# Model Checking

## Verifying Regular Properties

[Baier & Katoen, Chapter 4.1+4.2]

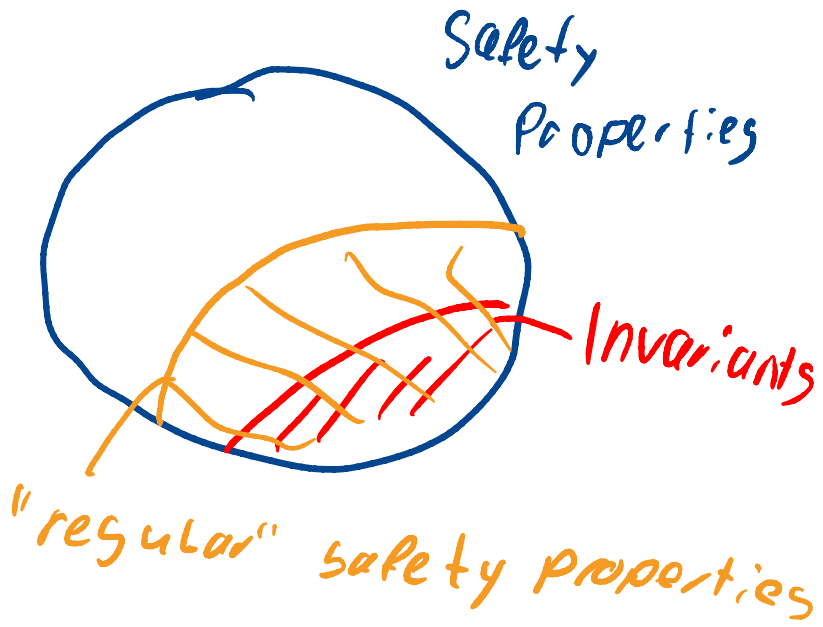
Joost-Pieter Katoen and Tim Quatmann

Software Modeling and Verification Group

RWTH Aachen, SoSe 2022

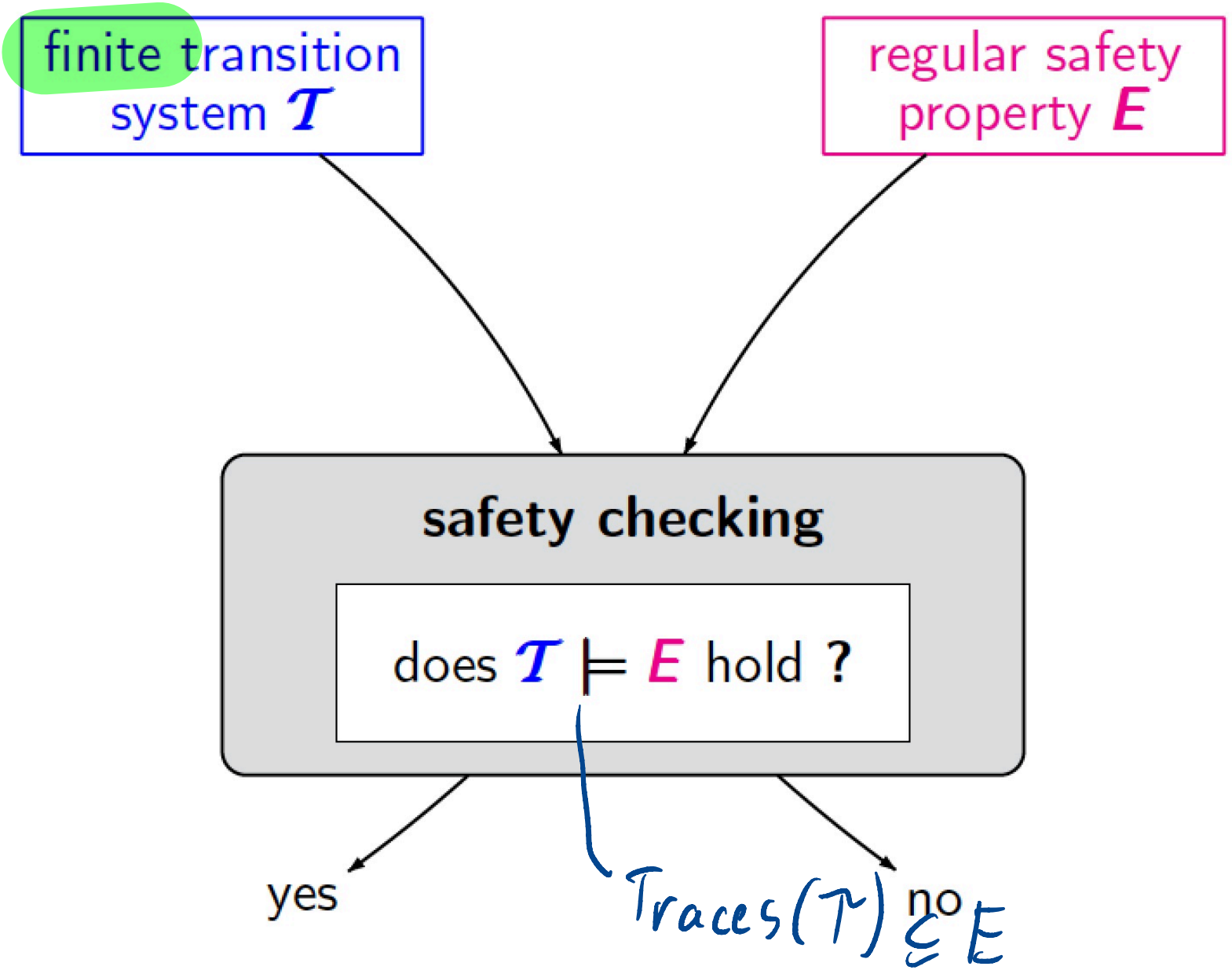
# Overview





# Overview

# Topic


finite transition system $\mathcal{T}$
regular safety property $E$
safety checking
does $\mathcal{T} \models E$ hold ?
yes
no
Traces($\mathcal{T}$) $\not\subseteq E$

# Safety Properties

## Definition: Safety Property

LT property $E_{safe}$ over $AP$ is a safety property if for all $\sigma \in (2^{AP})^\omega \setminus E_{safe}$:

$$E_{safe} \cap \{\sigma' \in (2^{AP})^\omega \mid \hat{\sigma} \text{ is a prefix of } \sigma'\} = \varnothing.$$

for some prefix $\hat{\sigma}$ of $\sigma$.

$\{\hat{\sigma}\sigma' \mid \sigma' \in (2^{AP})^\omega\}$

# Safety Properties

## Definition: Safety Property

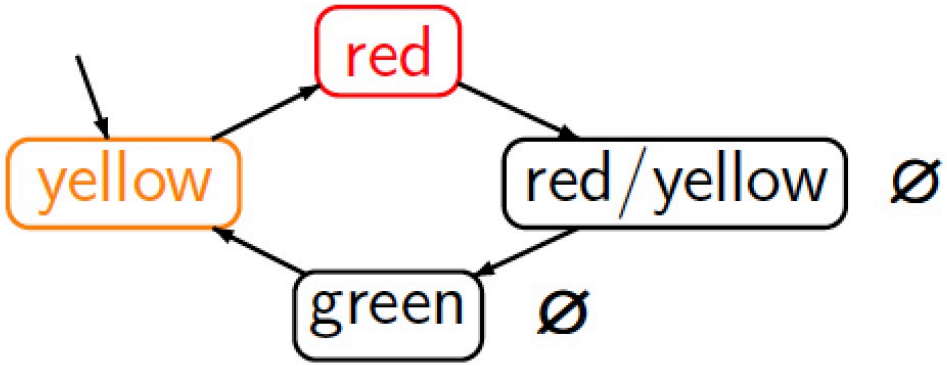
LT property $E_{safe}$ over $AP$ is a safety property if for all $\sigma \in \left(2^{AP}\right)^{\omega} \setminus E_{safe}$:

$$E_{safe} \cap \left\{\sigma' \in \left(2^{AP}\right)^{\omega} \mid \hat{\sigma} \text{ is a prefix of } \sigma'\right\} = \varnothing.$$

for some prefix $\hat{\sigma}$ of $\sigma$.

