

Introduction

Modelling parallel systems

Linear Time Properties

**Regular Properties**

Linear Temporal Logic

Computation-Tree Logic

Equivalences and Abstraction



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

*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  **$\omega$ -automata**, i.e.,  
acceptors for infinite words

Introduction

Modelling parallel systems

Linear Time Properties

**Regular Properties**

regular safety properties



$\omega$ -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

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

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

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

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

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





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.

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

$E$  is called regular iff the language

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

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

$E$  is called regular iff the language

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

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

is regular.



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

- $Q$  finite set of states
- $\Sigma$  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**

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

- $Q$  finite set of states
- $\Sigma$  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  $A_0 A_1 \dots A_{n-1} \in \Sigma^*$ :

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

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

- $Q$  finite set of states
- $\Sigma$  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  $A_0 A_1 \dots A_{n-1} \in \Sigma^*$ :

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

run  $\pi$  is called **accepting** if  $q_n \in F$

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

- $Q$  finite set of states
- $\Sigma$  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  $\Sigma$  that have an **accepting run** in  $\mathcal{A}$



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

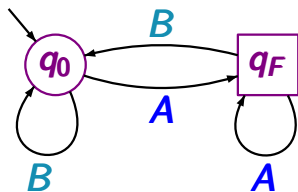
- $Q$  finite set of states
- $\Sigma$  alphabet  $\leftarrow$  here:  $\Sigma = 2^{AP}$
- $\delta : Q \times \Sigma \rightarrow 2^Q$  transition relation
- $Q_0 \subseteq Q$  set of initial states
- $F \subseteq Q$  set of **final states**, also called **accept states**

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

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

# Notations in pictures for NFA

is2.5-15A



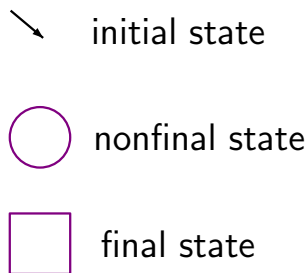
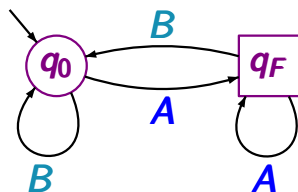
↘ initial state

○ nonfinal state

□ final state

# Notations in pictures for NFA

is2.5-15A



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

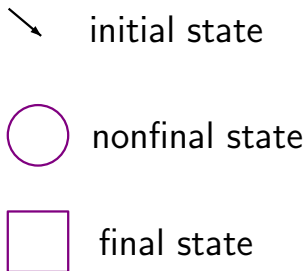
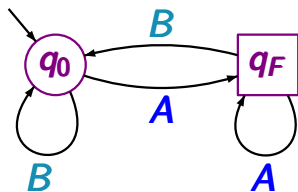
$q_0$  initial state

$q_F$  final state

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

# Notations in pictures for NFA

is2.5-15A



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

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

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

NFA  $\mathcal{A} = (Q, \Sigma, \delta, Q_0, F)$  over the alphabet  $\Sigma = 2^{AP}$   
*symbolic notation* for the labels of transitions:

If  $\phi$  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  $A \models \phi$

NFA  $\mathcal{A} = (Q, \Sigma, \delta, Q_0, F)$  over the alphabet  $\Sigma = 2^{AP}$   
*symbolic notation* for the labels of transitions:

If  $\Phi$  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

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

NFA  $\mathcal{A} = (Q, \Sigma, \delta, Q_0, F)$  over the alphabet  $\Sigma = 2^{AP}$   
*symbolic notation* for the labels of transitions:

If  $\Phi$  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

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

$q \xrightarrow{\text{true}} p \hat{=} \{ q \xrightarrow{A} p : A \subseteq AP \}$



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  $2^{AP}$

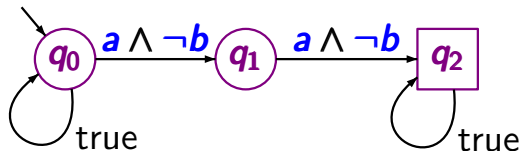
is regular.

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.



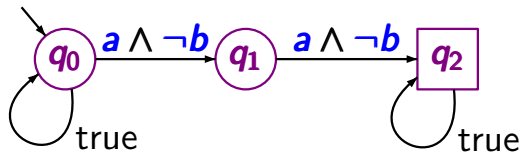
$AP = \{a, b\}$

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:

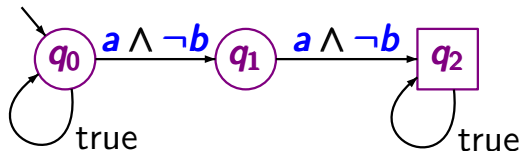
$a \wedge \neg b \hat{=} \{a\}$

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:

$a \wedge \neg b \hat{=} \{a\}$

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

*“Every red phase is preceded by a yellow phase”*

*“Every red phase is preceded by a yellow phase”*

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

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

## Example: regular safety property

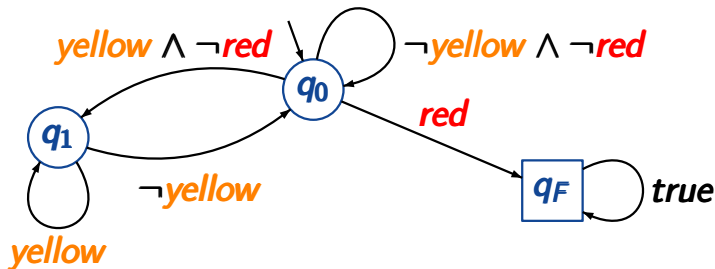
IS2.5-16

*“Every red phase is preceded by a yellow phase”*

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

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

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



## Example: regular safety property

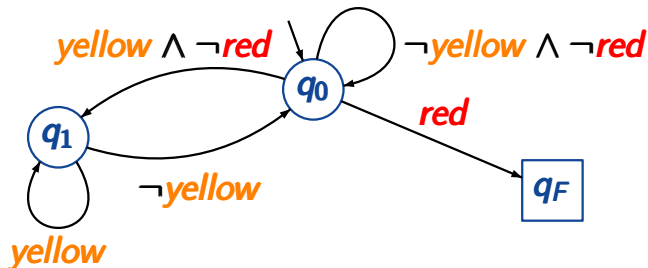
IS2.5-16

*“Every red phase is preceded by a yellow phase”*

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

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

DFA for 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  $\iff$  *MinBadPref* is regular*

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  $\iff$   $MinBadPref$  is regular*

“ $\Leftarrow$ ”: Let  $\mathcal{A}$  be an NFA for *MinBadPref*.

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  $\iff$   $MinBadPref$  is regular*

“ $\Leftarrow$ ”: 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  $p$ .

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  $\iff$   $MinBadPref$  is regular*

“ $\Leftarrow$ ”: Let  $\mathcal{A}$  be an NFA for *MinBadPref*.

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

“ $\Rightarrow$ ”: Let  $\mathcal{A}$  be a DFA for *BadPref*.

## Bad prefixes vs minimal bad prefixes

IS2.5-14A

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  $\iff$  *MinBadPref* is regular

“ $\Leftarrow$ ”: Let  $\mathcal{A}$  be an NFA for *MinBadPref*.

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

“ $\Rightarrow$ ”: Let  $\mathcal{A}$  be a DFA for *BadPref*.

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

Every **invariant** is regular.

Every **invariant** is regular.

**correct.**

Every **invariant** is regular.

**correct.**

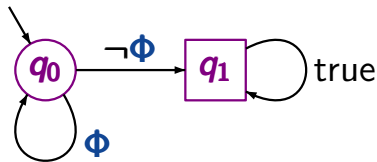
Let  **$E$**  be an invariant with invariant condition  $\Phi$



Every invariant is regular.

correct.

Let  $E$  be an invariant with invariant condition  $\Phi$

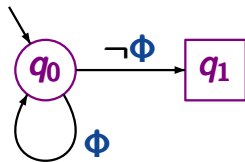


is a DFA for the language of all bad prefixes

Every invariant is regular.

**correct.**

Let  $E$  be an invariant with invariant condition  $\Phi$

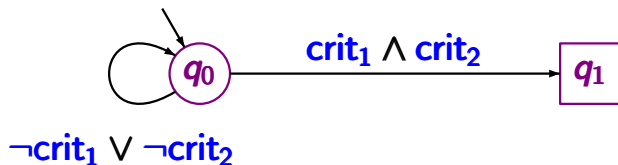


is a DFA for the language of all minimal bad prefixes

“The two processes are never simultaneously  
in their critical sections”

“The two processes are **never simultaneously**  
in their **critical sections**”

**DFA** for minimal bad prefixes over the alphabet  $2^{AP}$  where  $AP = \{\text{crit}_1, \text{crit}_2\}$



Every **safety property** is regular.

