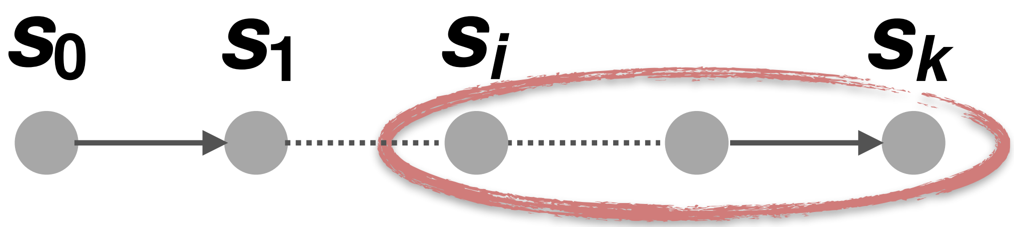
Bounded Length Paths – 1



$0 \leq i \leq k$, $\models^i_k$ satisfaction relation on subpath $s_i \cdots s_k$

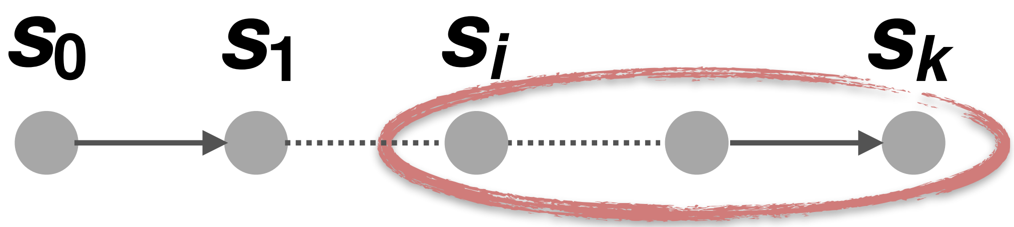
- $P(s)$ propositional formula: $\begin{cases} \rho \models^i_k P(s) \iff P(s_i) \\ \rho \models^i_k \neg P(s) \iff \neg P(s_i) \end{cases}$
- $\rho \models^i_k \varphi_1 \wedge \varphi_2 \iff \rho \models^i_k \varphi_1$ and $\rho \models^i_k \varphi_2$
- $\rho \models^i_k \mathrm{X}\varphi \iff 0 \leq i < k$ and $\rho \models^{i+1}_k \varphi_1$

Bounded Length Paths – 2



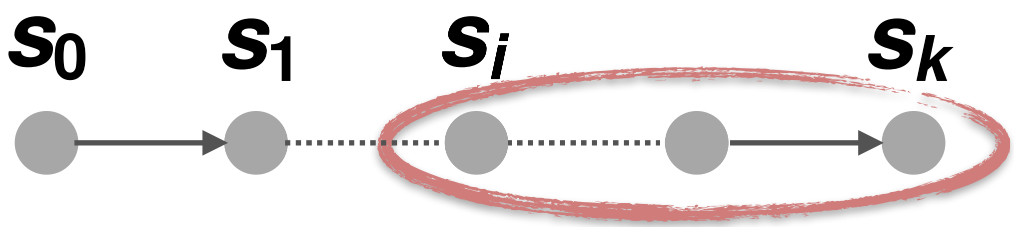
$0 \leq i \leq k, \models^i_k$ satisfaction relation on subpath $s_i \cdots s_k$

Bounded Length Paths – 2



$0 \leq i \leq k, \models^i_k$ satisfaction relation on subpath $s_i \cdots s_k$

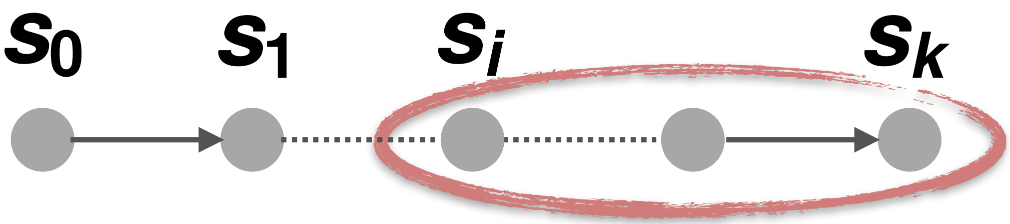
Bounded Length Paths – 2



$0 \leq i \leq k, \models^i_k$ satisfaction relation on subpath $s_i \cdots s_k$

- $\rho \models^i_k \varphi_1$ UNTIL $\varphi_2 \iff \begin{cases} \rho \models^i_k \exists i \leq j \leq k, \rho \models^j_k \varphi_2 \\ \forall i \leq n < j, \rho \models^n_k \varphi_1 \end{cases}$
- $\rho \models^i_k \mathrm{F}\,\varphi \iff \exists i \leq j \leq k, \rho \models^j_k \varphi$
- $\rho \models^i_k \mathrm{G}\,\varphi \iff \textit{False}$

Bounded Length Paths – 2



$\rho \models_k \varphi \iff \rho \models_k^0 \varphi$

$0 \leq i \leq k, \models_k^i$ satisfaction relation on subpath $s_i \cdots s_k$

- $\rho \models_k^i \varphi_1 \text{ UNTIL } \varphi_2 \iff \begin{cases} \rho \models_k^i \exists i \leq j \leq k, \rho \models_k^j \varphi_2 \\ \forall i \leq n < j, \rho \models_k^n \varphi_1 \end{cases}$
- $\rho \models_k^i \mathrm{F}\, \varphi \iff \exists i \leq j \leq k, \rho \models_k^j \varphi$
- $\rho \models_k^i \mathrm{G}\, \varphi \iff \textit{False}$

Exercise 2

LTL semantics

$$\rho \models \neg \mathrm{F} \neg \varphi \iff \rho \models \mathrm{G} \varphi$$

LTL bounded semantics

Is it true that $\rho \models_k \neg \mathrm{F} \neg \varphi \iff \rho \models_k \mathrm{G} \varphi$?

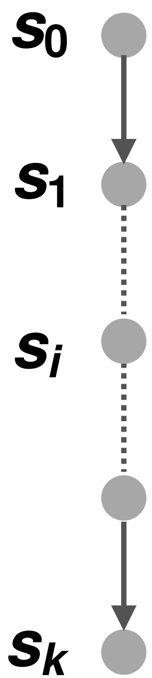
Bounded Semantics to SAT

Objective

$$\boldsymbol{A} \models_{\boldsymbol{k}} \mathrm{E}\varphi \iff \boldsymbol{For}(\boldsymbol{A}) \wedge \boldsymbol{For}(\varphi)_{\boldsymbol{k}} \text{ is SAT}$$

Objective

$$\boldsymbol{A} \models_k \mathrm{E}\varphi \iff \boldsymbol{For(A)} \wedge \boldsymbol{For(\varphi)_k} \text{ is SAT}$$



Finite paths

Objective

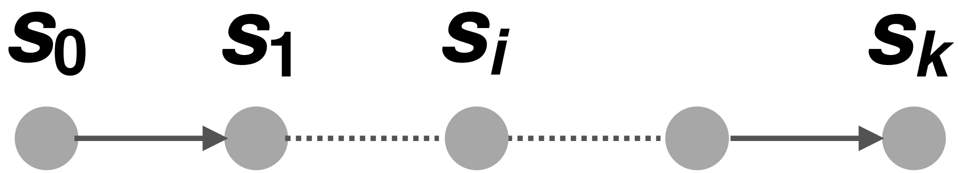
$$\boldsymbol{A} \models_k \mathrm{E}\varphi \iff \boldsymbol{For}(\boldsymbol{A}) \wedge \boldsymbol{For}(\varphi)_k \text{ is SAT}$$

$(\boldsymbol{k}, \boldsymbol{i})$-loop

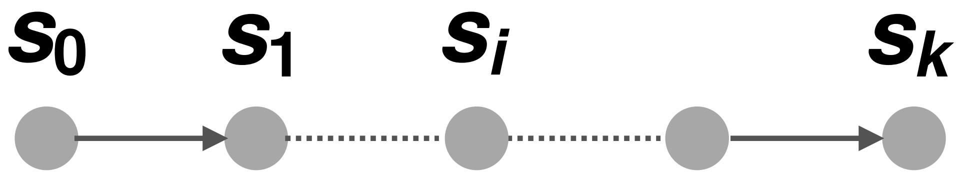
Finite paths

Lasso infinite paths

Finite Paths – 1

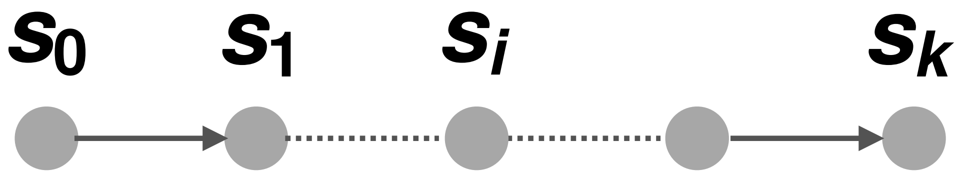


Finite Paths – 1



k fixed $\Longrightarrow$ ***For***(φ)