- Path fragment $\hat{\sigma}$ is called a bad prefix of $E_{safe}$
- Let $\mathit{BadPref}(E_{safe})$ denote the set of bad prefixes of $E_{safe}$
- $\hat{\sigma} \in E_{safe}$ is minimal if no proper prefix of it is in $\mathit{BadPref}(E_{safe})$
- Let $\mathit{MinBadPref}(E_{safe})$ denote the set of minimal bad prefixes of $E_{safe}$

# Examples




"every red phase is preceded by a yellow phase"

hence: $\mathcal{T} \models E$

$E$ = set of all infinite words $A_0\, A_1\, A_2 \ldots$ over $2^{AP}$ such that for all $i \in \mathbb{N}$:

$$red \in A_i \implies i \geq 1 \text{ and } yellow \in A_{i-1}$$

is a safety property over $AP = \{red, yellow\}$ with

$BadPref$ = set of all finite words $A_0\, A_1 \ldots A_n$ over $2^{AP}$ s.t. for some $i \in \{0, \ldots, n\}$:

$$red \in A_i \wedge (i{=}0 \vee yellow \notin A_{i-1})$$

# Regular Safety Properties

## Definition: regular safety property

Safety property $E_{safe}$ is regular if $BadPref(E_{safe})$ is a regular language.

set of finite words

Reg. Exp.
NFA
DFA

# Regular Safety Properties

**Definition: regular safety property**

Safety property $E_{safe}$ is regular if $BadPref(E_{safe})$ is a regular language.

Or, equivalently:

Safety property $E_{safe}$ is regular if there exists

a finite automaton over the alphabet $2^{AP}$ recognizing $BadPref(E_{safe})$

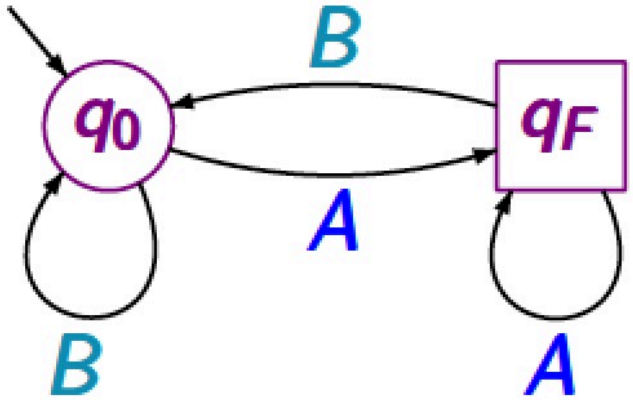
Or . . . . . MinBadPref

# Overview

# Finite Automata

A nondeterministic finite automaton (NFA) $\mathfrak{A} = (Q, \Sigma, \delta, Q_0, F)$ with:

- $Q$ is a finite set of states
- $\Sigma$ is an alphabet
- $\delta : Q \times \Sigma \to 2^Q$ is a transition function
- $Q_0 \subseteq Q$ a set of initial states
- $F \subseteq Q$ is a set of accept (or: final) states




$\delta(q_0, A) = \{q_F\}$

$\mathfrak{A}$ is a DFA iff $\forall q \in Q, A \in \Sigma : |\delta(q, A)| = 1$

# Language of an Finite Automaton

- NFA $\mathfrak{A} = (Q, \Sigma, \delta, Q_0, F)$ and finite word $w = A_1 \dots A_n \in \Sigma^*$
- A run for $w$ in $\mathfrak{A}$ is a finite sequence $q_0\, q_1\, \dots\, q_n \in Q^*$ such that:
  - $q_0 \in Q_0$ and $q_i \xrightarrow{A_{i+1}} q_{i+1}$ for all $0 \leq i < n$
- Run $q_0\, q_1\, \dots\, q_n$ is accepting if $q_n \in F$
- The accepted language of $\mathfrak{A}$:

$$\mathfrak{L}(\mathfrak{A}) = \left\{ w \in \Sigma^* \mid \mathfrak{A} \text{ has an accepting run for } w \right\}$$

  - $w \in \Sigma^*$ is accepted by $\mathfrak{A}$ if $\mathfrak{A}$ has an accepting run for $w$
- NFA $\mathfrak{A}$ and $\mathfrak{A}'$ are equivalent if $\mathfrak{L}(\mathfrak{A}) = \mathfrak{L}(\mathfrak{A}')$

# Facts about Finite Automata

- They are as expressive as regular languages (Kleene's theorem)
- They are closed under $\cup$, $\cap$, and complementation
  - NFA $\mathfrak{A} \otimes \mathfrak{B}$ ($=$ cross product) accepts $\mathfrak{L}(\mathfrak{A}) \cap \mathfrak{L}(\mathfrak{B})$
  - Total DFA $\overline{\mathfrak{A}}$ ($=$ swap all accept and normal states) accepts $\overline{\mathfrak{L}(\mathfrak{A})} = \Sigma^* \setminus \mathfrak{L}(\mathfrak{A})$
- They are closed under determinization ($=$ powerset construction)
  - although at an exponential cost ...
- $\mathfrak{L}(\mathfrak{A}) = \varnothing?$ $=$ check for a reachable accept state in NFA $\mathfrak{A}$
  - this can be done using a classical depth-first search
  - in linear-time complexity in the size of $\mathfrak{A}$
- For regular language $\mathfrak{L}$ there is a unique minimal DFA accepting $\mathfrak{L}$

# Regular Safety Properties Revisited

**Definition: regular safety property**

Safety property $E_{safe}$ is regular if $\mathit{BadPref}(E_{safe})$ is a regular language.

Or, equivalently:

if there exists a regular expression D over $2^{AP}$ with $\mathfrak{L}(\mathrm{D}) = \mathit{BadPref}(E_{safe})$

Or, equivalently:

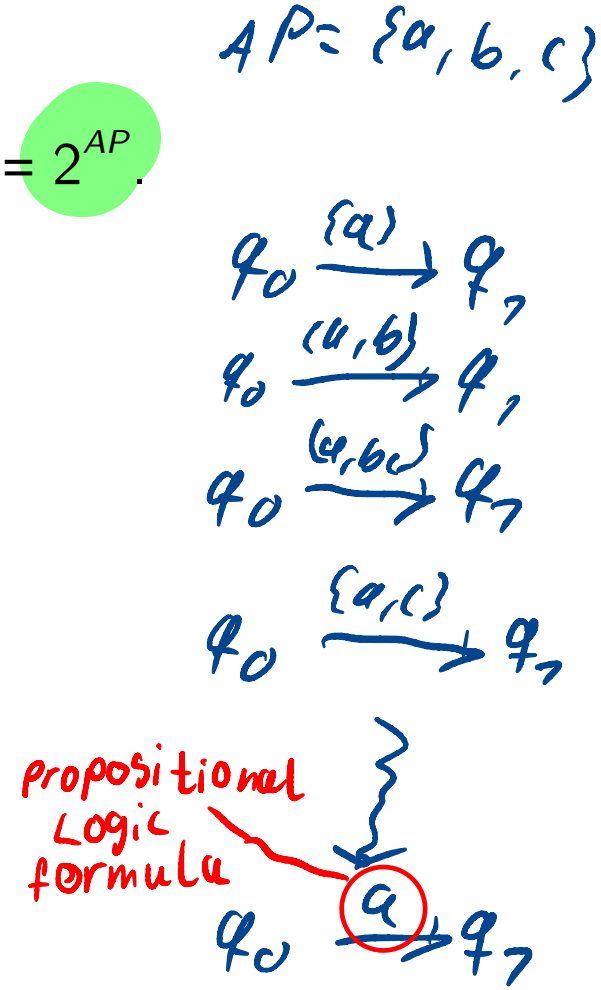
Safety property $E_{safe}$ is regular if there exists an NFA $\mathfrak{A}$ over the alphabet $2^{AP}$ with $\mathfrak{L}(\mathfrak{A}) = \mathit{BadPref}(E_{safe})$

Or, equivalently:

... if there exists a DFA $\mathfrak{A}$ over $2^{AP}$ with $\mathfrak{L}(\mathfrak{A}) = \mathit{BadPref}(E_{safe})$

# Sets as Formulas

Let NFA $\mathfrak{A} = (Q, \Sigma, \delta, Q_0, F)$ over the alphabet $\Sigma = 2^{AP}$.