Every **safety property** is regular.

**wrong.**

Every **safety property** is regular.

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

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

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

Every **safety property** is regular.

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

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

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

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





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

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

*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

$$\mathcal{L}(\mathcal{A}_1) \subseteq \mathcal{L}(\mathcal{A}_2) ?$$

verification of regular  
safety properties

$$\text{Traces}(\mathcal{T}) \subseteq E ?$$

language inclusion  
for NFA

$$\mathcal{L}(\mathcal{A}_1) \subseteq \mathcal{L}(\mathcal{A}_2) ?$$

check whether

$$\mathcal{L}(\mathcal{A}_1) \cap (\Sigma^* \setminus \mathcal{L}(\mathcal{A}_2))$$

is empty

verification of regular  
safety properties

$$\text{Traces}(\mathcal{T}) \subseteq E ?$$

language inclusion  
for NFA

verification of regular  
safety properties

$$\mathcal{L}(\mathcal{A}_1) \subseteq \mathcal{L}(\mathcal{A}_2) ?$$

$$\text{Traces}(\mathcal{T}) \subseteq E ?$$

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

$$\mathcal{L}(\mathcal{A}_1) \subseteq \mathcal{L}(\mathcal{A}_2) ?$$

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  
 $\mathcal{L}(\mathcal{A}) = \mathcal{L}(\mathcal{A}_1) \cap \mathcal{L}(\overline{\mathcal{A}_2})$

verification of regular  
safety properties

$$\text{Traces}(\mathcal{T}) \subseteq E ?$$

language inclusion  
for NFA

verification of regular  
safety properties

$$\mathcal{L}(\mathcal{A}_1) \subseteq \mathcal{L}(\mathcal{A}_2) ?$$

$$\text{Traces}(\mathcal{T}) \subseteq E ?$$

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  $\mathcal{L}(\mathcal{A}) = \emptyset$



language inclusion  
for NFA

$$\mathcal{L}(\mathcal{A}_1) \subseteq \mathcal{L}(\mathcal{A}_2) ?$$

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  $\mathcal{L}(\mathcal{A}) = \emptyset$

verification of regular  
safety properties

$$\text{Traces}(\mathcal{T}) \subseteq E ?$$

check whether

$$\text{Traces}_{fin}(\mathcal{T}) \cap \text{BadPref}$$

is empty

language inclusion  
for NFA

$$\mathcal{L}(\mathcal{A}_1) \subseteq \mathcal{L}(\mathcal{A}_2) ?$$

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  $\mathcal{L}(\mathcal{A}) = \emptyset$

verification of regular  
safety properties

$$\text{Traces}(\mathcal{T}) \subseteq E ?$$

check whether

$$\text{Traces}_{fin}(\mathcal{T}) \cap \text{BadPref}$$

is empty

1. construct NFA  $\mathcal{A}$   
for the bad prefixes  
 $\mathcal{L}(\overline{\mathcal{A}}) = \text{BadPref}$

language inclusion  
for NFA

$$\mathcal{L}(\mathcal{A}_1) \subseteq \mathcal{L}(\mathcal{A}_2) ?$$

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  $\mathcal{L}(\mathcal{A}) = \emptyset$

verification of regular  
safety properties

$$\text{Traces}(\mathcal{T}) \subseteq E ?$$

check whether

$$\text{Traces}_{fin}(\mathcal{T}) \cap \text{BadPref}$$

is empty

1. construct NFA  $\mathcal{A}$   
for the bad prefixes  
 $\mathcal{L}(\overline{\mathcal{A}}) = \text{BadPref}$
2. construct TS  $\mathcal{T}'$  with  
 $\text{Traces}_{fin}(\mathcal{T}') = \dots$

language inclusion  
for NFA

$$\mathcal{L}(\mathcal{A}_1) \subseteq \mathcal{L}(\mathcal{A}_2) ?$$

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  $\mathcal{L}(\mathcal{A}) = \emptyset$

verification of regular  
safety properties

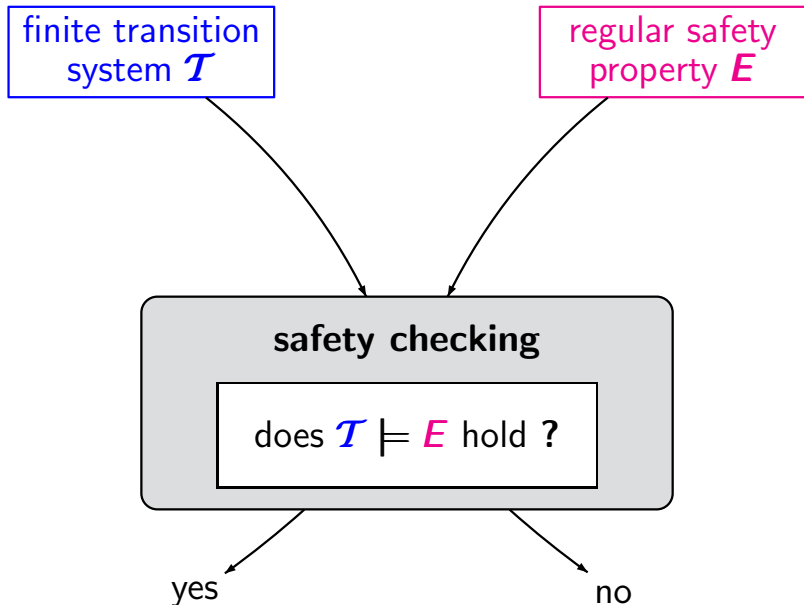
$$\text{Traces}(\mathcal{T}) \subseteq E ?$$

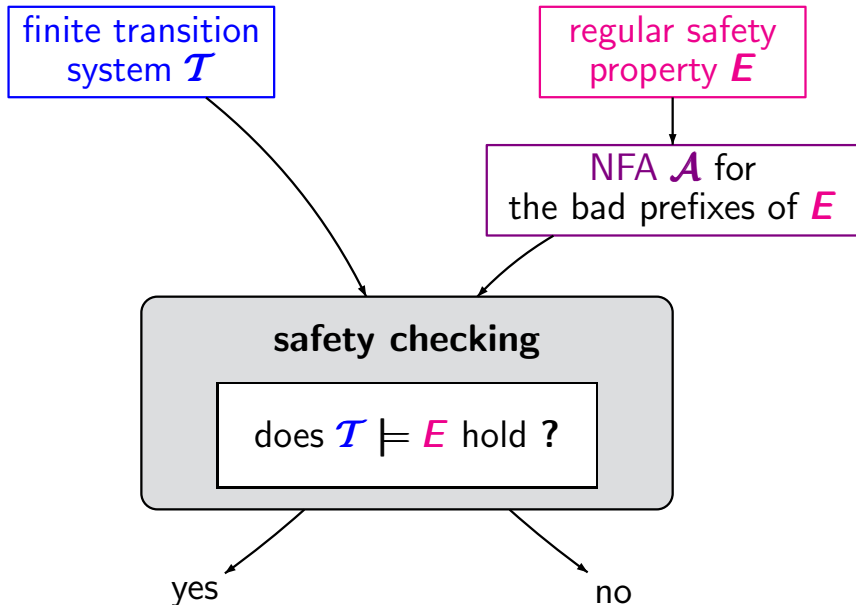
check whether

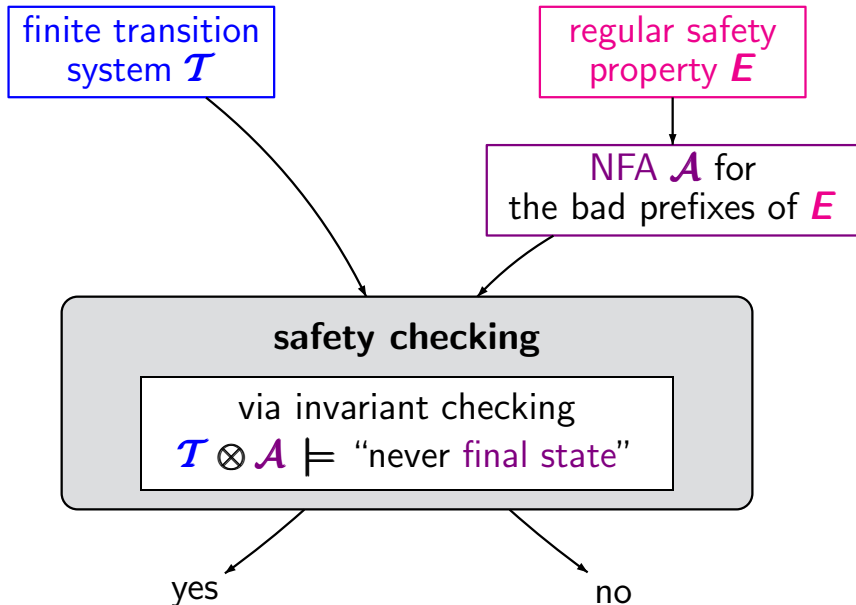
$$\text{Traces}_{fin}(\mathcal{T}) \cap \text{BadPref}$$