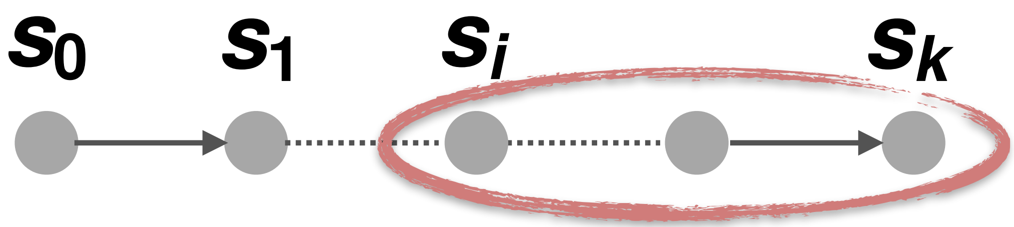
Finite Paths – 1



k fixed $\Longrightarrow$ $\boldsymbol{For}(\varphi)$

$\rho \models_k \varphi \iff \boldsymbol{For}(\varphi)$ is SAT

Finite Paths – 1

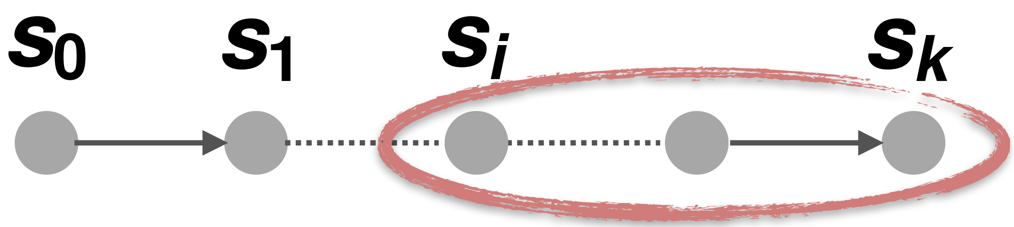


$\boldsymbol{k}$ fixed $\Longrightarrow$ $\boldsymbol{For}(\varphi)$

$\rho \models_k \varphi \iff \boldsymbol{For}(\varphi)$ is SAT

$\rho[i] \models_k \varphi \iff \boldsymbol{For}(\varphi)_i$ is SAT

Finite Paths – 1

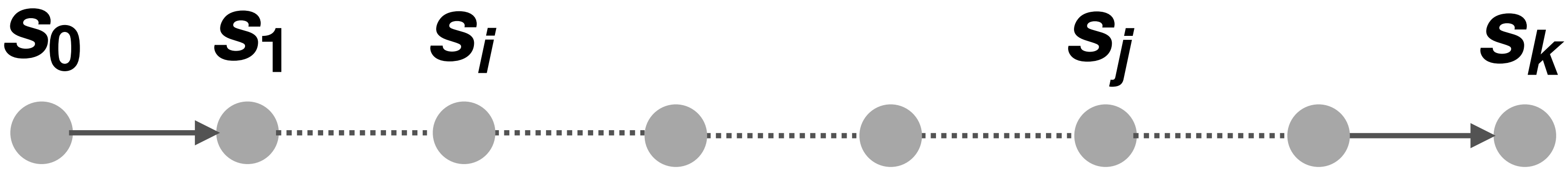


k fixed $\Longrightarrow$ $\mathit{For}(\varphi)$

$\rho \models_k \varphi \iff \mathit{For}(\varphi)$ is SAT $\qquad \rho[i] \models_k \varphi \iff \mathit{For}(\varphi)_i$ is SAT

- $P(s)$ propositional formula: $\mathit{For}(P(s))_i = P(s_i)$
- $\mathit{For}(\varphi_1 \wedge \varphi_2)_i = \mathit{For}(\varphi_1)_i \wedge \mathit{For}(\varphi_2)_i$
- $\mathit{For}(\mathrm{X}\varphi)_i =$ If $0 \leq i < k$ then $\mathit{For}(\varphi)_{i+1}$ else False
- $\mathit{For}(\mathrm{G}\varphi)_i = \mathit{False}$

Finite Paths – 2

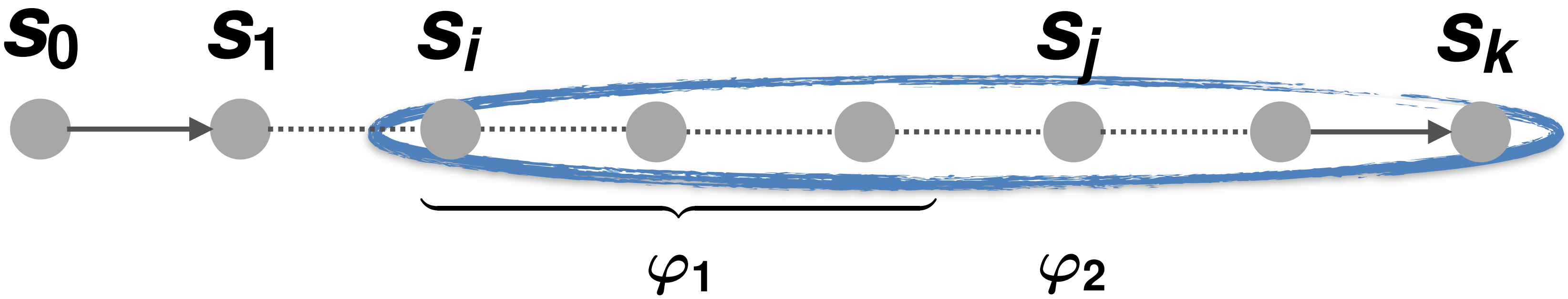


$\boldsymbol{k}$ fixed $\Longrightarrow$ $\boldsymbol{For}(\varphi)$

$\rho \models_k \varphi \iff \boldsymbol{For}(\varphi)$ is SAT $\qquad$ $\rho[i] \models_k \varphi \iff \boldsymbol{For}(\varphi)_i$ is SAT

- $\boldsymbol{For}(\mathrm{F}\,\varphi)_i = \bigvee_{j=i}^{k} \boldsymbol{For}(\varphi)_j$

Finite Paths – 2



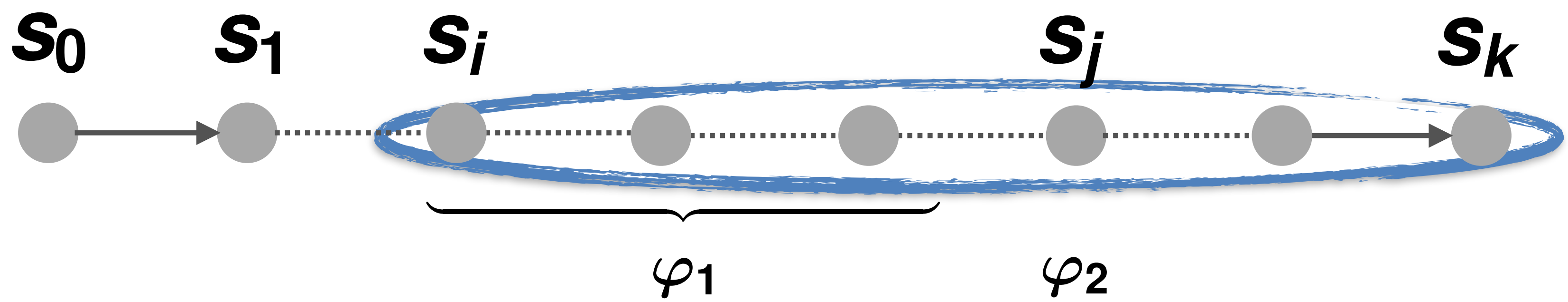
k fixed $\Longrightarrow$ $For(\varphi)$

$\rho \models_k \varphi \iff For(\varphi)$ is SAT $\quad\quad \rho[i] \models_k \varphi \iff For(\varphi)_i$ is SAT