# Sets as Formulas

Let NFA $\mathfrak{A} = (Q, \Sigma, \delta, Q_0, F)$ over the alphabet $\Sigma = 2^{AP}$.

We adopt a shorthand notation for the transitions using propositional logic. If $\Phi$ is a propositional logic formula over $AP$ then:

$p \xrightarrow{\Phi} q$ stands for the set of transitions $p \xrightarrow{A} q$ with $A \subseteq AP$ and $A \vDash \Phi$

where $A \subseteq AP$ such that $A \vDash \Phi$.

Examples. Let $A = \{ a, b, c \}$. Then:

- $p \xrightarrow{a \wedge \neg b} q$ stands for $\{ p \xrightarrow{\{a\}} q, p \xrightarrow{\{a,c\}} q \}$
- $p \xrightarrow{\text{false}} q$ stands for $\varnothing$
- $p \xrightarrow{\text{true}} q$ stands for $\{ p \xrightarrow{A} q \mid \forall A \subseteq AP \}$

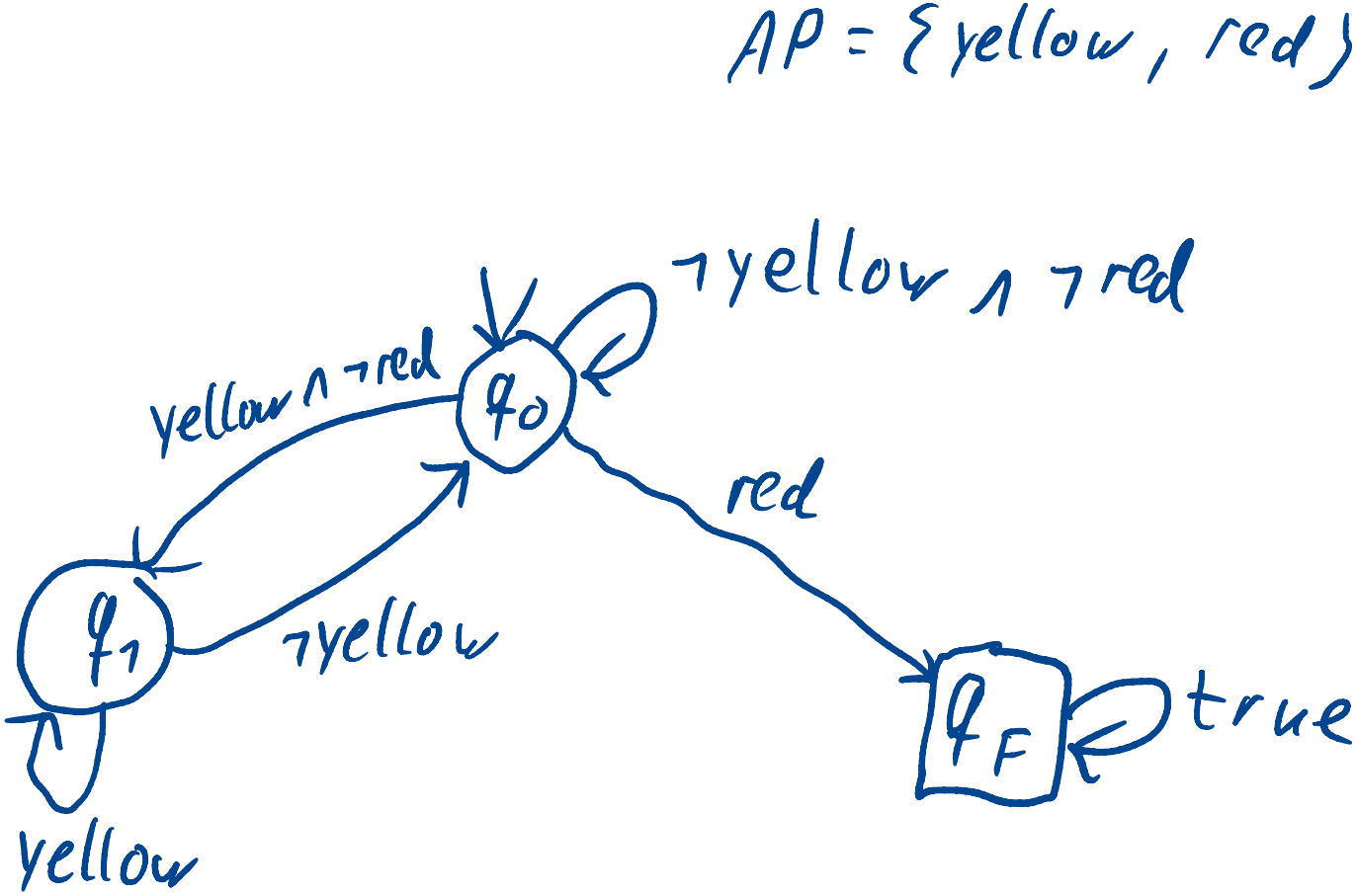
# Examples

- Every invariant (over $AP$) is a regular safety property
  - bad prefixes are of the form $\Phi^* (\neg\Phi)\,\mathsf{true}^*$
  - where $\Phi$ is the invariant condition
- A regular safety property which is not an invariant:

  "a red light is immediately preceded by a yellow light"
- A non-regular safety property:

  "the # inserted coins is at least the # of dispensed drinks"

# Details


AP = {yellow, red}
¬yellow ∧ ¬red
yellow ∧ ¬red
q0
red
q1
¬yellow
qF
true
yellow

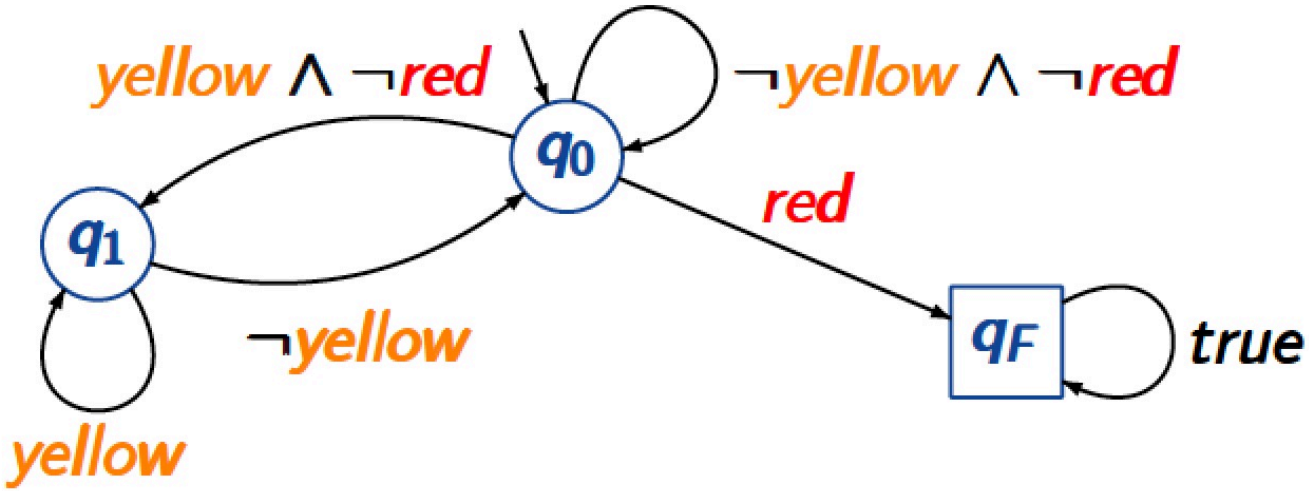
# Example

*"Every red phase is preceded by a yellow phase"*

set of all infinite words $A_0 A_1 A_2 \ldots$ s.t. for all $i \geq 0$:

$$red \in A_i \implies i \geq 1 \text{ and } yellow \in A_{i-1}$$

**DFA** for all (possibly non-minimal) bad prefixes



# Overview

# Peterson's Algorithm

```
P1   loop forever
     ⋮                                        (* non-critical actions *)
     ⟨b1 := true; x := 2⟩;                    (* request *)
     wait until (x = 1 ∨ ¬b2)
     do critical section od
     b1 := false                              (* release *)
     ⋮                                        (* non-critical actions *)
     end loop
```

atomic

$b_i$ is true if and only if process $P_i$ is waiting or in critical section

if both threads want to enter their critical section, $x$ decides who gets access

# Accessing a Bank Account

Thread Left behaves as follows:

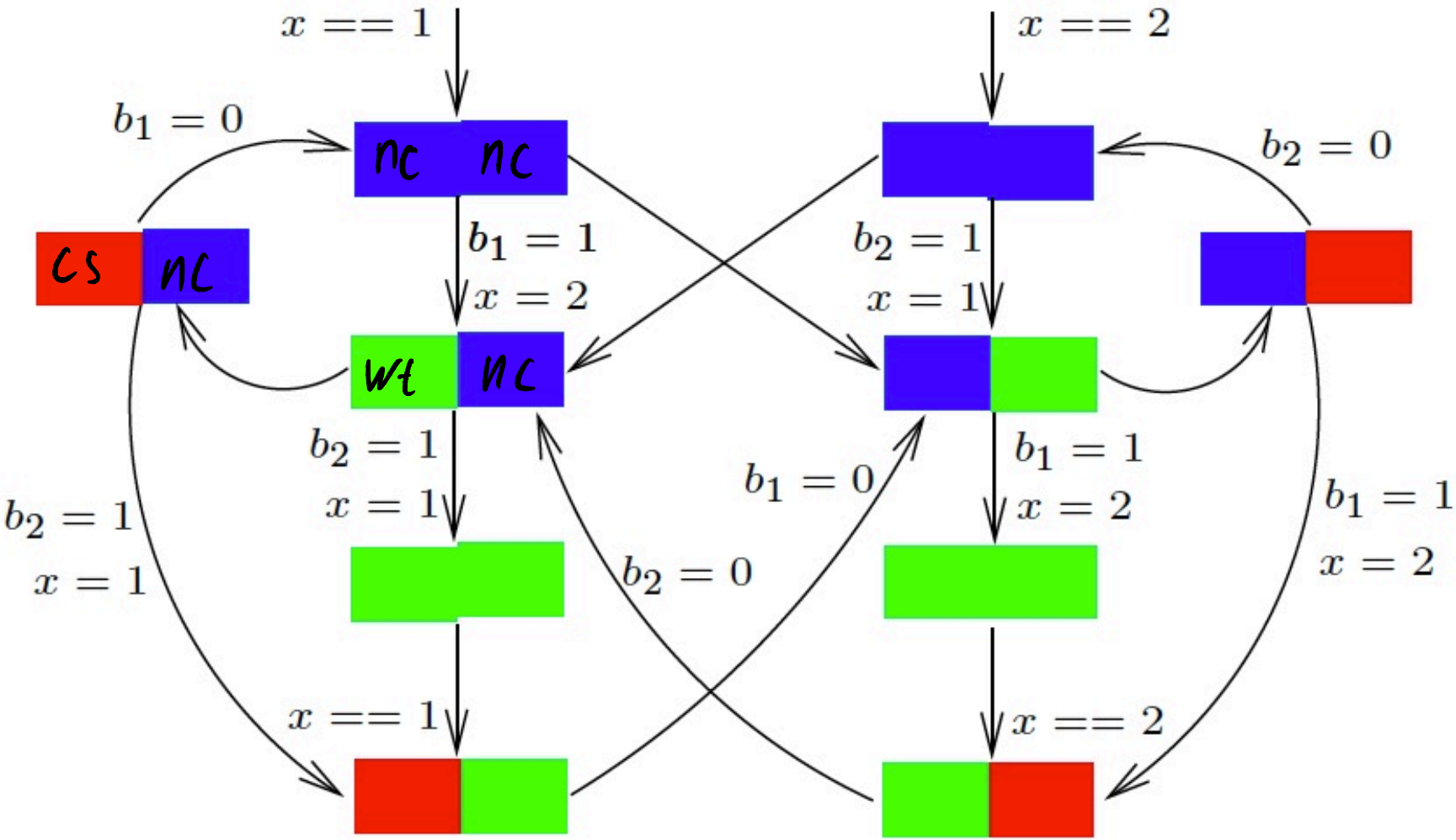
```
        while true {
            ......
nc :    ⟨b1 := true; x := 2;⟩
wt :    wait until(x == 1 || ¬ b2){
cs :        ...@account...}
            b1 = false;
            ......
        }
```

Thread Right behaves as follows:

```
        while true {
            ......
nc :    ⟨b2 := true; x := 1;⟩
wt :    wait until(x == 2 || ¬ b1){
cs :        ...@account...}
            b2 = false;
            ......
        }
```

Does only one thread at a time has access to the bank account?

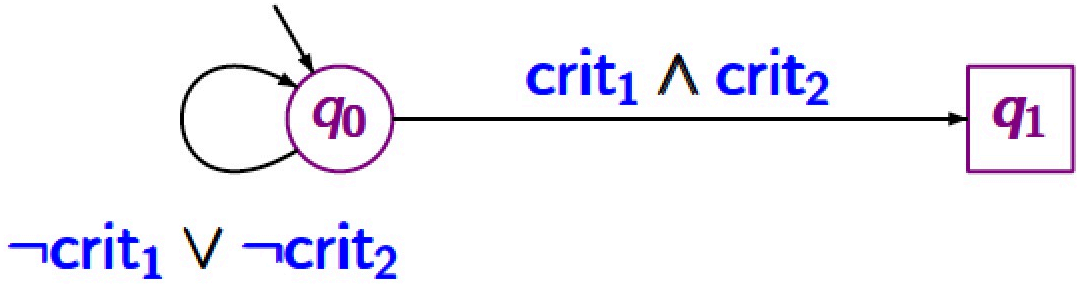
# Peterson's Transition System




Manual inspection reveals that mutual exclusion is guaranteed

# Verifying Mutual Exclusion

- Mutual exclusion = no simultaneous access to the account
- Minimal Bad prefix NFA $\mathfrak{A}$ :




- Checking mutual exclusion:

$$\underbrace{\mathit{Traces}_{\mathit{fin}}(\mathit{TS}_{\mathit{Pet}})}_{\text{finite traces}} \cap \underbrace{\mathit{BadPref}(E_{\mathit{safe}})}_{\mathfrak{L}(\mathfrak{A})} = \varnothing ?$$

- Intersection, complementation and emptiness of $\underbrace{\text{finite automata}}_{\text{accept finite words}}$

