# General Picture of LTL Model Checking with Büchi Automata

Luca Tesei

Reactive Systems Verification

MSc in Computer Science

University of Camerino

## Topics

- Automata-based LTL model checking. General picture.
- From LTL formulas to NBAs. Examples.
- NFA and NBA for safety properties.
- Examples of LTL model checking with NBA.
- LTL model checking complexity (without proof).

## Material

Reading:

Chapter 5 of the book, pages 225-243.

More:

The slides in the following pages are taken from the material of the course "Introduction to Model Checking" held by Prof. Dr. Ir. Joost-Pieter Katoen at Aachen University.

# Overview

# LTL model checking problem

# LTL model checking problem



*given:* finite transition system $\mathcal{T}$ over $AP$
(without terminal states)
LTL-formula $\varphi$ over $AP$

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

# LTL model checking problem



*given:* finite transition system $\mathcal{T}$ over $AP$
(without terminal states)
LTL-formula $\varphi$ over $AP$

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

*basic idea*: try to refute $\mathcal{T} \models \varphi$

# LTL model checking problem



*given:* finite transition system $\mathcal{T}$ over $AP$
(without terminal states)
LTL-formula $\varphi$ over $AP$

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

*basic idea*: try to refute $\mathcal{T} \models \varphi$ by searching for a path $\pi$ in $\mathcal{T}$ s.t.

$$\pi \not\models \varphi$$

# LTL model checking problem



*given:* finite transition system $\mathcal{T}$ over $AP$
(without terminal states)
LTL-formula $\varphi$ over $AP$

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

*basic idea*: try to refute $\mathcal{T} \models \varphi$ by searching for a path $\pi$ in $\mathcal{T}$ s.t.

$$\pi \not\models \varphi, \text{ i.e., } \pi \models \neg\varphi$$

# The LTL model checking problem



*given:* finite transition system $\mathcal{T}$ over $AP$
LTL-formula $\varphi$ over $AP$

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

1. construct an **NBA** $\mathcal{A}$ for $Words(\neg\varphi)$

# The LTL model checking problem



*given:* finite transition system $\mathcal{T}$ over $AP$
LTL-formula $\varphi$ over $AP$

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

1. construct an **NBA** $\mathcal{A}$ for $Words(\neg\varphi)$
2. search a path $\pi$ in $\mathcal{T}$ with
$$trace(\pi) \in Words(\neg\varphi)$$

# The LTL model checking problem



*given:* finite transition system $\mathcal{T}$ over $AP$
LTL-formula $\varphi$ over $AP$

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

1. construct an **NBA** $\mathcal{A}$ for $Words(\neg\varphi)$
2. search a path $\pi$ in $\mathcal{T}$ with

$$trace(\pi) \in Words(\neg\varphi) = \mathcal{L}_\omega(\mathcal{A})$$

# The LTL model checking problem



*given:* finite transition system $\mathcal{T}$ over $AP$
LTL-formula $\varphi$ over $AP$

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

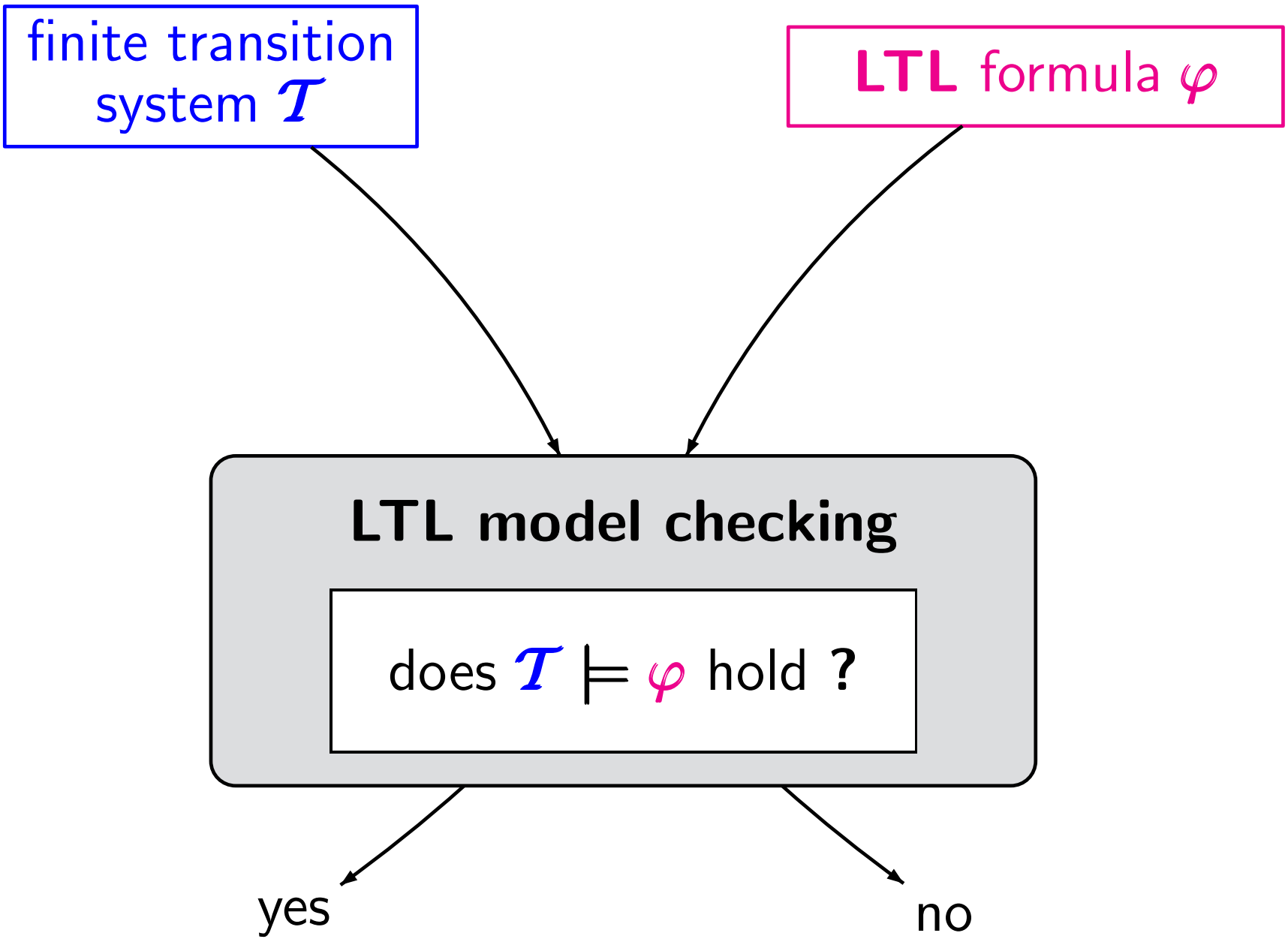
1. construct an **NBA** $\mathcal{A}$ for $Words(\neg\varphi)$
2. search a path $\pi$ in $\mathcal{T}$ with

$$trace(\pi) \in Words(\neg\varphi) = \mathcal{L}_\omega(\mathcal{A})$$

construct the product-TS $\mathcal{T} \otimes \mathcal{A}$
search a path in the product that meets the acceptance condition of $\mathcal{A}$