- $For(\mathrm{F}\,\varphi)_i = \bigvee_{j=i}^{k} For(\varphi)_j$

φ_1 **Until** φ_2

Finite Paths – 2

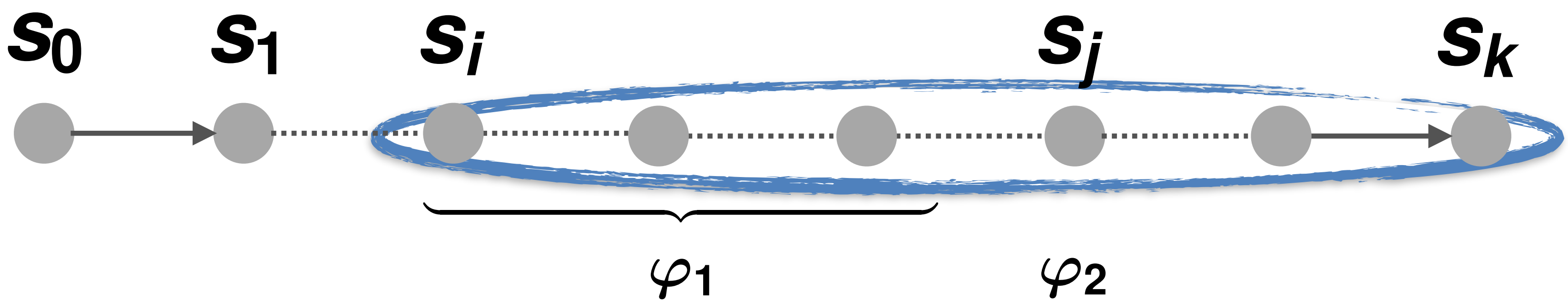


k fixed $\Longrightarrow$ $For(\varphi)$

$\rho \models_k \varphi \iff For(\varphi)$ is SAT $\qquad \rho[i] \models_k \varphi \iff For(\varphi)_i$ is SAT

- $For(\mathrm{F}\,\varphi)_i = \bigvee_{j=i}^{k} For(\varphi)_j$
- $For(\varphi_1 \text{ Until } \varphi_2)_i = \bigvee_{j=i}^{k} \left(For(\varphi_2)_j \wedge \bigwedge_{n=i}^{j-1} For(\varphi_1)_n\right)$

Finite Paths – 2



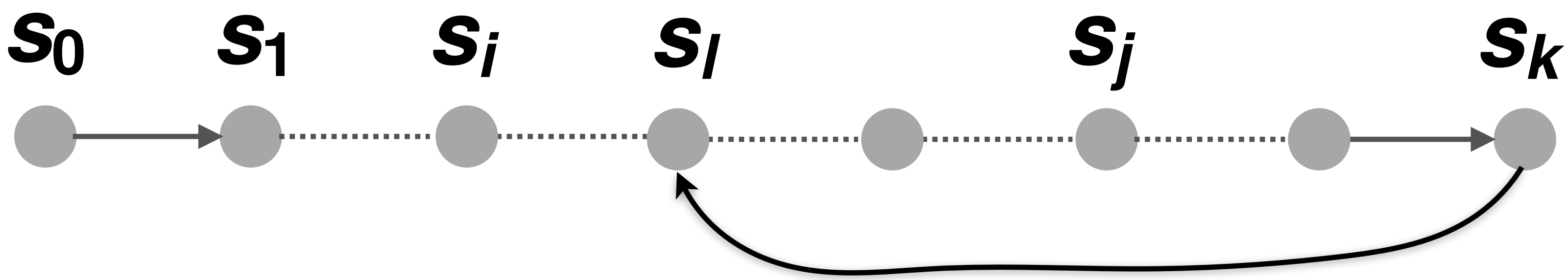
k fixed $\Longrightarrow$ $\mathit{For}(\varphi)$

$\rho \models_k \varphi \iff \mathit{For}(\varphi)$ is SAT $\qquad \rho[i] \models_k \varphi \iff \mathit{For}(\varphi)_i$ is SAT

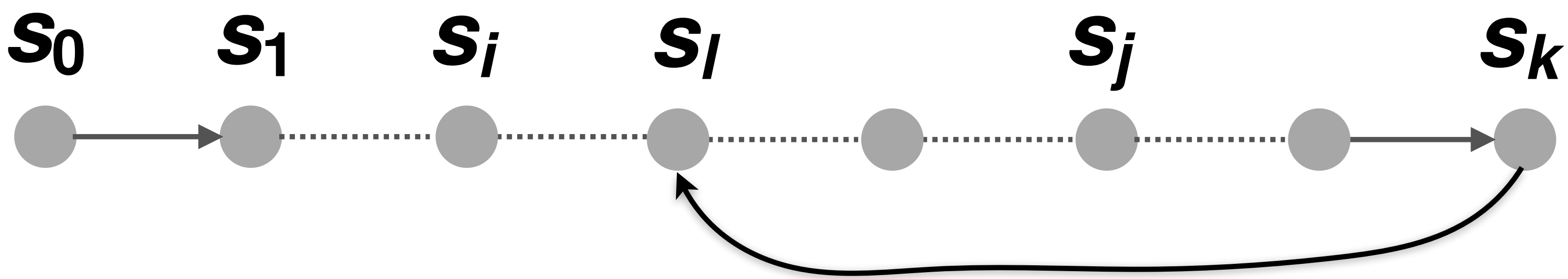
- $\mathit{For}(\mathrm{F}\,\varphi)_i = \bigvee_{j=i}^{k} \mathit{For}(\varphi)_j$
- $\mathit{For}(\varphi_1 \text{ Until } \varphi_2)_i = \bigvee_{j=i}^{k} \left(\mathit{For}(\varphi_2)_j \wedge \bigwedge_{n=i}^{j-1} \mathit{For}(\varphi_1)_n\right)$

$$\boxed{\mathit{For}(\varphi) = \mathit{For}(\varphi)_0}$$

Infinite Paths – 1

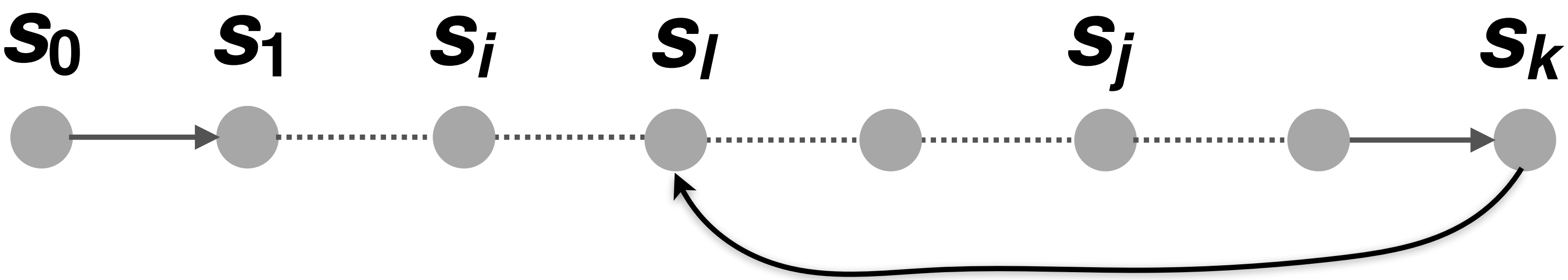


Infinite Paths – 1



$\rho[i] \models_k \varphi \iff \mathit{For}(\varphi)_i^l$ is SAT

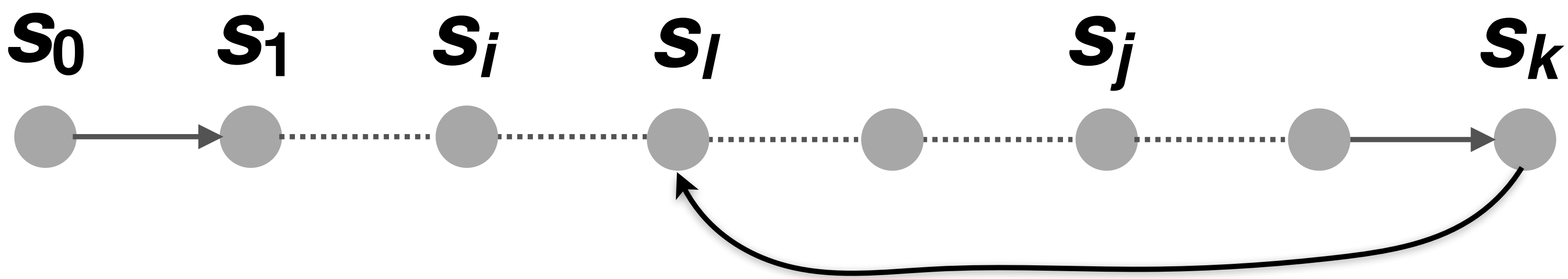
Infinite Paths – 1



$\rho = s_0 \cdots s_{l-1}(s_l \cdots s_k)^\omega$

$\rho[i] \models_k \varphi \iff For(\varphi)_i^l$ is SAT

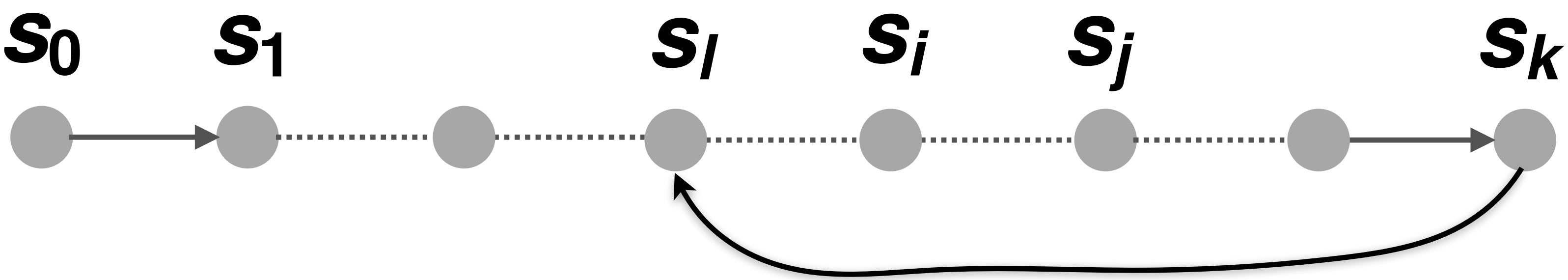
Infinite Paths – 1



$\rho = s_0 \cdots s_{l-1}(s_l \cdots s_k)^\omega$ $\qquad \rho[i] \models_k \varphi \iff For(\varphi)_i^l$ is SAT