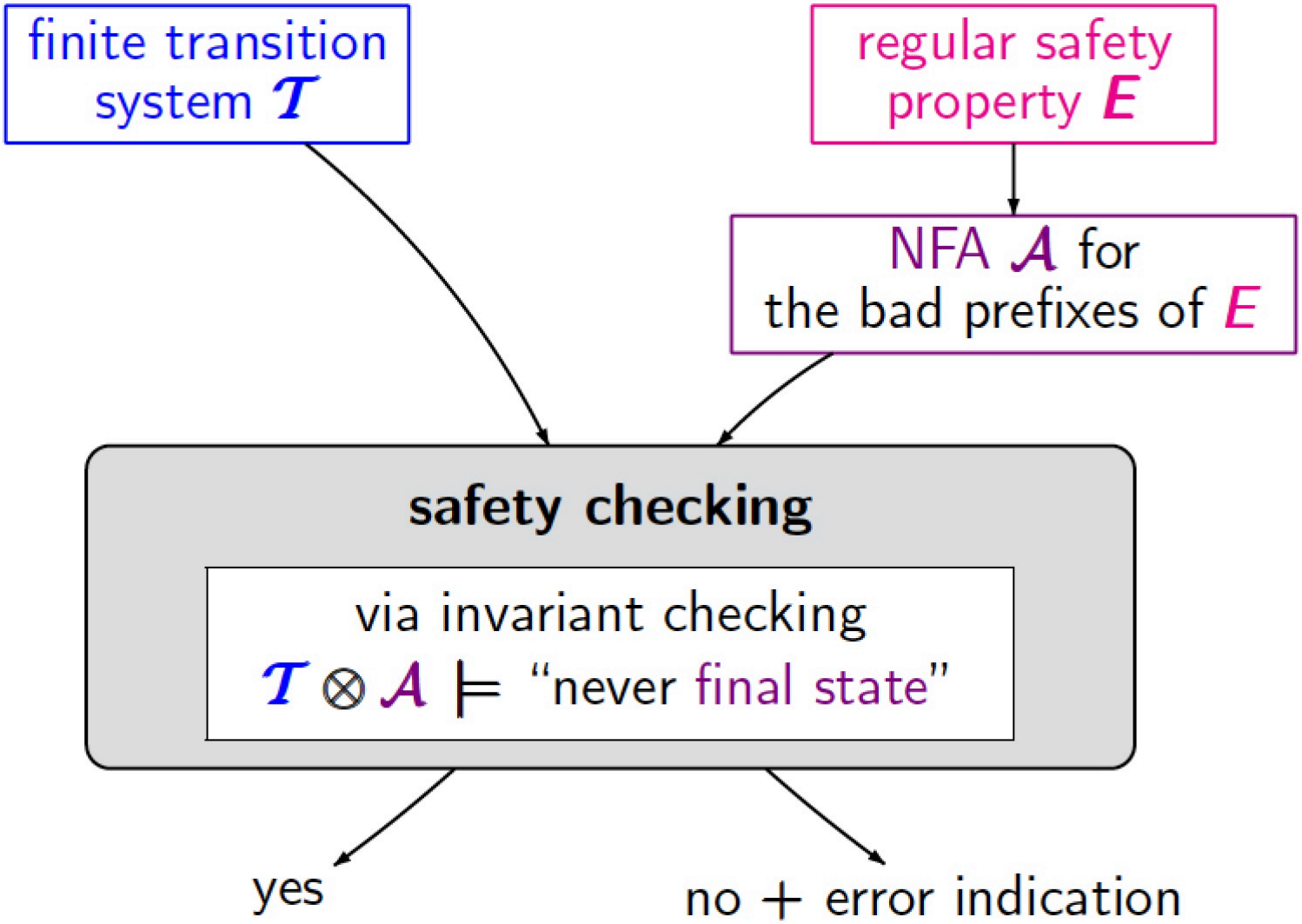
# Problem Statement

Let

1. $E_{safe}$ be a regular safety property over $AP$
2. $\mathfrak{A}$ be an NFA (or DFA) recognizing the bad prefixes of $E_{safe}$
   - with $\varepsilon \notin \mathfrak{L}(\mathfrak{A})$
   - otherwise all finite words over $2^{AP}$ are bad prefixes and $E_{safe} = \varnothing$
3. $TS$ be a finite transition system (over $AP$) without terminal states

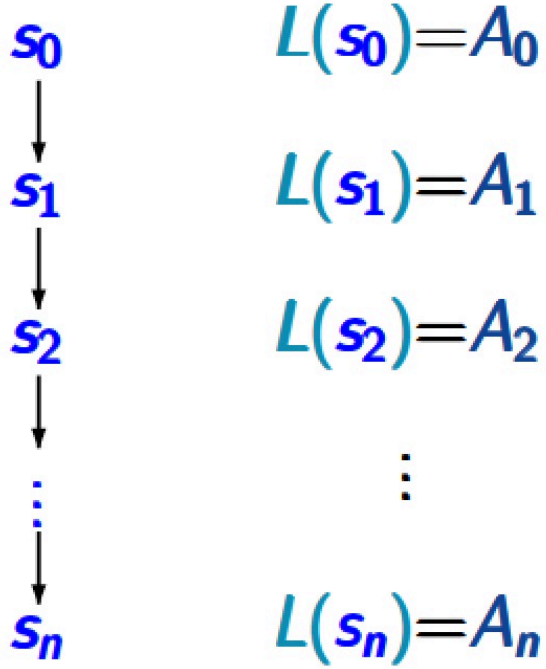
How to establish whether $TS \models E_{safe}$?

# Verifying Regular Safety Properties



# Product: Idea (1)

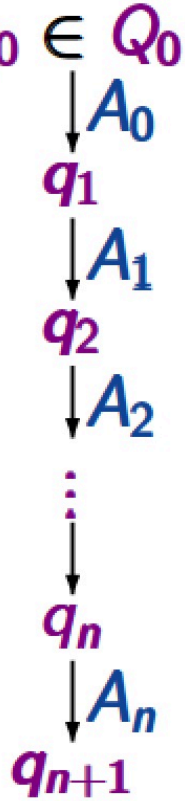
finite transition system
$\mathcal{T} = (S, Act, \rightarrow, S_0, AP, L)$




