Overview

Regular LT properties

Regular LT properties

Idea: define regular LT properties to be those languages of infinite words over the alphabet 2^{AP} that have a representation by a finite automata

Regular LT properties

Idea: define regular LT properties to be those languages of infinite words over the alphabet 2^{AP} that have a representation by a finite automata

- regular safety properties:
  NFA-representation for the bad prefixes

Idea: define regular LT properties to be those languages of infinite words over the alphabet 2^{AP} that have a representation by a finite automata

- regular safety properties:
  NFA-representation for the bad prefixes
- other regular LT properties:
  representation by ω-automata, i.e., acceptors for infinite words

Overview

Introduction

Modelling parallel systems

Linear Time Properties

Regular Properties

- regular safety properties ⟵
- ω-regular properties
- model checking with Büchi automata

Linear Temporal Logic

Computation-Tree Logic

Equivalences and Abstraction

Let E be a LT property over AP, i.e., $E \subseteq (2^{AP})^\omega$.

E is called a safety property if for all words

$$\sigma = A_0 A_1 A_2 \ldots \in (2^{AP})^\omega \setminus E$$

there exists a finite prefix $A_0 A_1 \ldots A_n$ of σ such that *none* of the words $A_0 A_1 \ldots A_n B_{n+1} B_{n+2} B_{n+3} \ldots$ belongs to E, i.e.,

$$E \cap \{\sigma' \in (2^{AP})^\omega : A_0 \ldots A_n \text{ is a prefix of } \sigma'\} = \varnothing$$

Such words $A_0 A_1 \ldots A_n$ are called bad prefixes for E.

$$BadPref \stackrel{\text{def}}{=} \text{set of bad prefixes for } E \subseteq (2^{AP})^+$$

Regular safety properties

Regular safety properties

Let $E \subseteq (2^{AP})^\omega$ be a safety property.

E is called regular iff the language

$BadPref$ = set of all bad prefixes for E

is regular.

Regular safety properties

Let $E \subseteq (2^{AP})^\omega$ be a safety property.

E is called regular iff the language

$$BadPref = \text{set of all bad prefixes for } E \subseteq (2^{AP})^+$$

is regular.

Let $E \subseteq (2^{AP})^\omega$ be a safety property.

E is called regular iff the language

$$BadPref = \text{set of all bad prefixes for } E \subseteq (2^{AP})^+$$

$$BadPref = \mathcal{L}(\mathcal{A}) \text{ for some NFA } \mathcal{A}$$
over the alphabet 2^{AP}

is regular.

Nondeterministic finite automata (NFA)

IS2.5-15

Nondeterministic finite automata (NFA)

NFA $\mathcal{A} = (Q, \Sigma, \delta, Q_0, F)$

- Q finite set of states
- Σ alphabet
- $\delta : Q \times \Sigma \to 2^Q$ transition relation
- $Q_0 \subseteq Q$ set of initial states
- $F \subseteq Q$ set of final states, also called accept states

NFA $\mathcal{A} = (Q, \Sigma, \delta, Q_0, F)$

- Q finite set of states
- Σ alphabet
- $\delta : Q \times \Sigma \to 2^Q$ transition relation
- $Q_0 \subseteq Q$ set of initial states
- $F \subseteq Q$ set of final states, also called accept states

run for a word $A_0 A_1 \ldots A_{n-1} \in \Sigma^*$:

state sequence $\pi = q_0\, q_1 \ldots q_n$ where $q_0 \in Q_0$
and $q_{i+1} \in \delta(q_i, A_i)$ for $0 \leq i < n$

Nondeterministic finite automata (NFA)

NFA $\mathcal{A} = (Q, \Sigma, \delta, Q_0, F)$

- Q finite set of states
- Σ alphabet
- $\delta : Q \times \Sigma \rightarrow 2^Q$ transition relation
- $Q_0 \subseteq Q$ set of initial states
- $F \subseteq Q$ set of final states, also called accept states

run for a word $A_0 A_1 \ldots A_{n-1} \in \Sigma^*$:

state sequence $\pi = q_0\, q_1 \ldots q_n$ where $q_0 \in Q_0$
and $q_{i+1} \in \delta(q_i, A_i)$ for $0 \leq i < n$

run π is called accepting if $q_n \in F$

NFA $\mathcal{A} = (Q, \Sigma, \delta, Q_0, F)$

- Q finite set of states
- Σ alphabet
- $\delta : Q \times \Sigma \rightarrow 2^Q$ transition relation
- $Q_0 \subseteq Q$ set of initial states
- $F \subseteq Q$ set of final states, also called accept states

accepted language $\mathcal{L}(\mathcal{A}) \subseteq \Sigma^*$ is given by:

$\mathcal{L}(\mathcal{A})$ = set of finite words over Σ that have an accepting run in $\mathcal{A}$

Nondeterministic finite automata (NFA)

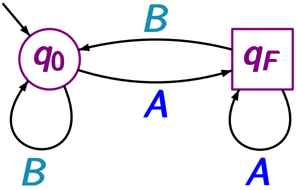
NFA $\mathcal{A} = (Q, \Sigma, \delta, Q_0, F)$

- Q finite set of states
- Σ alphabet $\longleftarrow$ here: $\Sigma = 2^{AP}$
- $\delta : Q \times \Sigma \rightarrow 2^Q$ transition relation
- $Q_0 \subseteq Q$ set of initial states
- $F \subseteq Q$ set of final states, also called accept states

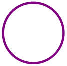
accepted language $\mathcal{L}(\mathcal{A}) \subseteq \Sigma^*$ is given by:

$\mathcal{L}(\mathcal{A})$ = set of finite words over Σ that have an accepting run in $\mathcal{A}$

Notations in pictures for NFA

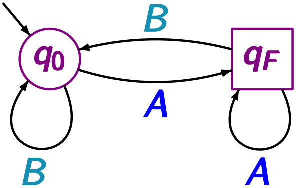


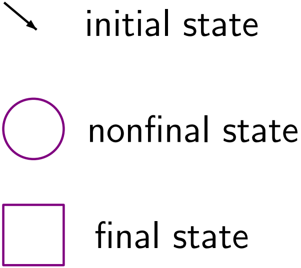
 initial state

 nonfinal state

 final state

Notations in pictures for NFA





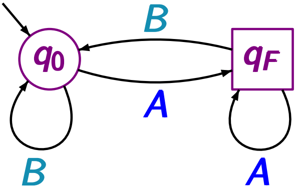
NFA $\mathcal{A}$ with state space $\{q_0, q_F\}$

q_0 initial state

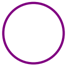
q_F final state

alphabet $\Sigma = \{A, B\}$

Notations in pictures for NFA



 initial state

 nonfinal state

 final state

accepted language $\mathcal{L}(\mathcal{A})$:

set of all finite words over $\{A, B\}$
ending with letter A

Symbolic notations

for transitions in **NFA** over the alphabet $\Sigma = 2^{AP}$

Symbolic notations

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

symbolic notation for the labels of transitions:

If Φ is a propositional formula over AP then

$q \xrightarrow{\Phi} p$ stands for the set of transitions $q \xrightarrow{A} p$

where $A \subseteq AP$ such that $A \models \Phi$

Symbolic notations

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

symbolic notation for the labels of transitions:

> If Φ is a propositional formula over AP then
>
> $q \xrightarrow{\Phi} p$ stands for the set of transitions $q \xrightarrow{A} p$
>
> where $A \subseteq AP$ such that $A \models \Phi$

Example: if $AP = \{a, b, c\}$ then

$$q \xrightarrow{a \wedge \neg b} p \;\widehat{=}\; \{\, q \xrightarrow{A} p : A{=}\{a, c\} \text{ or } A{=}\{a\} \,\}$$

Symbolic notations

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

symbolic notation for the labels of transitions:

> If Φ is a propositional formula over AP then
>
> $q \xrightarrow{\Phi} p$ stands for the set of transitions $q \xrightarrow{A} p$
>
> where $A \subseteq AP$ such that $A \models \Phi$

Example: if $AP = \{a, b, c\}$ then

$$q \xrightarrow{a \wedge \neg b} p \;\widehat{=}\; \left\{ q \xrightarrow{A} p : A=\{a, c\} \text{ or } A=\{a\} \right\}$$

$$q \xrightarrow{true} p \;\widehat{=}\; \left\{ q \xrightarrow{A} p : A \subseteq AP \right\}$$

Symbolic notations

$$\Sigma = \{A, B\}$$

$$q \xrightarrow{A} p$$

$$q \xrightarrow{B} p$$

$$AP = \{a, b\}$$

$$\Sigma = 2^{AP}$$

$$q \xrightarrow{\{\}} p$$

$$q \xrightarrow{\{a\}} p \qquad q \xrightarrow{\{b\}} p$$

$$q \xrightarrow{\{a, b\}} p$$

$$q \xrightarrow{a} p \ \widehat{=} \left\{ q \xrightarrow{\{a\}} p \quad q \xrightarrow{\{a, b\}} p \right\}$$

Regular safety properties

A safety property $E \subseteq (2^{AP})^\omega$ is called regular iff

$BadPref$ = set of all bad prefixes for $E \subseteq (2^{AP})^+$

$BadPref = \mathcal{L}(\mathcal{A})$ for some NFA $\mathcal{A}$ over the alphabet 2^{AP}

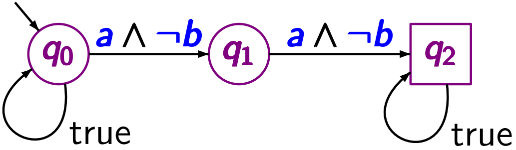
is regular.

A safety property $E \subseteq (2^{AP})^{\omega}$ is called regular iff

$BadPref$ = set of all bad prefixes for $E \subseteq (2^{AP})^{+}$

$BadPref = \mathcal{L}(\mathcal{A})$ for some NFA $\mathcal{A}$ over the alphabet 2^{AP}

is regular.



$AP = \{a, b\}$

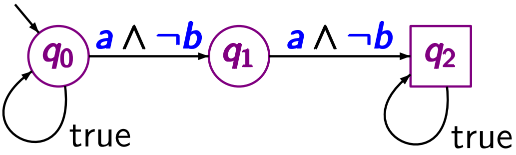
Regular safety properties

A safety property $E \subseteq (2^{AP})^\omega$ is called regular iff

$BadPref$ = set of all bad prefixes for $E \subseteq (2^{AP})^+$

$BadPref = \mathcal{L}(\mathcal{A})$ for some NFA $\mathcal{A}$ over the alphabet 2^{AP}

is regular.



$AP = \{a, b\}$

symbolic notation:

$a \wedge \neg b \mathrel{\widehat{=}} \{a\}$

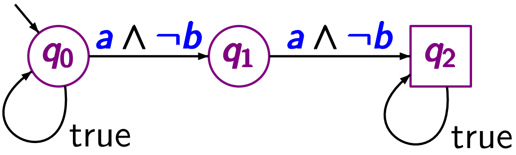
Regular safety properties

A safety property $E \subseteq (2^{AP})^{\omega}$ is called regular iff

$BadPref$ = set of all bad prefixes for $E \subseteq (2^{AP})^{+}$

$BadPref = \mathcal{L}(\mathcal{A})$ for some NFA $\mathcal{A}$ over the alphabet 2^{AP}

is regular.



