

# Satisfiability Checking

## Bounded Model Checking

Prof. Dr. Erika Ábrahám

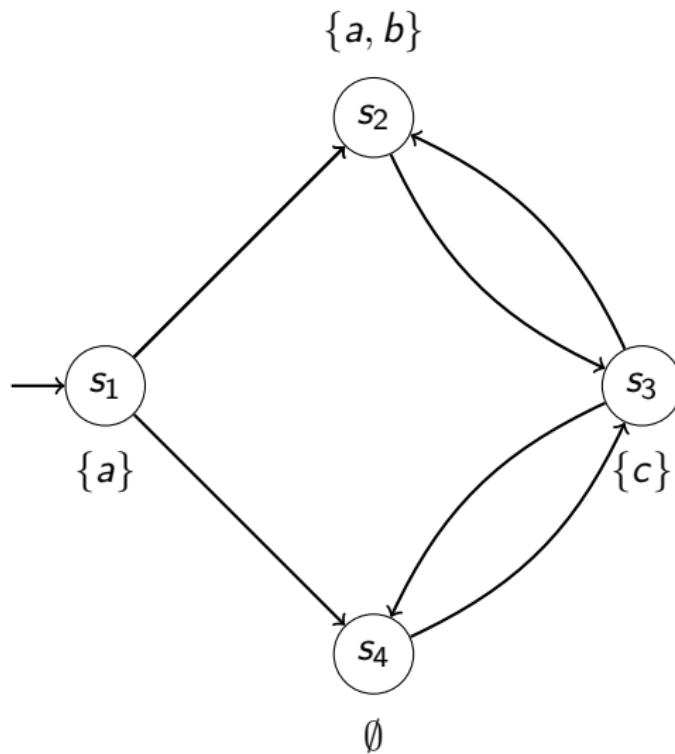
Theory of Hybrid Systems  
Informatik 2

WS 10/11

# Contents

- 1 Transition Systems
- 2 Logic
- 3 Bounded Model Checking

# Kripke Structure



## Definition

Let  $AP$  be a finite set of atomic propositions. A **Kripke structure** is a tuple  $M = (S, s_{\text{init}}, T, L)$  with

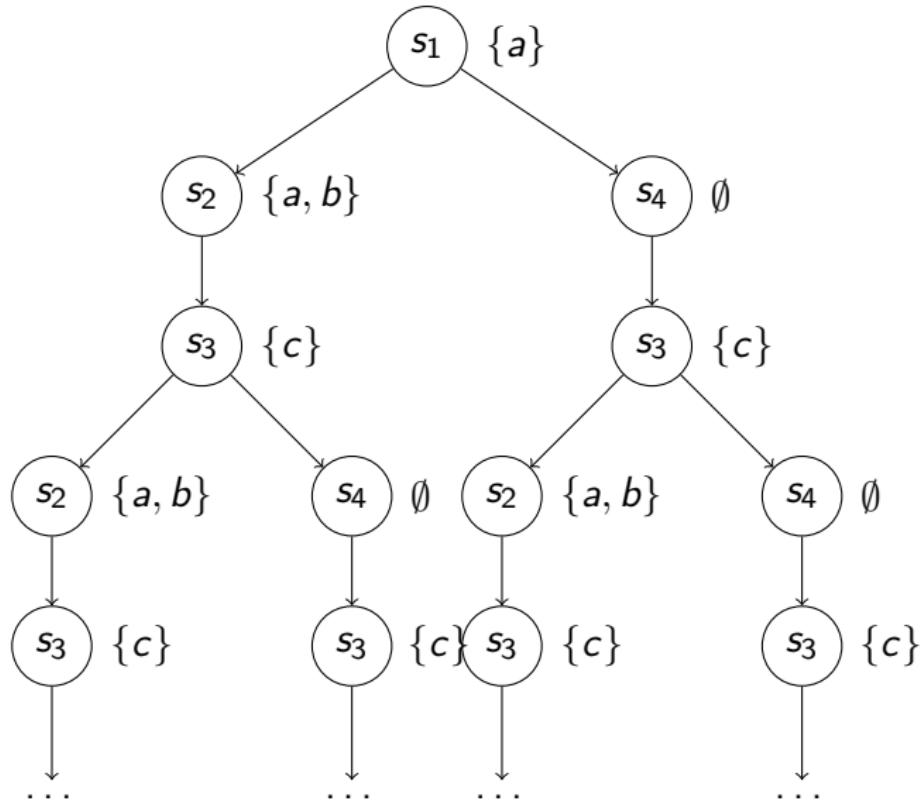
- $S$  a finite set of **states**,
- $s_{\text{init}} \in S$  an **initial state**,
- $T \subseteq S \times S$  a **transition relation**,
- $L : S \rightarrow 2^{AP}$  a **labeling function**  
( $2^{AP}$  denotes the powerset over  $AP$ ).