- $P(s)$ propositional formula: $For(P(s))_i^l = P(s_i)$
- $For(\varphi_1 \wedge \varphi_2)_i^l = For(\varphi_1)_i^l \wedge For(\varphi_2)_i^l$
- $For(\mathrm{G}\varphi)_i^l = \bigwedge_{j=min(i,l)}^{k} For(\varphi)_j^l$
- $For(\mathrm{F}\varphi)_i^l = \bigvee_{j=min(i,l)}^{k} For(\varphi)_j^l$

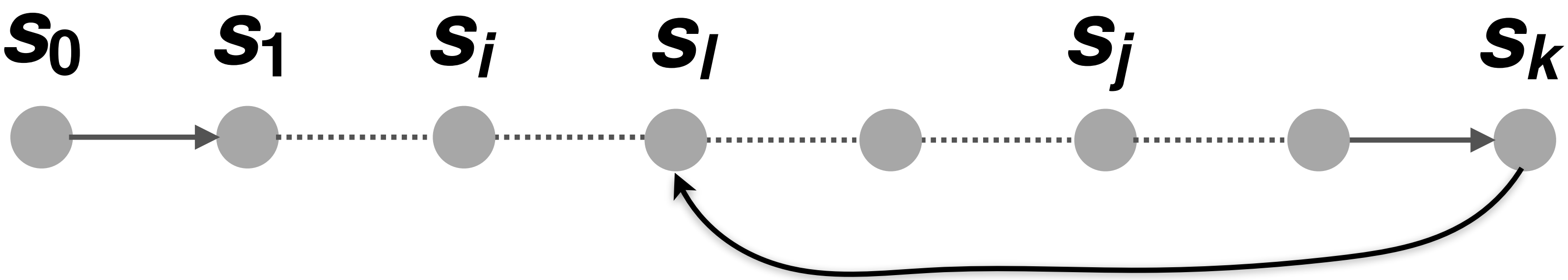
Infinite Paths – 1



$\rho = s_0 \cdots s_{l-1}(s_l \cdots s_k)^\omega$ $\qquad$ $\rho[i] \models_k \varphi \iff For(\varphi)^l_i$ is SAT

- $P(s)$ propositional formula: $For(P(s))^l_i = P(s_i)$
- $For(\varphi_1 \wedge \varphi_2)^l_i = For(\varphi_1)^l_i \wedge For(\varphi_2)^l_i$
- $For(\mathrm{G}\varphi)^l_i = \bigwedge^k_{j=min(i,l)} For(\varphi)^l_j$
- $For(\mathrm{F}\varphi)^l_i = \bigvee^k_{j=min(i,l)} For(\varphi)^l_j$

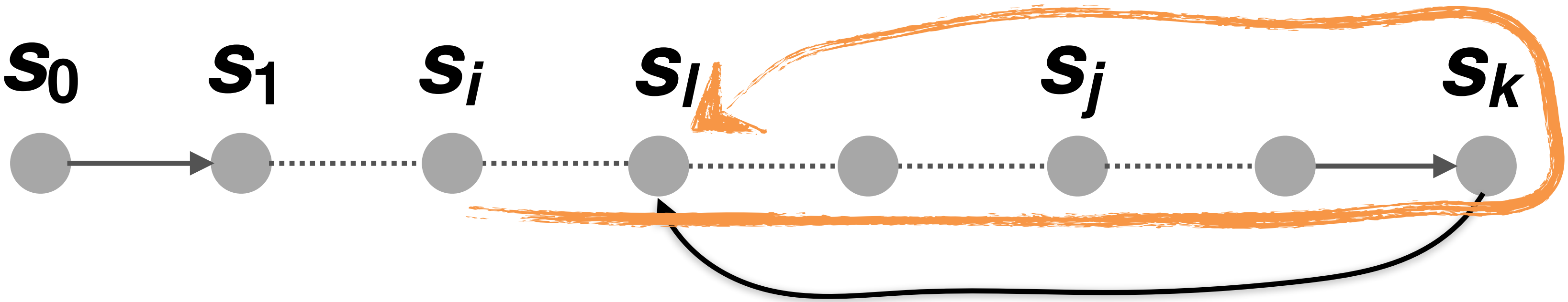
Infinite Paths – 2



$\rho = s_0 \cdots s_{l-1}(s_l \cdots s_k)^\omega$ $\rho[i] \models_k \varphi \iff For(\varphi)^l_i$ is SAT

- $For(\mathrm{X}\,\varphi)^l_i = For(\varphi)^l_{\mathrm{succ}(i)}$
- $For(\varphi_1 \text{ Until } \varphi_2)^l_i =$

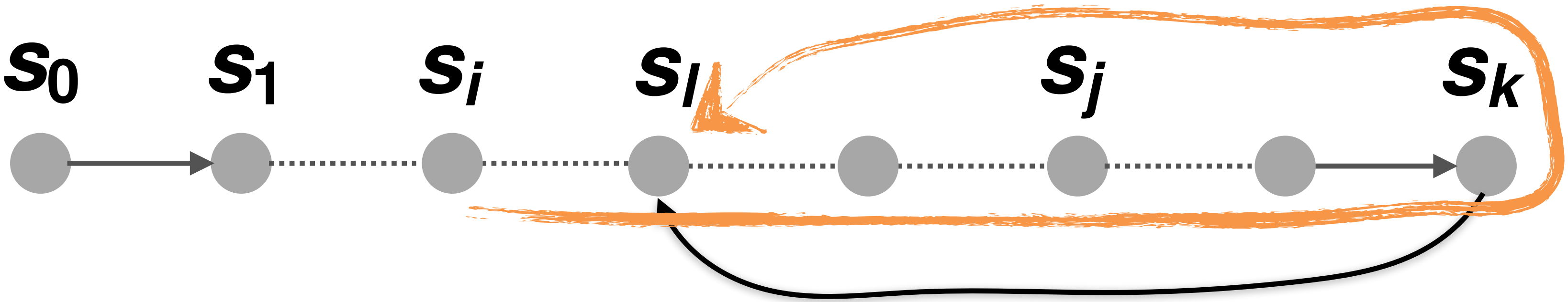
Infinite Paths – 2



$\rho = s_0 \cdots s_{l-1}(s_l \cdots s_k)^\omega$ $\qquad$ $\rho[i] \models_k \varphi \iff \mathit{For}(\varphi)_i^l$ is SAT

- $\mathit{For}(\mathrm{X}\,\varphi)_i^l = \mathit{For}(\varphi)_{\mathrm{succ}(i)}^l$
- $\mathit{For}(\varphi_1 \text{ Until } \varphi_2)_i^l =$

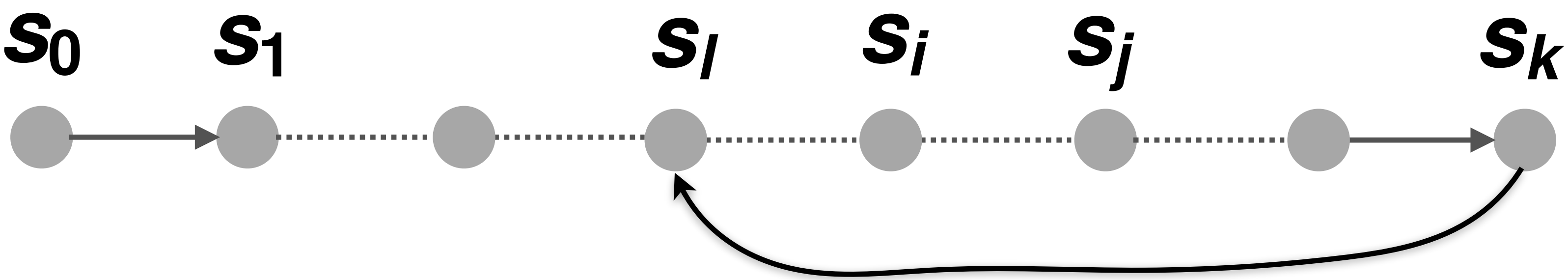
Infinite Paths – 2



$\rho = s_0 \cdots s_{l-1}(s_l \cdots s_k)^\omega$ $\quad\quad$ $\rho[i] \models_k \varphi \iff For(\varphi)^l_i$ is SAT