path fragment $\hat{\pi}$

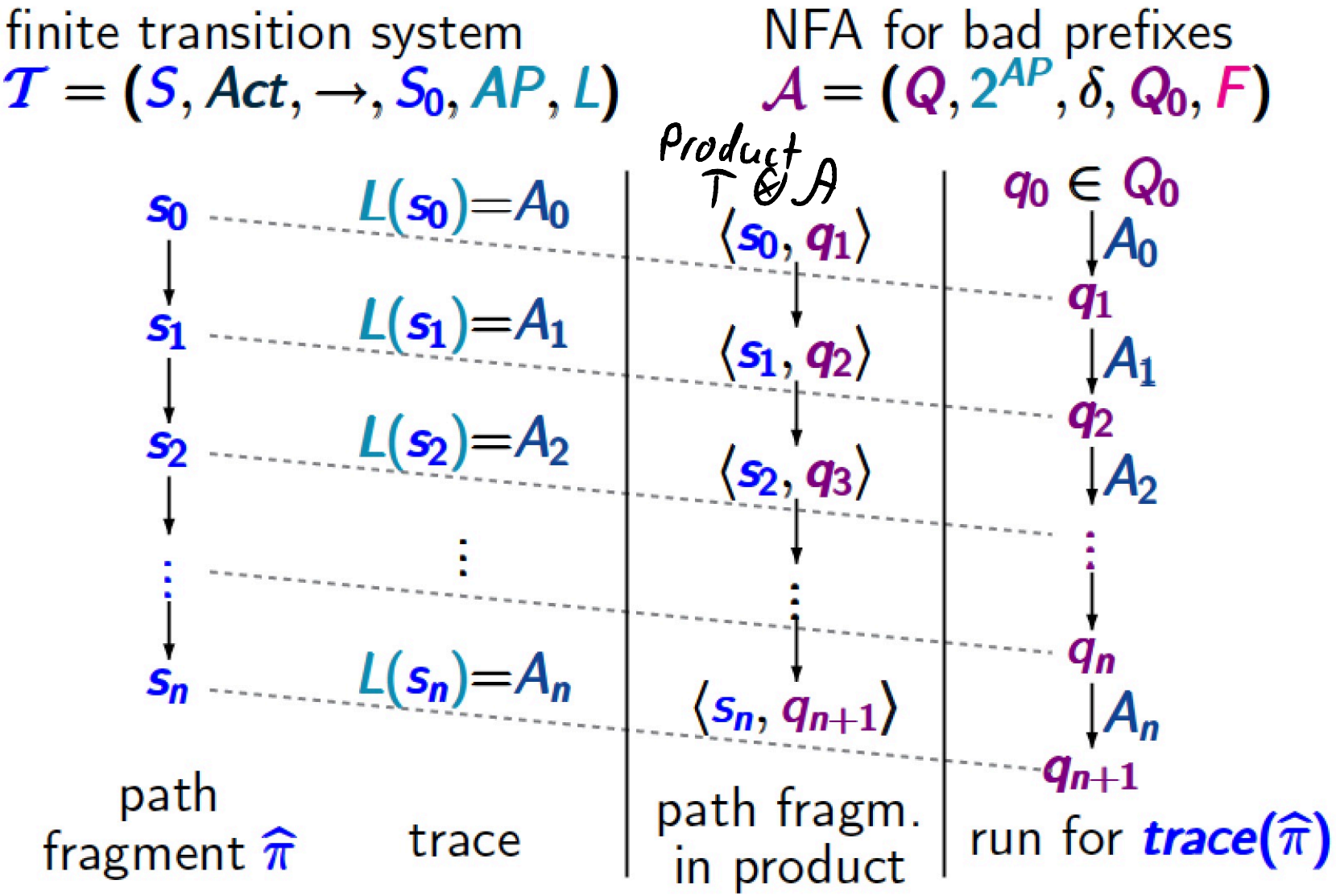
trace

NFA for bad prefixes
$\mathcal{A} = (Q, 2^{AP}, \delta, Q_0, F)$




run for $trace(\hat{\pi})$

# Product: Idea (2)



# Synchronous Product

## Definition: synchronous product of TS and NFA

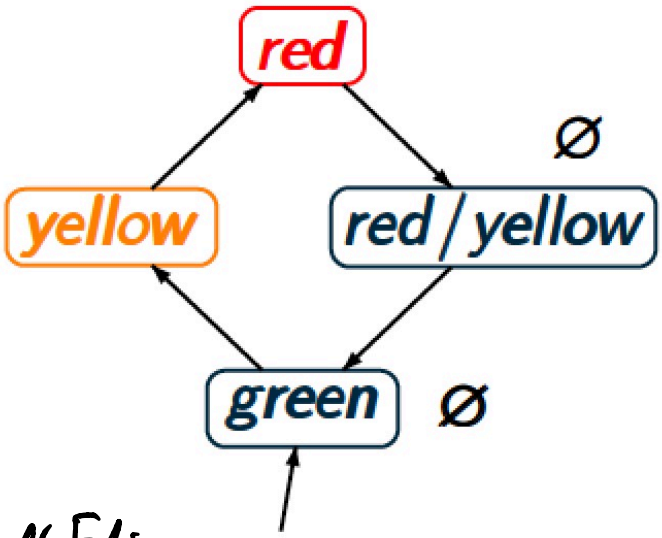
Let transition system $TS = (S, Act, \rightarrow, I, AP, L)$ without terminal states and $\mathfrak{A} = (Q, \Sigma, \delta, Q_0, F)$ an NFA with $\Sigma = 2^{AP}$ and $Q_0 \cap F = \varnothing$. The product of $TS$ and $\mathfrak{A}$ is the transition system:

$$TS \otimes \mathfrak{A} \;=\; (S', Act, \rightarrow', I', AP', L') \qquad \text{where}$$

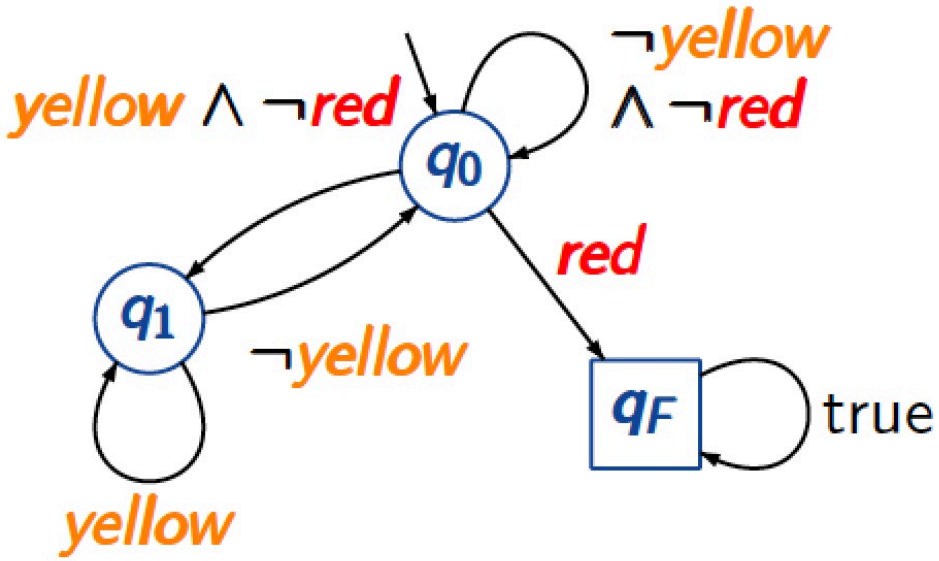
- $S' = S \times Q$, $AP' = Q$ and $L'(\langle s, q \rangle) \;=\; \{ q \}$
- $\rightarrow'$ is the smallest relation defined by: $$\frac{s \xrightarrow{\alpha} t \;\wedge\; q \xrightarrow{L(t)} p}{\langle s, q \rangle \xrightarrow{\alpha}' \langle t, p \rangle}$$
- $I' = \{ \langle s_0, q \rangle \mid s_0 \in I \wedge \exists q_0 \in Q_0.\; q_0 \xrightarrow{L(s_0)} q \}$.

# Example

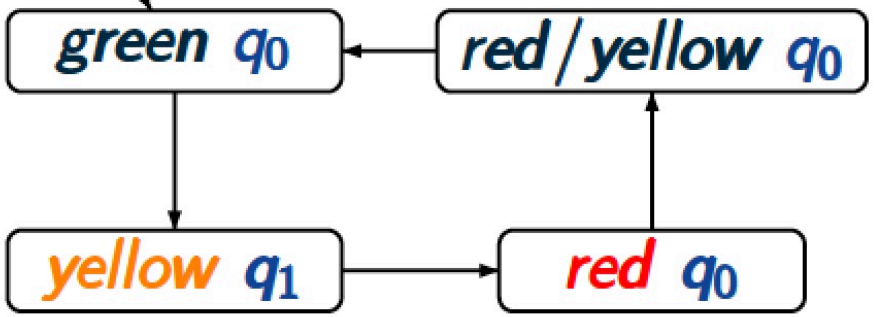
TS :




NFA:




TS ⊗ NFA:




product-TS

$\mathcal{T} \otimes \mathcal{A}$

$4 * 3 = 12$ states, but just 4 reachable states

# A Note on Terminal States

It may be safely assumed that $TS \otimes \mathfrak{A}$ has no terminal states

- Although $TS$ has no terminal state, $TS \otimes \mathfrak{A}$ may have one
- This can only occur if $\delta(q, A) = \varnothing$ for some $A \subseteq AP$
- Let NFA $\mathfrak{A}$ with some reachable state $q$ with $\delta(q, A) = \varnothing$
- Obtain an equivalent NFA $\mathfrak{A}'$ as follows:
  - introduce new state $q_{trap} \notin Q$
  - if $\delta(q, A) = \varnothing$ let $\delta'(q, A) = \{ q_{trap} \}$
  - set $\delta'(q_{trap}, A) = \{ q_{trap} \}$ for all $A \subseteq AP$
  - keep all other transitions that are present in $\mathfrak{A}$
- It follows $\mathfrak{L}(\mathfrak{A}) = \mathfrak{L}(\mathfrak{A}')$

# Verifying Regular Safety Properties

**Theorem**

Let $TS$ over $AP$, $E_{safe}$ a safety property such that $\mathfrak{L}(\mathfrak{A}) = BadPref(E_{safe})$ for some NFA $\mathfrak{A}$. Then:

$$\underbrace{TS \models E_{safe}}_{(1)} \text{ iff } \underbrace{Traces_{fin}(TS) \cap \mathfrak{L}(\mathfrak{A}) = \varnothing}_{(2)} \text{ iff } \underbrace{TS \otimes \mathfrak{A} \models \underbrace{\text{always } \neg F}_{\text{invariant}}}_{(3)}$$

where $F$ stands for $\bigvee_{q \in F} q$.

for some NFA $\mathfrak{A}$. Then:

$$\underbrace{TS \models E_{safe}}_{(1)} \text{ iff } \underbrace{Traces_{fin}(TS) \cap \mathfrak{L}(\mathfrak{A}) = \varnothing}_{(2)} \text{ iff } \underbrace{TS \otimes \mathfrak{A} \models \underbrace{\text{always } \neg F}_{\text{invariant}}}_{(3)}$$

where $F$ stands for $\bigvee_{a \in \ldots} a$

"(1) $\Rightarrow$ (2)"

$TS \not\models E_{safe}$

$\Leftrightarrow Traces(TS) \not\subseteq E_{safe}$

$\Leftrightarrow Traces(TS) \cap (2^{AP})^\omega \setminus E_{safe} \neq \varnothing$

$\Leftrightarrow Traces(TS) \cap BadPref(E_{safe}) \cdot (2^{AP})^\omega \neq \varnothing$

$\Leftrightarrow Traces_{fin}(TS) \cap BadPref(E_{safe}) \neq \varnothing$

$\Leftrightarrow Traces_{fin}(TS) \cap L(\mathfrak{A}) \neq \varnothing$

for some NFA $\mathfrak{A}$. Then:

$TS \models E_{safe}$ iff $Traces_{fin}(TS) \cap \mathfrak{L}(\mathfrak{A}) = \emptyset$ iff $TS \otimes \mathfrak{A} \models$ always $\neg F$ (invariant)

(1) (2) (3)

where $F$ stands for $V_{\ldots} q$

"(2) $\Rightarrow$ (3)", actually $\neg(3) \Rightarrow \neg(2)$

$\neg(3) \Rightarrow \exists \pi \in Paths_{fin}(TS \otimes \mathfrak{A})$ with

$\pi = \langle s_0, q_1\rangle \to \langle s_1, q_2\rangle \cdots \to \langle s_n, q_{n+1}\rangle$,

$q_{n+1} \in F$

$s_0 \to s_1 \to \ldots \to s_n \in Paths_{fin}(TS)$

$L(s_0) \to \ldots L(s_n) \in Traces_{fin}(TS)$

of $TS \otimes \mathfrak{A}$

- $\langle s_0, q_1\rangle \in I' \rightsquigarrow \exists q_0 \in Q_0 : q_0 \xrightarrow{L(s_0)} q_1$
- $\forall i > 0$: $q_i \xrightarrow{L(s_i)} q_{i+1}$ is a transition in $\mathfrak{A}$

$\rightsquigarrow q_0 q_1 \ldots q_{n+1}$ is an acc. ran in $\mathfrak{A}$

on $L(s_0) \ldots L(s_n) =: \sigma$

$\rightsquigarrow \sigma \in Traces_{fin}(TS) \cap L(\mathfrak{A}) \neq \emptyset$

"(3) $\Rightarrow$ (1)" is similar

# Counterexamples

For each initial path fragment $\langle s_0, q_1 \rangle \ldots \langle s_n, q_{n+1} \rangle$ of $TS \otimes \mathfrak{A}$:

$$q_1, \ldots, q_n \notin F \text{ and } q_{n+1} \in F \quad \Rightarrow \quad \underbrace{trace(s_0\, s_1 \ldots s_n)}_{\text{bad prefix for } E_{safe}} \in \mathfrak{L}(\mathfrak{A}).$$

# Complexity of Verifying Regular Safety Properties

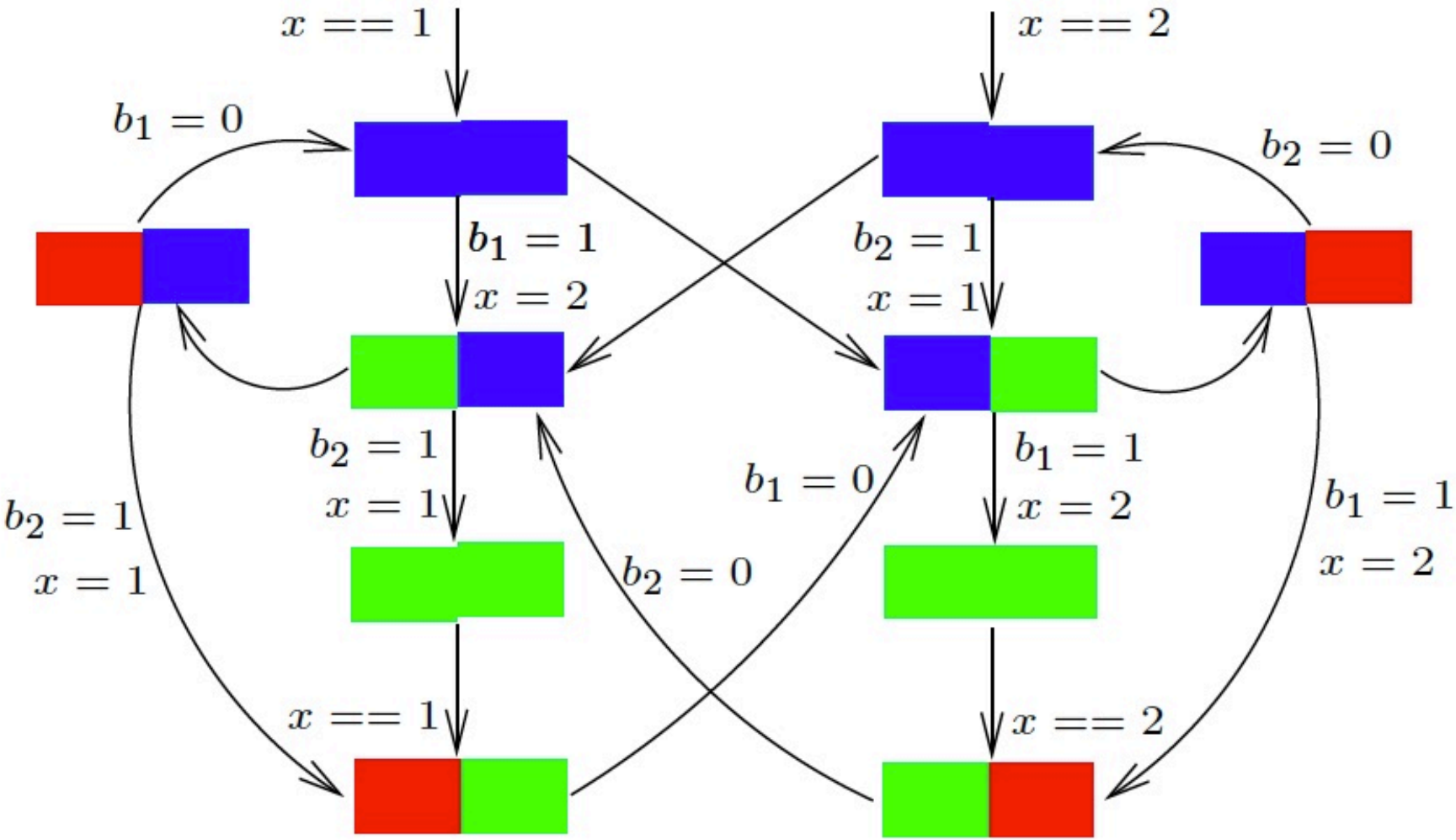
The time and space complexity of checking $TS \models E_{safe}$ is in $O(|TS| \cdot |\mathfrak{A}|)$ where $\mathfrak{A}$ is an NFA with $\mathfrak{L}(\mathfrak{A}) = BadPref(E_{safe})$ and $|\mathfrak{A}|$ is the size of $\mathfrak{A}$.