The labeling function attaches information to the system: for a state  $s \in S$  the set  $L(s)$  consists of those atomic propositions that hold in  $s$ .

# Kripke Structure: Semantics

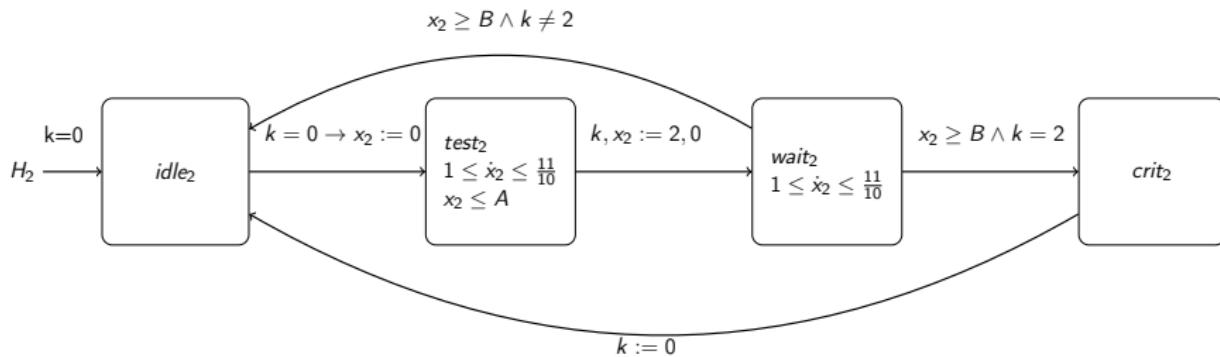
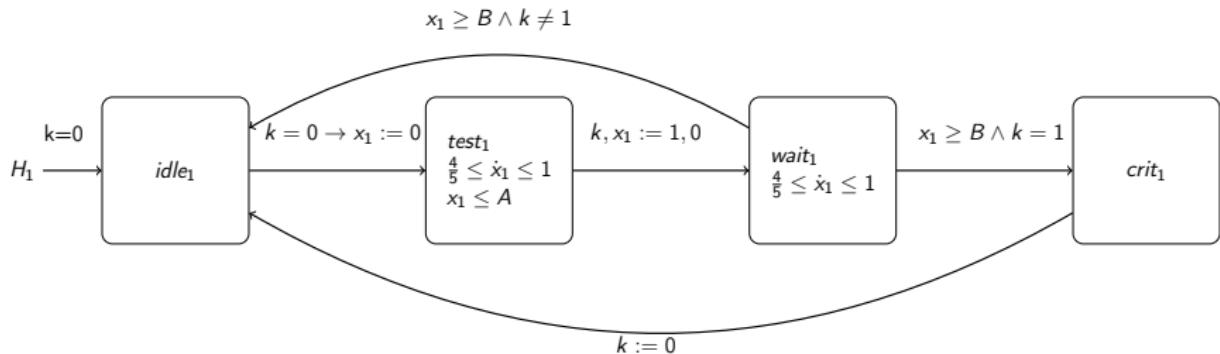
- An (infinite) path  $\pi = s_0 s_1 s_2 \dots$  of a Kripke structure  $M = (S, s_{\text{init}}, T, L)$  is a sequence of states such that
  - $s_0 = s_{\text{init}}$  and
  - for all  $i \geq 0$ ,  $(s_i, s_{i+1}) \in T$ .
- We write  $\pi(j)$  for the  $j$ th state (starting with 0) of the path  $\pi$ .
- $\pi_j$  denotes the postfix of  $\pi$  starting at  $\pi(j)$ .
- A finite path of  $M$  is a finite prefix of an infinite path of  $M$ .
- For a finite path  $\pi = s_0 \dots s_k$  we define  $|\pi| = k$ .
- The behaviour of  $M$  is given by the set of all of its infinite paths.