- $For(\mathrm{X}\,\varphi)^l_i = For(\varphi)^l_{\mathrm{succ}(i)}$
- $For(\varphi_1 \text{ Until } \varphi_2)^l_i = \bigvee_{j=i}^{k} \left(For(\varphi_2)^l_j \wedge \bigwedge_{n=i}^{j-1} For(\varphi_1)^l_n\right)$

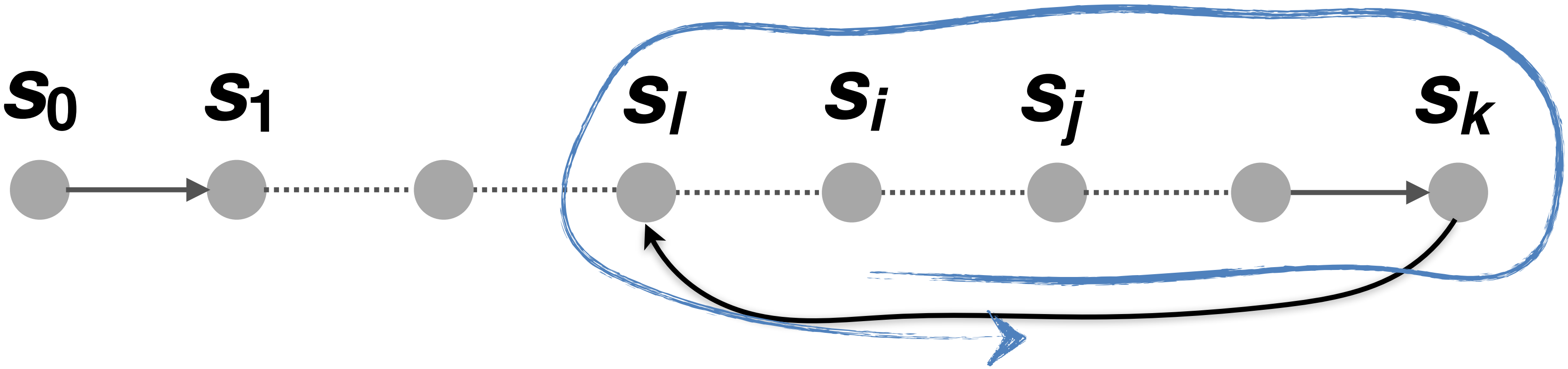
Infinite Paths – 2



$\rho = s_0 \cdots s_{l-1}(s_l \cdots s_k)^\omega$ $\qquad$ $\rho[i] \models_k \varphi \iff For(\varphi)^l_i$ is SAT

- $For(\mathrm{X}\,\varphi)^l_i = For(\varphi)^l_{\mathrm{succ}(i)}$
- $For(\varphi_1 \text{ Until } \varphi_2)^l_i = \bigvee_{j=i}^{k} \left(For(\varphi_2)^l_j \wedge \bigwedge_{n=i}^{j-1} For(\varphi_1)^l_n\right)$

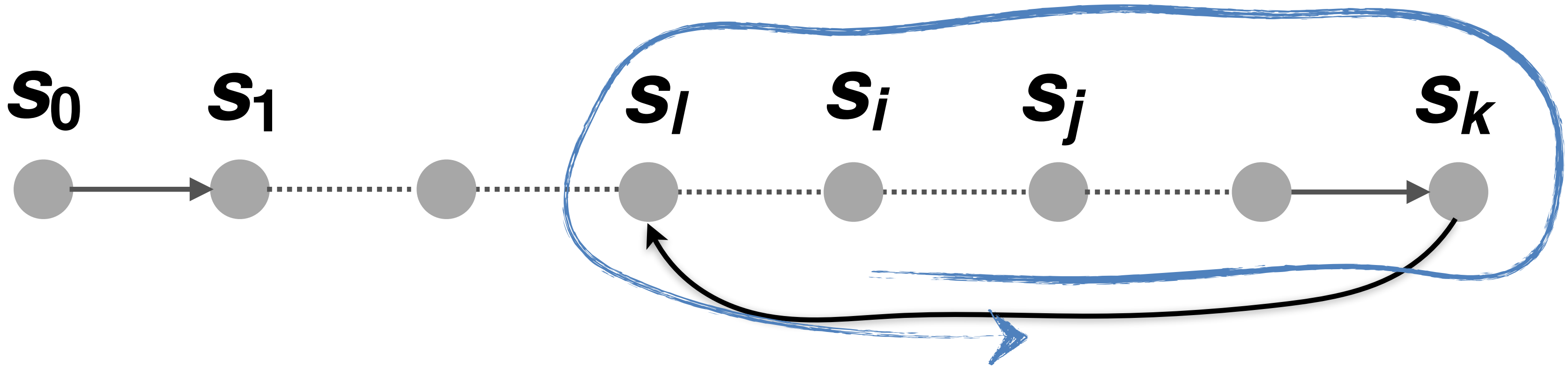
Infinite Paths – 2



$\rho = s_0 \cdots s_{l-1}(s_l \cdots s_k)^\omega$ $\quad\quad$ $\rho[i] \models_k \varphi \iff For(\varphi)^l_i$ is SAT

- $For(\mathrm{X}\,\varphi)^l_i = For(\varphi)^l_{\mathrm{succ}(i)}$
- $For(\varphi_1 \text{ Until } \varphi_2)^l_i = \bigvee_{j=i}^{k} \left(For(\varphi_2)^l_j \wedge \bigwedge_{n=i}^{j-1} For(\varphi_1)^l_n \right)$

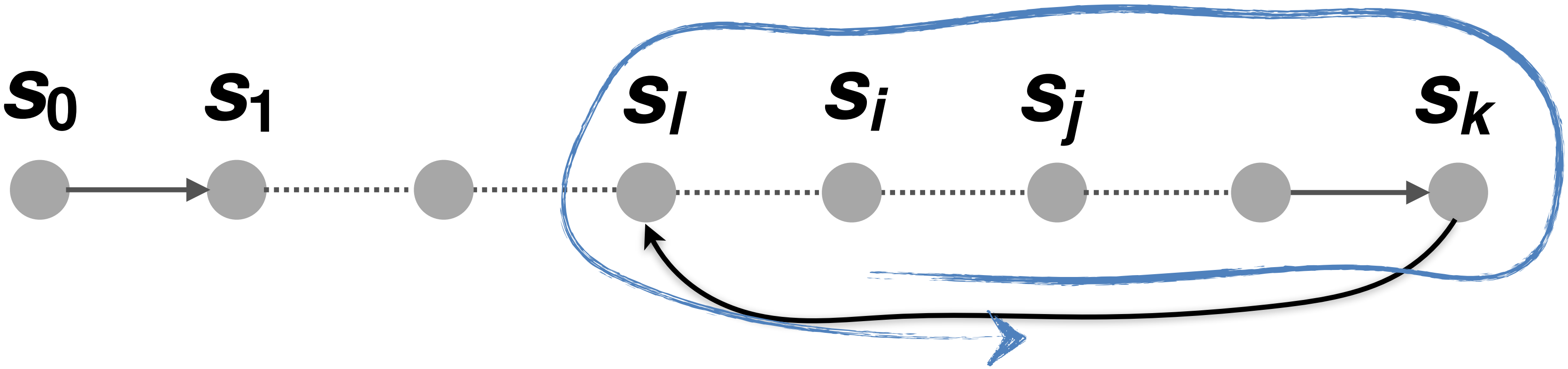
Infinite Paths – 2



$\rho = s_0 \cdots s_{l-1}(s_l \cdots s_k)^\omega$ $\qquad$ $\rho[i] \models_k \varphi \iff For(\varphi)^l_i$ is SAT

- $For(\mathrm{X}\,\varphi)^l_i = For(\varphi)^l_{\mathrm{succ}(i)}$
- $For(\varphi_1 \textbf{ Until } \varphi_2)^l_i = \bigvee_{j=i}^{k} \left(For(\varphi_2)^l_j \wedge \bigwedge_{n=i}^{j-1} For(\varphi_1)^l_n\right)$
  $\vee \bigvee_{j=l}^{i-1} \left(For(\varphi_2)^l_j \wedge \bigwedge_{n=i}^{k} For(\varphi_1)^l_n \wedge \bigwedge_{n=l}^{j-1} For(\varphi_1)^l_n\right)$