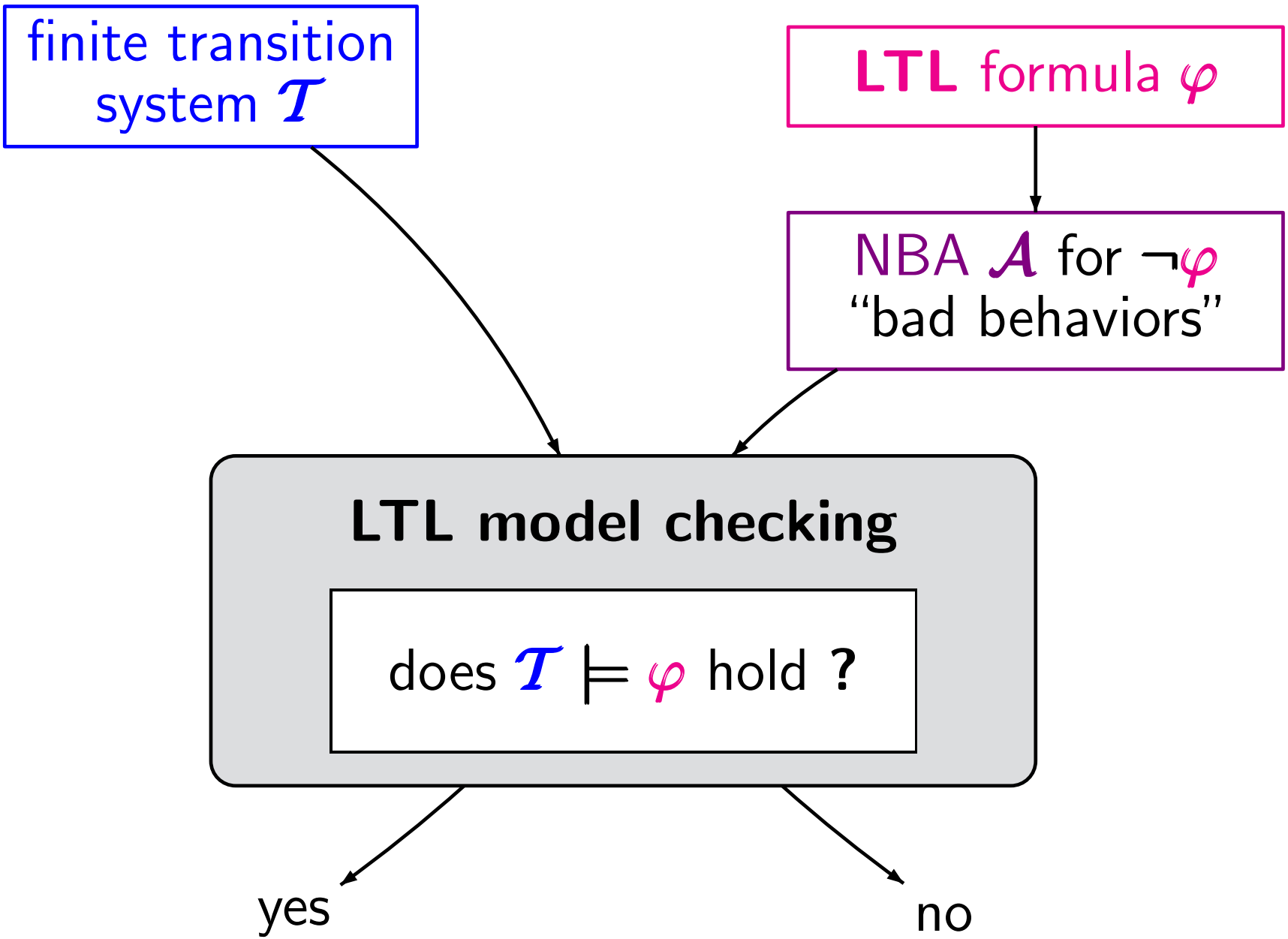
# Automata-based LTL model checking





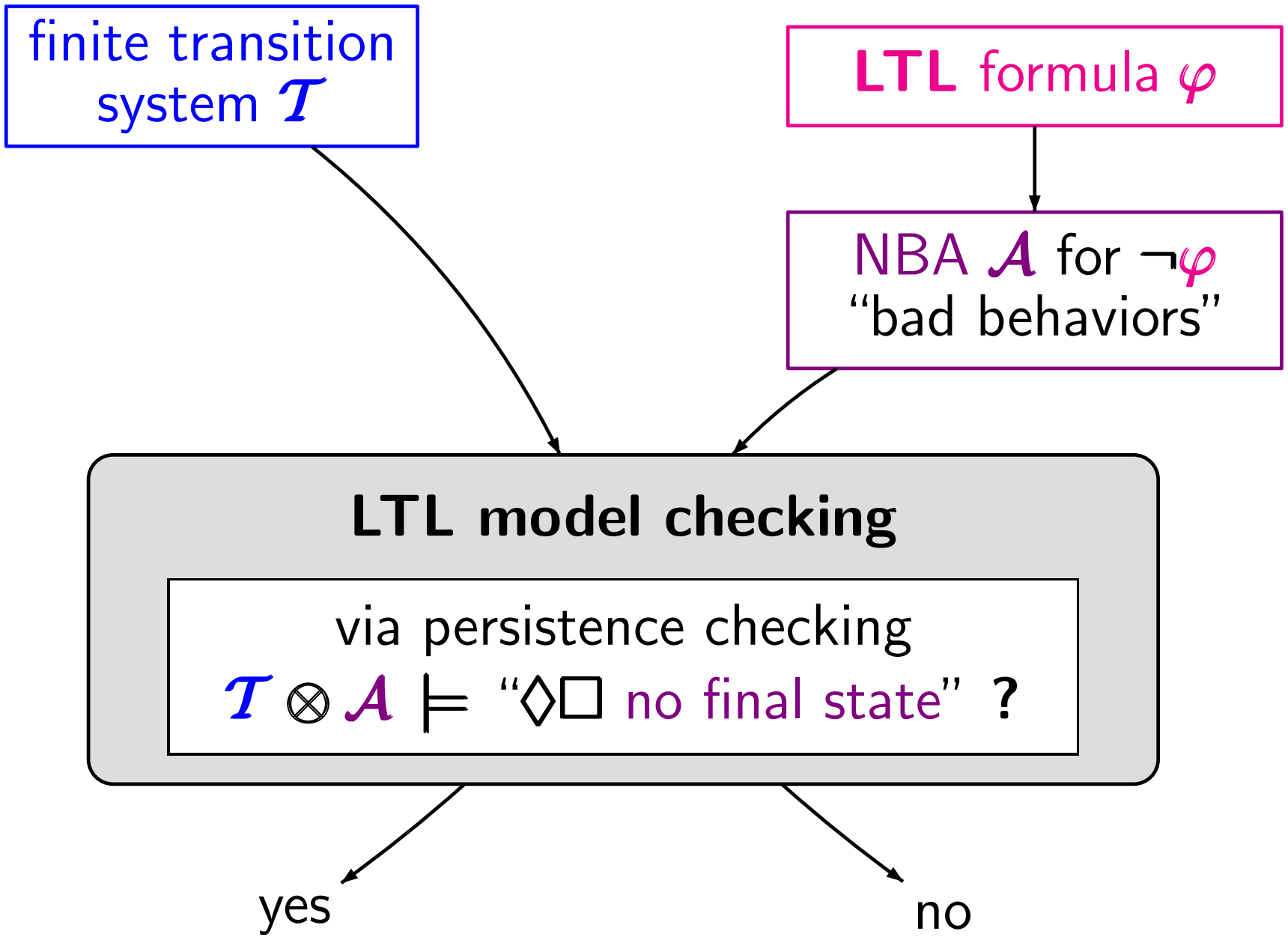
# Automata-based LTL model checking





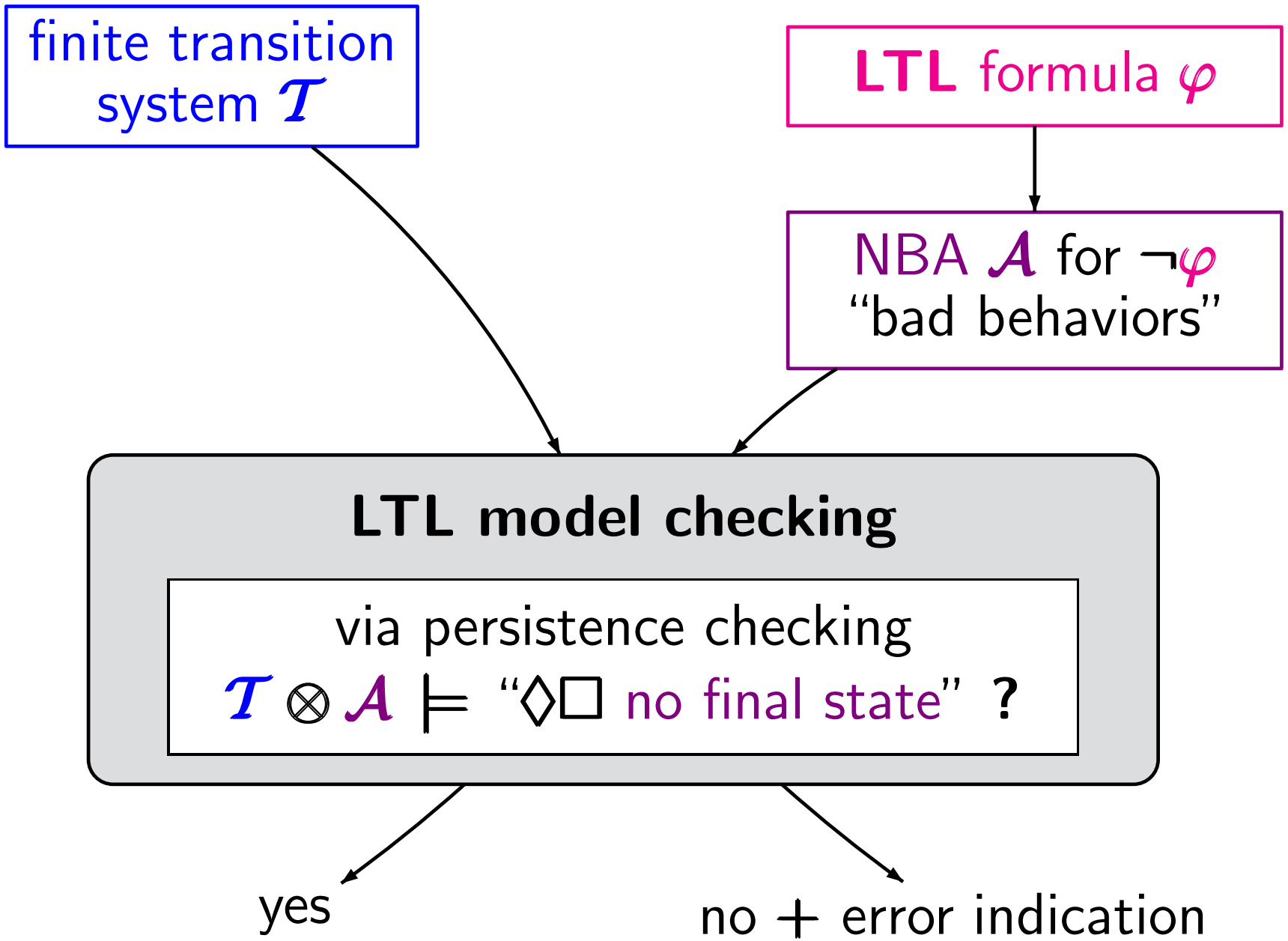
# Automata-based LTL model checking





# Automata-based LTL model checking





# Safety and LTL model checking

# Safety and LTL model checking



| safety property $E$ | LTL-formula $\varphi$ |
| --- | --- |
| | |

# Safety and LTL model checking



| safety property $E$ | LTL-formula $\varphi$ |
| --- | --- |
| **NFA** for the bad prefixes for $E$ $\mathcal{L}(\mathcal{A}) \subseteq (2^{AP})^+$ | |

# Safety and LTL model checking



| safety property $E$ | LTL-formula $\varphi$ |
|---|---|
| **NFA** for the bad prefixes for $E$ $\mathcal{L}(\mathcal{A}) \subseteq (2^{AP})^+$ | **NBA** for the "bad behaviors" $\mathcal{L}_\omega(\mathcal{A}) = \mathit{Words}(\neg\varphi)$ |

# Safety and LTL model checking



| safety property $E$ | LTL-formula $\varphi$ |
|---|---|
| **NFA** for the bad prefixes for $E$ $\mathcal{L}(\mathcal{A}) \subseteq (2^{AP})^+$ | **NBA** for the "bad behaviors" $\mathcal{L}_\omega(\mathcal{A}) = \mathit{Words}(\neg\varphi)$ |
| $\mathit{Traces}_{\mathit{fin}}(\mathcal{T}) \cap \mathcal{L}(\mathcal{A}) = \varnothing$ | |

# Safety and LTL model checking



| safety property $E$ | LTL-formula $\varphi$ |
|---|---|
| **NFA** for the bad prefixes for $E$ $\mathcal{L}(\mathcal{A}) \subseteq (2^{AP})^+$ | **NBA** for the "bad behaviors" $\mathcal{L}_\omega(\mathcal{A}) = \mathit{Words}(\neg\varphi)$ |
| $\mathit{Traces}_{\mathit{fin}}(\mathcal{T}) \cap \mathcal{L}(\mathcal{A}) = \varnothing$ | $\mathit{Traces}(\mathcal{T}) \cap \mathcal{L}_\omega(\mathcal{A}) = \varnothing$ |

# Safety and LTL model checking



| safety property $E$ | LTL-formula $\varphi$ |
|---|---|
| **NFA** for the bad prefixes for $E$ $\mathcal{L}(\mathcal{A}) \subseteq (2^{AP})^+$ | **NBA** for the "bad behaviors" $\mathcal{L}_\omega(\mathcal{A}) = \mathit{Words}(\neg\varphi)$ |
| $\mathit{Traces}_{\mathit{fin}}(\mathcal{T}) \cap \mathcal{L}(\mathcal{A}) = \varnothing$ | $\mathit{Traces}(\mathcal{T}) \cap \mathcal{L}_\omega(\mathcal{A}) = \varnothing$ |
| invariant checking in the product $\mathcal{T} \otimes \mathcal{A} \models \Box\neg F$ ? | |

# Safety and LTL model checking



| safety property $E$ | LTL-formula $\varphi$ |
|---|---|
| **NFA** for the bad prefixes for $E$ $\mathcal{L}(\mathcal{A}) \subseteq (2^{AP})^+$ | **NBA** for the "bad behaviors" $\mathcal{L}_\omega(\mathcal{A}) = \mathit{Words}(\neg\varphi)$ |
| $\mathit{Traces}_{\mathit{fin}}(\mathcal{T}) \cap \mathcal{L}(\mathcal{A}) = \varnothing$ | $\mathit{Traces}(\mathcal{T}) \cap \mathcal{L}_\omega(\mathcal{A}) = \varnothing$ |
| invariant checking in the product $\mathcal{T} \otimes \mathcal{A} \models \Box\neg F$ ? | persistence checking in the product $\mathcal{T} \otimes \mathcal{A} \models \Diamond\Box\neg F$ ? |

# Safety and LTL model checking



| safety property $E$ | LTL-formula $\varphi$ |
|---|---|
| **NFA** for the bad prefixes for $E$ $\mathcal{L}(\mathcal{A}) \subseteq (2^{AP})^+$ | **NBA** for the "bad behaviors" $\mathcal{L}_\omega(\mathcal{A}) = \mathit{Words}(\neg\varphi)$ |
| $\mathit{Traces}_{\mathit{fin}}(\mathcal{T}) \cap \mathcal{L}(\mathcal{A}) = \varnothing$ | $\mathit{Traces}(\mathcal{T}) \cap \mathcal{L}_\omega(\mathcal{A}) = \varnothing$ |
| invariant checking in the product $\mathcal{T} \otimes \mathcal{A} \models \Box\neg F$ ? | persistence checking in the product $\mathcal{T} \otimes \mathcal{A} \models \Diamond\Box\neg F$ ? |
| error indication: $\widehat{\pi} \in \mathit{Paths}_{\mathit{fin}}(\mathcal{T})$ s.t. $\mathit{trace}(\widehat{\pi}) \in \mathcal{L}(\mathcal{A})$ | |

# Safety and LTL model checking



| safety property $E$ | LTL-formula $\varphi$ |
|---|---|
| **NFA** for the bad prefixes for $E$ $\mathcal{L}(\mathcal{A}) \subseteq (2^{AP})^+$ | **NBA** for the "bad behaviors" $\mathcal{L}_\omega(\mathcal{A}) = \mathit{Words}(\neg\varphi)$ |
| $\mathit{Traces}_{\mathit{fin}}(\mathcal{T}) \cap \mathcal{L}(\mathcal{A}) = \varnothing$ | $\mathit{Traces}(\mathcal{T}) \cap \mathcal{L}_\omega(\mathcal{A}) = \varnothing$ |
| invariant checking in the product $\mathcal{T} \otimes \mathcal{A} \models \Box\neg F$ ? | persistence checking in the product $\mathcal{T} \otimes \mathcal{A} \models \Diamond\Box\neg F$ ? |
| error indication: $\widehat{\pi} \in \mathit{Paths}_{\mathit{fin}}(\mathcal{T})$ s.t. $\mathit{trace}(\widehat{\pi}) \in \mathcal{L}(\mathcal{A})$ | error indication: prefix of a path $\pi$ s.t. $\mathit{trace}(\pi) \in \mathcal{L}_\omega(\mathcal{A})$ |

# Safety vs LTL model checking

# Safety vs LTL model checking

$\mathcal{T} \models$ safety property $E$

iff $\mathit{Traces}_{\mathit{fin}}(\mathcal{T}) \cap \mathcal{L}(\mathcal{A}) = \varnothing$

where $\mathcal{A}$ is an NFA for the bad prefixes

---

$\mathcal{T} \models$ LTL-formula $\varphi$

iff $\mathit{Traces}(\mathcal{T}) \cap \mathcal{L}_\omega(\mathcal{A}) = \varnothing$

where $\mathcal{A}$ is an NBA for $\neg\varphi$

# Safety vs LTL model checking



$\mathcal{T} \models$ safety property $E$

iff $\mathit{Traces}_{\mathit{fin}}(\mathcal{T}) \cap \mathcal{L}(\mathcal{A}) = \varnothing$

iff there is no path fragment $\langle s_0, q_0\rangle \langle s_1, q_1\rangle \ldots \langle s_n, q_n\rangle$ in $\mathcal{T} \otimes \mathcal{A}$ s. t. $q_n \in F$

---

$\mathcal{T} \models$ LTL-formula $\varphi$

iff $\mathit{Traces}(\mathcal{T}) \cap \mathcal{L}_\omega(\mathcal{A}) = \varnothing$

iff there is no path $\langle s_0, q_0\rangle \langle s_1, q_1\rangle \langle s_2, q_2\rangle \ldots$ in $\mathcal{T} \otimes \mathcal{A}$ s.t. $q_i \in F$ for infinitely many $i \in \mathbb{N}$

# Safety vs LTL model checking

$\mathcal{T} \models$ safety property $E$

iff $\mathit{Traces}_{\mathit{fin}}(\mathcal{T}) \cap \mathcal{L}(\mathcal{A}) = \varnothing$

iff there is no path fragment $\langle s_0, q_0\rangle \langle s_1, q_1\rangle \ldots \langle s_n, q_n\rangle$ in $\mathcal{T} \otimes \mathcal{A}$ s. t. $q_n \in F$

iff $\mathcal{T} \otimes \mathcal{A} \models \Box\neg F$

---

$\mathcal{T} \models$ LTL-formula $\varphi$

iff $\mathit{Traces}(\mathcal{T}) \cap \mathcal{L}_\omega(\mathcal{A}) = \varnothing$

iff there is no path $\langle s_0, q_0\rangle \langle s_1, q_1\rangle \langle s_2, q_2\rangle \ldots$ in $\mathcal{T} \otimes \mathcal{A}$ s.t. $q_i \in F$ for infinitely many $i \in \mathbb{N}$

iff $\mathcal{T} \otimes \mathcal{A} \models \Diamond\Box\neg F$

# Safety vs LTL model checking

$\mathcal{T} \models$ safety property $E$

iff $\mathit{Traces}_{\mathit{fin}}(\mathcal{T}) \cap \mathcal{L}(\mathcal{A}) = \varnothing$

iff there is no path fragment $\langle s_0, q_0\rangle \langle s_1, q_1\rangle \ldots \langle s_n, q_n\rangle$ in $\mathcal{T} \otimes \mathcal{A}$ s. t. $q_n \in F$

iff $\mathcal{T} \otimes \mathcal{A} \models \Box\neg F$ $\longleftarrow$ invariant checking

---

$\mathcal{T} \models$ LTL-formula $\varphi$

iff $\mathit{Traces}(\mathcal{T}) \cap \mathcal{L}_\omega(\mathcal{A}) = \varnothing$

iff there is no path $\langle s_0, q_0\rangle \langle s_1, q_1\rangle \langle s_2, q_2\rangle \ldots$ in $\mathcal{T} \otimes \mathcal{A}$ s.t. $q_i \in F$ for infinitely many $i \in \mathbb{N}$

iff $\mathcal{T} \otimes \mathcal{A} \models \Diamond\Box\neg F$ $\longleftarrow$ persistence checking

# Recall: nondeterministic Büchi automata



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

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

# Recall: nondeterministic Büchi automata



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

- $Q$ finite set of states
- $\Sigma$ 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 $A_0 A_1 A_2 \ldots \in \Sigma^\omega$:

state sequence $\pi = q_0 q_1 q_2 \ldots$ where $q_0 \in Q_0$ and $q_{i+1} \in \delta(q_i, A_i)$ for $i \geq 0$

run $\pi$ is accepting if $\overset{\infty}{\exists}\, i \in \mathbb{N}.\ q_i \in F$

# Recall: nondeterministic Büchi automata



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

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

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

$$\mathcal{L}_\omega(\mathcal{A}) \stackrel{\text{def}}{=} \text{set of infinite words over } \Sigma \text{ that have an accepting run in } \mathcal{A}$$

# Recall: nondeterministic Büchi automata



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

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

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

$$\mathcal{L}_\omega(\mathcal{A}) \stackrel{\text{def}}{=} \text{set of infinite words over } \Sigma \text{ that have an accepting run in } \mathcal{A}$$

# From LTL to NBA

# From LTL to NBA



For each **LTL** formula $\varphi$ over $AP$ there is an **NBA** $\mathcal{A}$ over the alphabet $2^{AP}$ such that

$$Words(\varphi) = \mathcal{L}_\omega(\mathcal{A})$$

# From LTL to NBA



For each **LTL** formula $\varphi$ over $AP$ there is an **NBA** $\mathcal{A}$ over the alphabet $2^{AP}$ such that

- $Words(\varphi) = \mathcal{L}_\omega(\mathcal{A})$
- $size(\mathcal{A}) = \mathcal{O}(\exp(|\varphi|))$

# From LTL to NBA



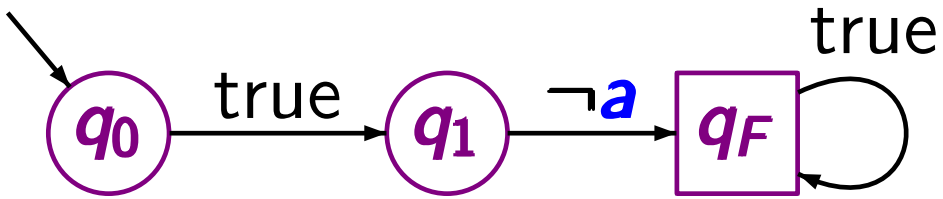
For each **LTL** formula $\varphi$ over $AP$ there is an **NBA** $\mathcal{A}$ over the alphabet $2^{AP}$ such that

- $Words(\varphi) = \mathcal{L}_\omega(\mathcal{A})$
- $size(\mathcal{A}) = \mathcal{O}(\exp(|\varphi|))$

*proof:* ... later ...