# Kripke Structure: Semantics



# Fischer's mutual exclusion protocol

There are also more complex systems we want to deal with later.



# LTL Syntax

Syntax of the Linear-Time Temporal Logic (LTL):

$$\varphi ::= a \mid \varphi \wedge \varphi \mid \neg \varphi \mid \mathcal{X} \varphi \mid \varphi \mathcal{U} \varphi \mid \mathcal{F} \varphi \mid \mathcal{G} \varphi$$

- $a \in AP$ : atomic proposition
- $\mathcal{X}$ : next time operator
- $\mathcal{U}$ : until operator

Syntactic sugar:

- $\vee, \rightarrow, \iff, \dots$
- $\mathcal{F}$ : finally (eventually) operator ( $\mathcal{F} \varphi := \text{true} \mathcal{U} \varphi$ )
- $\mathcal{G}$ : globally (always) operator ( $\mathcal{G} \varphi := \neg(\text{true} \mathcal{U} \neg \varphi)$ )

# LTL Semantics - Next

$$\pi_1: \quad \text{---} \quad \text{---} \quad \text{---} \quad \text{---} \quad \dots \quad \pi_1 \models \mathcal{X}p$$

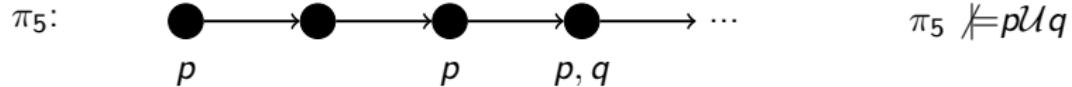
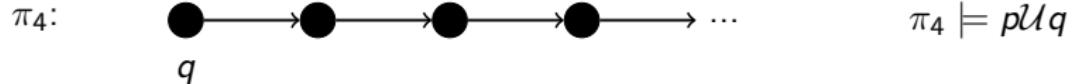
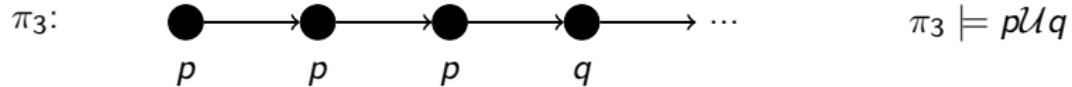
$p$

$$\pi_2: \quad \text{---} \quad \text{---} \quad \text{---} \quad \text{---} \quad \dots \quad \pi_2 \not\models \mathcal{X}p$$