Infinite Paths – 2



$\rho = s_0 \cdots s_{l-1}(s_l \cdots s_k)^\omega$ $\qquad$ $\rho[i] \models_k \varphi \iff For(\varphi)^l_i$ is SAT

- $For(\mathrm{X}\,\varphi)^l_i = For(\varphi)^l_{\mathrm{succ}(i)}$
- $For(\varphi_1 \text{ Until } \varphi_2)^l_i = \bigvee_{j=i}^{k} \left(For(\varphi_2)^l_j \wedge \bigwedge_{n=i}^{j-1} For(\varphi_1)^l_n \right)$
$$\vee \bigvee_{j=l}^{i-1} \left(For(\varphi_2)^l_j \wedge \bigwedge_{n=i}^{k} For(\varphi_1)^l_n \wedge \bigwedge_{n=l}^{j-1} For(\varphi_1)^l_n \right)$$

$$\boxed{For(\varphi) = For(\varphi)^l_0}$$

Encoding of LTL Formula

Given $\boldsymbol{A} = (\{\mathbf{0}, \mathbf{1}\}^{n}, \boldsymbol{q}_0, \delta)$

Encoding of LTL Formula

Given $\boldsymbol{A} = (\{0, 1\}^n, q_0, \delta)$

State = vector $(s[0], s[1], \cdots, s[n-1])$

Encoding of LTL Formula

Given $\boldsymbol{A} = (\{0,1\}^n, q_0, \delta)$

State = vector $(s[0], s[1], \cdots, s[n-1])$

Transition relation: $T(s, s') = \bigvee_{(s,s')\in\delta} f_\delta(s, s')$

Encoding of LTL Formula

Given $\boldsymbol{A} = (\{0,1\}^n, \boldsymbol{q_0}, \delta)$

State = vector $(\boldsymbol{s}[0], \boldsymbol{s}[1], \cdots, \boldsymbol{s}[n-1])$

Transition relation: $T(s, s') = \bigvee_{(s,s') \in \delta} f_\delta(s, s')$

$LOOP(k, l) = T(s_k, s_l)$ $\qquad$ $LOOP = \bigvee_{l=0}^{k} LOOP(k, l)$

Encoding of LTL Formula

Given $\boldsymbol{A} = (\{0,1\}^n, \boldsymbol{q_0}, \delta)$

State = vector $(\boldsymbol{s}[0], \boldsymbol{s}[1], \cdots, \boldsymbol{s}[\boldsymbol{n}-1])$

Transition relation: $\boldsymbol{T}(\boldsymbol{s}, \boldsymbol{s}') = \bigvee_{(\boldsymbol{s},\boldsymbol{s}')\in\delta} \boldsymbol{f}_\delta(\boldsymbol{s}, \boldsymbol{s}')$

$\boldsymbol{LOOP}(\boldsymbol{k}, \boldsymbol{l}) = \boldsymbol{T}(\boldsymbol{s_k}, \boldsymbol{s_l})$ $\qquad$ $\boldsymbol{LOOP} = \bigvee_{\boldsymbol{l}=0}^{\boldsymbol{k}} \boldsymbol{LOOP}(\boldsymbol{k}, \boldsymbol{l})$

Encoding of formula φ

Encoding of LTL Formula

Given $\boldsymbol{A} = (\{0,1\}^n, q_0, \delta)$

State = vector $(s[0], s[1], \cdots, s[n-1])$

Transition relation: $T(s, s') = \bigvee_{(s,s') \in \delta} f_\delta(s, s')$

$LOOP(k, l) = T(s_k, s_l)$ $\qquad$ $LOOP = \bigvee_{l=0}^{k} LOOP(k, l)$

Encoding of formula φ

finite paths

$\neg LOOP \wedge For(\varphi)_0$

Encoding of LTL Formula

Given $A = (\{0,1\}^n, q_0, \delta)$

State = vector $(s[0], s[1], \cdots, s[n-1])$

Transition relation: $T(s, s') = \bigvee_{(s,s') \in \delta} f_\delta(s, s')$

$LOOP(k, l) = T(s_k, s_l)$

$LOOP = \bigvee_{l=0}^{k} LOOP(k, l)$

Encoding of formula φ

finite paths

$\neg LOOP \wedge For(\varphi)_0$

infinite paths

$\bigvee_{l=0}^{k} LOOP(k, l) \wedge For(\varphi)_0^l$

Encoding of LTL Formula

Given $A = (\{0,1\}^n, q_0, \delta)$

State = vector $(s[0], s[1], \cdots, s[n-1])$

Transition relation: $T(s, s') = \bigvee_{(s,s')\in\delta} f_\delta(s, s')$

$LOOP(k,l) = T(s_k, s_l)$

$LOOP = \bigvee_{l=0}^{k} LOOP(k,l)$

Encoding of formula φ

$$\underbrace{\overset{\textbf{finite paths}}{\neg LOOP \wedge For(\varphi)_0} \quad \bigvee \quad \overset{\textbf{infinite paths}}{\bigvee_{l=0}^{k} LOOP(k,l) \wedge For(\varphi)_0^l}}_{For(\varphi)_k}$$

Model Encoding

Given $\boldsymbol{A} = (\{0, 1\}^n, q_0, \delta)$

Initial states: formula $\boldsymbol{I(s)}$

Transition relation: $\boldsymbol{T(s, s') = \bigvee_{(s,s')\in\delta} f_\delta(s, s')}$

Model Encoding

Given $A = (\{0, 1\}^n, q_0, \delta)$

Initial states: formula $I(s)$

Transition relation: $T(s, s') = \bigvee_{(s,s') \in \delta} f_\delta(s, s')$

Encoding of paths $\leq k + 1$ states

$$For(A) = I(s_0) \wedge \bigwedge_{i=0}^{k-1} T(s_i, s_{i+1})$$

Model Encoding

Given $\boldsymbol{A} = (\{0, 1\}^n, q_0, \delta)$

Initial states: formula $I(s)$

Transition relation: $T(s, s') = \bigvee_{(s,s') \in \delta} f_\delta(s, s')$

Encoding of paths $\leq k + 1$ states

$$For(A) = I(s_0) \wedge \bigwedge_{i=0}^{k-1} T(s_i, s_{i+1})$$

3 bit shift register

$x[2]$	$x[1]$	$x[0]$
1	1	0

$I(x) = True$

$T(x, x') \stackrel{def}{=} x'[0] = x[1] \wedge x'[1] = x[2] \wedge x'[2] = 1$

Summary

$$\boldsymbol{A} \models_{\boldsymbol{k}} \mathrm{E}\varphi \iff \boldsymbol{For(A)} \wedge \boldsymbol{For(\varphi)_k} \text{ is SAT}$$

$$\boldsymbol{A} \models \mathrm{A}\neg\varphi \iff \forall \boldsymbol{k} \in \mathbb{N}, \boldsymbol{For(A)} \wedge \boldsymbol{For(\varphi)_k} \text{ is UNSAT}$$

Properties of LTL Bounded Semantics

Properties of LTL Bounded Semantics

P_1: $\rho \models_k \varphi \implies \rho \models \varphi$

Properties of LTL Bounded Semantics

P_1: $\rho \models_k \varphi \implies \rho \models \varphi$ By definition of $\models_k$

Properties of LTL Bounded Semantics

$\boldsymbol{P_1}$: $\rho \models_k \varphi \implies \rho \models \varphi$ By definition of $\models_k$

$$\boldsymbol{A} \models_k \mathrm{E}\varphi \iff \exists\rho \in \boldsymbol{Runs}(\boldsymbol{A}) \text{ s.t. } \rho \models_k \varphi$$

$$\boldsymbol{A} \models_k \mathrm{A}\varphi \iff \forall\rho \in \boldsymbol{Runs}(\boldsymbol{A}) \text{ s.t. } \rho \models_k \varphi$$

Properties of LTL Bounded Semantics

$\boldsymbol{P_1}$: $\rho \models_k \varphi \implies \rho \models \varphi$ By definition of $\models_k$

$$\boldsymbol{A} \models_k \mathrm{E}\varphi \iff \exists \rho \in \boldsymbol{Runs}(\boldsymbol{A}) \text{ s.t. } \rho \models_k \varphi$$

$$\boldsymbol{A} \models_k \mathrm{A}\varphi \iff \forall \rho \in \boldsymbol{Runs}(\boldsymbol{A}) \text{ s.t. } \rho \models_k \varphi$$

$\boldsymbol{P_2}$: if $\boldsymbol{A} \models \mathrm{E}\varphi \implies \exists \boldsymbol{k} \in \mathbb{N}, \boldsymbol{A} \models_k \mathrm{E}\varphi$

Properties of LTL Bounded Semantics

P_1: $\rho \models_k \varphi \implies \rho \models \varphi$ By definition of $\models_k$

$$A \models_k \mathrm{E}\varphi \iff \exists \rho \in \mathit{Runs}(A) \text{ s.t. } \rho \models_k \varphi$$

$$A \models_k \mathrm{A}\varphi \iff \forall \rho \in \mathit{Runs}(A) \text{ s.t. } \rho \models_k \varphi$$

P_2: if $A \models \mathrm{E}\varphi \implies \exists k \in \mathbb{N}, A \models_k \mathrm{E}\varphi$??