# NBA for LTL formulas

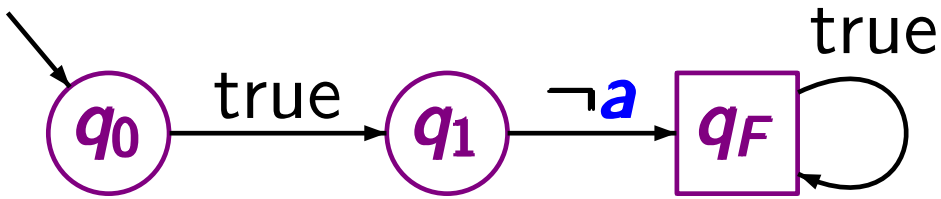





$$\mathcal{L}_\omega(\mathcal{A}) = \,?$$

# NBA for LTL formulas

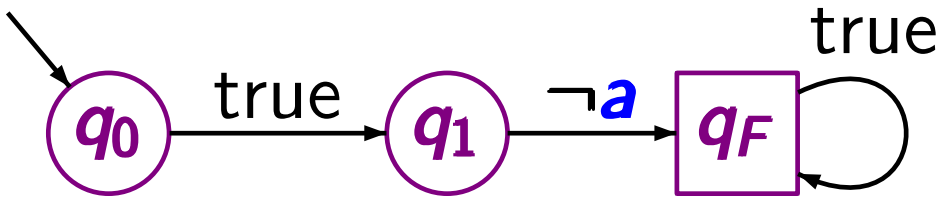





$$\mathcal{L}_\omega(\mathcal{A}) = \mathit{Words}(\bigcirc \neg a)$$

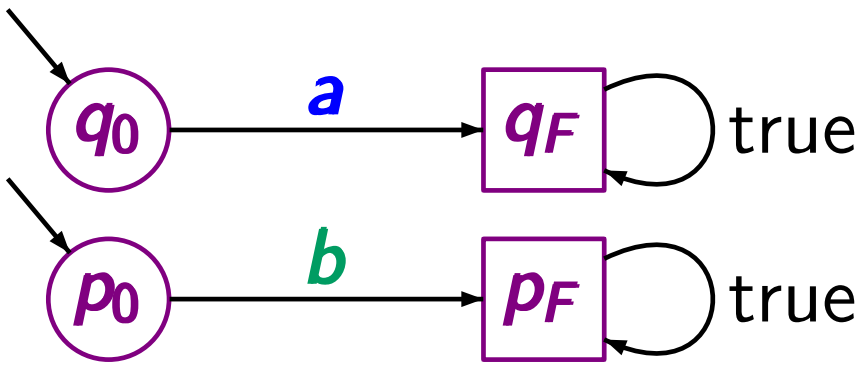
# NBA for LTL formulas






$$\mathcal{L}_\omega(\mathcal{A}) = \mathit{Words}(\bigcirc \neg a)$$

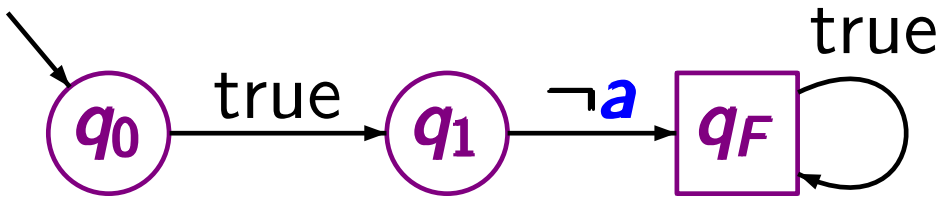
---




$$\mathcal{L}_\omega(\mathcal{A}) = \,?$$

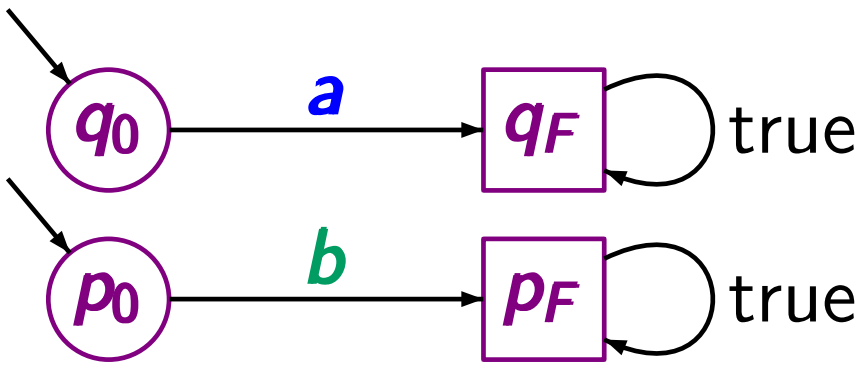
# NBA for LTL formulas






$$\mathcal{L}_\omega(\mathcal{A}) = \mathit{Words}(\bigcirc \neg a)$$

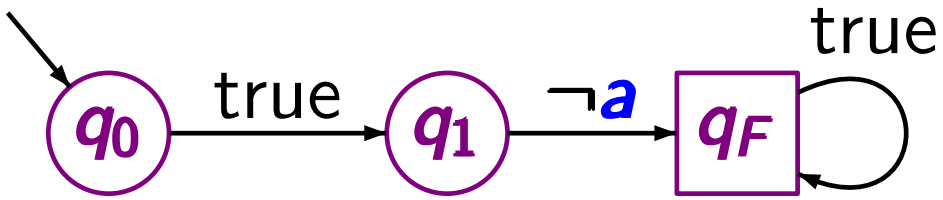
---




$$\mathcal{L}_\omega(\mathcal{A}) = \mathit{Words}(a \vee b)$$

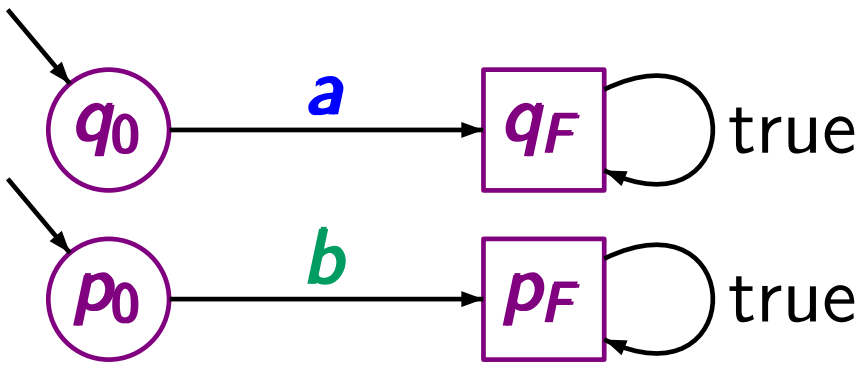
# NBA for LTL formulas





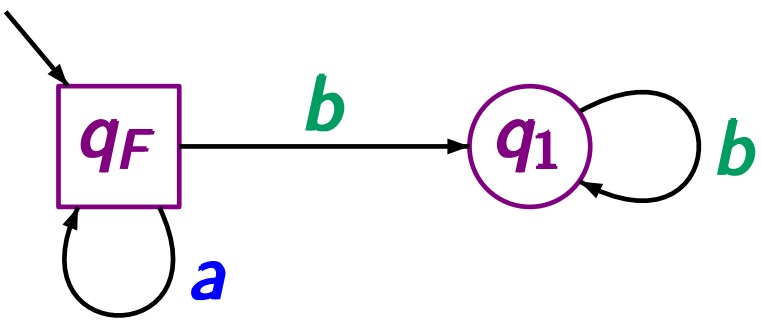

$$\mathcal{L}_\omega(\mathcal{A}) = \mathit{Words}(\bigcirc \neg a)$$

---




$$\mathcal{L}_\omega(\mathcal{A}) = \mathit{Words}(a \vee b)$$

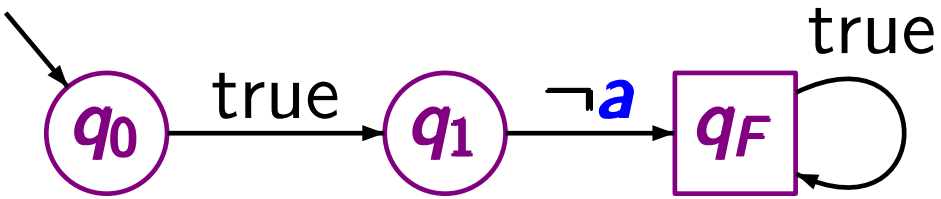
---




$$\mathcal{L}_\omega(\mathcal{A}) = \,?$$

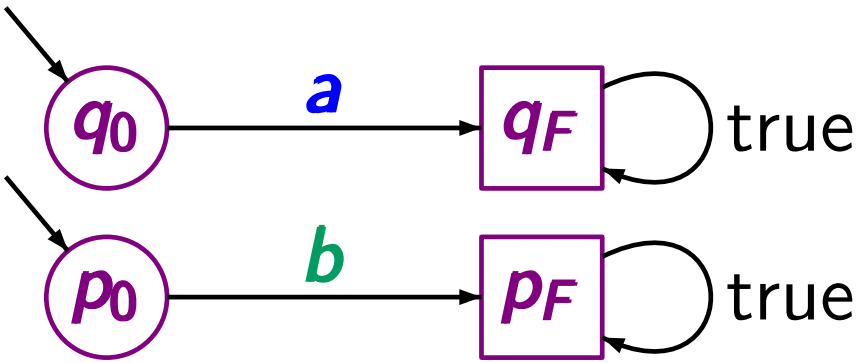
# NBA for LTL formulas





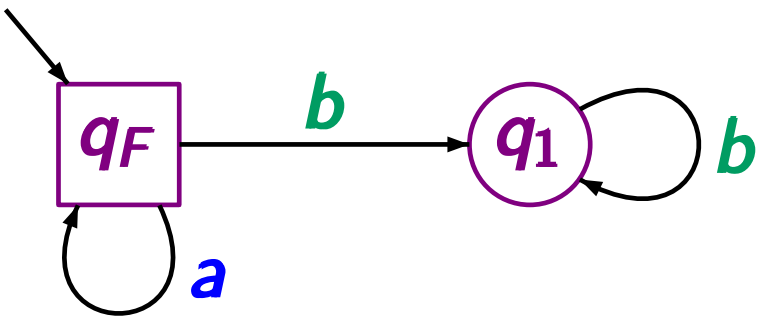

$$\mathcal{L}_\omega(\mathcal{A}) = \mathit{Words}(\bigcirc \neg a)$$

---




$$\mathcal{L}_\omega(\mathcal{A}) = \mathit{Words}(a \vee b)$$

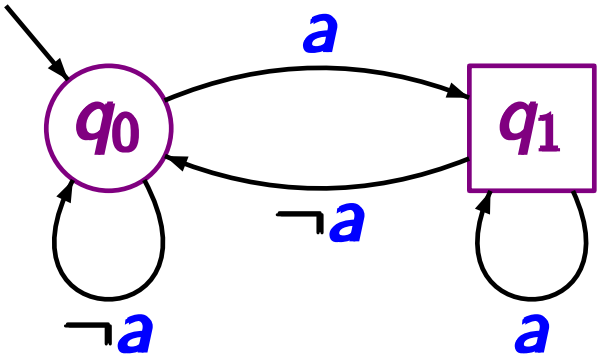
---




$$\mathcal{L}_\omega(\mathcal{A}) = \mathit{Words}(\Box a)$$

# NBA for LTL formulas

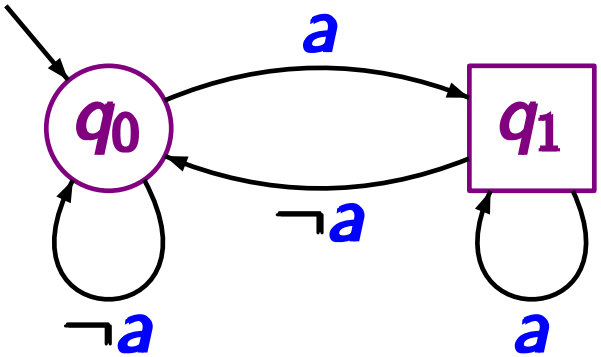





$$\mathcal{L}_\omega(\mathcal{A}) = ?$$

# NBA for LTL formulas

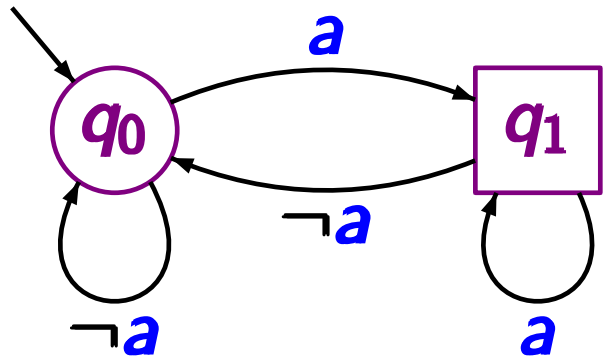





$$\mathcal{L}_\omega(\mathcal{A}) = \mathit{Words}(\Box\Diamond a)$$

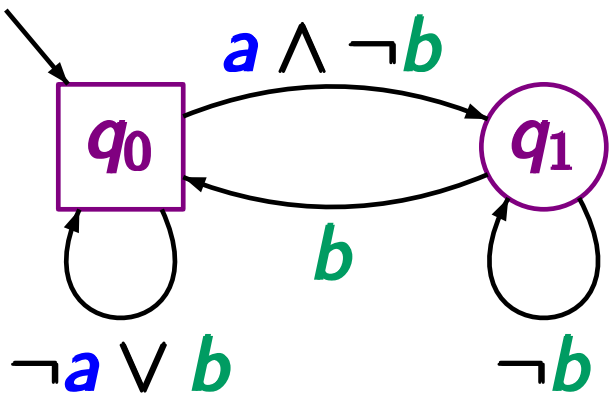
# NBA for LTL formulas






$$\mathcal{L}_\omega(\mathcal{A}) = \mathit{Words}(\Box\Diamond a)$$

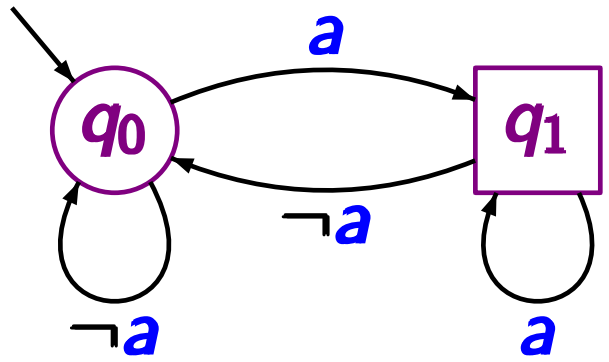
---




$$\mathcal{L}_\omega(\mathcal{A}) = ?$$

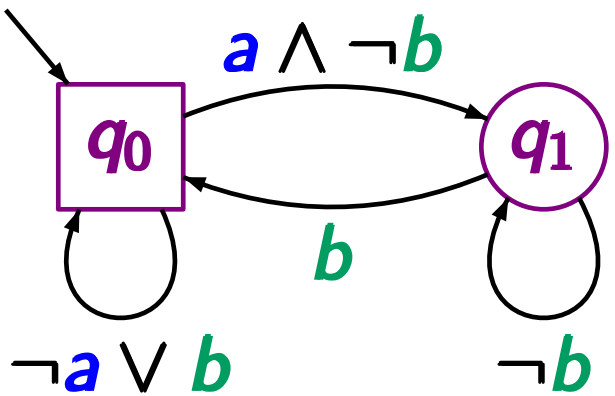
# NBA for LTL formulas






$$\mathcal{L}_\omega(\mathcal{A}) = \mathit{Words}(\Box\Diamond a)$$

---




$$\mathcal{L}_\omega(\mathcal{A}) = ?$$

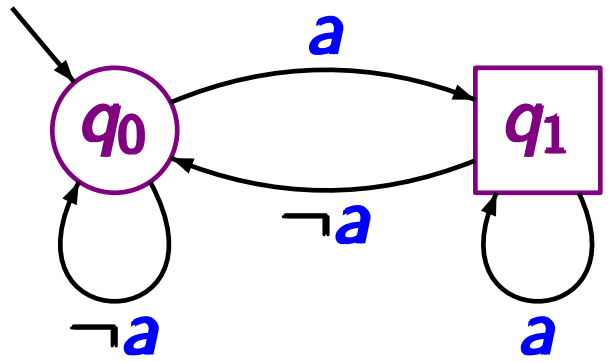
e.g., $\varnothing\,\varnothing\,\varnothing\,\varnothing\ldots = \varnothing^\omega$