$p$

$p$

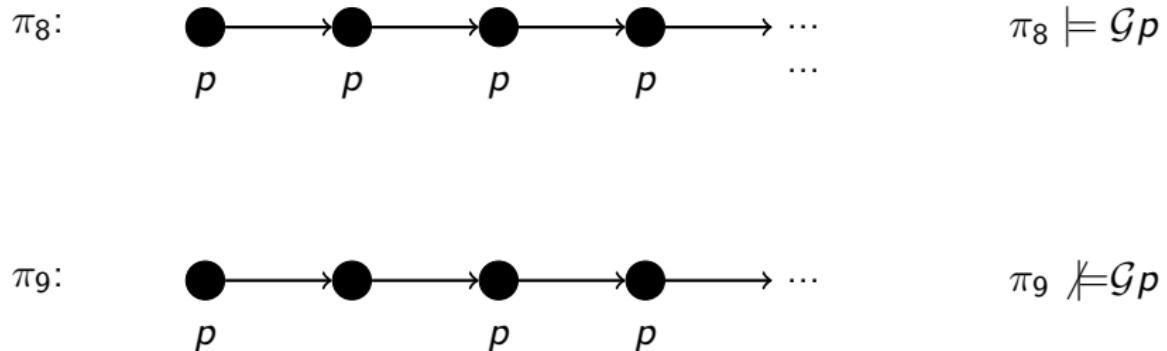
# LTL Semantics - Until



## LTL Semantics - Eventually



## LTL Semantics - Always



## Definition (LTL Semantics)

$\pi \models p$	iff	$p \in L(\pi(0))$
$\pi \models \varphi_1 \wedge \varphi_2$	iff	$\pi \models \varphi_1$ and $\pi \models \varphi_2$
$\pi \models \neg \varphi$	iff	$\pi \not\models \varphi$
$\pi \models \mathcal{X} \varphi$	iff	$\pi_1 \models \varphi$
$\pi \models \varphi_1 \mathcal{U} \varphi_2$	iff	$\pi_i \models \varphi_2$ for some $i \geq 0$ and $\pi_j \models \varphi_1$ for all $0 \leq j < i$
$\pi \models \mathcal{F} \varphi$	iff	$\pi_i \models \varphi$ for some $i \geq 0$
$\pi \models \mathcal{G} \varphi$	iff	$\pi_i \models \varphi$ for all $i \geq 0$

$M \models \mathbf{A}\varphi$

If all infinite paths of a Kripke structure  $M$  satisfy a property  $\varphi$ , then we say that **the property holds for  $M$** .

$M \models \mathbf{E}\neg\varphi$

If there is an infinite path of a Kripke structure  $M$  that does not satisfy a property  $\varphi$ , then we say that  **$M$  violates the property  $\varphi$** .

$M \models_k \mathbf{E}\neg\varphi$

Also **finite paths** can violate a property, if they contain enough information to assure the existence of an infinite path violating the property.

# Model Checking

Early 1980s: First implementations of Model Checking as verification technique

- Explicit representations of the transition graphs
- **Problem:** Due to the state explosion not applicable for most industrial settings

1990: Symbolic Model Checking

- BDDs represent *characteristic functions* of state sets
- **Problem:** Building the BDD may be expensive

1999: Bounded Model Checking *by Biere et al.*

- Incrementally finite paths of certain length are checked by SAT-Solver
- **Problem:** No completeness is guaranteed

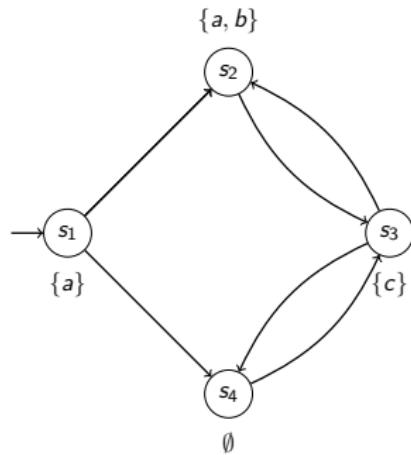
# Model Checking and Counterexamples

- Given a Kripke structure  $M$  and an LTL property  $\varphi$ , a **counterexample** is a path of  $M$  violating  $\varphi$ .
- If a system is buggy, counterexamples are extremely important for detecting and fixing the error.
- **Bounded model checking (BMC)** is a technique to search for finite counterexamples, not only for Kripke structures, but also for more complex systems.

# Finite and infinite Counterexamples

Property:  $\mathcal{G}\mathcal{F}a$

Negation:  $\neg\mathcal{G}\mathcal{F}a = \mathcal{F}\mathcal{G}\neg a$



Infinite counterexample:

$s_1 \ s_4 \ s_3 \ s_4 \ s_3 \ s_4 \ s_3 \ \dots$

Finite counterexample:

$s_1 \boxed{s_4} \ s_3 \ \boxed{s_4}$

“Loop detected”

## Definition of $\models_k$

Satisfaction relation for **finite paths**

$\pi \models_k^i \varphi$  : the finite segment of  $\pi$  consisting of its  $i$ th to  $k$ th states satisfies  $\varphi$

$\pi \models_k \varphi$  :  $\pi \models_k^0 \varphi$

# Definition of $\models_k$

Satisfaction relation for **finite paths with a loop**

## Definition

For  $l \leq k$  we call an infinite path  $\pi$  a  **$(k, l)$ -loop** iff  $T(\pi(k), \pi(l))$  and  $\pi = u \cdot v^\omega$  with  $u = \pi(0) \dots \pi(l-1)$  and  $v = \pi(l) \dots \pi(k)$ .