Properties of LTL Bounded Semantics

$\boldsymbol{P_1}$: $\rho \models_k \varphi \implies \rho \models \varphi$ By definition of $\models_k$

$$\boldsymbol{A} \models_k \mathrm{E}\varphi \iff \exists \rho \in \boldsymbol{Runs}(\boldsymbol{A}) \text{ s.t. } \rho \models_k \varphi$$

$$\boldsymbol{A} \models_k \mathrm{A}\varphi \iff \forall \rho \in \boldsymbol{Runs}(\boldsymbol{A}) \text{ s.t. } \rho \models_k \varphi$$

$\boldsymbol{P_2}$: if $\boldsymbol{A} \models \mathrm{E}\varphi \implies \exists \boldsymbol{k} \in \mathbb{N}, \boldsymbol{A} \models_k \mathrm{E}\varphi$??

$\boldsymbol{P_3}$: $\boldsymbol{A} \models \mathrm{E}\varphi \iff \exists \boldsymbol{k} \in \mathbb{N}, \boldsymbol{A} \models_k \mathrm{E}\varphi$

Properties of LTL Bounded Semantics

$\boldsymbol{P_1}$: $\rho \models_k \varphi \implies \rho \models \varphi$ By definition of $\models_k$

$$\boldsymbol{A} \models_k \mathrm{E}\varphi \iff \exists \rho \in \boldsymbol{Runs}(\boldsymbol{A}) \text{ s.t. } \rho \models_k \varphi$$
$$\boldsymbol{A} \models_k \mathrm{A}\varphi \iff \forall \rho \in \boldsymbol{Runs}(\boldsymbol{A}) \text{ s.t. } \rho \models_k \varphi$$

$\boldsymbol{P_2}$: if $\boldsymbol{A} \models \mathrm{E}\varphi \implies \exists \boldsymbol{k} \in \mathbb{N}, \boldsymbol{A} \models_k \mathrm{E}\varphi$??

$\boldsymbol{P_3}$: $\boldsymbol{A} \models \mathrm{E}\varphi \iff \exists \boldsymbol{k} \in \mathbb{N}, \boldsymbol{A} \models_k \mathrm{E}\varphi$ $\boldsymbol{P_1} \wedge \boldsymbol{P_2} \implies \boldsymbol{P_3}$

Completeness for Existential Model Checking

$\boldsymbol{P_2}$: if $\boldsymbol{A} \models \mathsf{E}\varphi \implies \exists \boldsymbol{k} \in \mathbb{N}, \boldsymbol{A} \models_{\boldsymbol{k}} \mathsf{E}\varphi$

Try $\boldsymbol{k} = \mathbf{0, 1, 2, 3, \cdots}$,

Completeness for Existential Model Checking

$\boldsymbol{P_2}$: if $\boldsymbol{A} \models \mathrm{E}\varphi \implies \exists \boldsymbol{k} \in \mathbb{N}, \boldsymbol{A} \models_{\boldsymbol{k}} \mathrm{E}\varphi$

Try $\boldsymbol{k} = \mathbf{0, 1, 2, 3, \cdots,}$

If $\boldsymbol{A} \not\models_{\boldsymbol{k}} \mathrm{E}\varphi$ for $\boldsymbol{k}$ sufficiently large

can we conclude that $\boldsymbol{A} \not\models \varphi$?

Proof

$\boldsymbol{P_2}$: if $\boldsymbol{A} \models \mathrm{E}\varphi \implies \exists \boldsymbol{k} \in \mathbb{N}, \boldsymbol{A} \models_{\boldsymbol{k}} \mathrm{E}\varphi$

Proof

$\boldsymbol{P_2}$: if $\boldsymbol{A} \models \mathrm{E}\varphi \implies \exists \boldsymbol{k} \in \mathbb{N}, \boldsymbol{A} \models_{\boldsymbol{k}} \mathrm{E}\varphi$

Büchi Automaton $\boldsymbol{A}$

Proof

$\boldsymbol{P_2}$: if $\boldsymbol{A} \models \mathrm{E}\varphi \implies \exists \boldsymbol{k} \in \mathbb{N}, \boldsymbol{A} \models_{\boldsymbol{k}} \mathrm{E}\varphi$

Büchi Automaton $\boldsymbol{A}$

$\boldsymbol{B_\varphi}$ Büchi automaton for φ

Proof

P_2: if $\boldsymbol{A} \models \mathrm{E}\varphi \implies \exists \boldsymbol{k} \in \mathbb{N}, \boldsymbol{A} \models_{\boldsymbol{k}} \mathrm{E}\varphi$

Büchi Automaton $\boldsymbol{A}$

$\boldsymbol{B}_\varphi$ Büchi automaton for φ

Proof

$\boldsymbol{P_2}$: if $\boldsymbol{A} \models \mathrm{E}\varphi \implies \exists \boldsymbol{k} \in \mathbb{N}, \boldsymbol{A} \models_{\boldsymbol{k}} \mathrm{E}\varphi$

$$\boldsymbol{A} \times \boldsymbol{B}_{\varphi}$$

Proof

$\boldsymbol{P_2}$: if $\boldsymbol{A} \models \mathrm{E}\varphi \implies \exists \boldsymbol{k} \in \mathbb{N}, \boldsymbol{A} \models_{\boldsymbol{k}} \mathrm{E}\varphi$

$$\boldsymbol{A} \times \boldsymbol{B}_\varphi$$

$$\emptyset \neq \mathcal{L}(\boldsymbol{A} \times \boldsymbol{B}_\varphi) \iff \boldsymbol{A} \models \mathrm{E}\varphi$$

Proof

$\boldsymbol{P_2}$: if $\boldsymbol{A} \models \mathrm{E}\varphi \implies \exists \boldsymbol{k} \in \mathbb{N}, \boldsymbol{A} \models_{\boldsymbol{k}} \mathrm{E}\varphi$

$\boldsymbol{A} \times \boldsymbol{B}_\varphi$

$$\emptyset \neq \mathcal{L}(\boldsymbol{A} \times \boldsymbol{B}_\varphi) \iff \boldsymbol{A} \models \mathrm{E}\varphi$$

$$\emptyset \neq \mathcal{L}(\boldsymbol{A} \times \boldsymbol{B}_\varphi) \iff \begin{cases} \boldsymbol{u}.\boldsymbol{v}^\omega \in \mathcal{L}(\boldsymbol{A} \times \boldsymbol{B}_\varphi) \\ |\boldsymbol{u}| + |\boldsymbol{v}| \leq |\boldsymbol{A} \times \boldsymbol{B}_\varphi| \end{cases}$$

Proof

P_2: if $A \models \mathrm{E}\varphi \implies \exists k \in \mathbb{N}, A \models_k \mathrm{E}\varphi$

$$A \times B_\varphi$$

$$\emptyset \neq \mathcal{L}(A \times B_\varphi) \iff A \models \mathrm{E}\varphi$$

$$\emptyset \neq \mathcal{L}(A \times B_\varphi) \iff \begin{cases} u.v^\omega \in \mathcal{L}(A \times B_\varphi) \\ |u| + |v| \leq |A \times B_\varphi| \end{cases}$$

$$|A \times B_\varphi| = |A| \cdot 2^{|\varphi|}$$

Proof

$\boldsymbol{P_2}$: if $\boldsymbol{A} \models \mathrm{E}\varphi \implies \exists \boldsymbol{k} \in \mathbb{N}, \boldsymbol{A} \models_{\boldsymbol{k}} \mathrm{E}\varphi$

$$\boldsymbol{A} \times \boldsymbol{B}_\varphi$$

$$\emptyset \neq \mathcal{L}(\boldsymbol{A} \times \boldsymbol{B}_\varphi) \iff \boldsymbol{A} \models \mathrm{E}\varphi$$

$$\emptyset \neq \mathcal{L}(\boldsymbol{A} \times \boldsymbol{B}_\varphi) \iff \begin{cases} \boldsymbol{u}.\boldsymbol{v}^\omega \in \mathcal{L}(\boldsymbol{A} \times \boldsymbol{B}_\varphi) \\ |\boldsymbol{u}| + |\boldsymbol{v}| \leq |\boldsymbol{A} \times \boldsymbol{B}_\varphi| \end{cases}$$

$$|\boldsymbol{A} \times \boldsymbol{B}_\varphi| = |\boldsymbol{A}| \cdot \mathbf{2}^{|\varphi|}$$