$AP = \{a, b\}$

symbolic notation:

$a \wedge \neg b \mathrel{\widehat{=}} \{a\}$

safety property E: "$a \wedge \neg b$ never holds twice in a row"

Example: regular safety property

"Every red phase is preceded by a yellow phase"

Example: regular safety property

"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 $i \geq 0$:

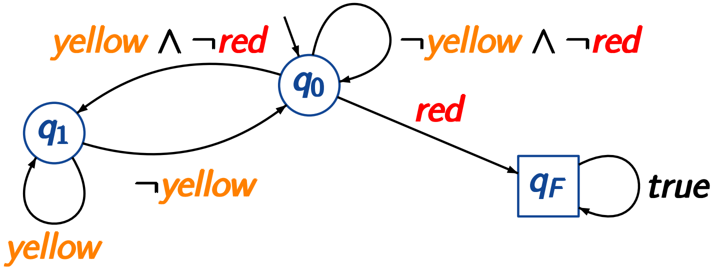
$$red \in A_i \implies i \geq 1 \text{ and } yellow \in A_{i-1}$$

"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



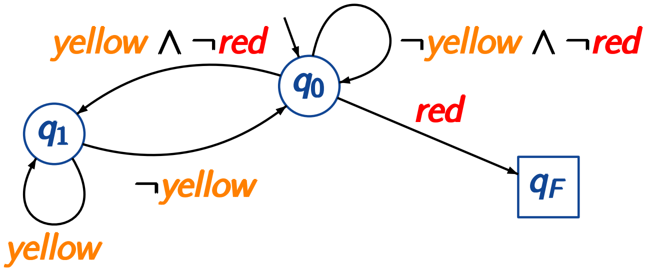
Example: regular safety property

"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 minimal bad prefixes



Bad prefixes vs minimal bad prefixes

Let $E \subseteq \left(2^{AP}\right)^{\omega}$ be a safety property.

$$\begin{array}{lcl} \mathit{BadPref} & = & \text{set of all bad prefixes for } E \\ \mathit{MinBadPref} & = & \text{set of minimal bad prefixes for } E \end{array}$$

Claim: $\mathit{BadPref}$ is regular $\iff$ $\mathit{MinBadPref}$ is regular

Bad prefixes vs minimal bad prefixes

Let $E \subseteq (2^{AP})^\omega$ be a safety property.

$BadPref$ = set of all bad prefixes for E

$MinBadPref$ = set of minimal bad prefixes for E

Claim: $BadPref$ is regular $\Longleftrightarrow$ $MinBadPref$ is regular

"$\Longleftarrow$": Let $\mathcal{A}$ be an NFA for $MinBadPref$.

Bad prefixes vs minimal bad prefixes

Let $E \subseteq (2^{AP})^\omega$ be a safety property.

$BadPref$ = set of all bad prefixes for E

$MinBadPref$ = set of minimal bad prefixes for E

Claim: $BadPref$ is regular $\Longleftrightarrow$ $MinBadPref$ is regular

"$\Longleftarrow$": Let $\mathcal{A}$ be an NFA for $MinBadPref$.

An NFA $\mathcal{A}'$ for $BadPref$ is obtained from $\mathcal{A}$ by adding self-loops $p \xrightarrow{true} p$ to all final states p.

Bad prefixes vs minimal bad prefixes

Let $E \subseteq (2^{AP})^\omega$ be a safety property.

$BadPref \quad = \quad$ set of all bad prefixes for E

$MinBadPref \quad = \quad$ set of minimal bad prefixes for E

Claim: $BadPref$ is regular $\Longleftrightarrow$ $MinBadPref$ is regular

"$\Longleftarrow$": Let $\mathcal{A}$ be an NFA for $MinBadPref$.

An NFA $\mathcal{A}'$ for $BadPref$ is obtained from $\mathcal{A}$ by adding self-loops $p \xrightarrow{true} p$ to all final states p.

"$\Longrightarrow$": Let $\mathcal{A}$ be a DFA for $BadPref$.

Bad prefixes vs minimal bad prefixes

Let $E \subseteq (2^{AP})^\omega$ be a safety property.

$$\begin{array}{lcl} \mathit{BadPref} & = & \text{set of all bad prefixes for } E \\ \mathit{MinBadPref} & = & \text{set of minimal bad prefixes for } E \end{array}$$

> *Claim:* $\mathit{BadPref}$ is regular $\iff$ $\mathit{MinBadPref}$ is regular

"$\Longleftarrow$": Let $\mathcal{A}$ be an NFA for $\mathit{MinBadPref}$.

An NFA $\mathcal{A}'$ for $\mathit{BadPref}$ is obtained from $\mathcal{A}$ by adding self-loops $p \xrightarrow{true} p$ to all final states p.

"$\Longrightarrow$": Let $\mathcal{A}$ be a DFA for $\mathit{BadPref}$.

A DFA $\mathcal{A}'$ for $\mathit{MinBadPref}$ is obtained from $\mathcal{A}$ by removing all outgoing transitions of final states.

Correct or wrong?

Every invariant is regular.

Correct or wrong?

Every invariant is regular.

correct.

Correct or wrong?

Every invariant is regular.

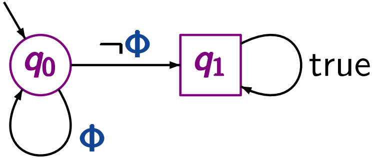
correct.

Let E be an invariant with invariant condition Φ

Every invariant is regular.

correct.

Let E be an invariant with invariant condition Φ



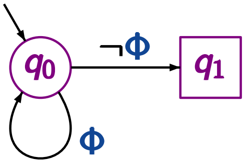
is a DFA for the language of all bad prefixes

Correct or wrong?

Every invariant is regular.

correct.

Let E be an invariant with invariant condition Φ



is a DFA for the language of all minimal bad prefixes

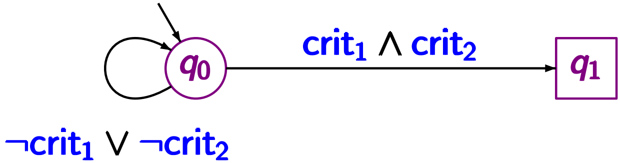
Example: DFA for *MUTEX*

"The two processes are never simultaneously
in their critical sections"

Example: DFA for *MUTEX*

"The two processes are never simultaneously in their critical sections"

DFA for minimal bad prefixes over the alphabet 2^{AP} where $AP = \{crit_1, crit_2\}$



Correct or wrong?

Every safety property is regular.

Correct or wrong?

Every safety property is regular.

wrong.

Correct or wrong?

Every safety property is regular.

wrong. e.g., $AP = \{pay, drink\}$

E = set of alle infinite words $A_0\,A_1\,A_2 \ldots \in (2^{AP})^\omega$ such that for all $j \in \mathbb{N}$:

$$\left|\{i \le j : pay \in A_i\}\right| \ge \left|\{i \le j : drink \in A_i\}\right|$$

Correct or wrong?

Every safety property is regular.

wrong. e.g., $AP = \{pay, drink\}$

E = set of alle infinite words $A_0 A_1 A_2 \ldots \in (2^{AP})^\omega$ such that for all $j \in \mathbb{N}$:

$$\big|\{i \leq j : pay \in A_i\}\big| \geq \big|\{i \leq j : drink \in A_i\}\big|$$

- E is a safety property, but
- the language of (minimal) bad prefixes is *not* regular

Verifying regular safety properties

Verifying regular safety properties

given: finite TS $\mathcal{T}$
regular safety property E
(represented by an **NFA** for its bad prefixes)

question: does $\mathcal{T} \models E$ hold **?**

Verifying regular safety properties

given: finite TS $\mathcal{T}$
regular safety property E
(represented by an **NFA** for its bad prefixes)

question: does $\mathcal{T} \models E$ hold **?**

method: relies on an analogy between the tasks:

- checking language inclusion for **NFA**
- model checking regular safety properties

language inclusion for NFA	verification of regular safety properties
$\mathcal{L}(\mathcal{A}_1) \subseteq \mathcal{L}(\mathcal{A}_2)$ **?**	$Traces(\mathcal{T}) \subseteq E$ **?**

language inclusion for NFA	verification of regular safety properties
$\mathcal{L}(\mathcal{A}_1) \subseteq \mathcal{L}(\mathcal{A}_2)$ **?**	$Traces(\mathcal{T}) \subseteq E$ **?**
check whether $\mathcal{L}(\mathcal{A}_1) \cap (\Sigma^* \setminus \mathcal{L}(\mathcal{A}_2))$ is empty	

language inclusion for NFA	verification of regular safety properties
$\mathcal{L}(\mathcal{A}_1) \subseteq \mathcal{L}(\mathcal{A}_2)$?	$Traces(\mathcal{T}) \subseteq E$?
check whether $\mathcal{L}(\mathcal{A}_1) \cap (\Sigma^* \setminus \mathcal{L}(\mathcal{A}_2))$ is empty	
1. complement $\mathcal{A}_2$, i.e., construct NFA $\overline{\mathcal{A}_2}$ with $\mathcal{L}(\overline{\mathcal{A}_2}) = \Sigma^* \setminus \mathcal{L}(\mathcal{A}_2)$	

language inclusion for NFA	verification of regular safety properties
$\mathcal{L}(\mathcal{A}_1) \subseteq \mathcal{L}(\mathcal{A}_2)$ **?**	$\mathit{Traces}(\mathcal{T}) \subseteq E$ **?**
check whether $\mathcal{L}(\mathcal{A}_1) \cap (\Sigma^* \setminus \mathcal{L}(\mathcal{A}_2))$ is empty	
1. complement $\mathcal{A}_2$, i.e., construct NFA $\overline{\mathcal{A}_2}$ with $\mathcal{L}(\overline{\mathcal{A}_2}) = \Sigma^* \setminus \mathcal{L}(\mathcal{A}_2)$ 2. construct NFA $\mathcal{A}$ with $\mathcal{L}(\mathcal{A}) = \mathcal{L}(\mathcal{A}_1) \cap \mathcal{L}(\overline{\mathcal{A}_2})$	

language inclusion for NFA	verification of regular safety properties
$\mathcal{L}(\mathcal{A}_1) \subseteq \mathcal{L}(\mathcal{A}_2)$ **?**	$\mathit{Traces}(\mathcal{T}) \subseteq E$ **?**
check whether $\mathcal{L}(\mathcal{A}_1) \cap (\Sigma^* \setminus \mathcal{L}(\mathcal{A}_2))$ is empty	
1. complement $\mathcal{A}_2$, i.e., construct NFA $\overline{\mathcal{A}_2}$ with $\mathcal{L}(\overline{\mathcal{A}_2}) = \Sigma^* \setminus \mathcal{L}(\mathcal{A}_2)$ 2. construct NFA $\mathcal{A}$ with $\mathcal{L}(\mathcal{A}) = \mathcal{L}(\mathcal{A}_1) \cap \mathcal{L}(\overline{\mathcal{A}_2})$ 3. check if $\mathcal{L}(\mathcal{A}) = \varnothing$	