We call  $\pi$  a  **$k$ -loop** iff  $\pi$  is a  $(k, l)$ -loop for some  $0 \leq l \leq k$ .

## Definition (Bounded Semantics for a loop)

Let  $k \geq 0$  and let  $\pi$  be a  $k$ -loop. Then an LTL formula  $\varphi$  is **valid along  $\pi$  with bound  $k$**  ( $\pi \models_k \varphi$ ) iff  $\pi \models \varphi$ .

# Definition of $\models_k$

## Definition (Bounded Semantics without a loop)

Let  $k \geq 0$  and let  $\pi$  be path that is not a  $k$ -loop. Then an LTL formula  $\varphi$  is valid along  $\pi$  with bound  $k$  ( $\pi \models_k \varphi$ ) iff  $\pi \models_k^0 \varphi$ , where

$\pi \models_k^i a$	: $a \in L(\pi(i))$
$\pi \models_k^i \neg a$	: $a \notin L(\pi(i))$
$\pi \models_k^i \varphi_1 \wedge \varphi_2$	: $\pi \models_k^i \varphi_1$ and $\pi \models_k^i \varphi_2$
$\pi \models_k^i \mathcal{X} \varphi$	: $i < k$ and $\pi \models_k^{i+1} \varphi$
$\pi \models_k^i \varphi_1 \mathcal{U} \varphi_2$	: $\pi \models_k^j \varphi_2$ for some $i \leq j \leq k$ and $\pi \models_k^n \varphi_1$ for all $i \leq n < j$
$\pi \models_k^i \mathcal{F} \varphi$	: $\pi \models_k^j \varphi$ for some $i \leq j \leq k$
$\pi \models_k^i \mathcal{G} \varphi$	: false

# Properties of $\models_k$

## Lemma

Let  $\varphi$  be an LTL formula and let  $\pi$  be a path. Then

$$\pi \models_k \varphi \Rightarrow \pi \models \varphi.$$

## Lemma

Let  $\varphi$  be an LTL formula and let  $M$  be a Kripke structure. Then

$$M \models \mathbf{E}\varphi \Rightarrow \exists k \geq 0. M \models_k \mathbf{E}\varphi.$$

# Bounded Model Checking

## Overview:

- Construction of a Boolean formula  $\varphi$  describing a finite path
  - through the underlying system
  - of length  $k$ , starting with 0,
  - and reaching a certain state of interest, i.e., violating a property.
- A SAT-solver searches for a satisfying assignment of  $\varphi$
- If SAT, the resulting assignment describes a counterexample
- If UNSAT,  $k$  is incremented and the procedure starts again

# Bounded Model Checking

Counterexamples of length  $k$  for a Kripke structure  $M$  and an LTL formula  $\varphi$  can be described by

$$[\![M, \varphi]\!]_k = I(s_0) \wedge T(s_0, s_1) \wedge \dots \wedge T(s_{k-1}, s_k) \wedge \neg \text{Prop}(s_0, \dots, s_k)$$

$[\![M, \varphi]\!]_k$  is satisfiable  $\iff$  there exists a finite counterexample of length  $k$

→ check  $[\![M, \varphi]\!]_k$  incrementally for  $k = 0, 1, \dots$  using a suitable solver

# Formula Encoding

$$\llbracket M, \varphi \rrbracket_k = I(s_0) \wedge T(s_0, s_1) \wedge \dots \wedge T(s_{k-1}, s_k) \wedge \neg \text{Prop}(s_0 \dots s_k)$$

How to build this formula?