$(\{a\}\,\{b\})^\omega$

} are accepted by $\mathcal{A}$

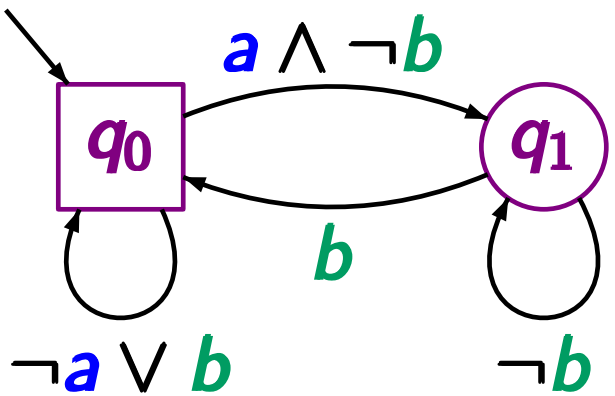
# NBA for LTL formulas






$$\mathcal{L}_\omega(\mathcal{A}) = \mathit{Words}(\Box\Diamond a)$$

---




$$\mathcal{L}_\omega(\mathcal{A}) = \mathit{Words}(\Box(a \rightarrow \Diamond b))$$

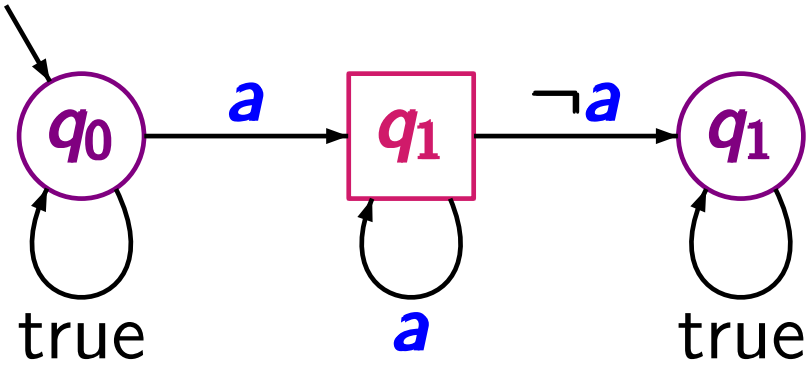
e.g., $\varnothing\,\varnothing\,\varnothing\,\varnothing\ldots = \varnothing^\omega$

$(\{a\}\,\{b\})^\omega$

} are accepted by $\mathcal{A}$

# NBA for LTL formula

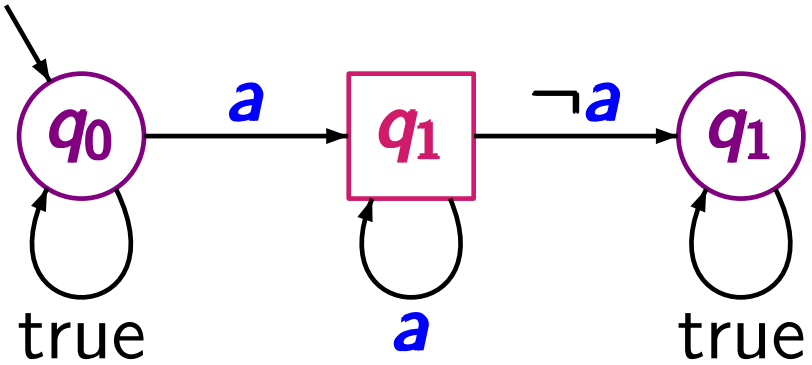





$$\mathcal{L}_\omega(\mathcal{A}) = \,?$$

# NBA for LTL formula

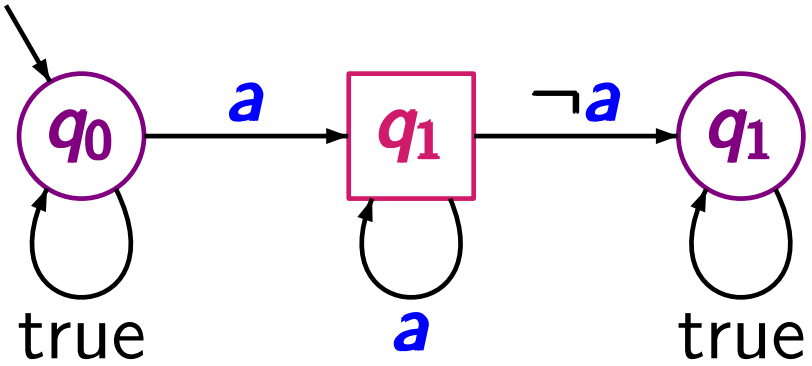





$$\mathcal{L}_\omega(\mathcal{A}) = \mathit{Words}(\Diamond\Box a)$$

# NBA for LTL formula






$$\mathcal{L}_\omega(\mathcal{A}) = \mathit{Words}(\Diamond\Box a)$$

possible runs for $\{a\}^\omega$

| | |
|---|---|
| $q_0\ q_0\ q_0\ q_0\ q_0\ q_0\ \ldots$ | not accepting |
| $q_0\ q_1\ q_1\ q_1\ q_1\ q_1\ \ldots$ | accepting |
| $q_0\ q_0\ q_1\ q_1\ q_1\ q_1\ \ldots$ | accepting |
| $q_0\ q_0\ q_0\ q_1\ q_1\ q_1\ \ldots$ | accepting |
| $\vdots$ | |

# NFA and NBA for safety properties

# NFA and NBA for safety properties



Let $\mathcal{A}$ be an **NFA** for the language of all bad prefixes for a safety property $E$.

# NFA and NBA for safety properties



Let $\mathcal{A}$ be an **NFA** for the language of all bad prefixes for a safety property $E$. Then:

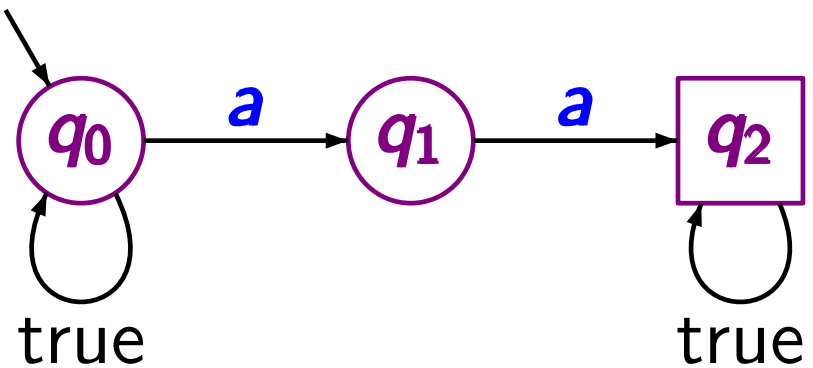
$$\mathcal{L}_\omega(\mathcal{A}) = \overline{E} = (2^{AP})^\omega \setminus E$$

# NFA and NBA for safety properties

Let $\mathcal{A}$ be an **NFA** for the language of all bad prefixes for a safety property $E$. Then:

$$\mathcal{L}_\omega(\mathcal{A}) = \overline{E} = (2^{AP})^\omega \setminus E$$

Example: $E \mathrel{\widehat{=}}$ "never $a$ twice in a row"



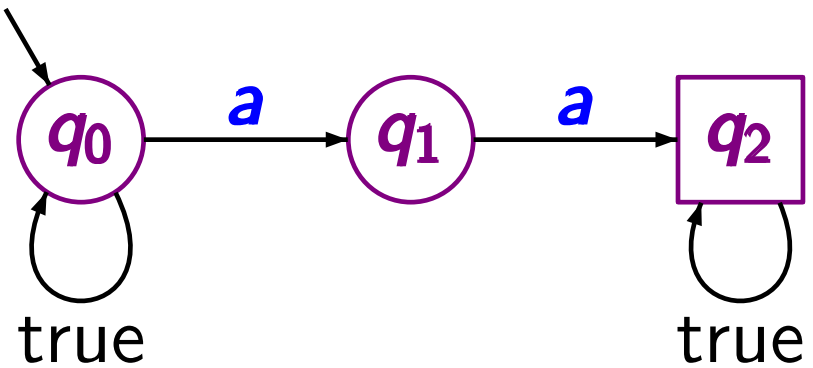
# NFA and NBA for safety properties



Let $\mathcal{A}$ be an **NFA** for the language of all bad prefixes for a safety property $E$. Then:

$$\mathcal{L}_\omega(\mathcal{A}) = \overline{E} = (2^{AP})^\omega \setminus E = \mathit{Words}(\neg\varphi)$$

Example: $E \mathrel{\widehat{=}}$ "never $a$ twice in a row"




$$\varphi = \Box(a \rightarrow \bigcirc\neg a)$$

# NFA and NBA for safety properties

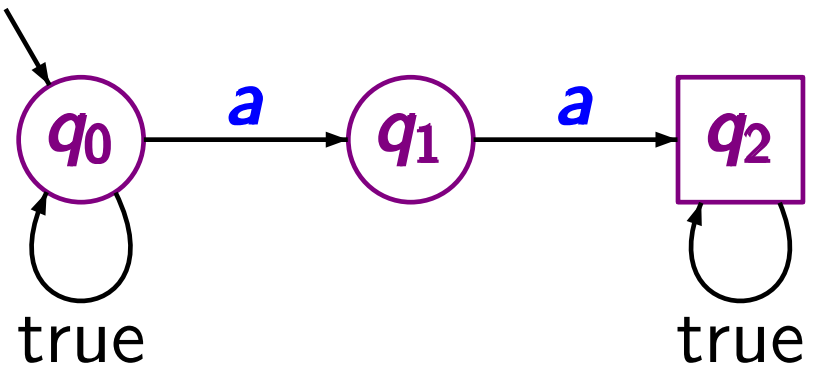


Let $\mathcal{A}$ be an **NFA** for the language of all bad prefixes for a safety property $E$. Then:

$$\mathcal{L}_\omega(\mathcal{A}) = \overline{E} = (2^{AP})^\omega \setminus E = \mathit{Words}(\neg\varphi)$$

**wrong**, if $\mathcal{L}(\mathcal{A})$ = language of minimal bad prefixes

Example: $E \mathrel{\widehat{=}}$ "never $a$ twice in a row"




$$\varphi = \Box(a \rightarrow \bigcirc\neg a)$$

# NFA and NBA for safety properties

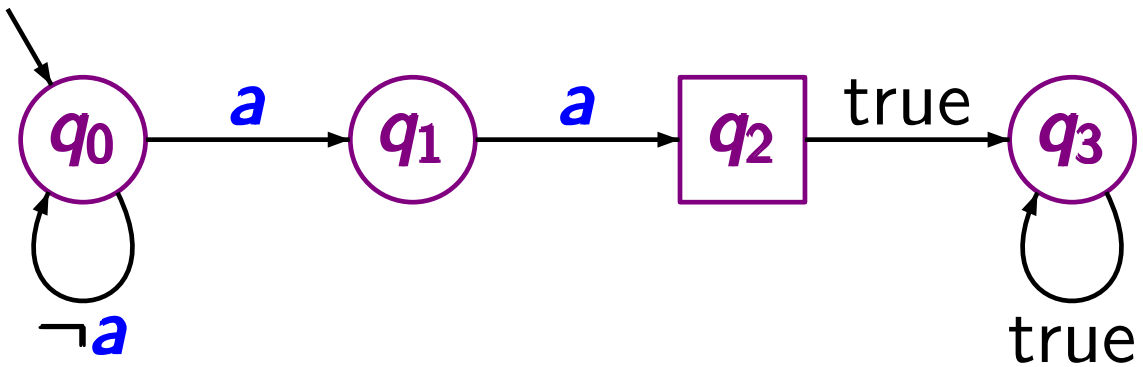


Let $\mathcal{A}$ be an **NFA** for the language of all bad prefixes for a safety property $E$. Then:

$$\mathcal{L}_\omega(\mathcal{A}) = \overline{E} = (2^{AP})^\omega \setminus E = \mathit{Words}(\neg\varphi)$$

**wrong**, if $\mathcal{L}(\mathcal{A})$ = language of minimal bad prefixes

Example: $E \mathrel{\widehat{=}}$ "never $a$ twice in a row"




$$\mathcal{L}_\omega(\mathcal{A}) = \varnothing$$

# NFA and NBA for safety properties

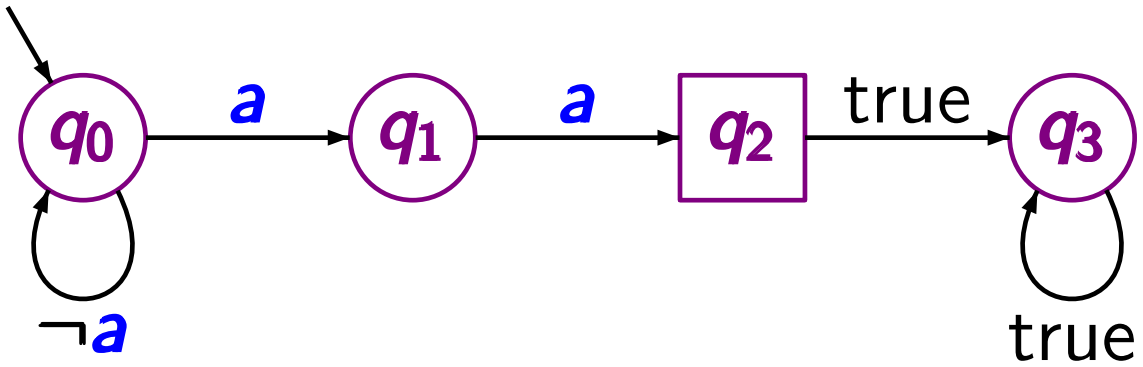


Let $\mathcal{A}$ be an **NFA** for the language of all bad prefixes for a safety property $E$. Then:

$$\mathcal{L}_\omega(\mathcal{A}) = \overline{E} = (2^{AP})^\omega \setminus E = \mathit{Words}(\neg\varphi)$$

**wrong**, if $\mathcal{L}(\mathcal{A})$ = language of minimal bad prefixes
even if $\mathcal{A}$ is a non-blocking DFA

Example: $E \mathrel{\widehat{=}}$ "never $a$ twice in a row"




$$\mathcal{L}_\omega(\mathcal{A}) = \varnothing$$

# NFA and NBA for safety properties

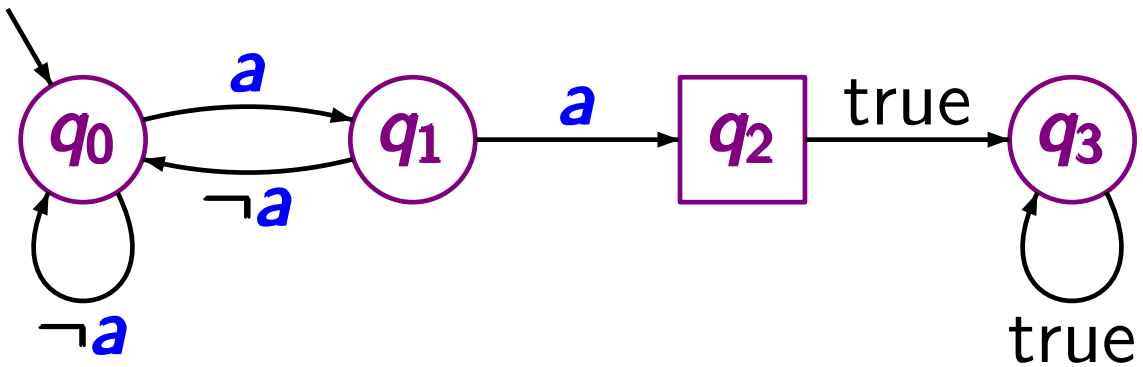


Let $\mathcal{A}$ be an **NFA** for the language of all bad prefixes for a safety property $E$. Then:

$$\mathcal{L}_\omega(\mathcal{A}) = \overline{E} = (2^{AP})^\omega \setminus E = \mathit{Words}(\neg\varphi)$$