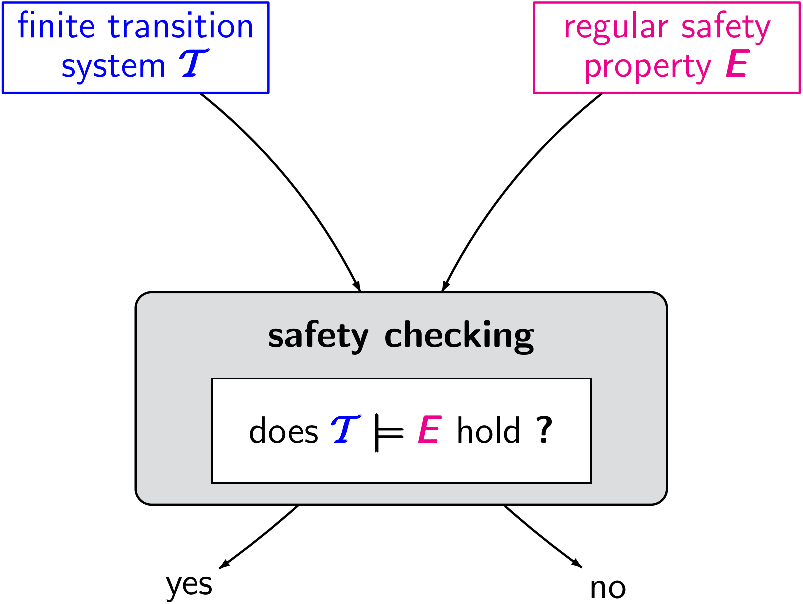
language inclusion for NFA	verification of regular safety properties
$\mathcal{L}(\mathcal{A}_1) \subseteq \mathcal{L}(\mathcal{A}_2)$ **?**	$\mathit{Traces}(\mathcal{T}) \subseteq E$ **?**
check whether $\mathcal{L}(\mathcal{A}_1) \cap (\Sigma^* \setminus \mathcal{L}(\mathcal{A}_2))$ is empty	check whether $\mathit{Traces}_{\mathit{fin}}(\mathcal{T}) \cap \mathit{BadPref}$ is empty
1. complement $\mathcal{A}_2$, i.e., construct NFA $\overline{\mathcal{A}_2}$ with $\mathcal{L}(\overline{\mathcal{A}_2}) = \Sigma^* \setminus \mathcal{L}(\mathcal{A}_2)$ 2. construct NFA $\mathcal{A}$ with $\mathcal{L}(\mathcal{A}) = \mathcal{L}(\mathcal{A}_1) \cap \mathcal{L}(\overline{\mathcal{A}_2})$ 3. check if $\mathcal{L}(\mathcal{A}) = \varnothing$	

language inclusion for NFA	verification of regular safety properties
$\mathcal{L}(\mathcal{A}_1) \subseteq \mathcal{L}(\mathcal{A}_2)$ **?**	$\mathit{Traces}(\mathcal{T}) \subseteq E$ **?**
check whether $\mathcal{L}(\mathcal{A}_1) \cap (\Sigma^* \setminus \mathcal{L}(\mathcal{A}_2))$ is empty	check whether $\mathit{Traces}_{fin}(\mathcal{T}) \cap \mathit{BadPref}$ is empty
1. complement $\mathcal{A}_2$, i.e., construct NFA $\overline{\mathcal{A}_2}$ with $\mathcal{L}(\overline{\mathcal{A}_2}) = \Sigma^* \setminus \mathcal{L}(\mathcal{A}_2)$ 2. construct NFA $\mathcal{A}$ with $\mathcal{L}(\mathcal{A}) = \mathcal{L}(\mathcal{A}_1) \cap \mathcal{L}(\overline{\mathcal{A}_2})$ 3. check if $\mathcal{L}(\mathcal{A}) = \varnothing$	1. construct NFA $\mathcal{A}$ for the bad prefixes $\mathcal{L}(\overline{\mathcal{A}}) = \mathit{BadPref}$

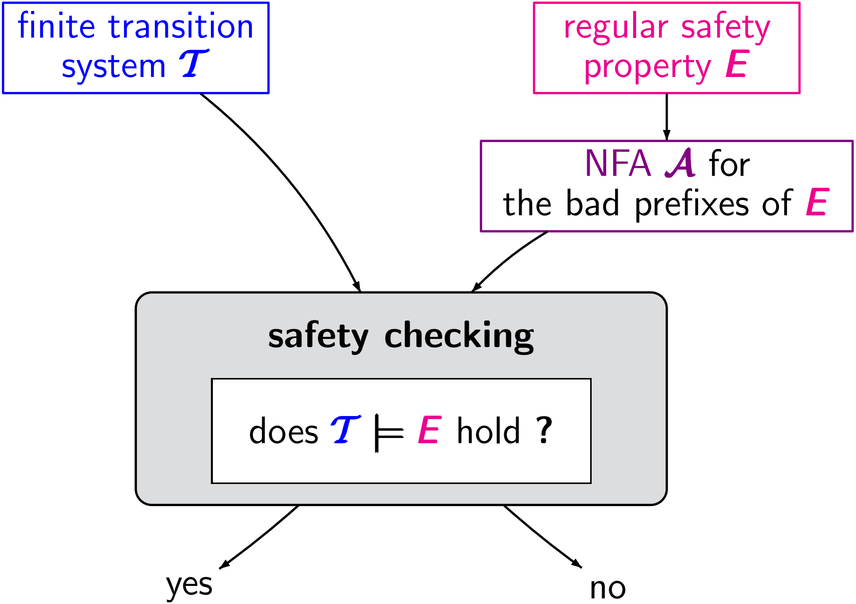
language inclusion for NFA	verification of regular safety properties
$\mathcal{L}(\mathcal{A}_1) \subseteq \mathcal{L}(\mathcal{A}_2)$ **?**	$Traces(\mathcal{T}) \subseteq E$ **?**
check whether $\mathcal{L}(\mathcal{A}_1) \cap (\Sigma^* \setminus \mathcal{L}(\mathcal{A}_2))$ is empty	check whether $Traces_{fin}(\mathcal{T}) \cap BadPref$ is empty
1. complement $\mathcal{A}_2$, i.e., construct NFA $\overline{\mathcal{A}_2}$ with $\mathcal{L}(\overline{\mathcal{A}_2}) = \Sigma^* \setminus \mathcal{L}(\mathcal{A}_2)$ 2. construct NFA $\mathcal{A}$ with $\mathcal{L}(\mathcal{A}) = \mathcal{L}(\mathcal{A}_1) \cap \mathcal{L}(\overline{\mathcal{A}_2})$ 3. check if $\mathcal{L}(\mathcal{A}) = \varnothing$	1. construct NFA $\overline{\mathcal{A}}$ for the bad prefixes $\mathcal{L}(\overline{\mathcal{A}}) = BadPref$ 2. construct TS $\mathcal{T}'$ with $Traces_{fin}(\mathcal{T}') = \ldots$

language inclusion for NFA	verification of regular safety properties
$\mathcal{L}(\mathcal{A}_1) \subseteq \mathcal{L}(\mathcal{A}_2)$ **?**	$Traces(\mathcal{T}) \subseteq E$ **?**
check whether $\mathcal{L}(\mathcal{A}_1) \cap (\Sigma^* \setminus \mathcal{L}(\mathcal{A}_2))$ is empty	check whether $Traces_{fin}(\mathcal{T}) \cap BadPref$ is empty
1. complement $\mathcal{A}_2$, i.e., construct NFA $\overline{\mathcal{A}_2}$ with $\mathcal{L}(\overline{\mathcal{A}_2}) = \Sigma^* \setminus \mathcal{L}(\mathcal{A}_2)$	1. construct NFA $\mathcal{A}$ for the bad prefixes $\mathcal{L}(\overline{\mathcal{A}}) = BadPref$
2. construct NFA $\mathcal{A}$ with $\mathcal{L}(\mathcal{A}) = \mathcal{L}(\mathcal{A}_1) \cap \mathcal{L}(\overline{\mathcal{A}_2})$	2. construct TS $\mathcal{T}'$ with $Traces_{fin}(\mathcal{T}') = \ldots$
3. check if $\mathcal{L}(\mathcal{A}) = \varnothing$	3. invariant checking for $\mathcal{T}'$

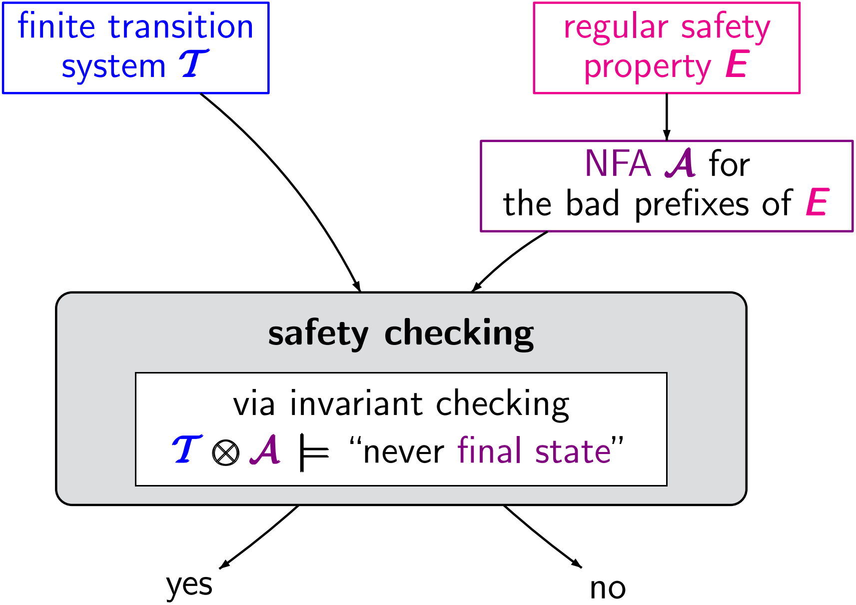
Checking regular safety properties



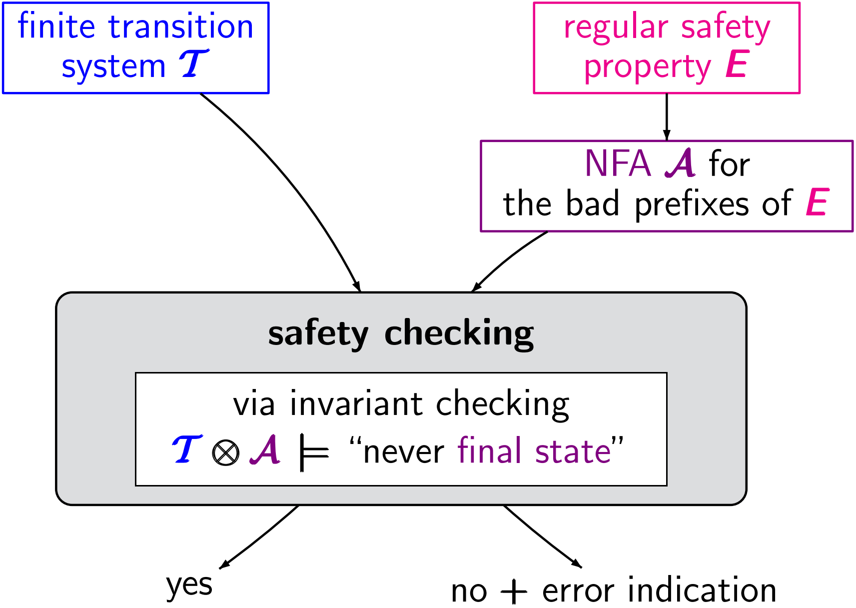
Checking regular safety properties



Checking regular safety properties



Checking regular safety properties



Product of a TS and an NFA

Product of a TS and an NFA

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

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

path
fragment $\widehat{\pi}$

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

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

s_0	$L(s_0) = A_0$
$\downarrow$	
s_1	$L(s_1) = A_1$
$\downarrow$	
s_2	$L(s_2) = A_2$
$\downarrow$	
$\vdots$	$\vdots$
$\downarrow$	
s_n	$L(s_n) = A_n$
path fragment $\widehat{\pi}$	trace