- $I$  and  $T$  are (nearly) straightforward for Kripke structures. We build a sub-formula describing initial paths of length  $k$ :

$$\llbracket M \rrbracket_k := I(s_0) \wedge \bigwedge_{i=0}^{k-1} T(s_i, s_{i+1})$$

- This formula is called the **unfolding of the transition relation**

# Formula Encoding

$$[\![M, \varphi]\!]_k = I(s_0) \wedge T(s_0, s_1) \wedge \dots \wedge T(s_{k-1}, s_k) \wedge \neg \text{Prop}(s_0 \dots s_k)$$

How to build this formula?

- To get a counterexample for an LTL formula  $\varphi$  we have to find a **witness** for  $\neg\varphi$ .
- This will be encoded for paths of length  $k$  within the formula  $\neg \text{Prop}(s_0, \dots, s_k)$
- The translation of the formula depends on the fact whether the considered path has a loop or not.

# Encoding of loops

Loop condition: Is there a transition from  $s_k$  to a previous state?

Loop successor: Successor state of a state inside a loop

## Definition

The loop condition  $L_k$  is true iff there exists a back loop from state  $s_k$  to a previous state or to itself:  $L_k := \bigvee_{l=0}^k T(s_k, s_l)$

## Definition

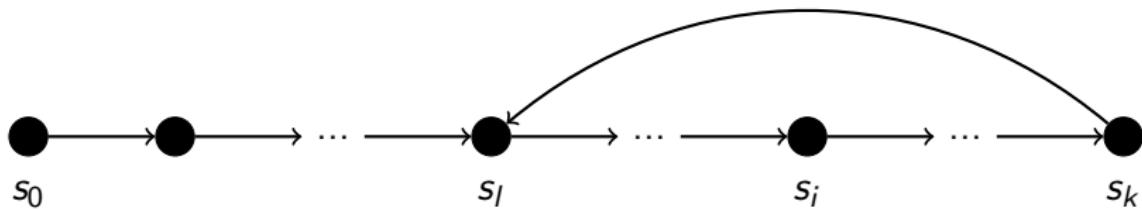
Let  $k, l$  and  $i$  be non-negative integers with  $l, i \leq k$ .

- $\text{succ}(i) := i + 1$ , if  $i$  is inside a  $(k, l)$ -loop, i.e.  $i < k$
- $\text{succ}(i) := l$  for  $i = k$

## Encoding of loops - Always

- Given: LTL formula  $\varphi$  and path  $\pi$  with  $(k, l)$ -loop
- Recursive translation over the sub-terms of  $\varphi$  and states in  $\pi$
- Introduce intermediate formula of the form  ${}_l[\![\cdot]\!]_k^i$ 
  - $l$  start-state of the loop
  - $k$  bound
  - $i$  current position
- Translation rule for  $\mathcal{G}\varphi$ :

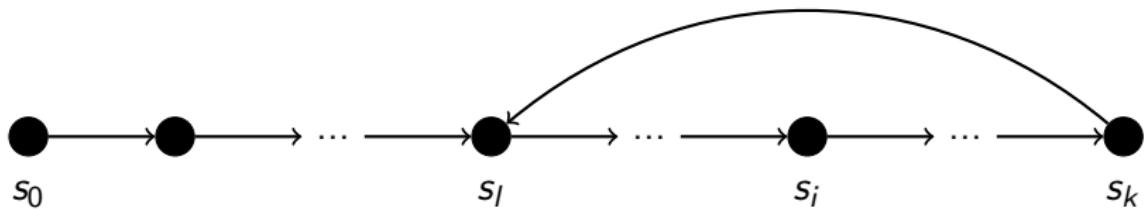
$${}_l[\![\mathcal{G}\varphi]\!]_k^i := {}_l[\![\varphi]\!]_k^i \wedge {}_l[\![\mathcal{G}\varphi]\!]_k^{\text{succ}(i)}$$