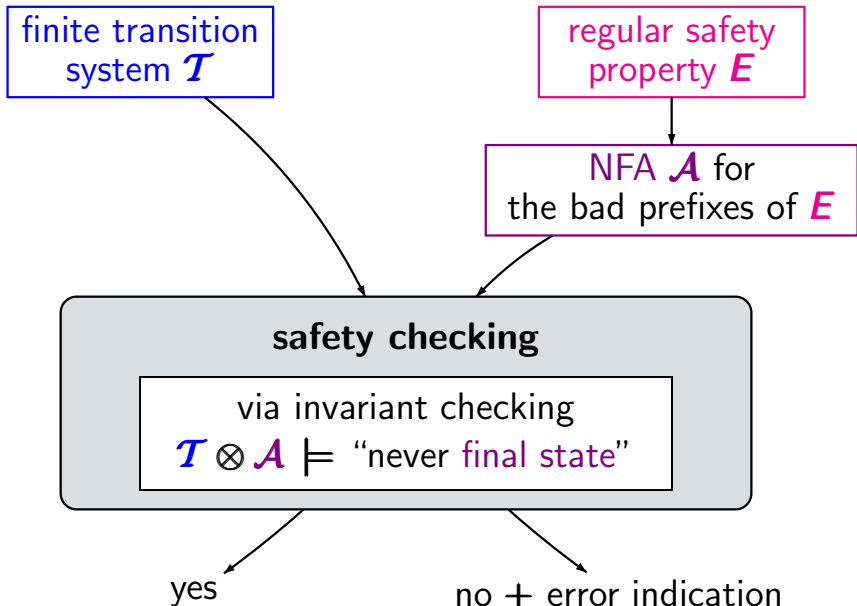
is empty

1. construct NFA  $\mathcal{A}$   
for the bad prefixes  
 $\mathcal{L}(\overline{\mathcal{A}}) = \text{BadPref}$
2. construct TS  $\mathcal{T}'$  with  
 $\text{Traces}_{fin}(\mathcal{T}') = \dots$
3. invariant checking  
for  $\mathcal{T}'$













# Product of a TS and an NFA

IS2.5-22

finite transition system

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

NFA for bad prefixes

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

$s_0$



$s_1$



$s_2$



$\vdots$



$s_n$

path  
fragment  $\hat{\pi}$

# Product of a TS and an NFA

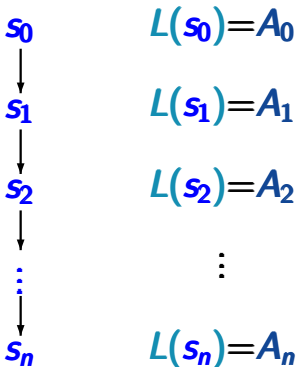
IS2.5-22

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  $\hat{\pi}$

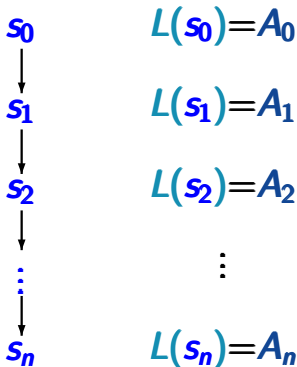
trace

# Product of a TS and an NFA

IS2.5-22

finite transition system

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

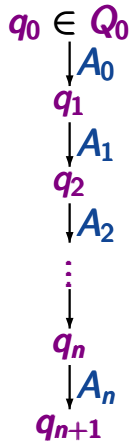


path  
fragment  $\hat{\pi}$

trace

NFA for bad prefixes

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



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

# Product of a TS and an NFA

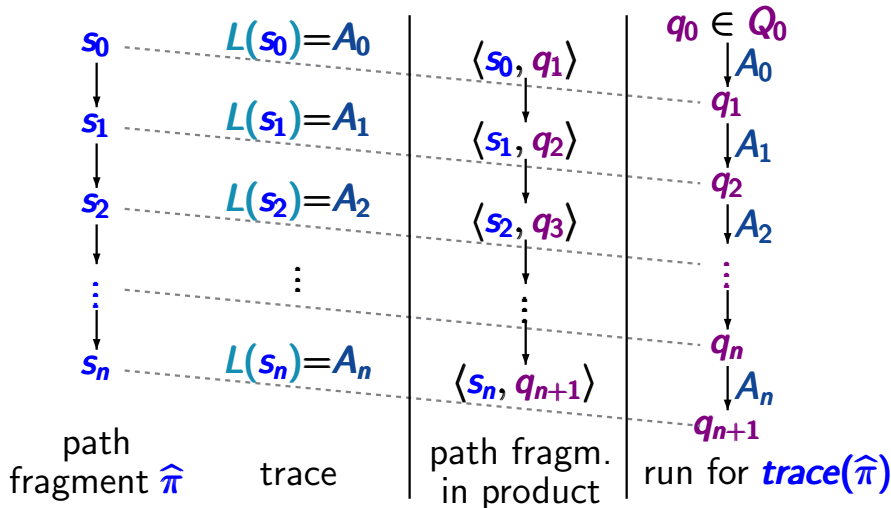
IS2.5-22

finite transition system

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

NFA for bad prefixes

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





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

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

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

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

product-TS  $\mathcal{T} \otimes \mathcal{A} \stackrel{\text{def}}{=} (S \times Q, Act, \longrightarrow', S'_0, AP', L')$



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

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

product-TS  $\mathcal{T} \otimes \mathcal{A} \stackrel{\text{def}}{=} (S \times Q, Act, \longrightarrow', S'_0, AP', L')$

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

$\mathcal{T} = (S, Act, \rightarrow, S_0, AP, L)$  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:  $S'_0 = \{ \langle s_0, q \rangle : s_0 \in S_0, q \in \delta(Q_0, L(s_0)) \}$

# Product transition system

IS2.5-25

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

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

product-TS  $\mathcal{T} \otimes \mathcal{A} \stackrel{\text{def}}{=} (S \times Q, \text{Act}, \longrightarrow', S'_0, AP', L')$

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

initial states:  $S'_0 = \{ \langle s_0, q \rangle : s_0 \in S_0, q \in \delta(Q_0, L(s_0)) \}$

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

$\mathcal{T} = (S, Act, \rightarrow, S_0, AP, L)$  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:  $S'_0 = \{ \langle s_0, q \rangle : s_0 \in S_0, q \in \delta(Q_0, L(s_0)) \}$

set of atomic propositions:  $AP' = Q$

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

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

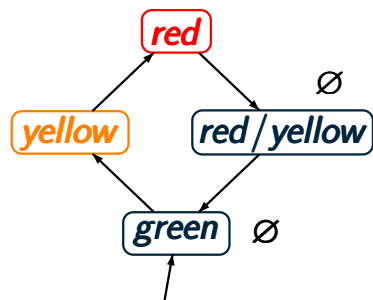
product-TS  $\mathcal{T} \otimes \mathcal{A} \stackrel{\text{def}}{=} (S \times Q, Act, \longrightarrow', S'_0, AP', L')$