Product of a TS and an NFA

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

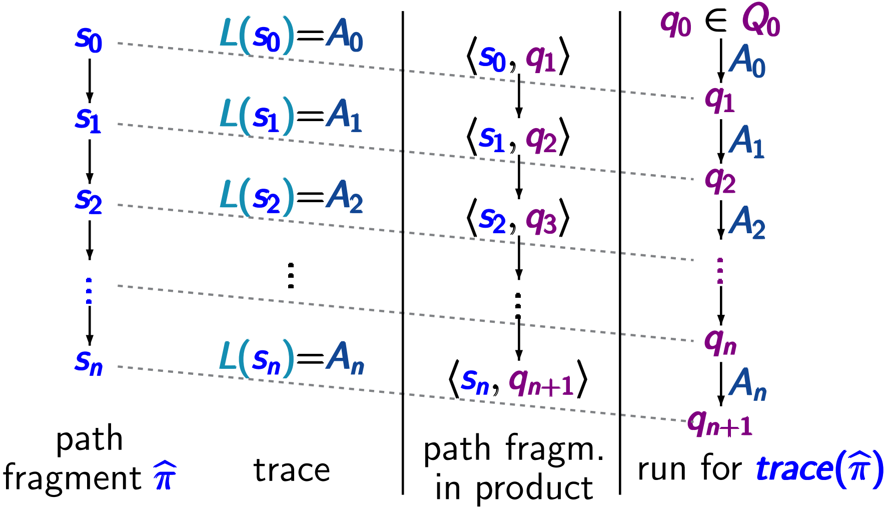
NFA for bad prefixes
$\mathcal{A} = (Q, 2^{AP}, \delta, Q_0, F)$

path fragment $\widehat{\pi}$	trace	run for $trace(\widehat{\pi})$
s_0	$L(s_0) = A_0$	$q_0 \in Q_0$
$\downarrow$		$\downarrow A_0$
s_1	$L(s_1) = A_1$	q_1
$\downarrow$		$\downarrow A_1$
s_2	$L(s_2) = A_2$	q_2
$\downarrow$		$\downarrow A_2$
$\vdots$	$\vdots$	$\vdots$
$\downarrow$		$\downarrow$
s_n	$L(s_n) = A_n$	q_n
		$\downarrow A_n$
		q_{n+1}

Product of a TS and an NFA

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

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



Product transition system

Product transition system

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

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

Product transition system

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

$$\mathcal{A} = (Q, 2^{AP}, \delta, Q_0, F) \quad \text{NFA}$$

product-TS $\mathcal{T} \otimes \mathcal{A} \stackrel{\text{def}}{=} (S \times Q, Act, \longrightarrow', S_0', AP', L')$

Product transition system

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

$$\mathcal{A} = (Q, 2^{AP}, \delta, Q_0, F) \quad \text{NFA}$$

product-TS $\mathcal{T} \otimes \mathcal{A} \overset{\text{def}}{=} (S \times Q, Act, \longrightarrow', S_0', AP', L')$

$$\frac{s \xrightarrow{\alpha} s' \quad \wedge \quad q' \in \delta(q, L(s'))}{\langle s, q \rangle \xrightarrow{\alpha}{}' \langle s', q' \rangle}$$

Product transition system

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

$$\mathcal{A} = (Q, 2^{AP}, \delta, Q_0, F) \quad \text{NFA}$$

product-TS $\mathcal{T} \otimes \mathcal{A} \stackrel{\text{def}}{=} (S \times Q, Act, \longrightarrow', S_0', AP', L')$

$$\frac{s \xrightarrow{\alpha} s' \quad \wedge \quad q' \in \delta(q, L(s'))}{\langle s, q \rangle \xrightarrow{\alpha}{}' \langle s', q' \rangle}$$

initial states: $S_0' = \{ \langle s_0, q \rangle : s_0 \in S_0, q \in \delta\big(Q_0, L(s_0)\big) \}$

Product transition system

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

$$\mathcal{A} = (Q, 2^{AP}, \delta, Q_0, F) \quad \text{NFA}$$

product-TS $\mathcal{T} \otimes \mathcal{A} \stackrel{\text{def}}{=} (S \times Q, Act, \longrightarrow', S_0', AP', L')$

$$\frac{s \xrightarrow{\alpha} s' \quad \wedge \quad q' \in \delta(q, L(s'))}{\langle s, q \rangle \xrightarrow{\alpha}{}' \langle s', q' \rangle}$$

initial states: $S_0' = \{ \langle s_0, q \rangle : s_0 \in S_0, q \in \delta\big(Q_0, L(s_0)\big) \}$

for $P \subseteq Q$ and $A \subseteq AP$: $\delta(P, A) = \bigcup_{p \in P} \delta(p, A)$

Product transition system

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

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

product-TS $\mathcal{T} \otimes \mathcal{A} \stackrel{\text{def}}{=} (S \times Q, Act, \longrightarrow', S_0', AP', L')$

$$\frac{s \xrightarrow{\alpha} s' \quad \wedge \quad q' \in \delta(q, L(s'))}{\langle s, q \rangle \xrightarrow{\alpha}{}' \langle s', q' \rangle}$$

initial states: $S_0' = \{ \langle s_0, q \rangle : s_0 \in S_0, q \in \delta\big(Q_0, L(s_0)\big) \}$

set of atomic propositions: $AP' = Q$

Product transition system

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

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

product-TS $\mathcal{T} \otimes \mathcal{A} \stackrel{\text{def}}{=} (S \times Q, Act, \longrightarrow', S_0', AP', L')$

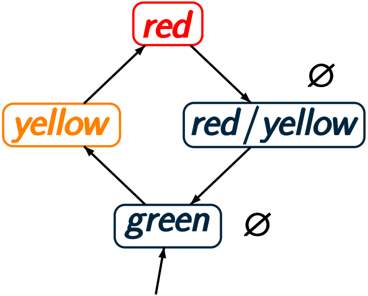
$$\frac{s \stackrel{\alpha}{\longrightarrow} s' \quad \wedge \quad q' \in \delta(q, L(s'))}{\langle s, q \rangle \stackrel{\alpha}{\longrightarrow}' \langle s', q' \rangle}$$

initial states: $S_0' = \{ \langle s_0, q \rangle : s_0 \in S_0, q \in \delta\big(Q_0, L(s_0)\big) \}$

set of atomic propositions: $AP' = Q$

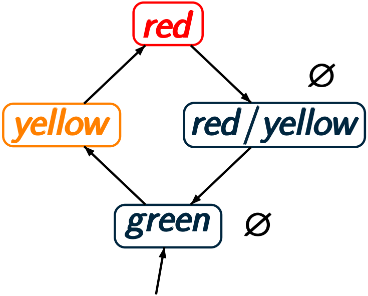
labeling function: $L'(\langle s, q \rangle) = \{q\}$

Example: product-TS



transition system $\mathcal{T}$ over
$AP = \{red, yellow\}$

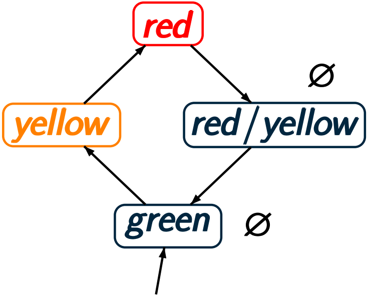
Example: product-TS

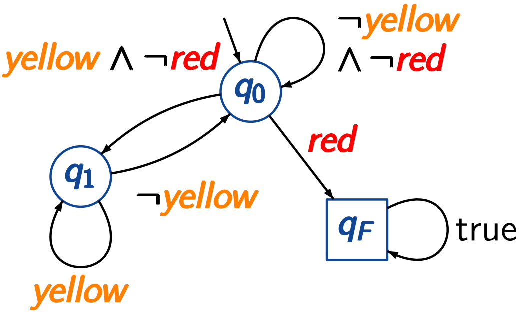


transition system $\mathcal{T}$ over
$AP = \{red, yellow\}$

> $\mathcal{T}$ satisfies the safety property E
> *"every red phase is preceded by a yellow phase"*

Example: product-TS



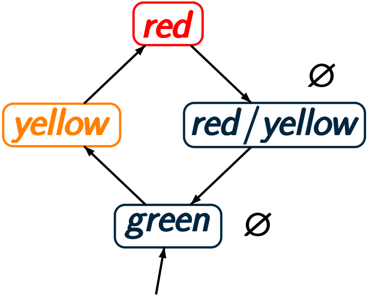


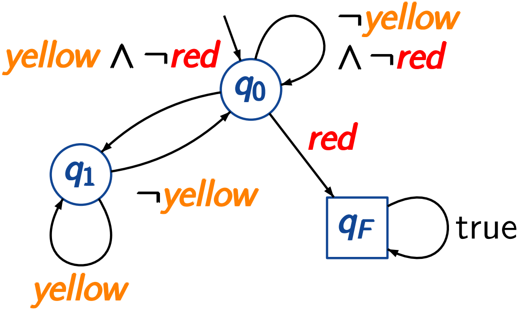
transition system $\mathcal{T}$ over $AP = \{red, yellow\}$

DFA $\mathcal{A}$ for the bad prefixes for E

$\mathcal{T}$ satisfies the safety property E
"every red phase is preceded by a yellow phase"

Example: product-TS





green q_0 red / yellow q_0

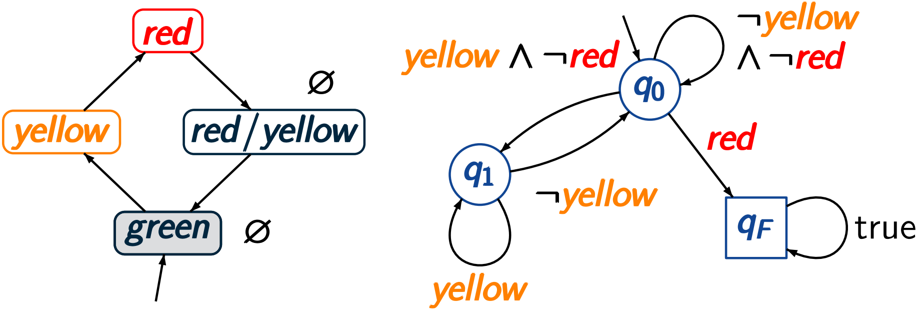
yellow q_1 red q_0

...

product-TS
$\mathcal{T} \otimes \mathcal{A}$

$(4 * 3 = 12$ states$)$

Example: product-TS



green q_0	red / yellow q_0
yellow q_1	red q_0

...