Let $\boldsymbol{k} = |\boldsymbol{A}| \cdot \mathbf{2}^{|\varphi|}$, then $\boldsymbol{A} \models_{\boldsymbol{k}} \mathrm{E}\varphi$

Towards Completeness

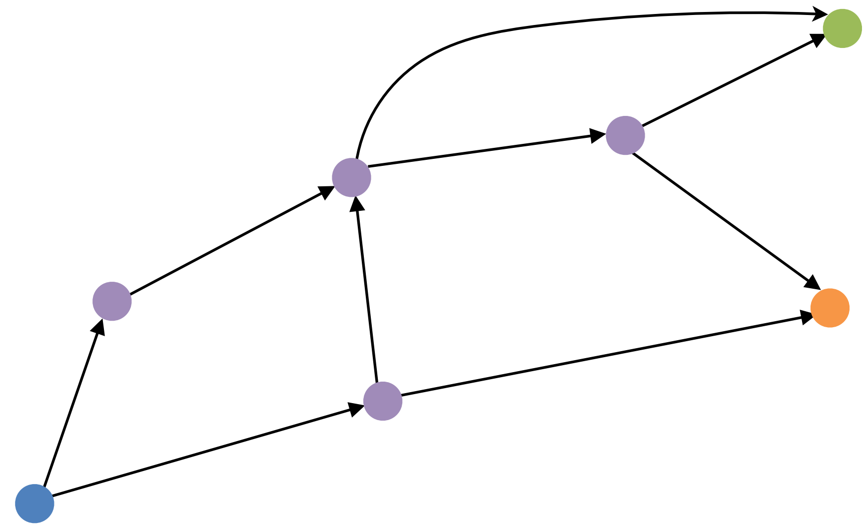
Focus on F$\boldsymbol{p}$, $\boldsymbol{p}$ atomic proposition

Computing a better bound

Towards Completeness

Focus on F***p***, ***p*** atomic proposition

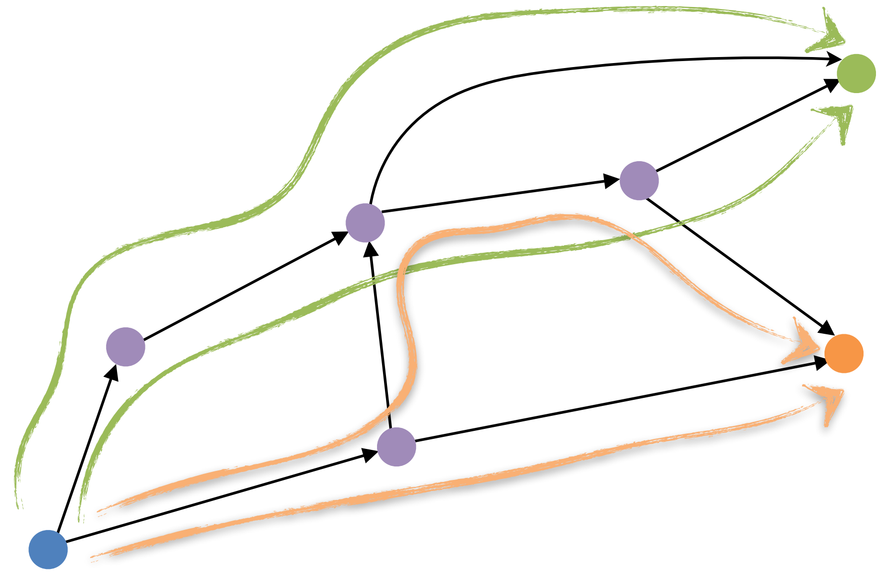
Computing a better bound



Towards Completeness

Focus on F***p***, ***p*** atomic proposition

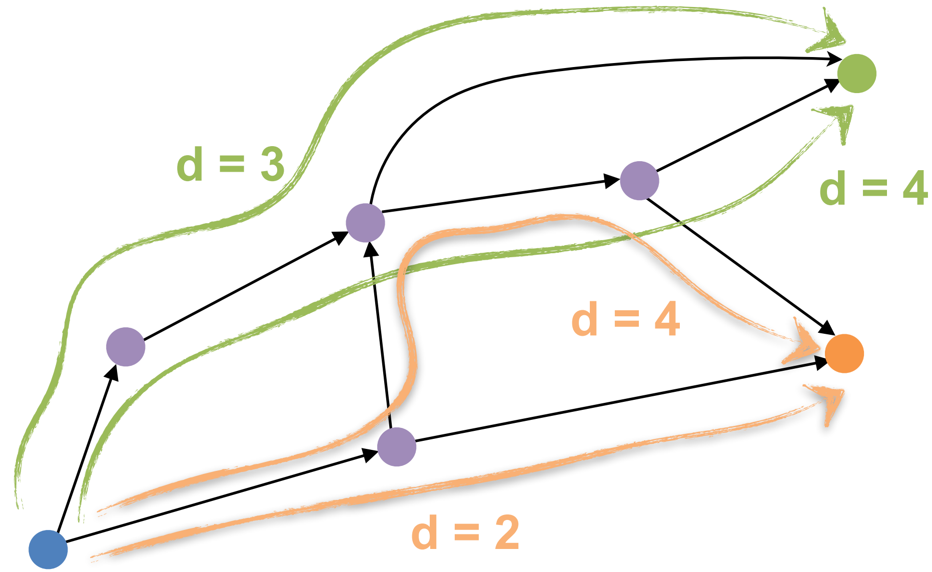
Computing a better bound



Towards Completeness

Focus on F$\boldsymbol{p}$, $\boldsymbol{p}$ atomic proposition

Computing a better bound

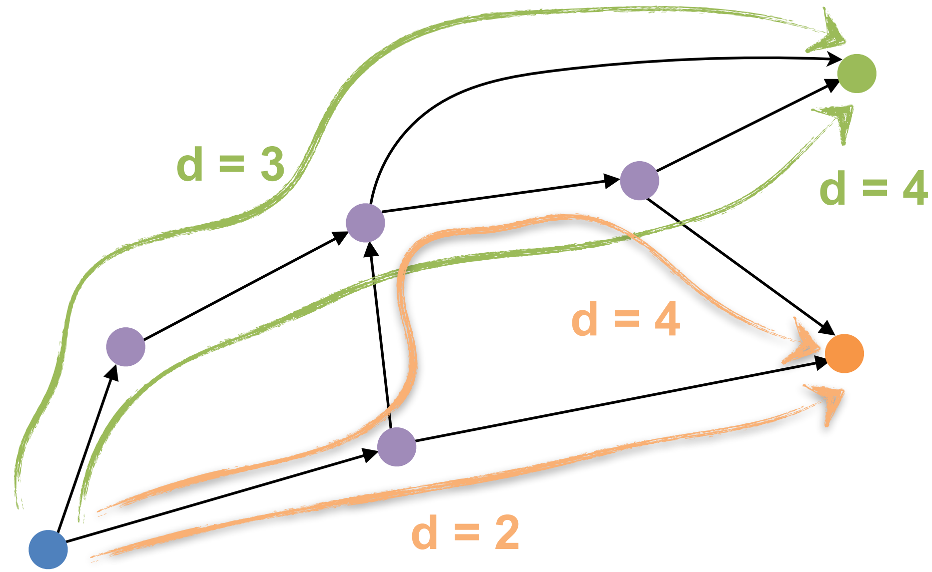


Towards Completeness

Focus on F***p***, ***p*** atomic proposition

Computing a better bound

Diameter = 3



Towards Completeness

Focus on F$\boldsymbol{p}$, $\boldsymbol{p}$ atomic proposition

Computing a better bound

$\boldsymbol{s} \models \boldsymbol{p}$ reachable $\implies$ $\boldsymbol{s}$ reachable on a shortest path

Towards Completeness

Focus on F$\boldsymbol{p}$, $\boldsymbol{p}$ atomic proposition

Computing a better bound

$$\boldsymbol{s} \models \boldsymbol{p} \text{ reachable} \implies \boldsymbol{s} \text{ reachable on a shortest path}$$

$$\boldsymbol{A} \models \mathrm{EF}\boldsymbol{p} \iff \boldsymbol{A} \models_{\boldsymbol{k}} \mathrm{EF}\boldsymbol{p} \text{ for some } \boldsymbol{k} \leq \mathrm{Diameter}(\boldsymbol{A})$$