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

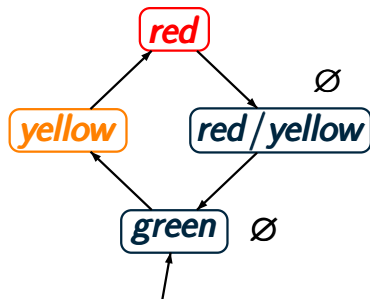
initial states:  $S'_0 = \{ \langle s_0, q \rangle : s_0 \in S_0, q \in \delta(Q_0, L(s_0)) \}$

set of atomic propositions:  $AP' = Q$

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



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

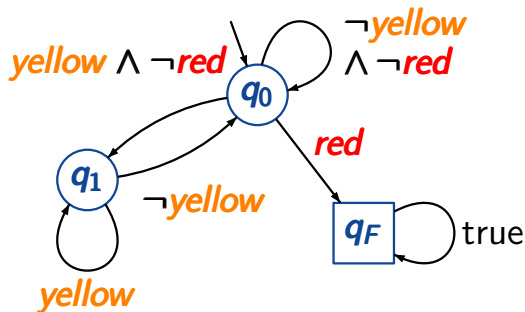
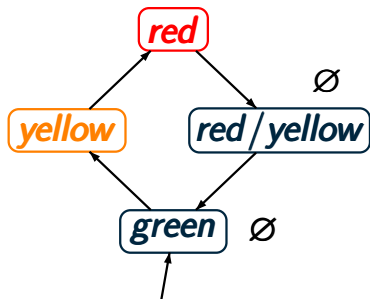


transition system  $\mathcal{T}$  over  
 $AP = \{\text{red}, \text{yellow}\}$

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

# Example: product-TS

IS2.5-26



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

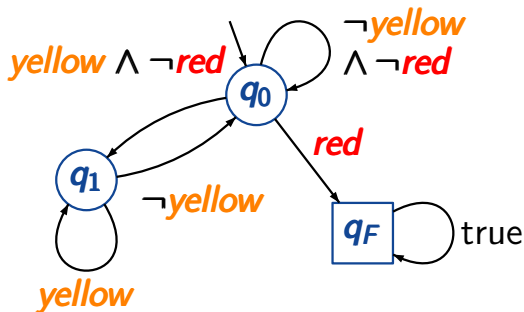
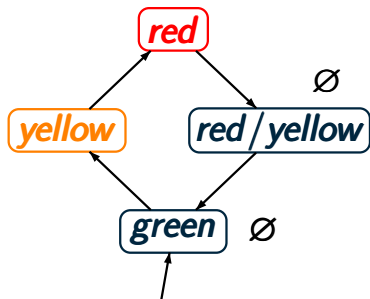
DFA  $\mathcal{A}$  for the  
bad prefixes for  $E$

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



# Example: product-TS

IS2.5-26



green  $q_0$

red/yellow  $q_0$

yellow  $q_1$

red  $q_0$

...

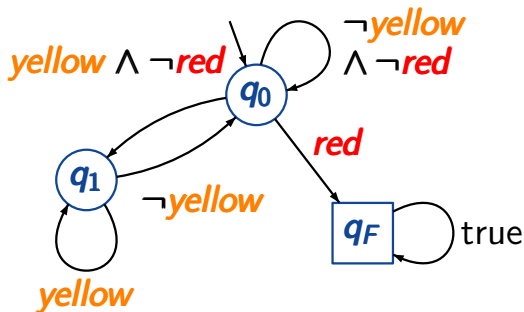
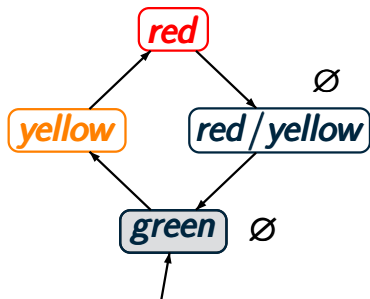
product-TS

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

(4 \* 3 = 12 states)

# Example: product-TS

IS2.5-26



green  $q_0$

red/yellow  $q_0$

yellow  $q_1$

red  $q_0$

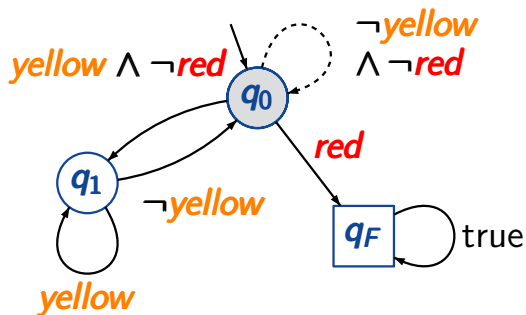
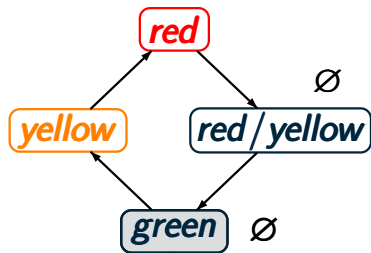
...

initial state  
 $\langle \text{green}, \delta(q_0, \emptyset) \rangle$

$$L(\text{green}) \stackrel{\uparrow}{=} \emptyset$$

# Example: product-TS

IS2.5-26



**green**  $q_0$

**red/yellow**  $q_0$

**yellow**  $q_1$

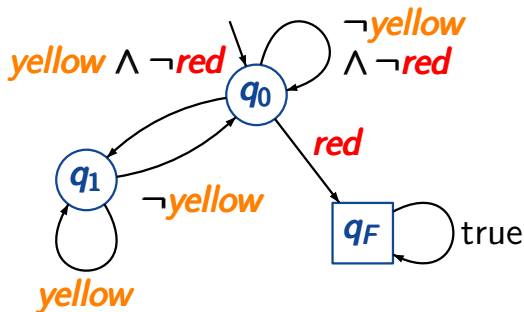
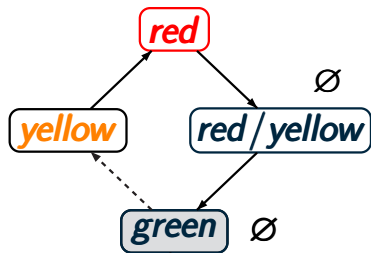
**red**  $q_0$

...

initial state  
 $\langle \text{green}, \underbrace{\delta(q_0, \emptyset)}_{= q_0} \rangle$

# Example: product-TS

IS2.5-26



green  $q_0$

red/yellow  $q_0$

yellow  $q_1$

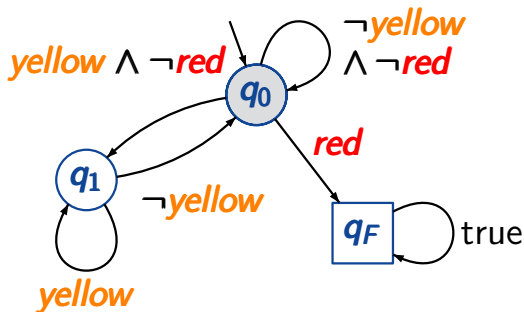
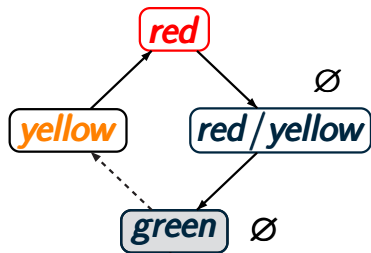
red  $q_0$

...

lifting the transition  
 $green \rightarrow yellow$

# Example: product-TS

IS2.5-26



green  $q_0$

red/yellow  $q_0$

yellow  $q_1$

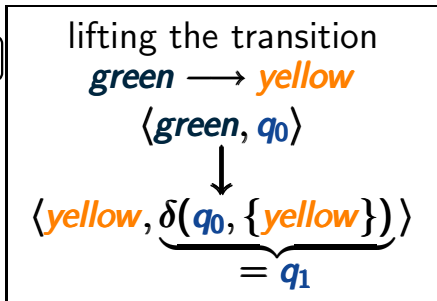
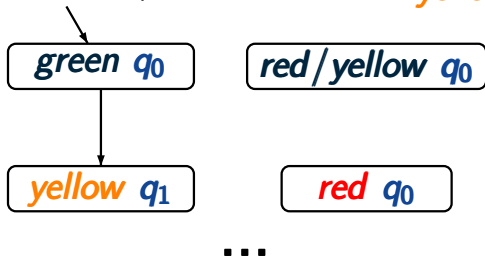
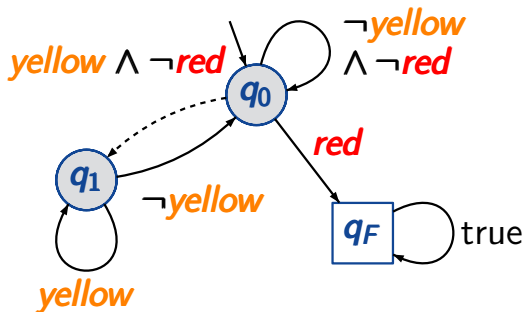
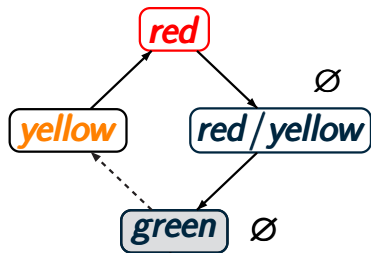
red  $q_0$