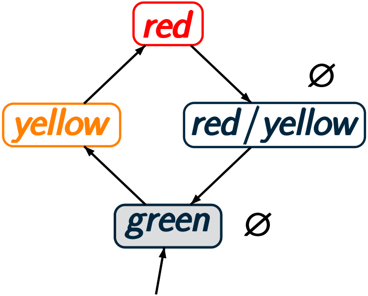
initial state

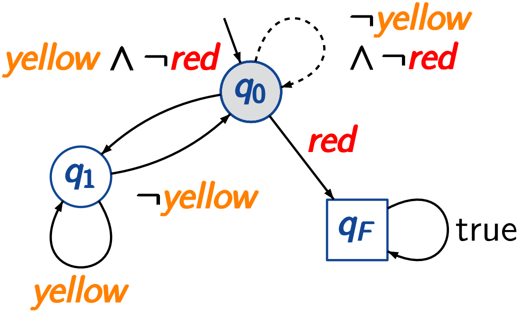
$\langle green, \delta(q_0, \varnothing)\rangle$

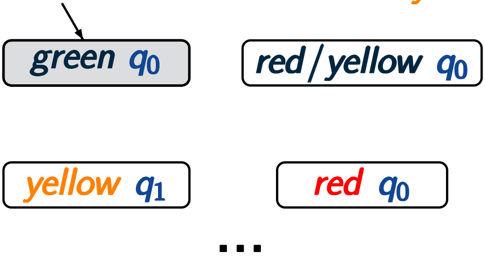
↑

$L(green) = \varnothing$

Example: product-TS



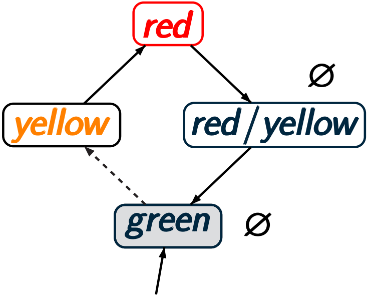


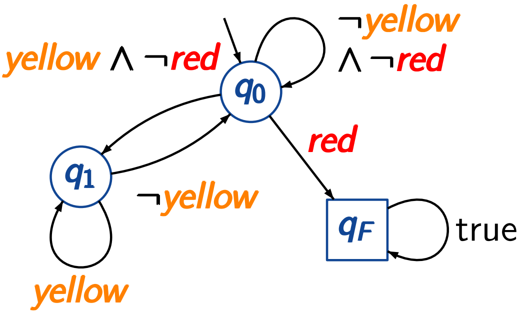


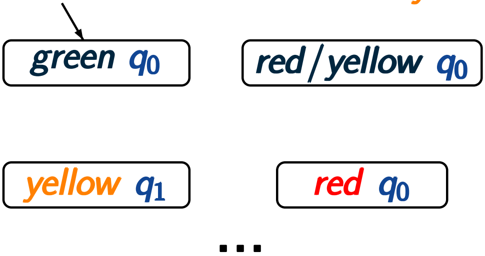
initial state

$$\langle green, \underbrace{\delta(q_0, \varnothing)}_{= q_0} \rangle$$

Example: product-TS

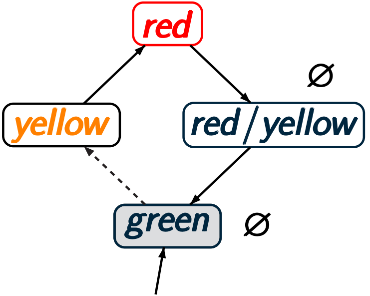


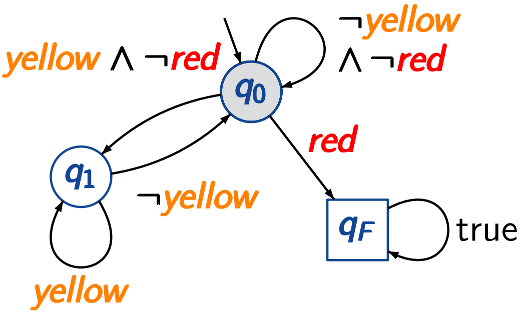


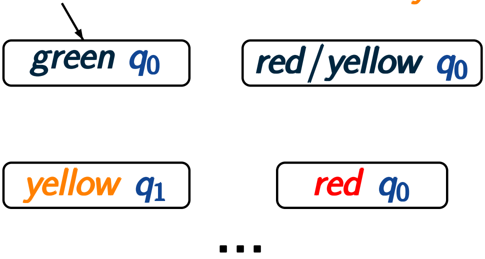


lifting the transition
green ⟶ yellow

Example: product-TS







lifting the transition

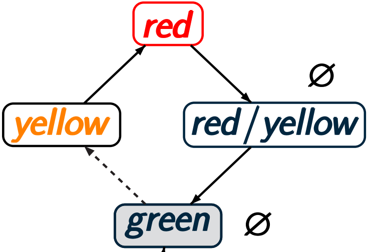
green $\longrightarrow$ *yellow*

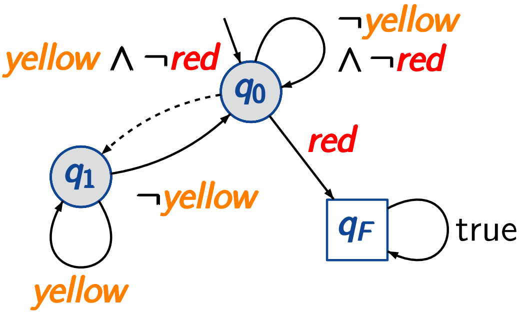
$\langle$ *green*, q_0 $\rangle$

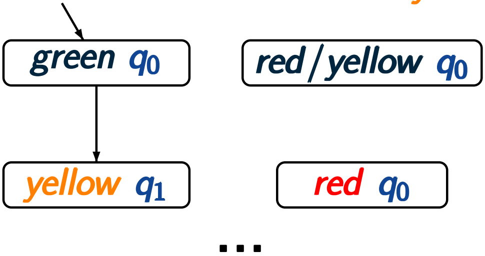
$\downarrow$

$\langle$ *yellow*, ? $\rangle$

Example: product-TS







lifting the transition

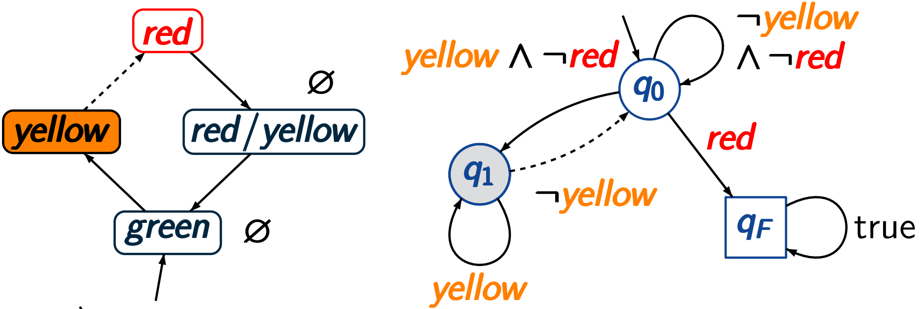
green $\longrightarrow$ yellow

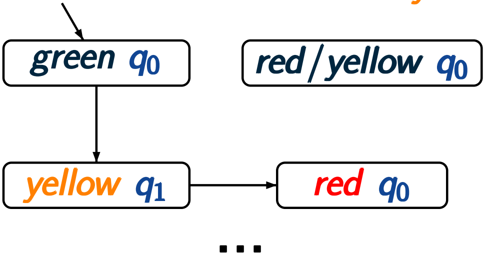
$$\langle green, q_0 \rangle$$

$$\downarrow$$

$$\langle yellow, \underbrace{\delta(q_0, \{yellow\})}_{= q_1} \rangle$$

Example: product-TS





lifting the transition

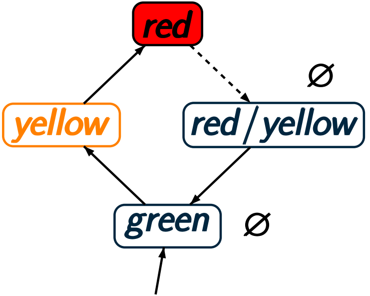
$$yellow \longrightarrow red$$

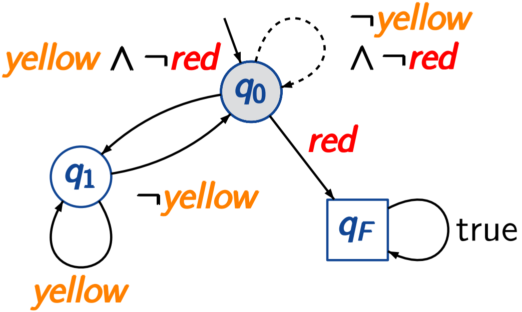
$$\langle yellow, q_1 \rangle$$

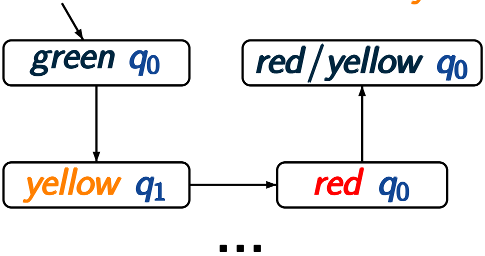
$$\downarrow$$

$$\langle red, \underbrace{\delta(q_1, \{red\})}_{= q_0} \rangle$$

Example: product-TS







lifting the transition

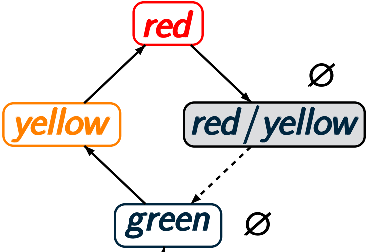
red $\longrightarrow$ *red / yellow*

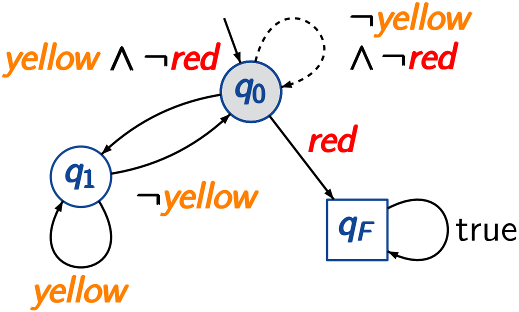
$\langle$*red*, $q_0\rangle$

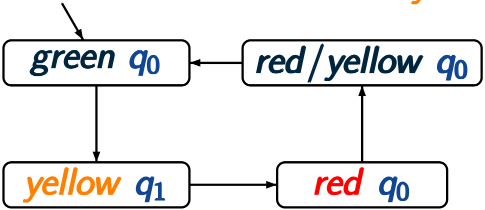
$\Downarrow$

$\langle$*red / yellow*, $\underbrace{\delta(q_0, \varnothing)}_{= q_0}\rangle$

Example: product-TS





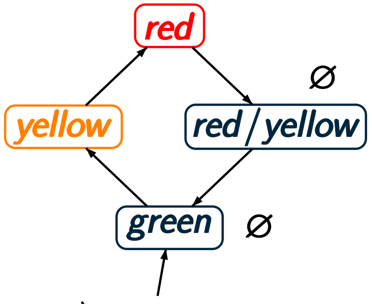


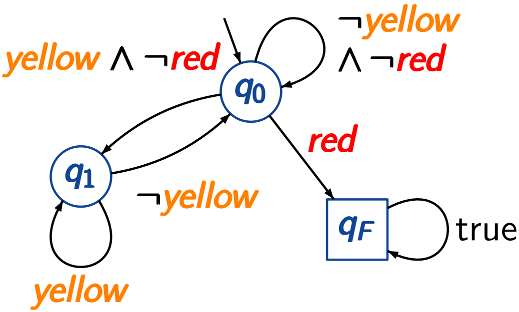
lifting the transition

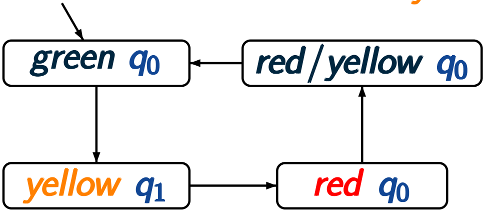
red / yellow $\longrightarrow$ *green*

$$\langle \mathit{red/yellow}, q_0 \rangle$$
$$\downarrow$$
$$\langle \mathit{green}, \underbrace{\delta(q_0, \varnothing)}_{=\, q_0} \rangle$$