**wrong**, if $\mathcal{L}(\mathcal{A})$ = language of minimal bad prefixes
even if $\mathcal{A}$ is a non-blocking DFA

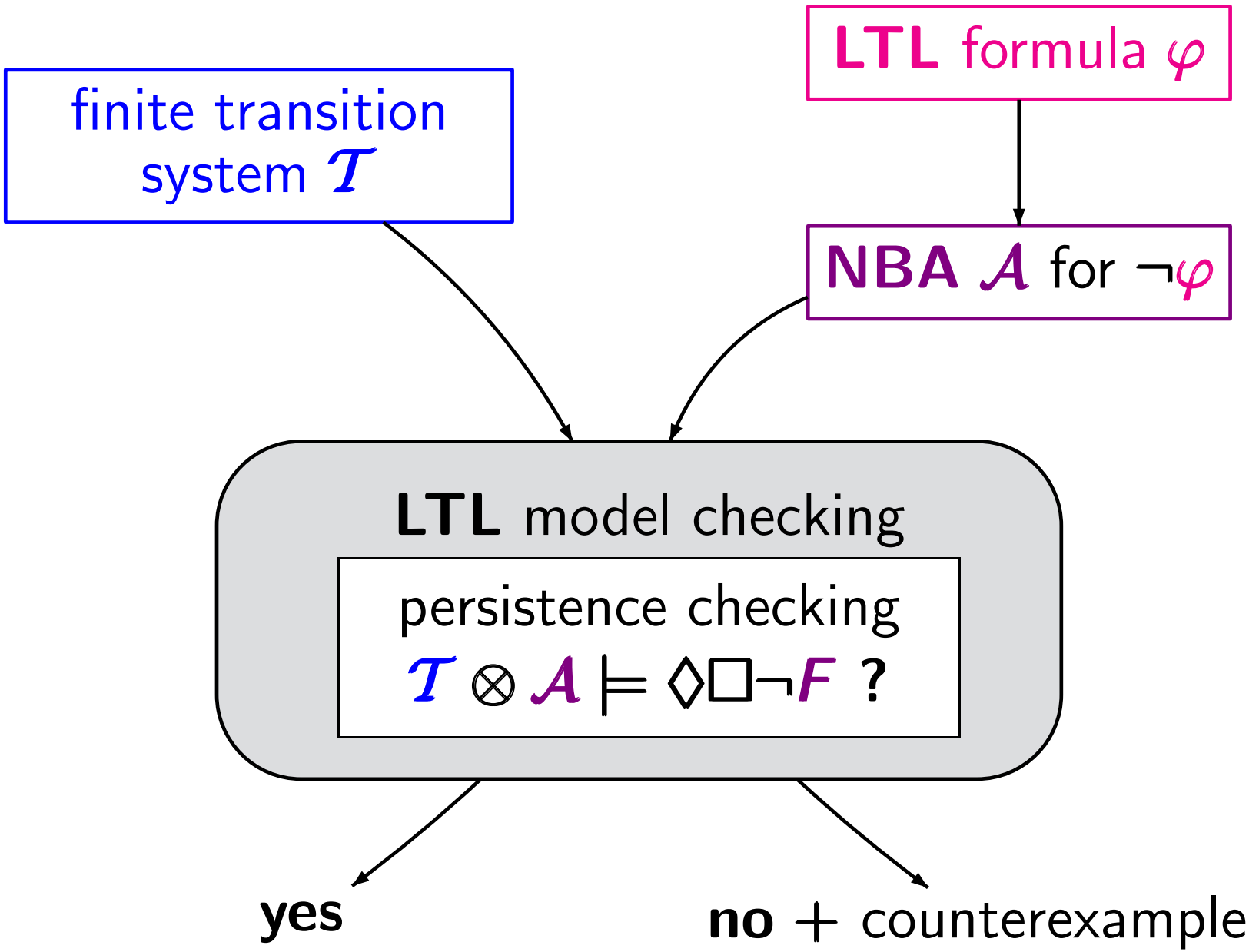
Example: $E \mathrel{\widehat{=}}$ "never $a$ twice in a row"




$$\mathcal{L}_\omega(\mathcal{A}) = \varnothing$$

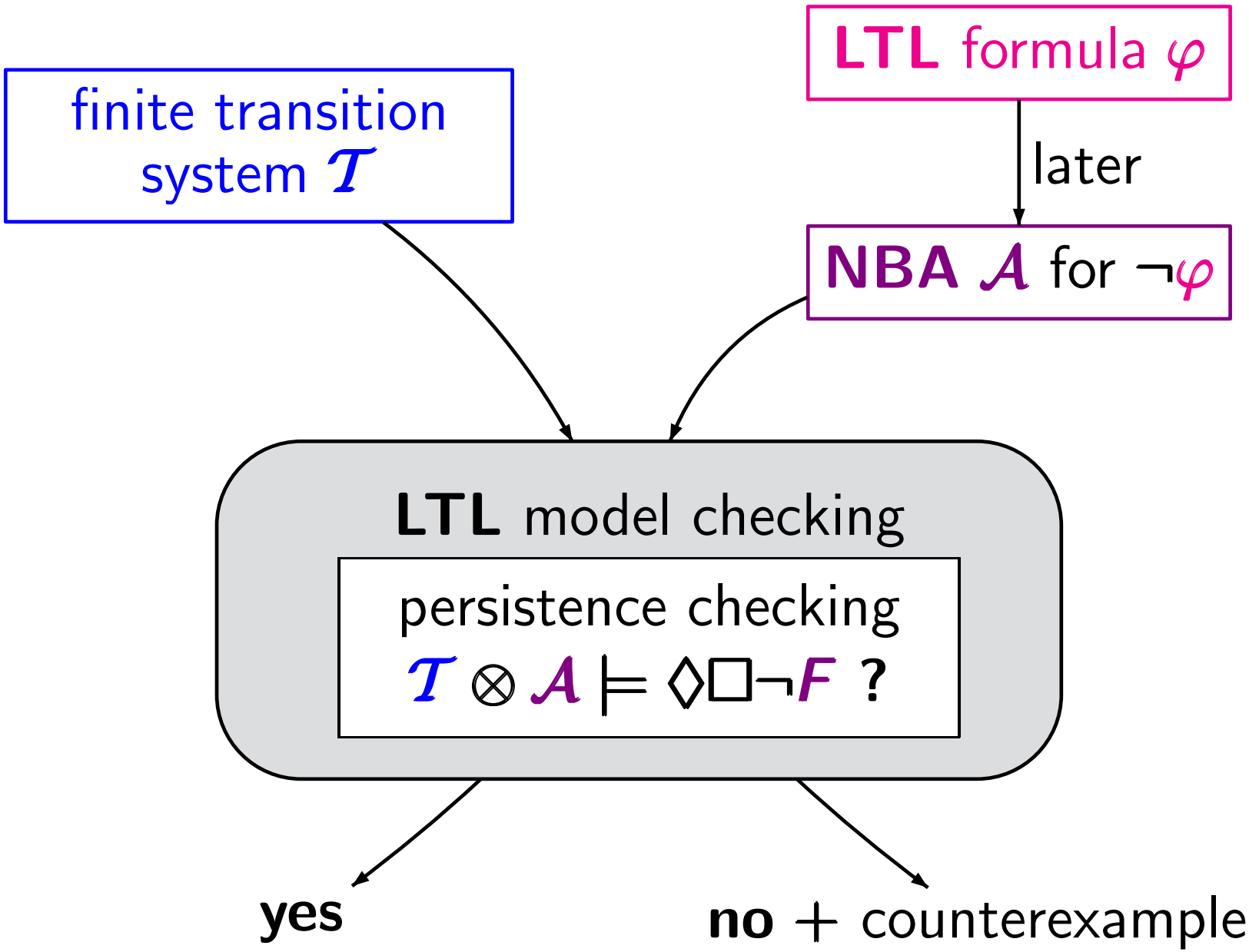
# LTL model checking





# LTL model checking





# Recall: product transition system



$\mathcal{T} = (S, Act, \rightarrow, S_0, AP, L)$ TS without terminal states

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

non-blocking, $Q_0 \cap F = \varnothing$

# Recall: product transition system



$\mathcal{T} = (S, Act, \rightarrow, S_0, AP, L)$ TS without terminal states

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

non-blocking, $Q_0 \cap F = \varnothing$

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

# Recall: product transition system



$\mathcal{T} = (S, Act, \rightarrow, S_0, AP, L)$ TS without terminal states

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

non-blocking, $Q_0 \cap F = \varnothing$

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

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

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

# Recall: product transition system



$\mathcal{T} = (S, Act, \rightarrow, S_0, AP, L)$ TS without terminal states

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

non-blocking, $Q_0 \cap F = \varnothing$

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

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

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

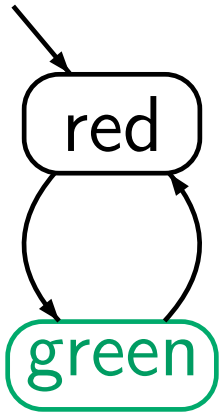
transition relation:

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

# Example: LTL model checking



TS $\mathcal{T}$

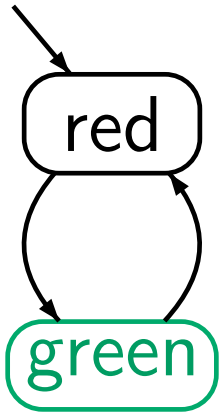



**LTL** formula $\varphi = \Box\Diamond$green

# Example: LTL model checking



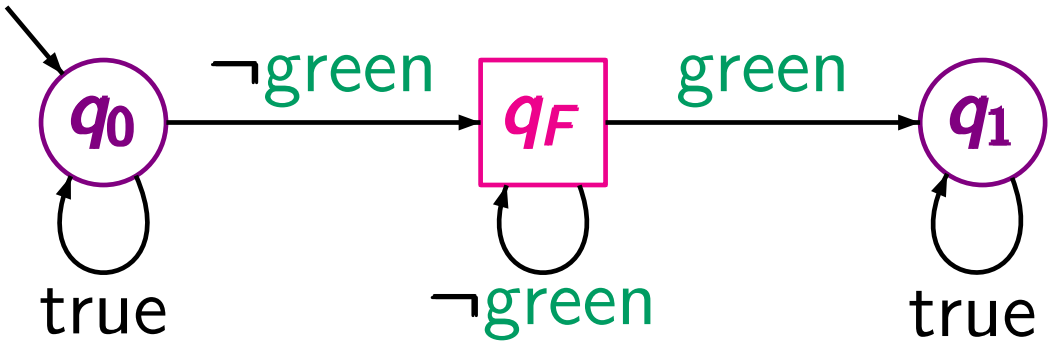
TS $\mathcal{T}$




**LTL** formula $\varphi = \Box\Diamond$green

**NBA** $\mathcal{A}$ for the complement

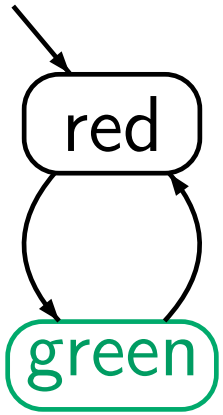
$\neg\varphi \equiv \Diamond\Box\neg$green



# Example: LTL model checking



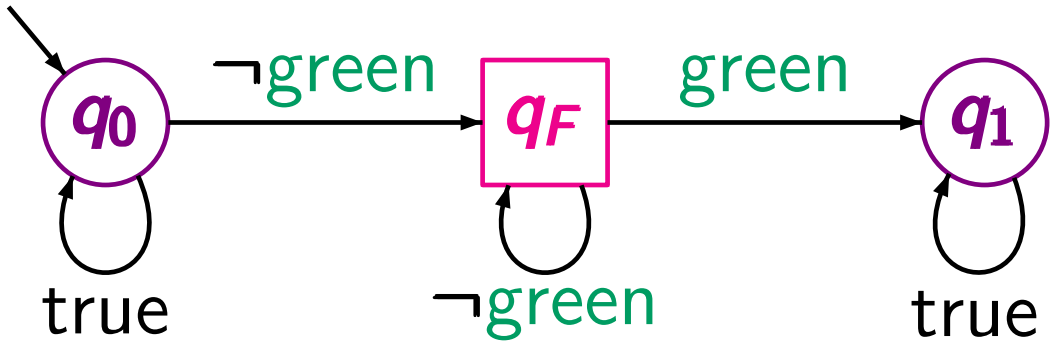
TS $\mathcal{T}$

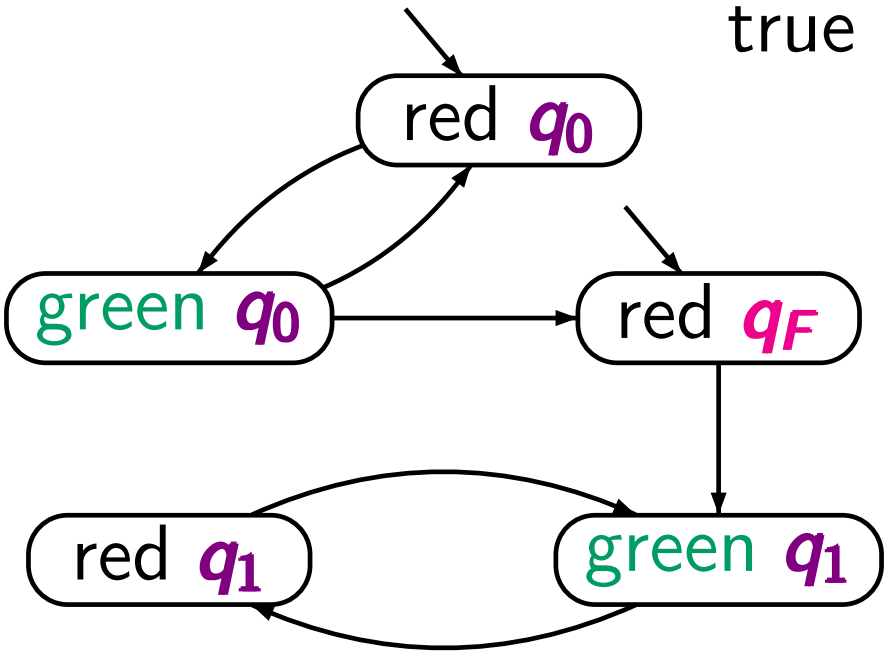



**LTL** formula $\varphi = \Box\Diamond$green

**NBA** $\mathcal{A}$ for the complement

$\neg\varphi \equiv \Diamond\Box\neg$green



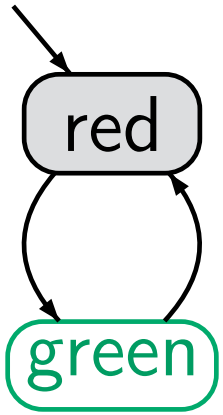




reachable fragment of the product TS $\mathcal{T} \otimes \mathcal{A}$

# Example: LTL model checking



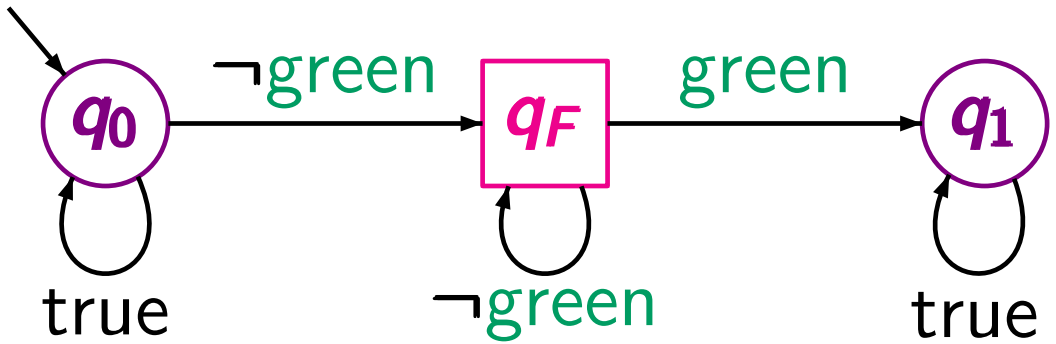
TS $\mathcal{T}$

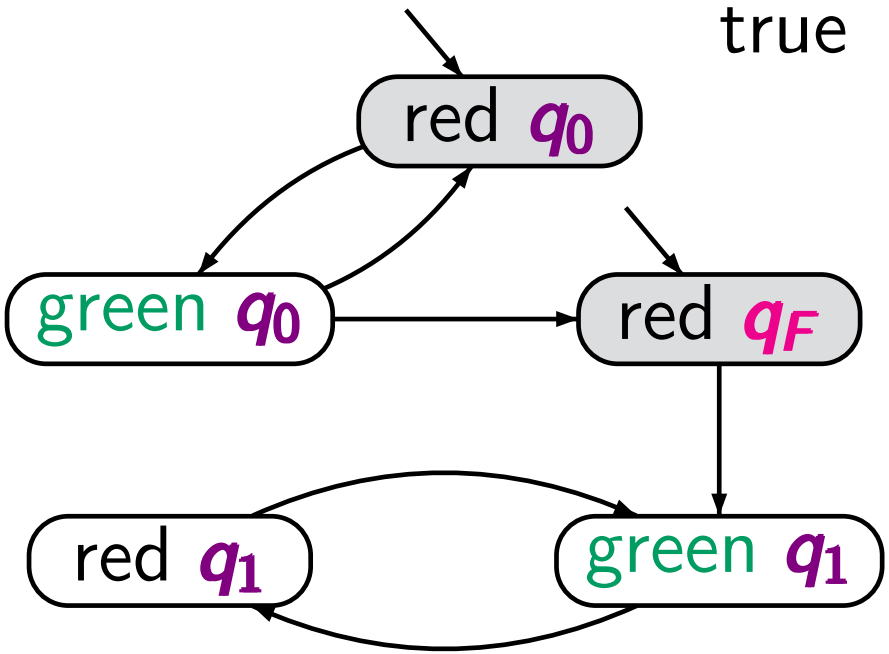



**LTL** formula $\varphi = \Box\Diamond\text{green}$

**NBA** $\mathcal{A}$ for the complement

$\neg\varphi \equiv \Diamond\Box\neg\text{green}$




initial states:

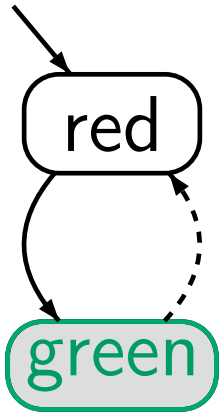
$\langle \text{red}, q \rangle$ where

$$q \in \delta(q_0, L(\text{red})) = \delta(q_0, \varnothing) = \{q_0, q_F\}$$

# Example: LTL model checking



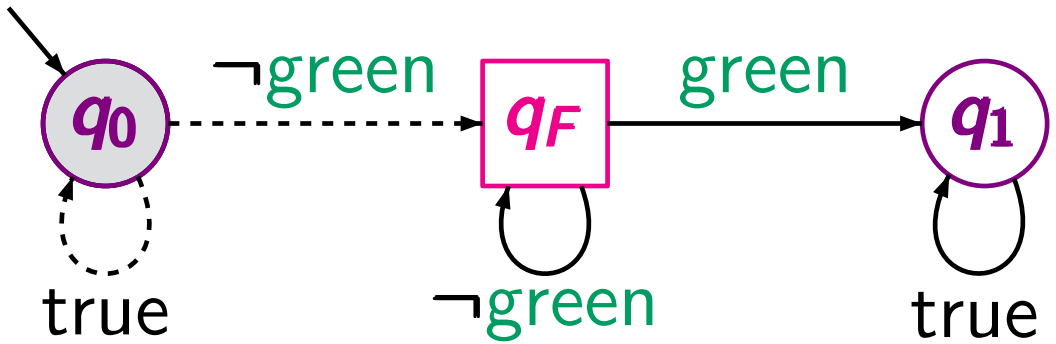
TS $\mathcal{T}$

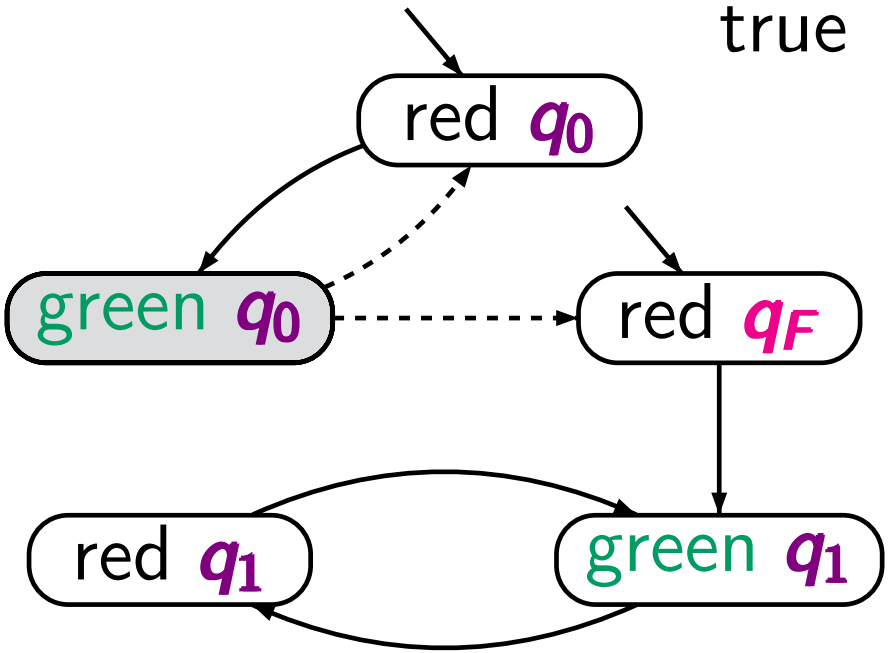



**LTL** formula $\varphi = \Box\Diamond\text{green}$

**NBA** $\mathcal{A}$ for the complement

$\neg\varphi \equiv \Diamond\Box\neg\text{green}$




transition

$\langle \text{green}, q_0 \rangle \rightarrow \langle \text{red}, q \rangle$

$q \in \delta(q_0, L(\text{red}))$

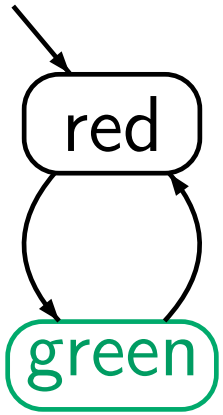
$= \delta(q_0, \varnothing)$

$= \{q_0, q_F\}$

# Example: LTL model checking



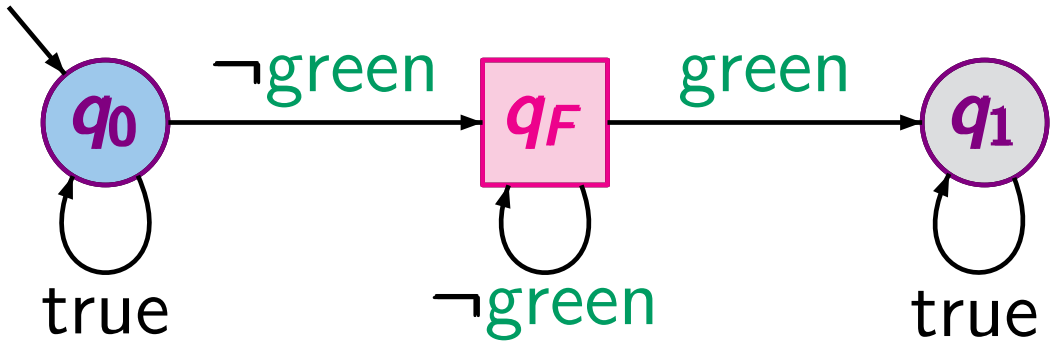
TS $\mathcal{T}$

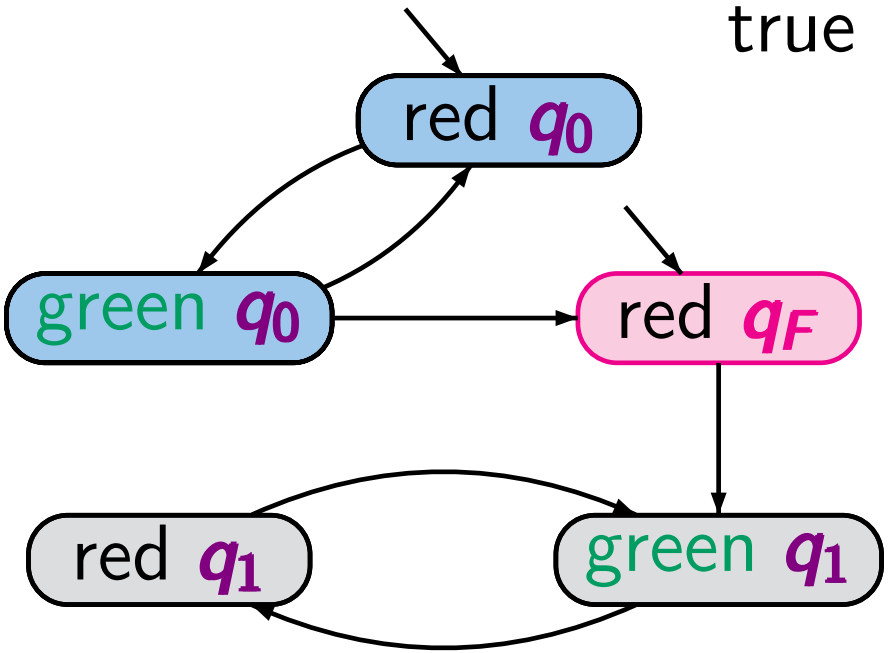



**LTL** formula $\varphi = \Box\Diamond\text{green}$

**NBA** $\mathcal{A}$ for the complement

$\neg\varphi \equiv \Diamond\Box\neg\text{green}$




atomic propositions

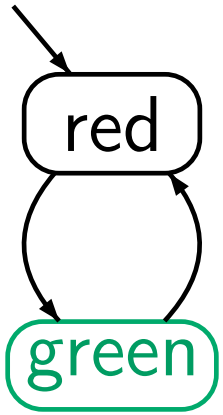
$AP' = \{q_0, q_F, q_1\}$

obvious labeling function

# Example: LTL model checking



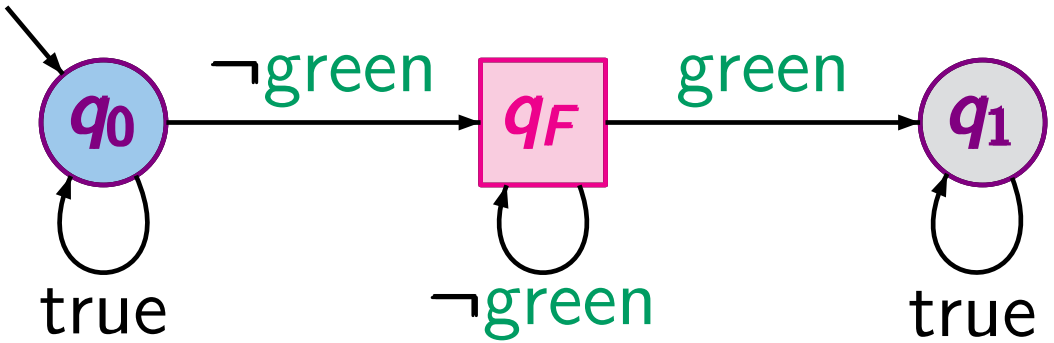
TS $\mathcal{T}$

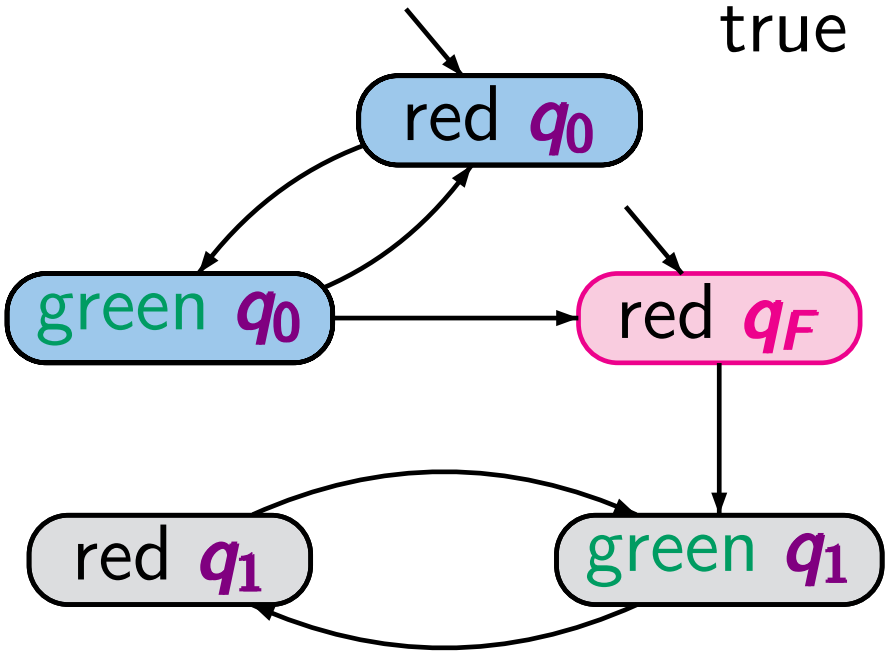



**LTL** formula $\varphi = \Box\Diamond$green

**NBA** $\mathcal{A}$ for the complement

$\neg\varphi \equiv \Diamond\Box\neg$green