The size of NFA $\mathfrak{A}$ is the number of states and transitions in $\mathfrak{A}$:

$$|\mathfrak{A}| \;=\; |Q| + \sum_{q \in Q} \sum_{A \in \Sigma} |\,\delta(q, A)\,|$$

# Overview

# Peterson's Transition System

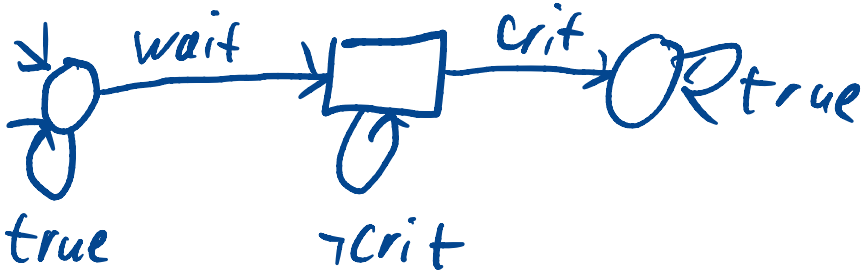



If a thread wants to update the account, does it ever get the opportunity to do so?

# Verifying Starvation Freedom

- Starvation freedom = when a thread wants access to account, it eventually gets it

- "Infinite bad prefix" automaton: once a thread wants access to the account, it never gets it




not an NFA

# Verifying Starvation Freedom

- Starvation freedom = when a thread wants access to account, it eventually gets it
- "Infinite bad prefix" automaton: once a thread wants access to the account, it never gets it
- Checking starvation freedom:

$$\underbrace{\mathit{Traces}(\mathit{TS}_{\mathit{Pet}})}_{\text{infinite traces}} \cap \mathfrak{L}_{\omega}(\overline{E_{\mathit{live}}}) = \varnothing?$$

- Intersection, complementation and emptiness of $\underbrace{\text{Büchi automata}}_{\text{accept infinite words}}$

# $\omega$-Regular Expressions: Syntax

## Definition: $\omega$-regular expression

An $\omega$-regular expression G over the alphabet $\Sigma$ has the form:

$$G = E_1.F_1^\omega + \ldots + E_n.F_n^\omega \quad \text{for } n \in \mathbb{N}_{>0}$$

where $E_i, F_i$ are regular expressions over $\Sigma$ with $\varepsilon \notin \mathfrak{L}(F_i)$.

- $\omega$-Regular expressions denote languages of infinite words
- Examples over the alphabet $\Sigma = \{ A, B \}$:
  - language of all words with infinitely many $A$s: $(B^*.A)^\omega$
  - language of all words with finitely many $A$s: $(A + B)^*.B^\omega$
  - the empty language $\varnothing^\omega$

# $\omega$-Regular Expressions: Semantics

## Definition: semantics of $\omega$-regular expressions

The semantics of $\omega$-regular expression

$$G \;=\; E_1.F_1^{\omega} \;+\; \ldots \;+\; E_n.F_n^{\omega}$$

is the language $\mathfrak{L}(G) \subseteq \Sigma^{\omega}$ defined by:

$$\mathfrak{L}_{\omega}(G) \;=\; \mathfrak{L}(E_1).\mathfrak{L}(F_1)^{\omega} \;\cup\; \ldots \;\cup\; \mathfrak{L}(E_n).\mathfrak{L}(F_n)^{\omega}.$$

where for $\mathfrak{L} \subseteq \Sigma^{*}$, we have $\mathfrak{L}^{\omega} \;=\; \{\, w_1 w_2 w_3 \ldots \mid \forall i \geq 0.\, w_i \in \mathfrak{L} \,\}$.

The $\omega$-regular expression $G_1$ and $G_2$ are equivalent,

denoted $G_1 \equiv G_2$, if $\mathfrak{L}_{\omega}(G_1) = \mathfrak{L}_{\omega}(G_2)$.

# $\omega$-Regular Properties

## Definition: $\omega$-regular language

The set $\mathfrak{L}$ of infinite words over the alphabet $\Sigma$ is $\omega$-regular if $\mathfrak{L} = \mathfrak{L}_\omega(G)$ for some $\omega$-regular expression G over $\Sigma$.

## Definition: $\omega$-regular properties

LT property $E$ over $AP$ is $\omega$-regular if $E$ is an $\omega$-regular language over $2^{AP}$.

We will see that this is equivalent to:

LT property $E$ over $AP$ is $\omega$-regular if $E$ is accepted by a non-deterministic Büchi automaton (over the alphabet $2^{AP}$).

But not by a deterministic Büchi automaton.

# Example $\omega$-Regular Properties

- Any invariant $E$ is an $\omega$-regular property
  - $\Phi^\omega$ describes $E$ with invariant condition $\Phi$

- Any regular safety property $E$ is an $\omega$-regular property
  - $\overline{E} = \mathit{BadPref}(E).\left(2^{AP}\right)^\omega$ is $\omega$-regular
  - and $\omega$-regular languages are closed under complement

- Let $\Sigma = \{ a, b \}$ Then:
  - Infinitely often $a$:

$$\left((\varnothing + \{ b \})^*.(\{ a \} + \{ a, b \})\right)^\omega$$

  - eventually $a$:

$$\left(2^{AP}\right)^* .(\{ a \} + \{ a, b \}).\left(2^{AP}\right)^\omega$$

co-Safety Property

# Shorthand Notation

*Examples* for $AP = \{a, b\}$

- invariant with invariant condition $a \vee \neg b$

$$(a \vee \neg b)^\omega \quad \widehat{=} \quad (\emptyset + \{a\} + \{a, b\})^\omega$$

- "infinitely often $a$"

$$((\neg a)^*.a)^\omega \quad \widehat{=} \quad ((\emptyset + \{b\})^*.(\{a\} + \{a, b\}))^\omega$$

- "from some moment on $a$":

$$true^*.a^\omega$$

- "whenever $a$ then $b$ will hold somewhen later"

$$((\neg a)^*.a.true^*.b)^*.(\neg a)^\omega + ((\neg a)^*.a.true^*.b)^\omega$$

a finitely often

a inf. often

# Overview

# Verifying $\omega$-Regular Properties

**Theorem**

Let $TS$ over $AP$, $E$ an $\omega$-regular property and NBA $\mathfrak{A}$ with $\mathfrak{L}_\omega(\mathfrak{A}) = \overline{E}$. Then:

$$TS \models E \text{ iff } \mathit{Traces}(TS) \cap \mathfrak{L}_\omega(\mathfrak{A}) = \varnothing \text{ iff } TS \otimes \mathfrak{A} \models \underbrace{\text{eventually forever} \neg F}_{\text{persistence property}}$$

where $F$ stands for $\bigvee_{q \in F} q$.

**Proof.**

Next lecture. □

# Next Lecture

Thursday April 28, 12:30