Example: product-TS



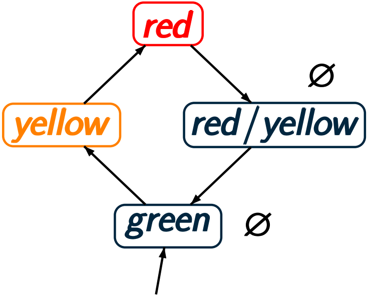


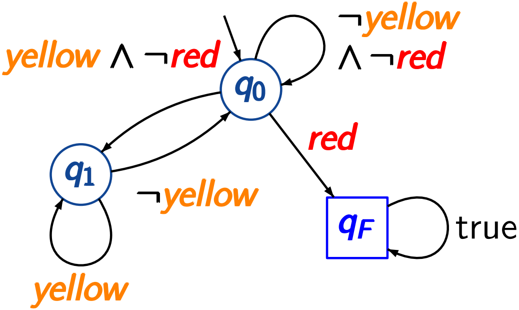


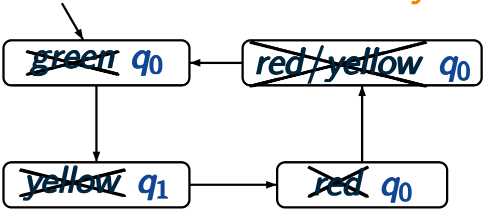
product-TS
$\mathcal{T} \otimes \mathcal{A}$

$\mathbf{4} * \mathbf{3} = \mathbf{12}$ states, but just **4** reachable states

Example: product-TS

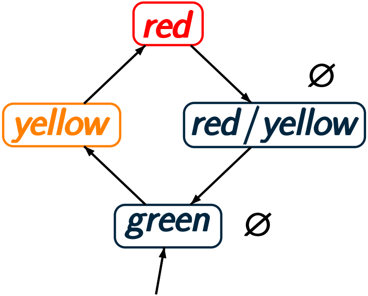


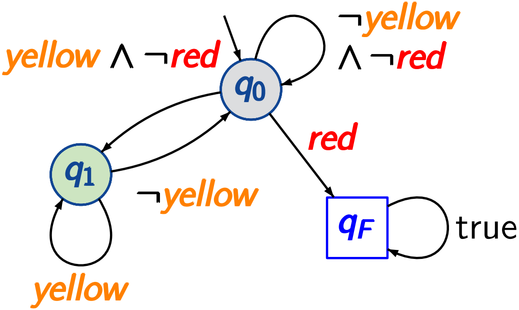


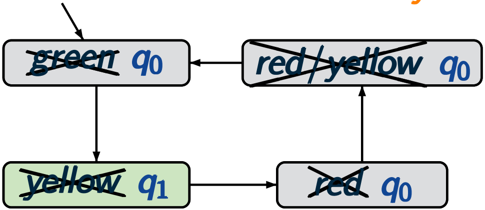


set of propositions
$AP' = \{q_0, q_1, q_F\}$

Example: product-TS







set of propositions
$AP' = \{q_0, q_1, q_F\}$

invariant condition $\neg q_F$ holds
for all reachable states

Technical remark on the product-TS

definition of the product of

- a transition system $\mathcal{T} = (S, Act, \rightarrow, S_0, AP, L)$
- an NFA $\mathcal{A} = (Q, 2^{AP}, \delta, Q_0, F)$

then the product $\mathcal{T} \otimes \mathcal{A} = (S \times Q, Act, \rightarrow', \ldots)$ is a TS

Technical remark on the product-TS

definition of the product of

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

  without terminal states

- an NFA $\mathcal{A} = (Q, 2^{AP}, \delta, Q_0, F)$

then the product $\mathcal{T} \otimes \mathcal{A} = (S \times Q, Act, \rightarrow', \ldots)$ is a TS

Technical remark on the product-TS

definition of the product of

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

  ↑ without terminal states

- an NFA $\mathcal{A} = (Q, 2^{AP}, \delta, Q_0, F)$

then the product $\mathcal{T} \otimes \mathcal{A} = (S \times Q, Act, \rightarrow', \ldots)$ is a TS

↑ without terminal states

Technical remark on the product-TS

definition of the product of

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

  without terminal states

- an NFA $\mathcal{A} = (Q, 2^{AP}, \delta, Q_0, F)$

then the product $\mathcal{T} \otimes \mathcal{A} = (S \times Q, Act, \rightarrow', \ldots)$ is a TS

without terminal states

assumptions on the NFA $\mathcal{A}$:

Technical remark on the product-TS

definition of the product of

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

  without terminal states

- an NFA $\mathcal{A} = (Q, 2^{AP}, \delta, Q_0, F)$

then the product $\mathcal{T} \otimes \mathcal{A} = (S \times Q, Act, \rightarrow', \ldots)$ is a TS

without terminal states

assumptions on the NFA $\mathcal{A}$:

- $\mathcal{A}$ is non-blocking, i.e.,

$$Q_0 \neq \varnothing \;\wedge\; \forall q \in Q \, \forall A \in 2^{AP}. \, \delta(q, A) \neq \varnothing$$

Technical remark on the product-TS

definition of the product of

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

  without terminal states

- an NFA $\mathcal{A} = (Q, 2^{AP}, \delta, Q_0, F)$

then the product $\mathcal{T} \otimes \mathcal{A} = (S \times Q, Act, \rightarrow', \ldots)$ is a TS

without terminal states

assumptions on the NFA $\mathcal{A}$:

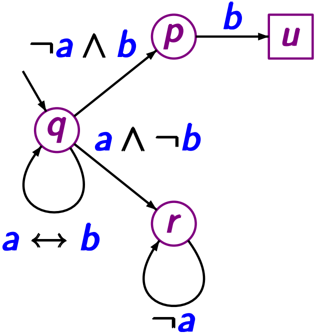
- $\mathcal{A}$ is non-blocking, i.e.,

  $$Q_0 \neq \varnothing \ \wedge \ \forall q \in Q \, \forall A \in 2^{AP}. \, \delta(q, A) \neq \varnothing$$

- no initial state of $\mathcal{A}$ is final, i.e., $Q_0 \cap F = \varnothing$

Non-blocking NFA

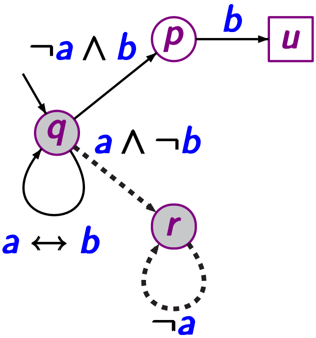
NFA $\mathcal{A}$



alphabet $\Sigma = 2^{AP}$ where $AP = \{a, b\}$

Non-blocking NFA

NFA $\mathcal{A}$

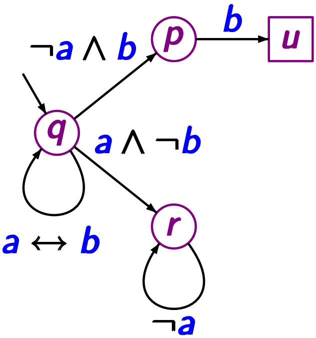


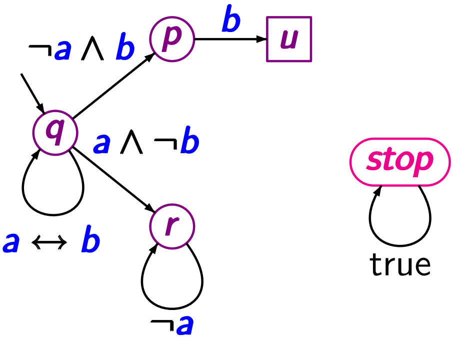
blocks for input
$\{a\}\,\varnothing\,\{a\}$

alphabet $\Sigma = 2^{AP}$ where $AP = \{a, b\}$

Non-blocking NFA

NFA $\mathcal{A}$ $\rightsquigarrow$ equivalent NFA $\mathcal{A}'$



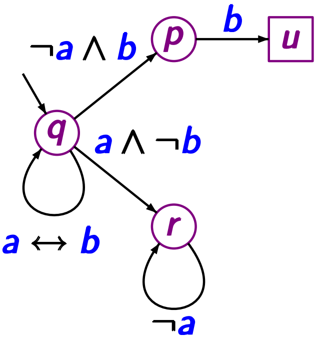


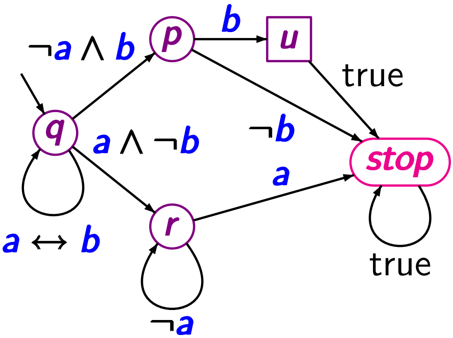
blocks for input $\{a\}\varnothing\{a\}$

add a trap state *stop*

Non-blocking NFA

NFA $\mathcal{A}$ $\rightsquigarrow$ equivalent NFA $\mathcal{A}'$



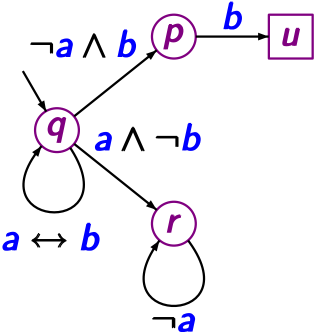


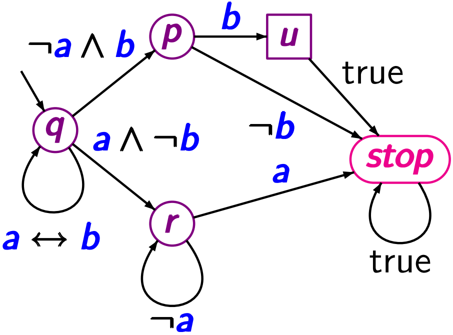
blocks for input
$\{a\}\,\varnothing\,\{a\}$

add a trap state *stop*

Non-blocking NFA

NFA $\mathcal{A}$ $\rightsquigarrow$ equivalent NFA $\mathcal{A}'$



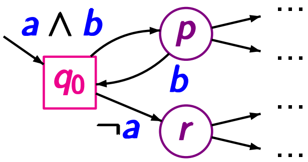


blocks for input
$\{a\} \varnothing \{a\}$

non-blocking

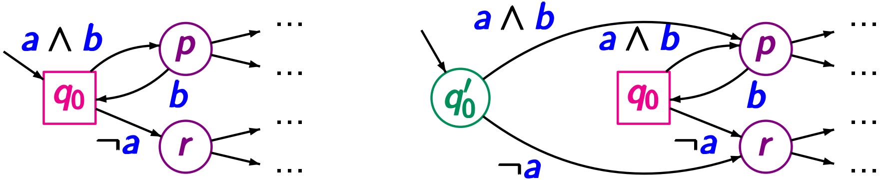
NFA where no initial state is final

NFA $\mathcal{A}$ with $Q_0 \cap F \neq \varnothing$



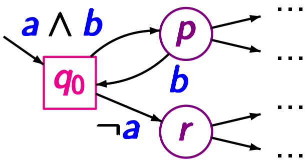
NFA where no initial state is final

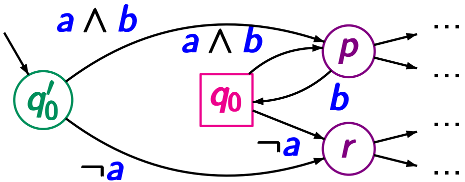
NFA $\mathcal{A}$ with $Q_0 \cap F \neq \varnothing$ $\rightsquigarrow$ NFA $\mathcal{A}'$ with $Q_0 \cap F = \varnothing$



NFA where no initial state is final

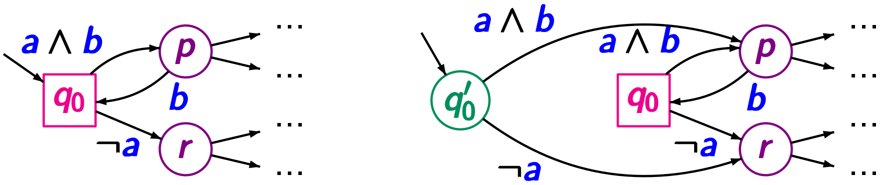
NFA $\mathcal{A}$ with $Q_0 \cap F \neq \varnothing$ $\rightsquigarrow$ NFA $\mathcal{A}'$ with $Q_0 \cap F = \varnothing$





$$\mathcal{L}(\mathcal{A}') = \mathcal{L}(\mathcal{A}) \setminus \{\varepsilon\}$$

NFA $\mathcal{A}$ with $Q_0 \cap F \neq \varnothing$ $\rightsquigarrow$ NFA $\mathcal{A}'$ with $Q_0 \cap F = \varnothing$



$$\mathcal{L}(\mathcal{A}') = \mathcal{L}(\mathcal{A}) \setminus \{\varepsilon\}$$

note: if $\mathcal{A}$ is an NFA for the bad prefixes of a safety property then

$$\varepsilon \notin \mathcal{L}(\mathcal{A}) = \mathit{BadPref}$$

Model checking regular safety properties

IS2.5-26A

Model checking regular safety properties

… via a reduction to invariant checking …..