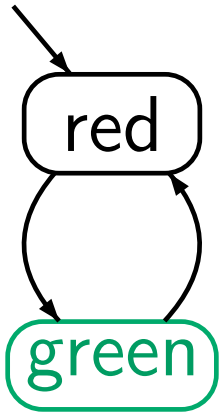




$\mathcal{T} \otimes \mathcal{A} \models \Diamond\Box\neg F$

# Example: LTL model checking



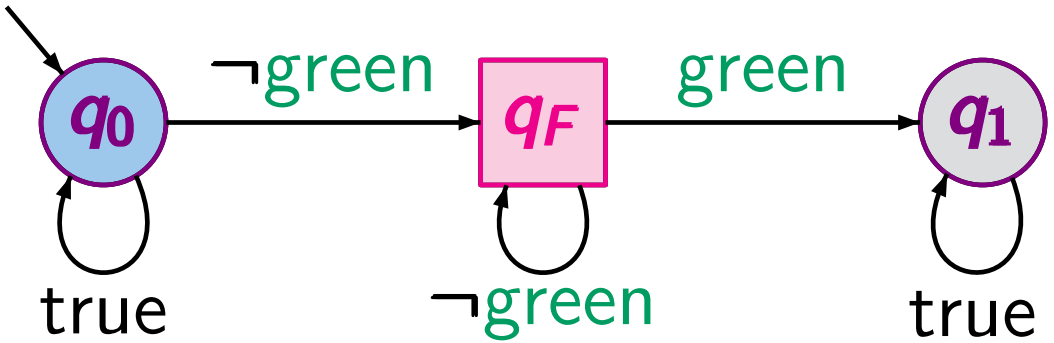
TS $\mathcal{T}$

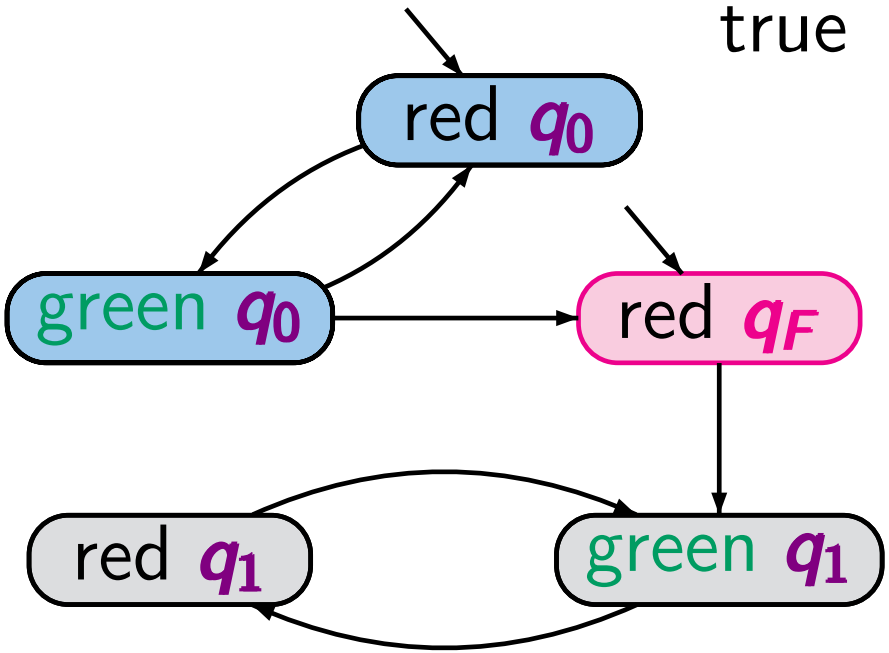



**LTL** formula $\varphi = \Box\Diamond$green

**NBA** $\mathcal{A}$ for the complement

$\neg\varphi \equiv \Diamond\Box\neg$green



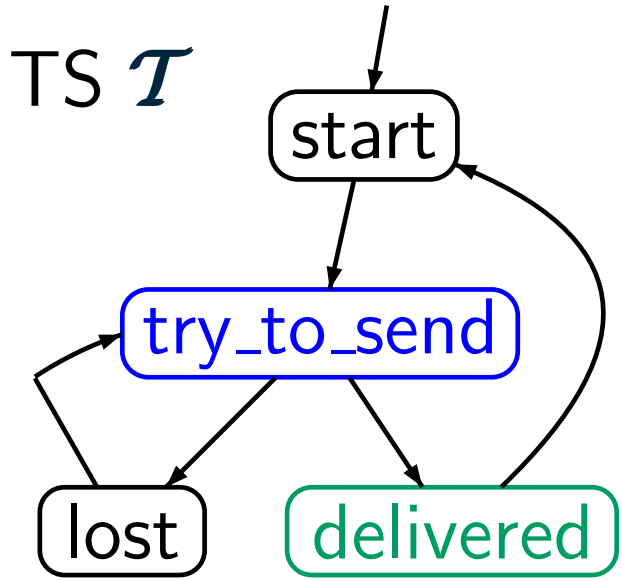




$\mathcal{T} \otimes \mathcal{A} \models \Diamond\Box\neg F$

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

# Example: LTL model checking



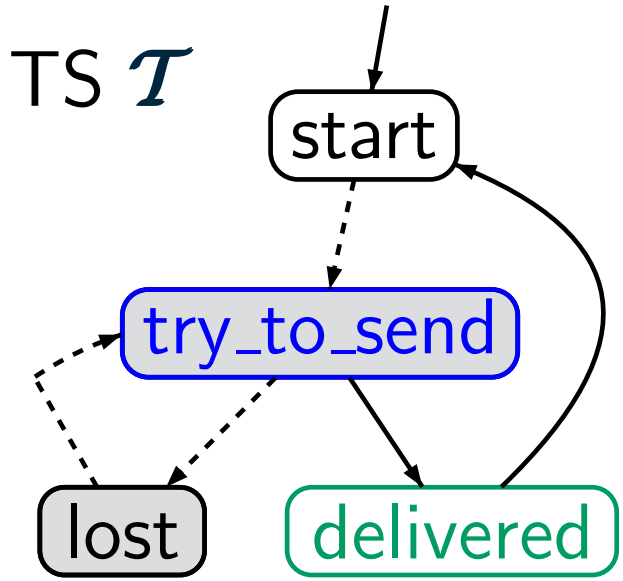



**LTL** formula $\varphi = \Box(\mathit{try} \rightarrow \Diamond \mathit{del})$

"each (repeatedly) sent message will eventually be delivered"

# Example: LTL model checking





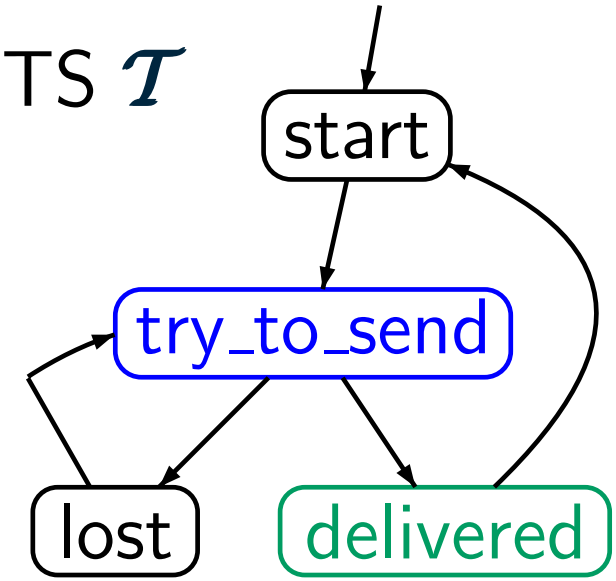

**LTL** formula $\varphi = \Box(\textit{try} \rightarrow \Diamond \textit{del})$

"each (repeatedly) sent message will eventually be delivered"

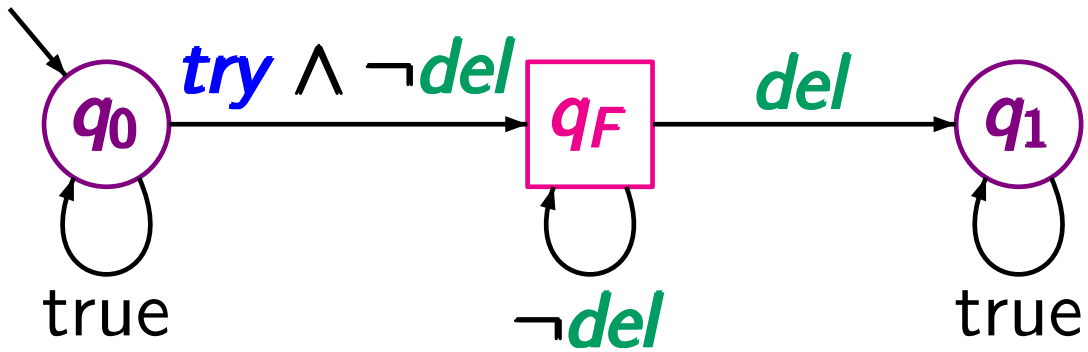
$\mathcal{T} \not\models \varphi$

# Example: LTL model checking






NBA $\mathcal{A}$ for $\neg\varphi \equiv \Diamond(\mathit{try} \wedge \Box\neg\mathit{del})$



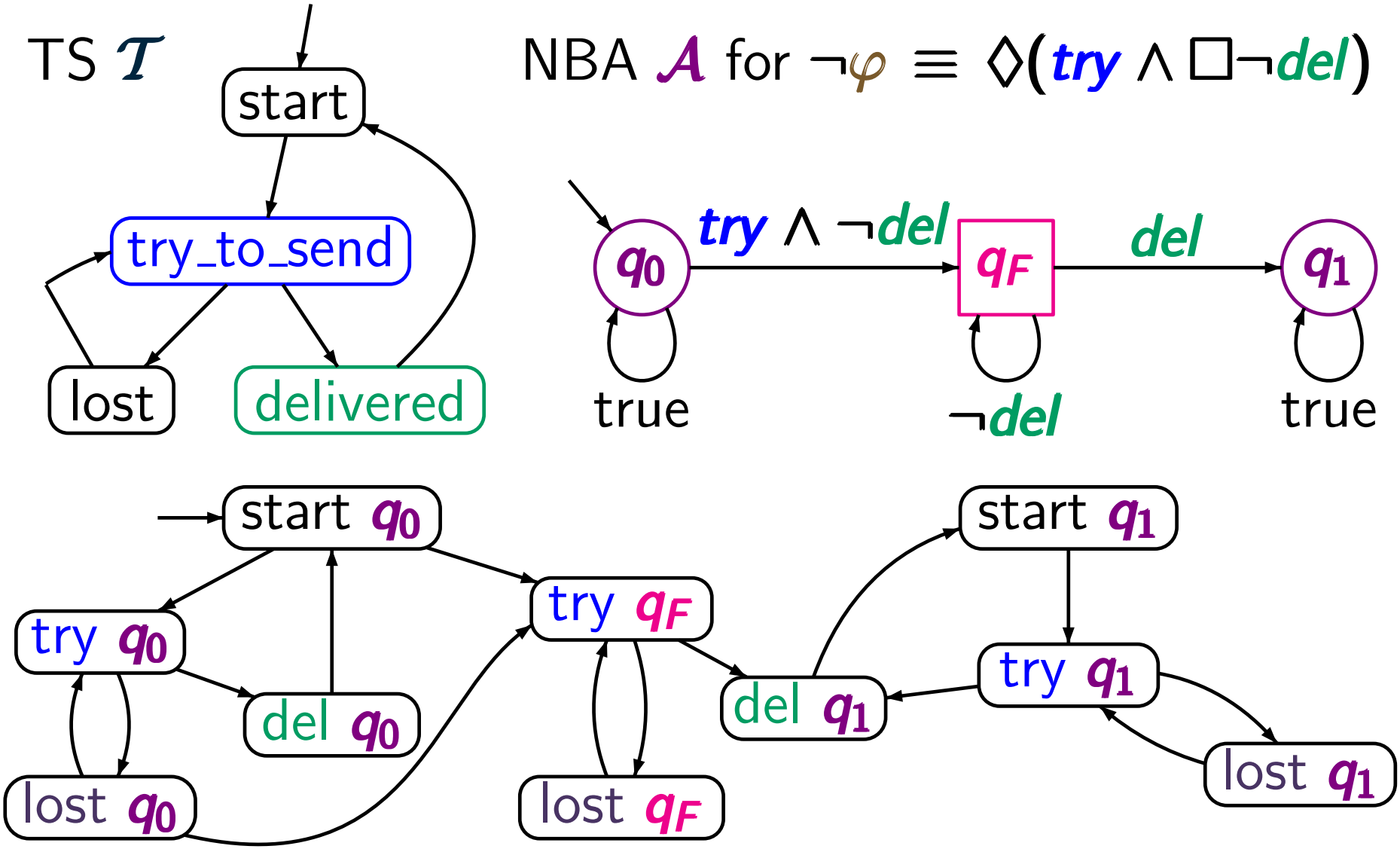

**LTL** formula $\varphi = \Box(\mathit{try} \rightarrow \Diamond\mathit{del})$

"each (repeatedly) sent message will eventually be delivered"