...

lifting the transition  
 $green \rightarrow yellow$   
 $\langle green, q_0 \rangle$   
 $\downarrow$   
 $\langle yellow, ? \rangle$

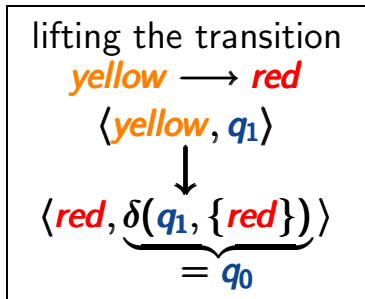
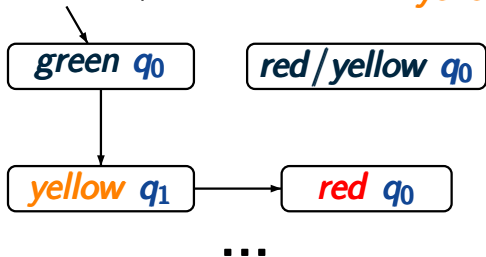
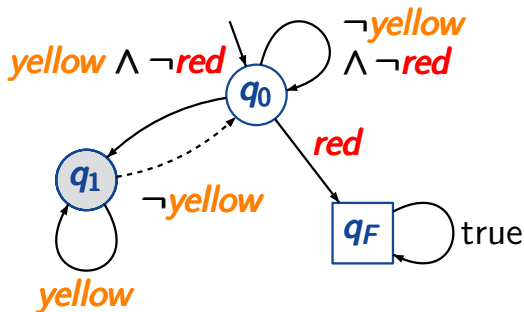
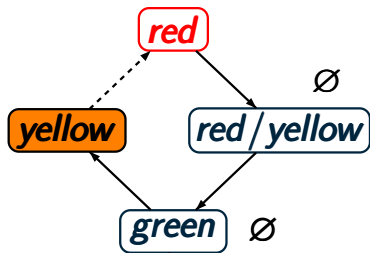
# Example: product-TS

IS2.5-26

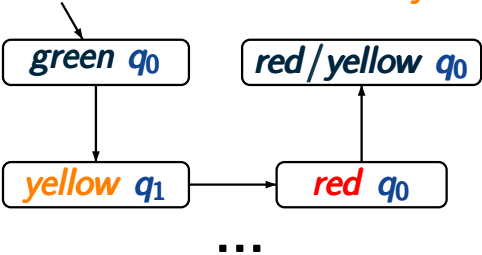
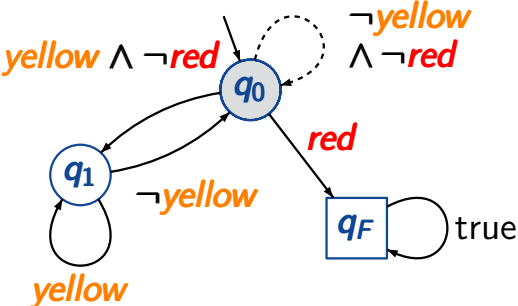
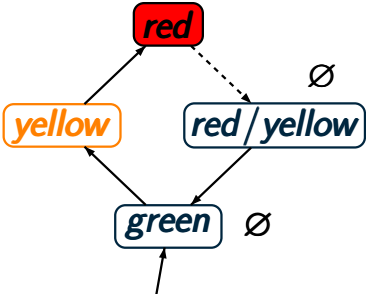


# Example: product-TS

IS2.5-26



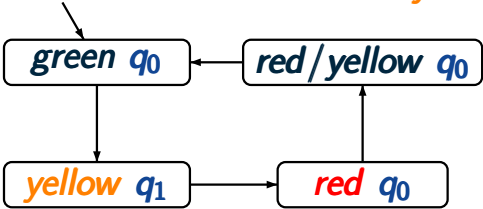
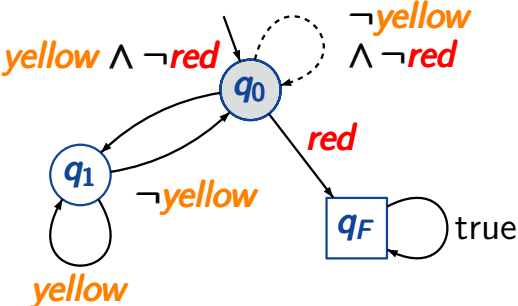
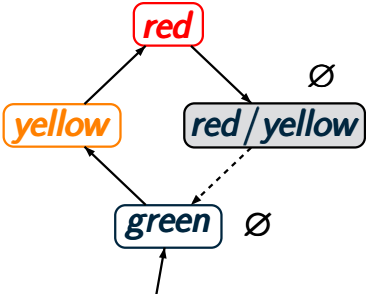
# Example: product-TS



lifting the transition  
 $red \rightarrow red/yellow$   
 $\langle red, q_0 \rangle$   
 $\downarrow$   
 $\langle red/yellow, \underbrace{\delta(q_0, \emptyset)}_{= q_0} \rangle$



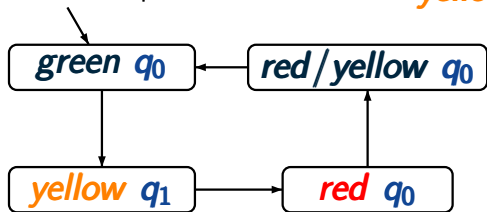
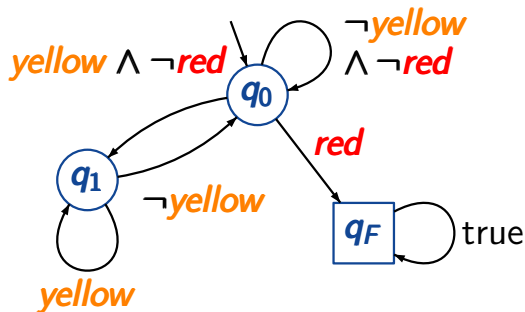
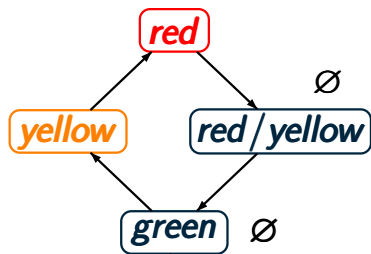
# Example: product-TS



lifting the transition  
 $red/yellow \rightarrow green$   
 $\langle red/yellow, q_0 \rangle$   
 $\downarrow$   
 $\langle green, \underbrace{\delta(q_0, \emptyset)}_{= q_0} \rangle$

# Example: product-TS

IS2.5-26



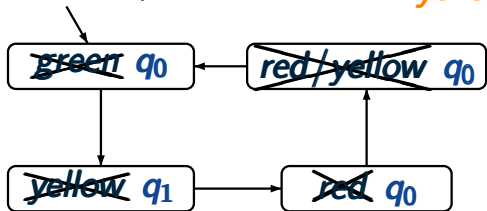
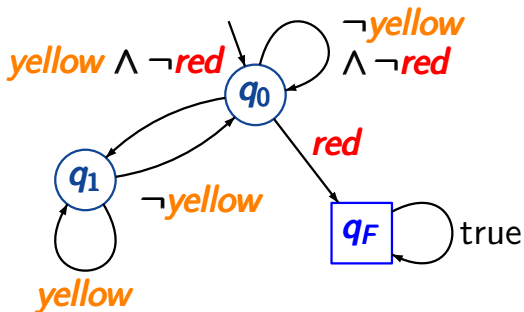
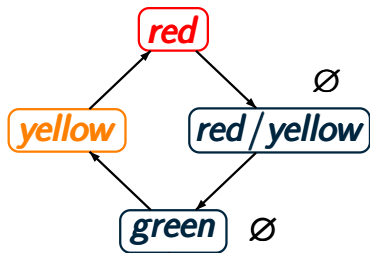
product-TS