Model checking regular safety properties

Let $\mathcal{T} = (S, Act, \to, S_0, AP, L)$ be a transition system

$\mathcal{A} = (Q, 2^{AP}, \delta, Q_0, F)$ be an NFA
for the bad prefixes of a regular safety property E

Model checking regular safety properties

Let $\mathcal{T} = (S, Act, \rightarrow, S_0, AP, L)$ be a transition system (without terminal states)

$\mathcal{A} = (Q, 2^{AP}, \delta, Q_0, F)$ be an NFA for the bad prefixes of a regular safety property E

Model checking regular safety properties

Let $\mathcal{T} = (S, Act, \rightarrow, S_0, AP, L)$ be a transition system
(without terminal states)

$\mathcal{A} = (Q, 2^{AP}, \delta, Q_0, F)$ be an NFA
for the bad prefixes of a regular safety property E
(non-blocking and $Q_0 \cap F = \varnothing$)

Model checking regular safety properties

Let $\mathcal{T} = (S, Act, \rightarrow, S_0, AP, L)$ be a transition system (without terminal states)

$\mathcal{A} = (Q, 2^{AP}, \delta, Q_0, F)$ be an NFA for the bad prefixes of a regular safety property E (non-blocking and $Q_0 \cap F = \varnothing$)

The following statements are equivalent:

(1) $\mathcal{T} \models E$

(2) $Traces_{fin}(\mathcal{T}) \cap \mathcal{L}(\mathcal{A}) = \varnothing$

Model checking regular safety properties

Let $\mathcal{T} = (S, Act, \to, S_0, AP, L)$ be a transition system
(without terminal states)

$\mathcal{A} = (Q, 2^{AP}, \delta, Q_0, F)$ be an NFA
for the bad prefixes of a regular safety property E
(non-blocking and $Q_0 \cap F = \varnothing$)

The following statements are equivalent:

(1) $\mathcal{T} \models E$

(2) $Traces_{fin}(\mathcal{T}) \cap \mathcal{L}(\mathcal{A}) = \varnothing$

(3) $\mathcal{T} \otimes \mathcal{A} \models$ invariant "always $\neg F$"

Model checking regular safety properties

Let $\mathcal{T} = (S, Act, \rightarrow, S_0, AP, L)$ be a transition system (without terminal states)

$\mathcal{A} = (Q, 2^{AP}, \delta, Q_0, F)$ be an NFA for the bad prefixes of a regular safety property E (non-blocking and $Q_0 \cap F = \varnothing$)

The following statements are equivalent:

(1) $\mathcal{T} \models E$

(2) $Traces_{fin}(\mathcal{T}) \cap \mathcal{L}(\mathcal{A}) = \varnothing$

(3) $\mathcal{T} \otimes \mathcal{A} \models$ invariant "always $\neg F$"

where "$\neg F$" denotes $\bigwedge_{q \in F} \neg q$

Product transition system

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

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

product-TS $\mathcal{T} \otimes \mathcal{A} \stackrel{\text{def}}{=} (S \times Q, Act, \longrightarrow', S_0', AP', L')$

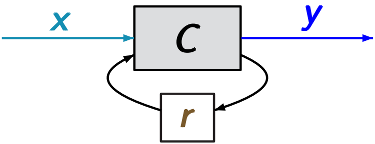
$$\frac{s \xrightarrow{\alpha} s' \quad \wedge \quad q' \in \delta(q, L(s'))}{\langle s, q \rangle \xrightarrow{\alpha}{}' \langle s', q' \rangle}$$

initial states: $S_0' = \{ \langle s_0, q \rangle : s_0 \in S_0, q \in \delta\big(Q_0, L(s_0)\big) \}$

set of atomic propositions: $AP' = Q$

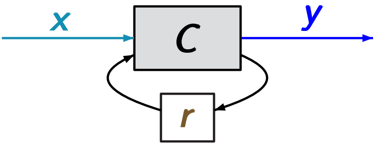
labeling function: $L'(\langle s, q \rangle) = \{q\}$

Example: sequential circuit



$$\lambda_y = \delta_r = x \oplus r$$

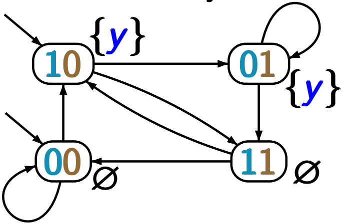
Example: sequential circuit



$\lambda_y = \delta_r = x \oplus r$

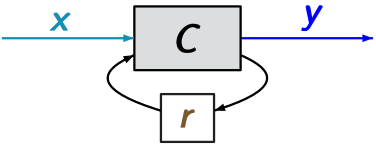
initially $r = 0$

transition system $\mathcal{T}$



over $AP = \{y\}$

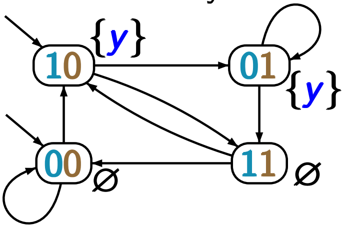
Example: sequential circuit



$\lambda_y = \delta_r = x \oplus r$

initially $r = 0$

transition system $\mathcal{T}$

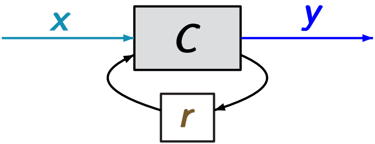


over $AP = \{y\}$

safety property E

The circuit will never ouput two ones after each other

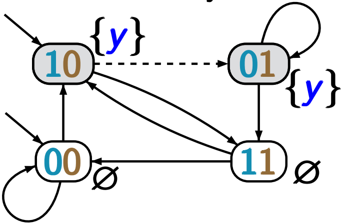
Example: sequential circuit



$\lambda_y = \delta_r = x \oplus r$

initially $r = 0$

transition system $\mathcal{T}$

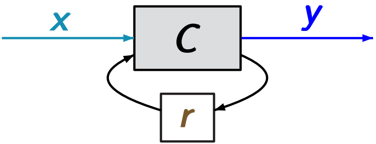


$\mathcal{T} \not\models E$

safety property E

The circuit will never ouput two ones after each other

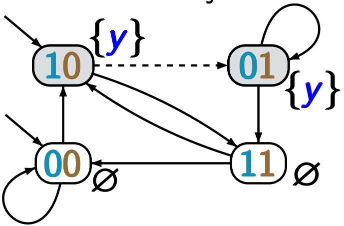
Example: sequential circuit



$\lambda_y = \delta_r = x \oplus r$

initially $r = 0$

transition system $\mathcal{T}$



$\mathcal{T} \not\models E$

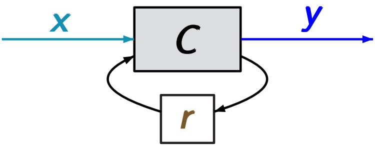
error indication, e.g.,

$\langle 10 \rangle \langle 01 \rangle$

safety property E

The circuit will never ouput two ones after each other

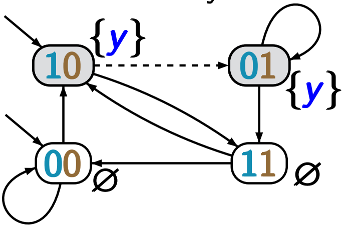
Example: sequential circuit



$\lambda_y = \delta_r = x \oplus r$

initially $r = 0$

transition system $\mathcal{T}$



$\mathcal{T} \not\models E$

error indication, e.g.,

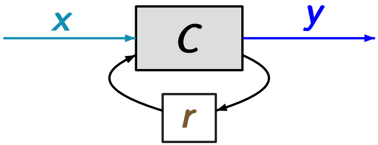
$\langle 10 \rangle \langle 01 \rangle$

bad prefix: $\{y\}\{y\}$

safety property E

The circuit will never ouput two ones after each other

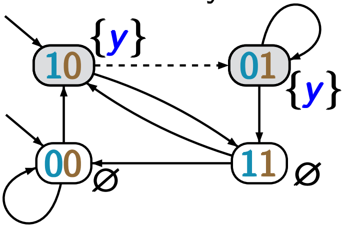
Example: sequential circuit



$\lambda_y = \delta_r = x \oplus r$

initially $r = 0$

transition system $\mathcal{T}$

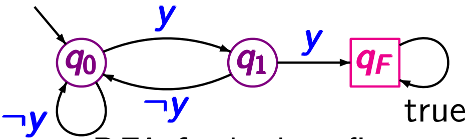


$\mathcal{T} \not\models E$

error indication, e.g.,

$\langle 10 \rangle \langle 01 \rangle$

bad prefix: $\{y\}\{y\}$



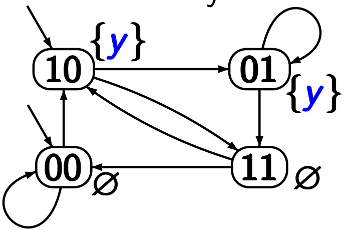
DFA for bad prefixes

safety property E

The circuit will never ouput two ones after each other

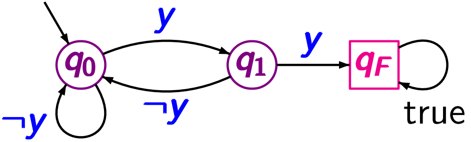
Example: product-TS

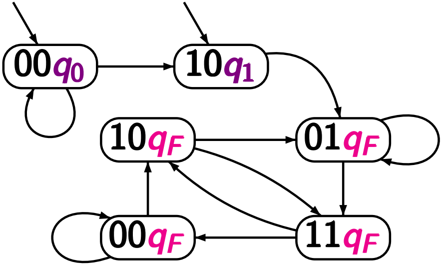
transition system $\mathcal{T}$



safety property E

… *never two ones in a row* …

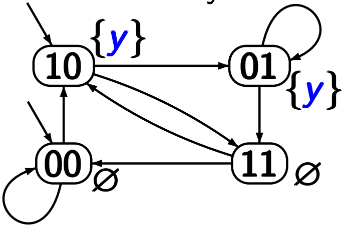




product-TS $\mathcal{T} \otimes \mathcal{A}$

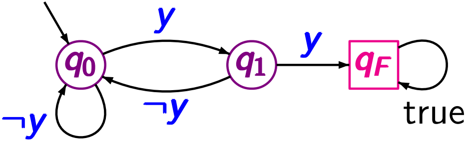
Example: product-TS

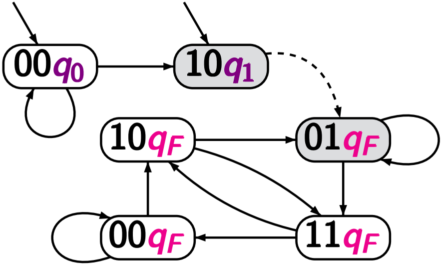
transition system $\mathcal{T}$



safety property E

... *never two ones in a row* ...