## Encoding of loops - Eventually

- Given: LTL formula  $\varphi$  and path  $\pi$  with  $(k, l)$ -loop
- Recursive translation over the sub-terms of  $\varphi$  and states in  $\pi$
- Introduce intermediate formula of the form  ${}_l[\![\cdot]\!]_k^i$ 
  - $l$  start-state of the loop
  - $k$  bound
  - $i$  current position
- Translation rule for  $\mathcal{F}\varphi$ :

$${}_l[\![\mathcal{F}\varphi]\!]_k^i := {}_l[\![\varphi]\!]_k^i \vee {}_l[\![\mathcal{F}\varphi]\!]_k^{\text{succ}(i)}$$



# Encoding of loops

$$\begin{aligned}{}_I[\![p]\!]_k^i &:= p(s_i) \\ {}_I[\![\neg p]\!]_k^i &:= \neg p(s_i) \\ {}_I[\![\varphi \vee \psi]\!]_k^i &:= {}_I[\![\varphi]\!]_k^i \vee {}_I[\![\psi]\!]_k^i \\ {}_I[\![\varphi \wedge \psi]\!]_k^i &:= {}_I[\![\varphi]\!]_k^i \wedge {}_I[\![\psi]\!]_k^i \\ {}_I[\![\mathcal{G}\varphi]\!]_k^i &:= {}_I[\![\varphi]\!]_k^i \wedge {}_I[\![\mathcal{G}\varphi]\!]_k^{\text{succ}(i)} \\ {}_I[\![\mathcal{F}\varphi]\!]_k^i &:= {}_I[\![\varphi]\!]_k^i \vee {}_I[\![\mathcal{F}\varphi]\!]_k^{\text{succ}(i)} \\ {}_I[\![\varphi \mathcal{U} \psi]\!]_k^i &:= {}_I[\![\psi]\!]_k^i \vee ({}_I[\![\varphi]\!]_k^i \wedge {}_I[\![\varphi \mathcal{U} \psi]\!]_k^{\text{succ}(i)}) \\ {}_I[\![\mathcal{X}\varphi]\!]_k^i &:= {}_I[\![\varphi]\!]_k^{\text{succ}(i)} \end{aligned}$$

## Encoding without loops - Always

- Given: LTL formula  $\varphi$  and path  $\pi$  without  $(k, l)$ -loop
- Special case of loop translation
- Extension to infinite path with considering all properties beyond  $s_k$  as **false**
  - $k$  bound
  - $i$  current position
- Translation rule for  $\mathcal{G}\varphi$ ,  $i \leq k$ :

$$\llbracket \mathcal{G}\varphi \rrbracket_k^i := \llbracket \varphi \rrbracket_k^i \vee \llbracket \mathcal{G}\varphi \rrbracket_k^{\text{succ}(i)}$$

## Encoding without loops - Eventually

- Given: LTL formula  $\varphi$  and path  $\pi$  without  $(k, l)$ -loop
- Special case of loop translation
- Extension to infinite path with considering all properties beyond  $s_k$  as **false**
  - $k$  bound
  - $i$  current position
- Translation rule for  $\mathcal{F}\varphi$ ,  $i \leq k$ :

$$\llbracket \mathcal{F}\varphi \rrbracket_k^i := \llbracket \varphi \rrbracket_k^i \vee \llbracket \mathcal{F}\varphi \rrbracket_k^{\text{succ}(i)}$$

# Encoding without loops

$$\begin{aligned}\llbracket p \rrbracket_k^i &:= p(s_i) \\ \llbracket \neg p \rrbracket_k^i &:= \neg p(s_i) \\ \llbracket \varphi \vee \psi \rrbracket_k^i &:= \llbracket \varphi \rrbracket_k^i \vee \llbracket \psi \rrbracket_k^i \\ \llbracket \varphi \wedge \psi \rrbracket_k^i &:= \llbracket \varphi \rrbracket_k^i \wedge \llbracket \psi \rrbracket_k^i \\ \llbracket \mathcal{G} \varphi \rrbracket_k^i &:= \llbracket \varphi \rrbracket_k^i \wedge \llbracket \mathcal{G} \varphi \rrbracket_k^{i+1} \\ \llbracket \mathcal{F} \varphi \rrbracket_k^i &:= \llbracket \varphi \rrbracket_k^i \vee \llbracket \mathcal{F} \varphi \rrbracket_k^{i+1} \\ \llbracket \varphi \mathcal{U} \psi \rrbracket_k^i &:= \llbracket \psi \rrbracket_k^i \vee (\llbracket \varphi \rrbracket_k^i \wedge \llbracket \varphi \mathcal{U} \psi \rrbracket_k^{i+1}) \\ \llbracket \mathcal{X} \varphi \rrbracket_k^i &:= \llbracket \varphi \rrbracket_k^{i+1}\end{aligned}$$