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

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

# Example: product-TS

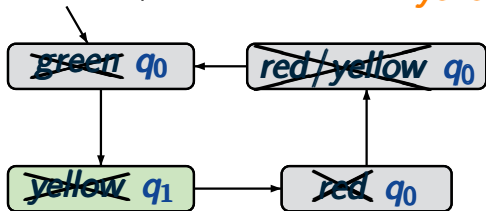
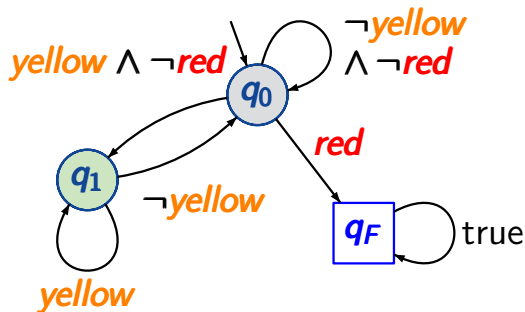
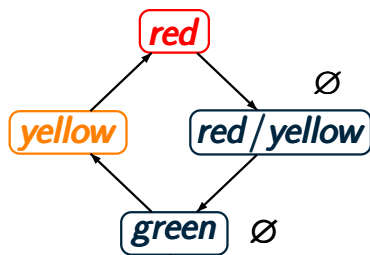
IS2.5-26



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

# Example: product-TS

IS2.5-26



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

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

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

definition of the product of

- a transition system  $\mathcal{T} = (\mathcal{S}, \text{Act}, \rightarrow, \mathcal{S}_0, \text{AP}, L)$

↑  
without terminal states

- an NFA  $\mathcal{A} = (\mathcal{Q}, 2^{\text{AP}}, \delta, \mathcal{Q}_0, F)$

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

definition of the product of

- a transition system  $\mathcal{T} = (\mathcal{S}, \mathit{Act}, \rightarrow, \mathcal{S}_0, \mathit{AP}, \mathit{L})$

without terminal states

- an NFA  $\mathcal{A} = (\mathcal{Q}, 2^{\mathit{AP}}, \delta, \mathcal{Q}_0, \mathit{F})$

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

without terminal states

definition of the product of

- a transition system  $\mathcal{T} = (\mathcal{S}, \mathit{Act}, \rightarrow, \mathcal{S}_0, \mathit{AP}, \mathit{L})$

without terminal states

- an NFA  $\mathcal{A} = (\mathcal{Q}, 2^{\mathit{AP}}, \delta, \mathcal{Q}_0, \mathit{F})$

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

without terminal states

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



definition of the product of

- a transition system  $\mathcal{T} = (\mathcal{S}, \text{Act}, \rightarrow, \mathcal{S}_0, \text{AP}, L)$

without terminal states

- an NFA  $\mathcal{A} = (\mathcal{Q}, 2^{\text{AP}}, \delta, \mathcal{Q}_0, F)$

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

without terminal states

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

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

$$\mathcal{Q}_0 \neq \emptyset \wedge \forall q \in \mathcal{Q} \forall A \in 2^{\text{AP}}. \delta(q, A) \neq \emptyset$$

definition of the product of

- a transition system  $\mathcal{T} = (\mathcal{S}, \text{Act}, \rightarrow, \mathcal{S}_0, \text{AP}, L)$

without terminal states

- an NFA  $\mathcal{A} = (\mathcal{Q}, 2^{\text{AP}}, \delta, \mathcal{Q}_0, F)$

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

without terminal states

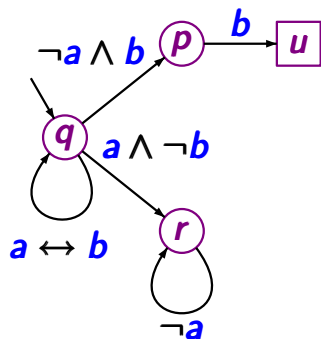
assumptions on the NFA  $\mathcal{A}$ :

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

$$\mathcal{Q}_0 \neq \emptyset \wedge \forall q \in \mathcal{Q} \forall A \in 2^{\text{AP}}. \delta(q, A) \neq \emptyset$$

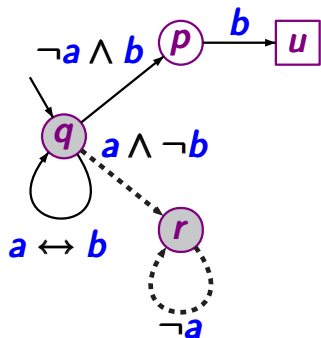
- no initial state of  $\mathcal{A}$  is final, i.e.,  $\mathcal{Q}_0 \cap F = \emptyset$

NFA  $\mathcal{A}$



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

NFA  $\mathcal{A}$



blocks for input  
 $\{a\} \not\subseteq \{a\}$

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

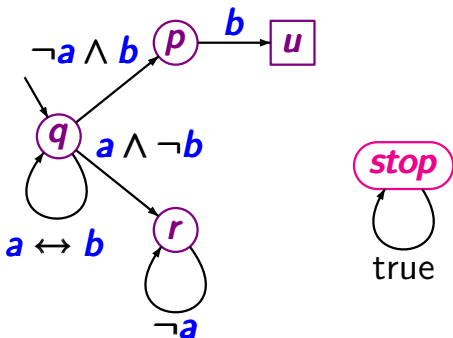
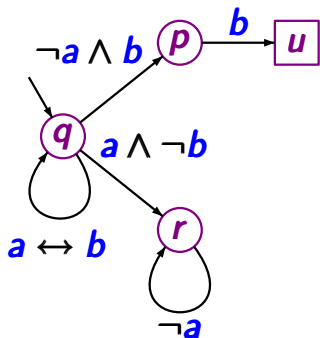
# Non-blocking NFA

IS2.5-23

NFA  $\mathcal{A}$



equivalent NFA  $\mathcal{A}'$



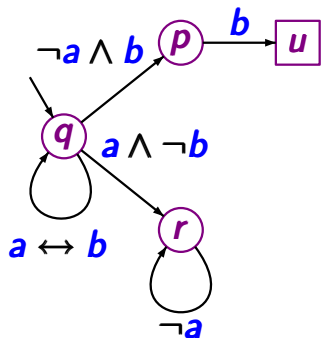
blocks for input  
 $\{a\} \emptyset \{a\}$

add a trap state *stop*

# Non-blocking NFA

IS2.5-23

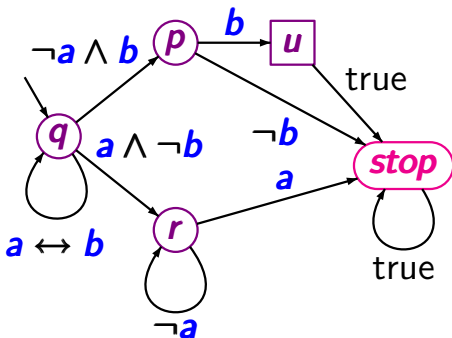
NFA  $\mathcal{A}$



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

$\rightsquigarrow$

equivalent NFA  $\mathcal{A}'$

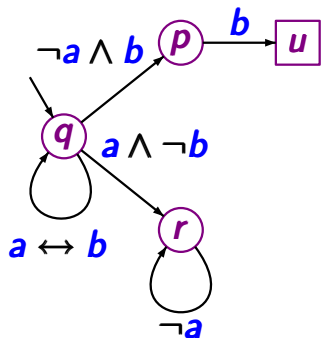


add a trap state *stop*

# Non-blocking NFA

IS2.5-23

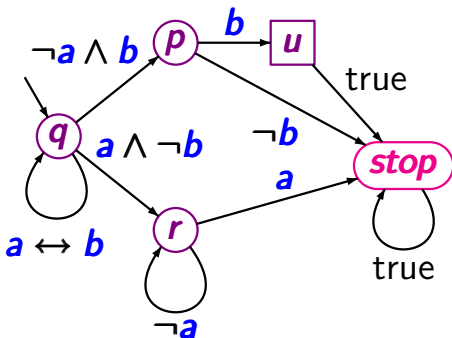
NFA  $\mathcal{A}$



blocks for input  
 $\{a\} \not\subseteq \{a\}$

$\rightsquigarrow$

equivalent NFA  $\mathcal{A}'$

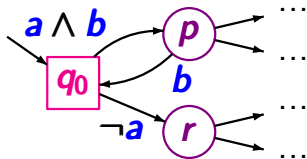


non-blocking

# NFA where no initial state is final

IS2.5-24

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

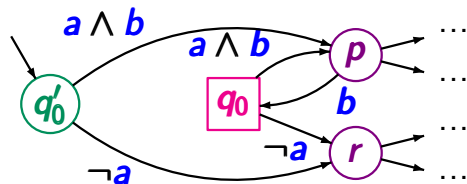
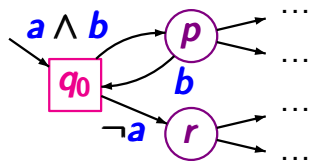