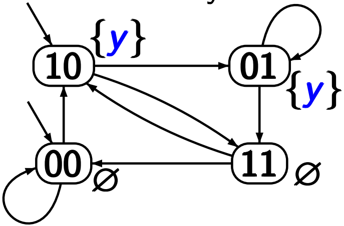




$\mathcal{T} \otimes \mathcal{A} \not\models$ "never q_F"

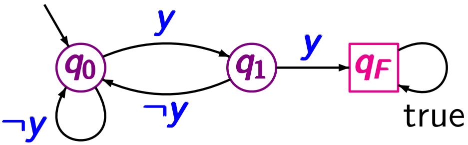
Example: product-TS

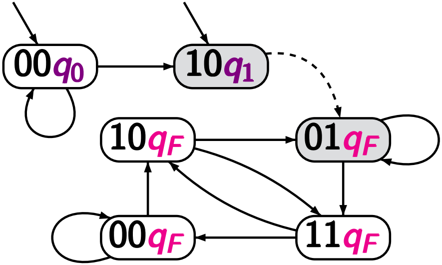
transition system $\mathcal{T}$



safety property E

... never two ones in a row ...





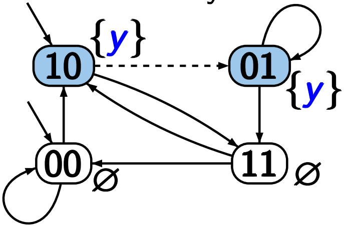
error indication for

$\mathcal{T} \otimes \mathcal{A} \not\models$ "never q_F"

$10q_1 \quad 01q_F$

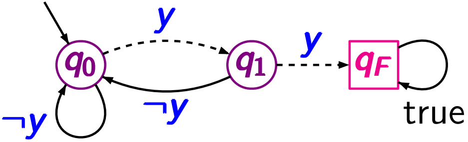
Example: product-TS

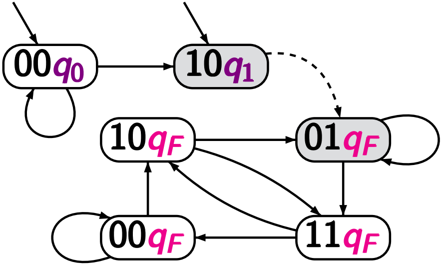
transition system $\mathcal{T}$



safety property E

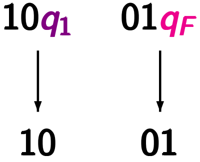
... never two ones in a row ...





error indication for $\mathcal{T} \not\models E$

error indication for $\mathcal{T} \otimes \mathcal{A} \not\models$ "never q_F"



Model checking regular safety properties

Model checking regular safety properties

input: finite TS $\mathcal{T}$,
NFA $\mathcal{A}$ for the bad prefixes of E

output: "yes" if $\mathcal{T} \models E$
otherwise "no"

Model checking regular safety properties

input: finite TS $\mathcal{T}$,
NFA $\mathcal{A}$ for the bad prefixes of E

output: "yes" if $\mathcal{T} \models E$
otherwise "no"

construct product transition system $\mathcal{T} \otimes \mathcal{A}$
check whether $\mathcal{T} \otimes \mathcal{A} \models$ "always $\neg F$"

where F = set of final states in $\mathcal{A}$

Model checking regular safety properties

input: finite TS $\mathcal{T}$,
NFA $\mathcal{A}$ for the bad prefixes of E

output: "yes" if $\mathcal{T} \models E$
otherwise "no"

construct product transition system $\mathcal{T} \otimes \mathcal{A}$

check whether $\mathcal{T} \otimes \mathcal{A} \models$ "always $\neg F$"

if so, then return "yes"

if not, then return "no"

where F = set of final states in $\mathcal{A}$

input: finite TS $\mathcal{T}$,
NFA $\mathcal{A}$ for the bad prefixes of E

output: "yes" if $\mathcal{T} \models E$
otherwise "no" + error indication

construct product transition system $\mathcal{T} \otimes \mathcal{A}$
check whether $\mathcal{T} \otimes \mathcal{A} \models$ "always $\neg F$"
if so, then return "yes"
if not, then return "no" $\longleftarrow$ and an error indication

where F = set of final states in $\mathcal{A}$

Model checking regular safety properties

construct product transition system $\mathcal{T} \otimes \mathcal{A}$

```
IF T ⊗ A ⊨ "always ¬F"
 THEN  return "yes"
 ELSE



FI
```

Model checking regular safety properties

construct product transition system $\mathcal{T} \otimes \mathcal{A}$

IF $\mathcal{T} \otimes \mathcal{A} \models$ “always $\neg F$”

THEN return “yes”

ELSE compute a counterexample for $\mathcal{T} \otimes \mathcal{A}$ and the invariant “always $\neg F$”,

FI

Model checking regular safety properties

construct product transition system $\mathcal{T} \otimes \mathcal{A}$

IF $\mathcal{T} \otimes \mathcal{A} \models$ "always $\neg F$"

THEN return "yes"

ELSE compute a counterexample for $\mathcal{T} \otimes \mathcal{A}$ and the invariant "always $\neg F$",

i.e., an initial path fragment in the product

$\langle s_0, p_0 \rangle \langle s_1, p_1 \rangle \ldots \langle s_n, p_n \rangle$ where $p_n \in F$

FI

Model checking regular safety properties

construct product transition system $\mathcal{T} \otimes \mathcal{A}$

IF $\mathcal{T} \otimes \mathcal{A} \models$ "always $\neg F$"

THEN return "yes"

ELSE compute a counterexample for $\mathcal{T} \otimes \mathcal{A}$ and the invariant "always $\neg F$",

i.e., an initial path fragment in the product

$\langle s_0, p_0 \rangle \langle s_1, p_1 \rangle \ldots \langle s_n, p_n \rangle$ where $p_n \in F$

return "no" and $s_0\, s_1 \ldots s_n$

FI

Model checking regular safety properties

construct product transition system $\mathcal{T} \otimes \mathcal{A}$

IF $\mathcal{T} \otimes \mathcal{A} \models$ "always $\neg F$"

THEN return "yes"

ELSE compute a counterexample for $\mathcal{T} \otimes \mathcal{A}$ and the invariant "always $\neg F$",

i.e., an initial path fragment in the product

$$\langle s_0, p_0 \rangle \langle s_1, p_1 \rangle \ldots \langle s_n, p_n \rangle \text{ where } p_n \in F$$

return "no" and $s_0 \, s_1 \ldots s_n$

FI

time complexity: $\mathcal{O}\big(size(\mathcal{T}) \cdot size(\mathcal{A})\big)$

Correct or wrong?

If $\mathcal{T}$ is a finite transition system then $\mathit{Traces}_{\mathit{fin}}(\mathcal{T})$ is regular.

Correct or wrong?

If $\mathcal{T}$ is a finite transition system then $\mathit{Traces}_{\mathit{fin}}(\mathcal{T})$ is regular.

correct.

Correct or wrong?

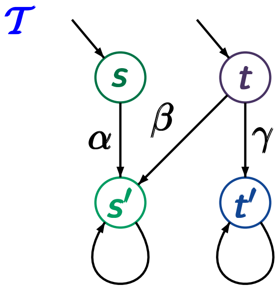
If $\mathcal{T}$ is a finite transition system then $\mathit{Traces}_{fin}(\mathcal{T})$ is regular.

correct. $\mathcal{T}$ can be transformed into an **NFA**.

Correct or wrong?

If $\mathcal{T}$ is a finite transition system then $Traces_{fin}(\mathcal{T})$ is regular.

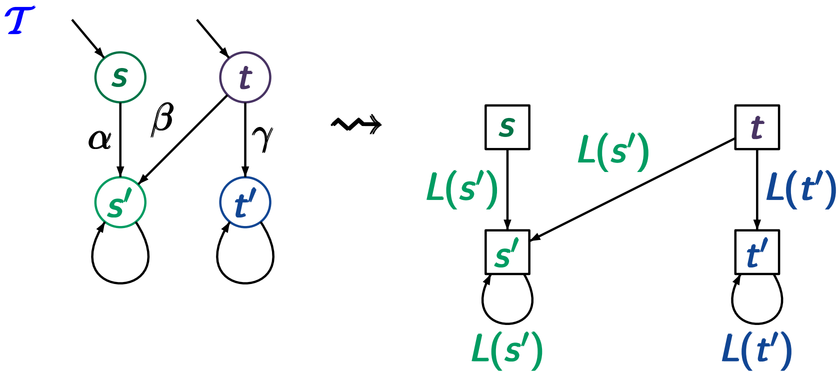
correct. $\mathcal{T}$ can be transformed into an **NFA**.



Correct or wrong?

If $\mathcal{T}$ is a finite transition system then $Traces_{fin}(\mathcal{T})$ is regular.

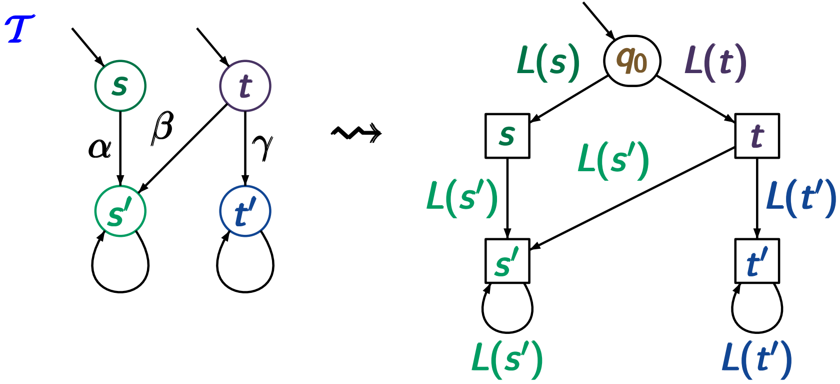
correct. $\mathcal{T}$ can be transformed into an **NFA**.



Correct or wrong?

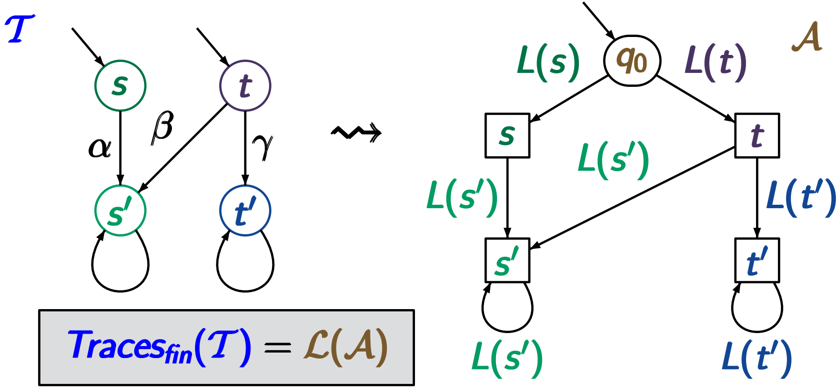
> If $\mathcal{T}$ is a finite transition system then $Traces_{fin}(\mathcal{T})$ is regular.

correct. $\mathcal{T}$ can be transformed into an **NFA**.



If $\mathcal{T}$ is a finite transition system then $Traces_{fin}(\mathcal{T})$ is regular.

correct. $\mathcal{T}$ can be transformed into an **NFA**.



$Traces_{fin}(\mathcal{T}) = \mathcal{L}(\mathcal{A})$