# General Translation to SAT-Problem

- Combining the components, BMC is encoded in propositional logic
- Given: LTL formula  $\varphi$ , Kripke structure  $M$ , bound  $k$

$$[\![M, \varphi]\!]_k := [\![M]\!]_k \wedge \left( (\neg L_k \wedge [\![\varphi]\!]_k^0) \vee \bigvee_{l=0}^k (T(s_k, s_l) \wedge {}_l[\![\varphi]\!]_k^0) \right)$$

- Unfolding of the transition relation
- There is no back loop  $\rightsquigarrow$  Translation without loops
- All possible starting points of a loop are considered  $\rightsquigarrow$  Translation for  $(k, l)$ -loop together with loop condition

## Theorem

$[\![M, \varphi]\!]_k$  is satisfiable iff  $M \models_k \mathbf{E}\varphi$

## BMC is not complete

- Application: Start with  $k = 0$  and increment until witness is found
- Termination is guaranteed iff witness exists ( $M \models E\varphi$ )
- If no witness exists, procedure does not terminate ( $M \not\models E\varphi$ )
- Upper bound for  $k$  to ensure property: **Completeness threshold**

# Completeness Threshold

- For each (finite state) system  $M$ , property  $p$  and given translation scheme there exists a number  $\mathcal{CT}$ , called **completeness threshold**.
- Considering  $\mathcal{G}\varphi$  formulas,  $\mathcal{CT}$  is equal to the **reachability diameter**, i.e., the minimal distance required to reach all (reachable) states of the system.

## Definition (Reachability Diameter)

$$rd(M) := \min \left\{ i \mid \forall n > i. \forall s_0, \dots, s_n. \exists t \leq i. \exists s'_0, \dots, s'_t. \right.$$

$$\left( I(s_0) \wedge \bigwedge_{j=0}^{n-1} T(s_j, s_{j+1}) \right) \rightarrow \left( I(s'_0) \wedge \bigwedge_{j=0}^{t-1} T(s'_j, s'_{j+1}) \wedge s'_t = s_n \right) \right\}$$

“Every state that is reachable in  $n$  steps, is also reachable in  $i$  steps.”

- This yields maximal shortest paths in the system.

# Completeness Threshold

- Problem: One has to choose  $n$
- Let  $V$  be the set of variables defining the states. Worst case:  $n = 2^{|V|}$
- Better: Choose  $n = i + 1$ .

## Definition (Reachability Diameter)

$$rd(M) := \min \left\{ i \mid \forall s_0, \dots, s_{i+1}. \exists s'_0, \dots, s'_i. \right. \\ \left( I(s_0) \wedge \bigwedge_{j=0}^i T(s_j, s_{j+1}) \right) \rightarrow \left( I(s'_0) \wedge \bigwedge_{j=0}^{i-1} T(s'_j, s'_{j+1}) \wedge \bigvee_{j=0}^i s'_j = s_{i+1} \right) \right\}$$

“Every state that is reachable in  $i + 1$  steps, it is also reachable in  $i$  steps.”

# Completeness Threshold

- Problem: Formula contains alternation of quantifiers
- Solution: Over-approximation of  $rd(M)$

## Definition (Recurrence Diameter)

$rdr(M) :=$

$$\max \left\{ i \mid \exists s_0 \dots s_i : I(s_0) \wedge \bigwedge_{j=0}^{i-1} T(s_j, s_{j+1}) \wedge \bigwedge_{j=0}^{i-1} \bigwedge_{k=j+1}^i s_j \neq s_k \right\}$$

“Longest loop-free initial path in  $M$ .”

- As every shortest path is a loop-free path, this is an over-approximation of  $rd(M)$ .