# NFA where no initial state is final

IS2.5-24

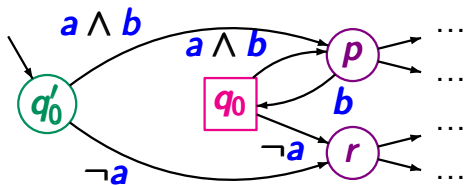
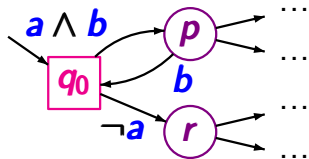
NFA  $\mathcal{A}$  with  $Q_0 \cap F \neq \emptyset \rightsquigarrow$  NFA  $\mathcal{A}'$  with  $Q_0 \cap F = \emptyset$



# NFA where no initial state is final

IS2.5-24

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

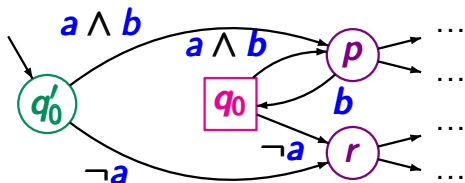
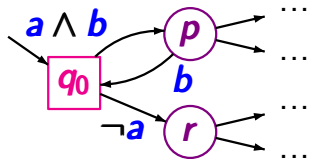


$$\mathcal{L}(\mathcal{A}') = \mathcal{L}(\mathcal{A}) \setminus \{\epsilon\}$$

# NFA where no initial state is final

IS2.5-24

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



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

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

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



... via a reduction to invariant checking .....

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

Let  $\mathcal{T} = (\mathcal{S}, \text{Act}, \rightarrow, \mathcal{S}_0, \text{AP}, L)$  be a transition system  
(without terminal states)

$\mathcal{A} = (\mathcal{Q}, 2^{\text{AP}}, \delta, \mathcal{Q}_0, F)$  be an NFA  
for the bad prefixes of a regular safety property  $E$

Let  $\mathcal{T} = (\mathcal{S}, \text{Act}, \rightarrow, \mathcal{S}_0, \text{AP}, L)$  be a transition system  
(without terminal states)

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



Let  $\mathcal{T} = (\mathcal{S}, \text{Act}, \rightarrow, \mathcal{S}_0, \text{AP}, \mathcal{L})$  be a transition system  
(without terminal states)

$\mathcal{A} = (\mathcal{Q}, \Sigma^{\text{AP}}, \delta, \mathcal{Q}_0, \mathcal{F})$  be an NFA  
for the bad prefixes of a regular safety property  $E$   
(non-blocking and  $\mathcal{Q}_0 \cap \mathcal{F} = \emptyset$ )

The following statements are equivalent:

- (1)  $\mathcal{T} \models E$
- (2)  $\text{Traces}_{\text{fin}}(\mathcal{T}) \cap \mathcal{L}(\mathcal{A}) = \emptyset$

Let  $\mathcal{T} = (\mathcal{S}, \text{Act}, \rightarrow, \mathcal{S}_0, \text{AP}, \mathcal{L})$  be a transition system  
(without terminal states)

$\mathcal{A} = (\mathcal{Q}, \Sigma^{\text{AP}}, \delta, \mathcal{Q}_0, \mathcal{F})$  be an NFA

for the bad prefixes of a regular safety property  $E$   
(non-blocking and  $\mathcal{Q}_0 \cap \mathcal{F} = \emptyset$ )

The following statements are equivalent:

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

(2)  $\text{Traces}_{\text{fin}}(\mathcal{T}) \cap \mathcal{L}(\mathcal{A}) = \emptyset$

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

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

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

The following statements are equivalent:

- (1)  $\mathcal{T} \models E$
- (2)  $\text{Traces}_{fin}(\mathcal{T}) \cap \mathcal{L}(\mathcal{A}) = \emptyset$
- (3)  $\mathcal{T} \otimes \mathcal{A} \models$  invariant “always  $\neg F$ ”

where “ $\neg F$ ” denotes  $\bigwedge_{q \in F} \neg q$

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

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

product-TS  $\mathcal{T} \otimes \mathcal{A} \stackrel{\text{def}}{=} (S \times Q, Act, \longrightarrow', S'_0, AP', L')$

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

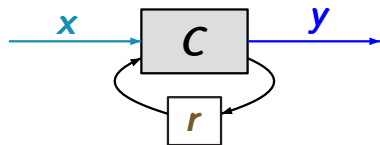
initial states:  $S'_0 = \{ \langle s_0, q \rangle : s_0 \in S_0, q \in \delta(Q_0, L(s_0)) \}$

set of atomic propositions:  $AP' = Q$

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

# Example: sequential circuit

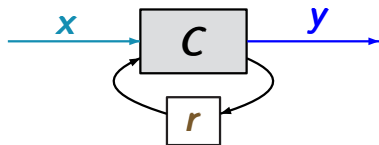
IS2.5-27



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

# Example: sequential circuit

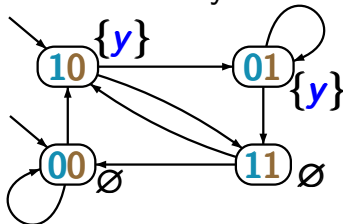
IS2.5-27



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

initially  $r = 0$

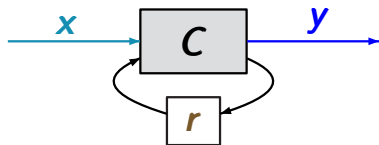
transition system  $\mathcal{T}$



over  $AP = \{y\}$

# Example: sequential circuit

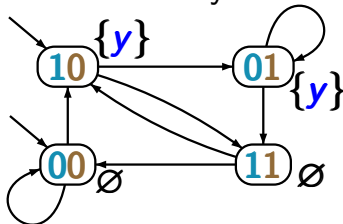
IS2.5-27



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

initially  $r = 0$

transition system  $\mathcal{T}$



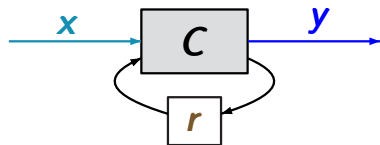
over  $AP = \{y\}$

safety property  $E$

*The circuit will never output two ones after each other*

# Example: sequential circuit

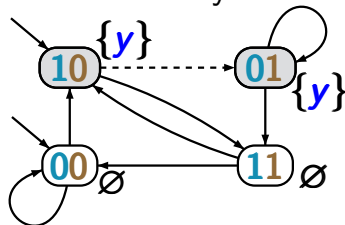
IS2.5-27



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

initially  $r = 0$

transition system  $\mathcal{T}$



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

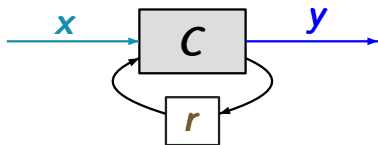
safety property  $E$

*The circuit will never output two ones after each other*



# Example: sequential circuit

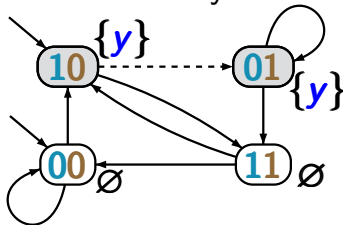
is2.5-27



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

initially  $r = 0$

transition system  $\mathcal{T}$



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

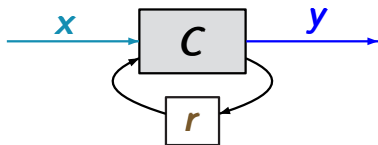
error indication, e.g.,  
 $\langle 10 \rangle \langle 01 \rangle$

safety property  $E$

*The circuit will never  
output two ones  
after each other*

# Example: sequential circuit

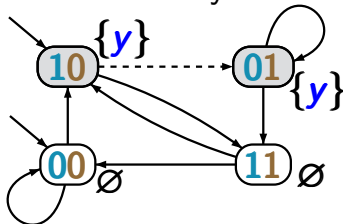
IS2.5-27



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

initially  $r = 0$

transition system  $\mathcal{T}$



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

error indication, e.g.,  
 $\langle 10 \rangle \langle 01 \rangle$

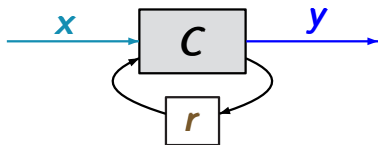
bad prefix:  $\{y\} \{y\}$

safety property  $E$

*The circuit will never  
output two ones  
after each other*

# Example: sequential circuit

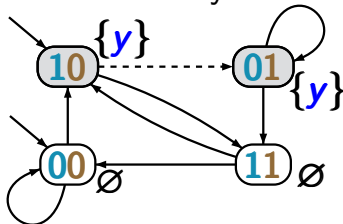
IS2.5-27



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

initially  $r = 0$

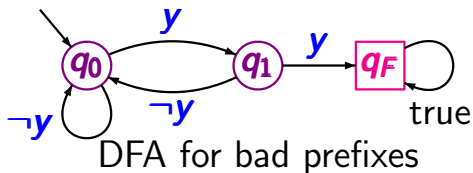
transition system  $\mathcal{T}$



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

error indication, e.g.,  
 $\langle 10 \rangle \langle 01 \rangle$

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

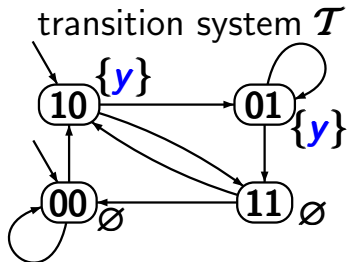


safety property  $E$

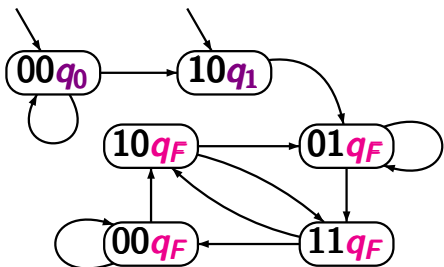
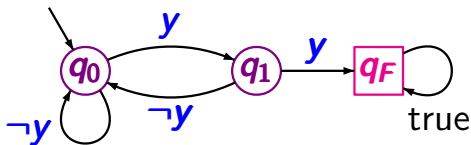
*The circuit will never output two ones after each other*

# Example: product-TS

IS2.5-28



safety property  $E$   
... never two ones in a row ...

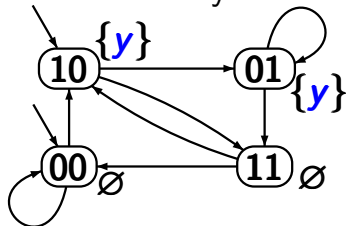


product-TS  $\mathcal{T} \otimes \mathcal{A}$

# Example: product-TS

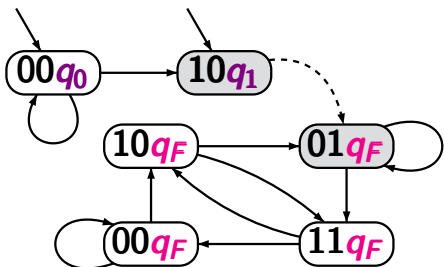
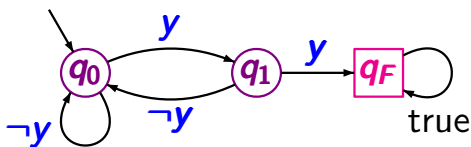
IS2.5-28

transition system  $\mathcal{T}$



safety property  $E$

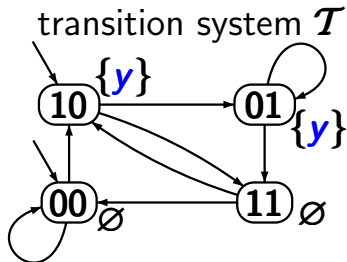
... never two ones in a row ...



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

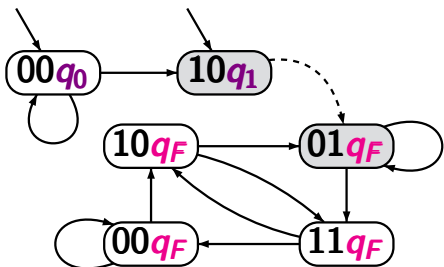
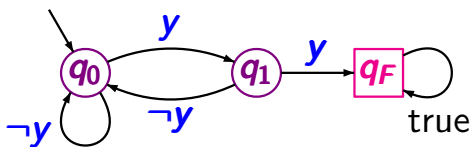
# Example: product-TS

IS2.5-28



safety property  $E$

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

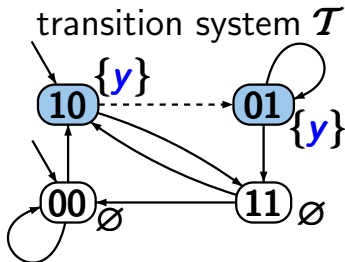


error indication for  $\mathcal{T} \otimes \mathcal{A} \not\models$  "never  $q_F$ "

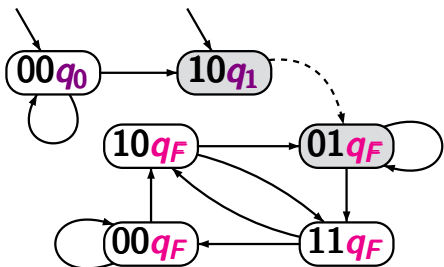
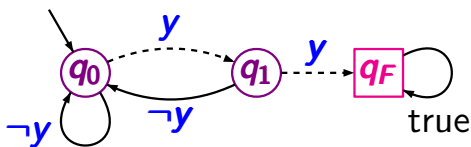
$10q_1$      $01q_F$

# Example: product-TS

IS2.5-28

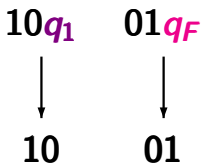


safety property  $E$   
 ... never two ones in a row ...



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

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







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

*output:* “yes” if  $\mathcal{T} \models E$   
otherwise “no”

*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  $\mathcal{A}$

*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” ← and an error indication

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

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

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

THEN return “yes”

ELSE

FI

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

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

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

FI

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

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

return “no” and  $s_0 s_1 \dots s_n$

FI



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

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

return “no” and  $s_0 s_1 \dots s_n$

FI

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

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

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

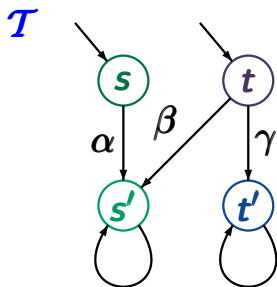
**correct.**

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

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

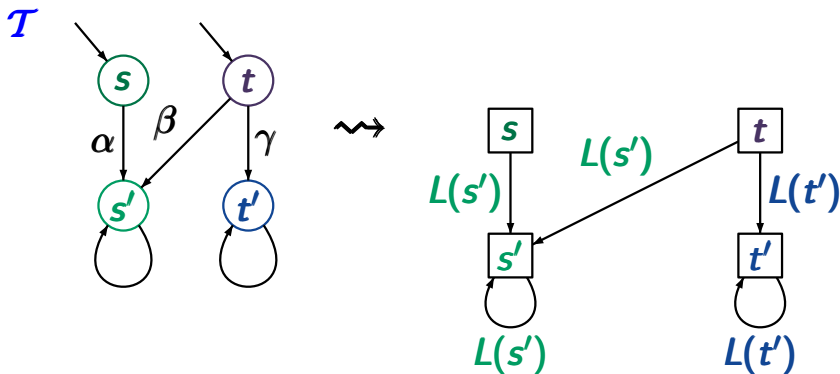
If  $\mathcal{T}$  is a finite transition system then  
 $Traces_{fin}(\mathcal{T})$  is regular.

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



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

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

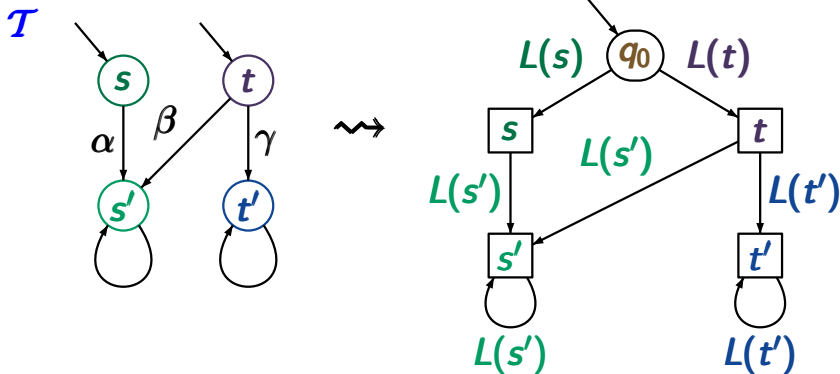


# Correct or wrong?

IS2.5-35

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?

IS2.5-35

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

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