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

# Example: LTL model checking

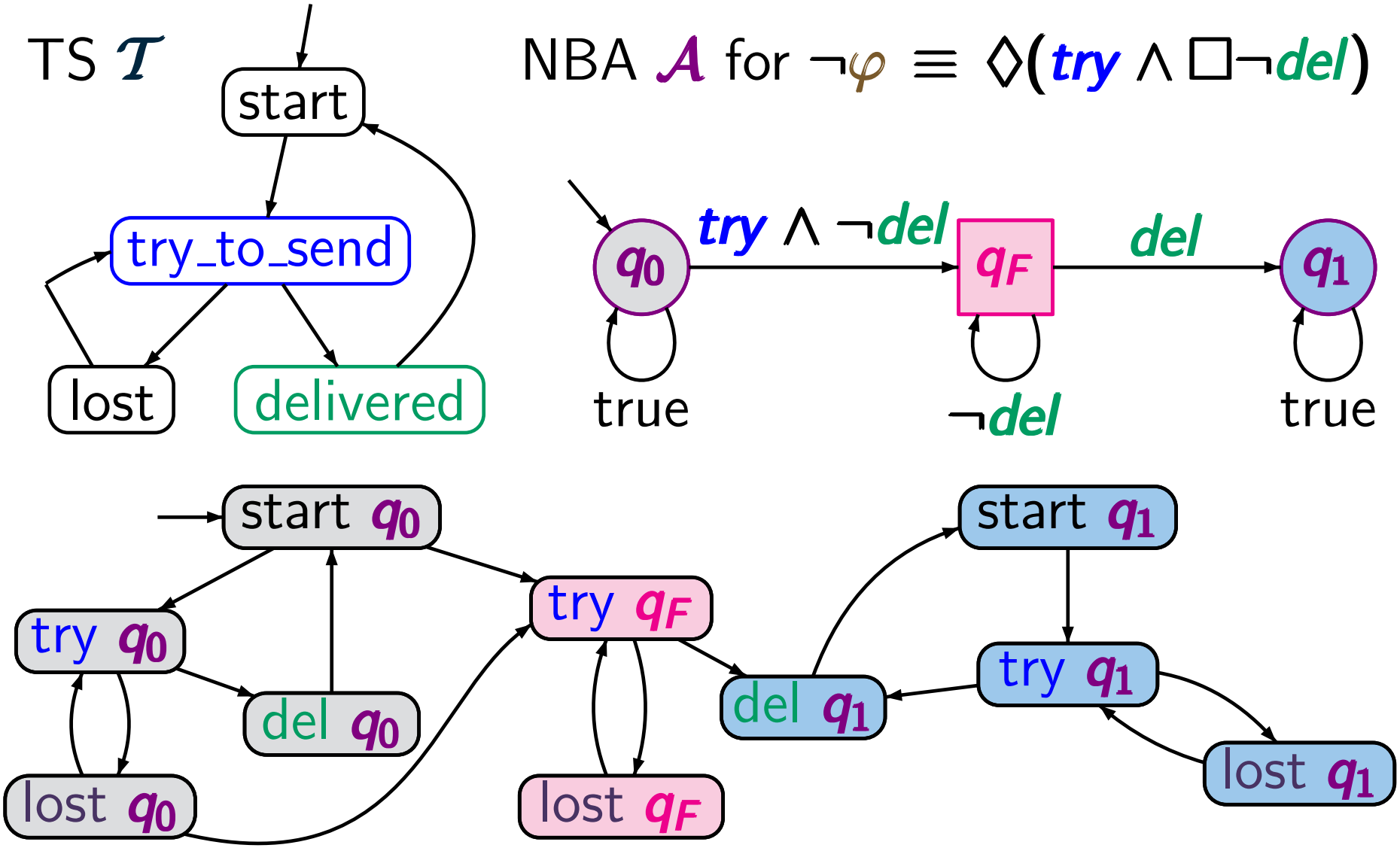





reachable fragment of the product-TS

# Example: LTL model checking

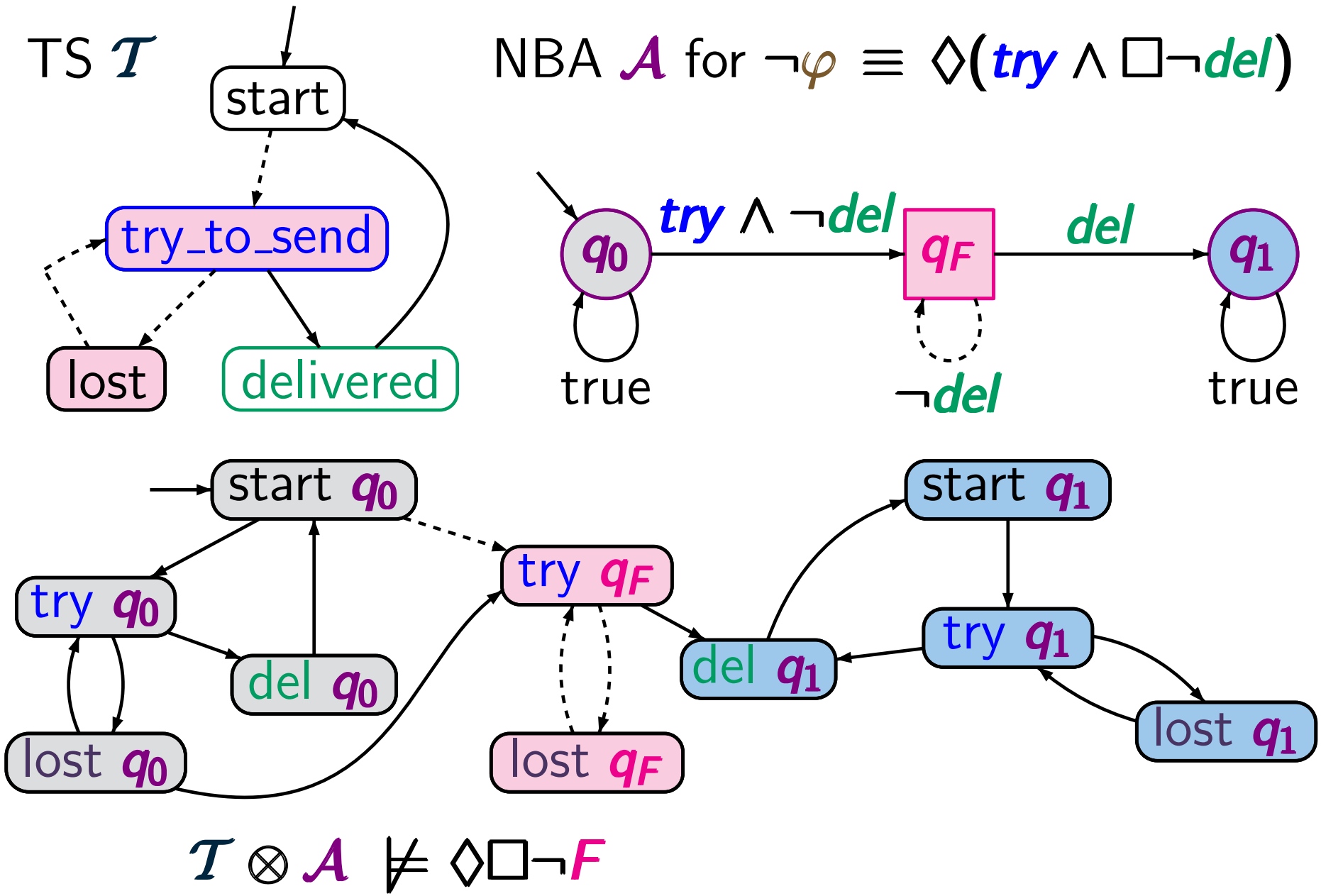





set of atomic propositions $AP' = \{q_0, q_1, q_F\}$

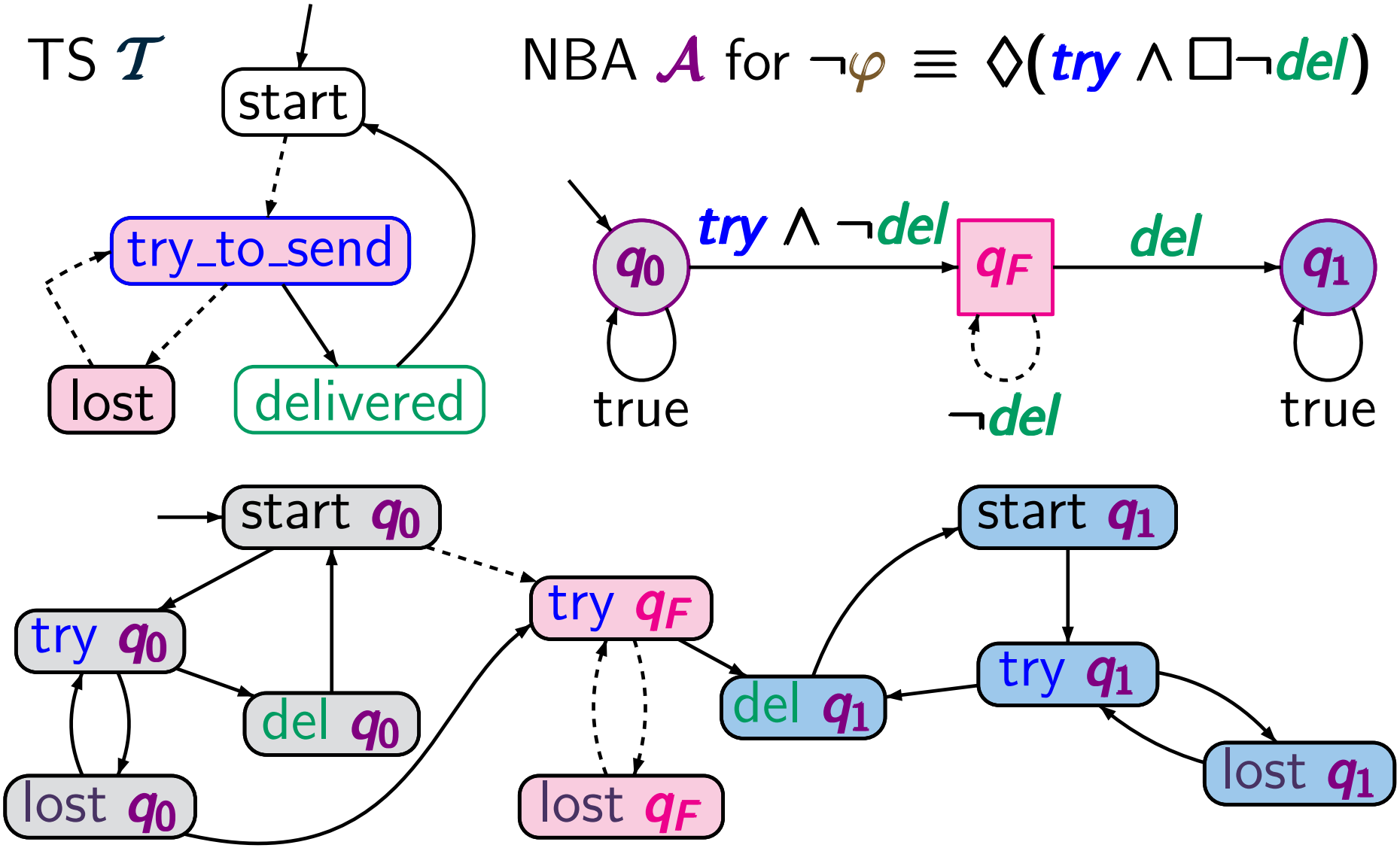
# Example: LTL model checking





# Example: LTL model checking






$\mathcal{T} \otimes \mathcal{A} \not\models \Diamond\Box\neg F$ hence: $\mathcal{T} \not\models \varphi$

# Complexity of LTL model checking



main steps of automata-based LTL model checking:

construction of an NBA $\mathcal{A}$ for $\neg\varphi$

persistence checking in the product $\mathcal{T} \otimes \mathcal{A}$

# Complexity of LTL model checking



main steps of automata-based LTL model checking:

| construction of an NBA $\mathcal{A}$ for $\neg\varphi$ | $\longleftarrow \mathcal{O}(\exp(\lvert\varphi\rvert))$ |
|---|---|

| persistence checking in the product $\mathcal{T} \otimes \mathcal{A}$ |
|---|

# Complexity of LTL model checking



main steps of automata-based LTL model checking:

| | |
|---|---|
| construction of an NBA $\mathcal{A}$ for $\neg\varphi$ | $\longleftarrow \mathcal{O}(\exp(\lvert\varphi\rvert))$ |
| persistence checking in the product $\mathcal{T} \otimes \mathcal{A}$ | $\longleftarrow \mathcal{O}(\mathit{size}(\mathcal{T}) \cdot \mathit{size}(\mathcal{A}))$ |

# Complexity of LTL model checking



main steps of automata-based LTL model checking:

| | |
|---|---|
| construction of an NBA $\mathcal{A}$ for $\neg\varphi$ | $\longleftarrow \mathcal{O}(\exp(\lvert\varphi\rvert))$ |
| persistence checking in the product $\mathcal{T} \otimes \mathcal{A}$ | $\longleftarrow \mathcal{O}(\mathit{size}(\mathcal{T}) \cdot \mathit{size}(\mathcal{A}))$ |

*complexity:* $\mathcal{O}(\mathit{size}(\mathcal{T}) \cdot \exp(\lvert\varphi\rvert))$

# Complexity of LTL model checking



main steps of automata-based LTL model checking:

| | |
|---|---|
| construction of an NBA $\mathcal{A}$ for $\neg\varphi$ | $\longleftarrow \mathcal{O}(\exp(\lvert\varphi\rvert))$ |
| persistence checking in the product $\mathcal{T} \otimes \mathcal{A}$ | $\longleftarrow \mathcal{O}(\mathit{size}(\mathcal{T}) \cdot \mathit{size}(\mathcal{A}))$ |

*complexity:* $\mathcal{O}(\mathit{size}(\mathcal{T}) \cdot \exp(\lvert\varphi\rvert))$

The **LTL** model checking problem is **PSPACE**-complete

# Complexity of LTL model checking



main steps of automata-based LTL model checking:

| construction of an NBA $\mathcal{A}$ for $\neg\varphi$ | $\longleftarrow \mathcal{O}(\exp(\lvert\varphi\rvert))$ |
|---|---|

| persistence checking in the product $\mathcal{T} \otimes \mathcal{A}$ |
|---|

# Complexity of LTL model checking



main steps of automata-based LTL model checking:

| | |
|---|---|
| construction of an NBA $\mathcal{A}$ for $\neg\varphi$ | $\longleftarrow \mathcal{O}(\exp(\lvert\varphi\rvert))$ |
| persistence checking in the product $\mathcal{T} \otimes \mathcal{A}$ | $\longleftarrow \mathcal{O}(\mathit{size}(\mathcal{T}) \cdot \mathit{size}(\mathcal{A}))$ |

# Complexity of LTL model checking



main steps of automata-based LTL model checking:

| | |
|---|---|
| construction of an NBA $\mathcal{A}$ for $\neg\varphi$ | $\longleftarrow \mathcal{O}(\exp(\lvert\varphi\rvert))$ |
| persistence checking in the product $\mathcal{T} \otimes \mathcal{A}$ | $\longleftarrow \mathcal{O}(\mathit{size}(\mathcal{T}) \cdot \mathit{size}(\mathcal{A}))$ |

*complexity:* $\mathcal{O}(\mathit{size}(\mathcal{T}) \cdot \exp(\lvert\varphi\rvert))$

# Complexity of LTL model checking



main steps of automata-based LTL model checking:

| | |
|---|---|
| construction of an NBA $\mathcal{A}$ for $\neg\varphi$ | $\longleftarrow \mathcal{O}(\exp(\lvert\varphi\rvert))$ |
| persistence checking in the product $\mathcal{T} \otimes \mathcal{A}$ | $\longleftarrow \mathcal{O}(\mathit{size}(\mathcal{T}) \cdot \mathit{size}(\mathcal{A}))$ |

*complexity:* $\mathcal{O}(\mathit{size}(\mathcal{T}) \cdot \exp(\lvert\varphi\rvert))$

The **LTL** model checking problem is **PSPACE**-complete

# Complexity of LTL model checking



**LTL** model checking problem

| | |
|---|---|
| *given*: | finite transition system $\mathcal{T}$ |
| | LTL-formula $\varphi$ |
| *question*: | does $\mathcal{T} \models \varphi$ hold **?** |

# Complexity of LTL model checking



**LTL** model checking problem

*given*: finite transition system $\mathcal{T}$
LTL-formula $\varphi$

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

we show

- just for fun: **coNP**-hardness
- **PSPACE**-completeness

# Recall: complexity classes

# Complexity classes *P*, *NP*

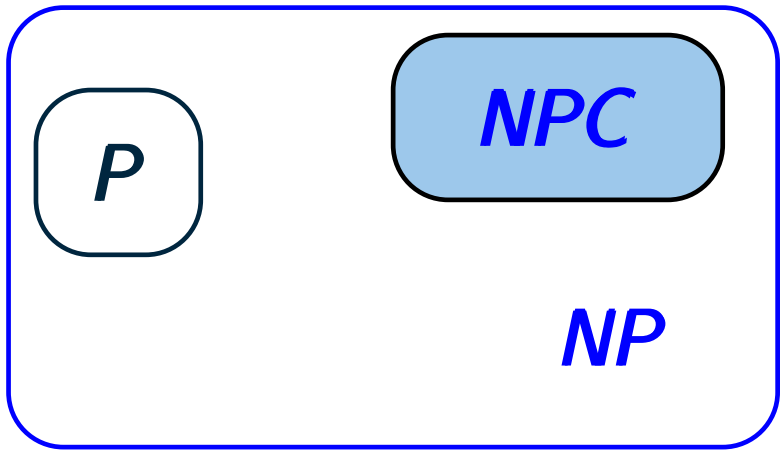



***P*** = class of decision problem solvable in deterministic polynomial time

***NP*** = class of decision problem solvable in nondeterministic polynomial time

# Complexity classes *P*, *NP*

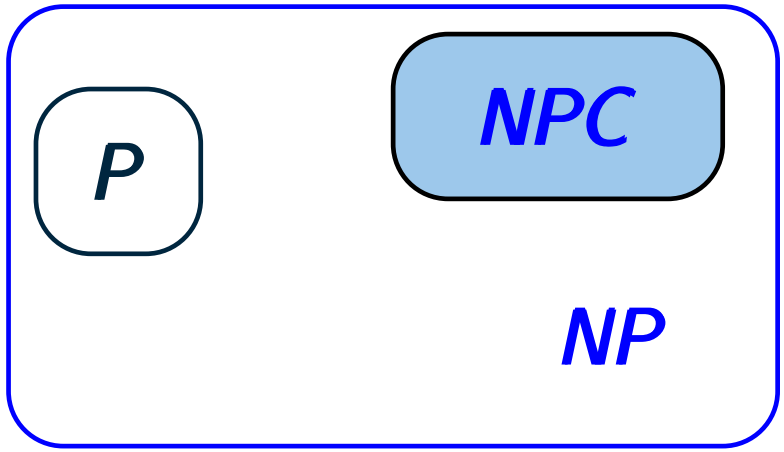





***NPC*** = class of ***NP***-complete problems

# Complexity classes *P*, *NP*





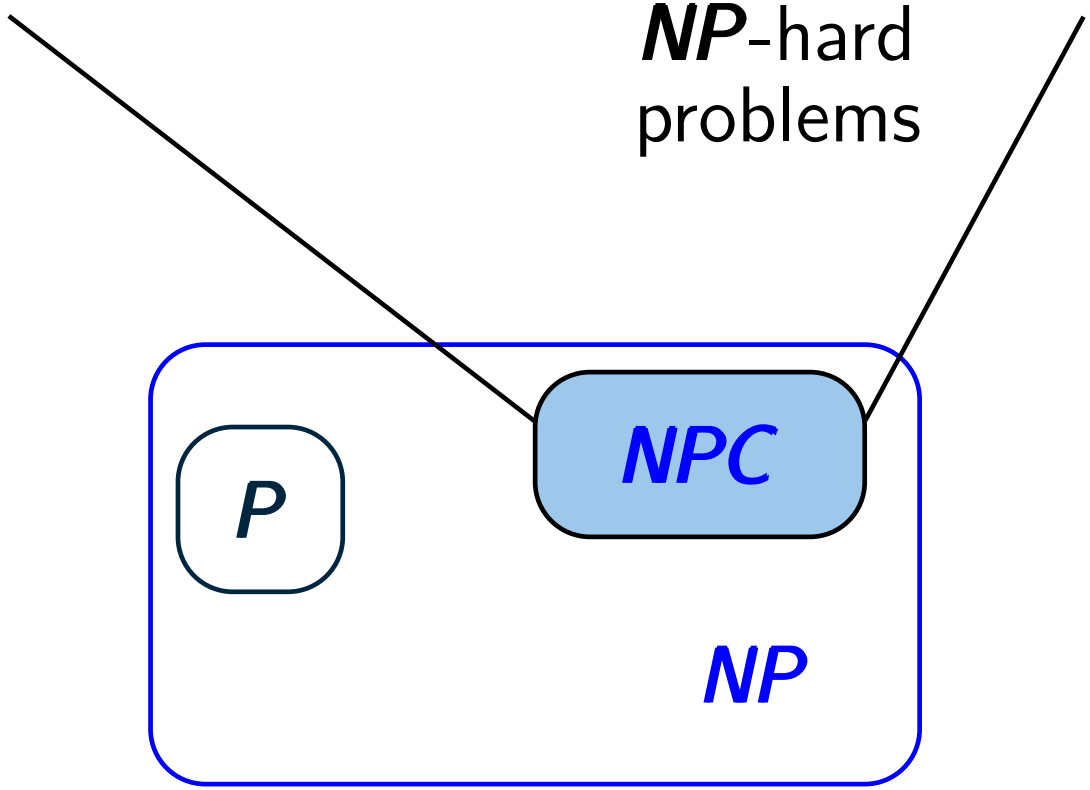

***NPC*** = class of ***NP***-complete problems

(1) $L \in NP$

(2) $L$ is ***NP***-hard, i.e., $K \leq_{poly} L$ for all $K \in NP$

# Complexity classes *P*, *NP*





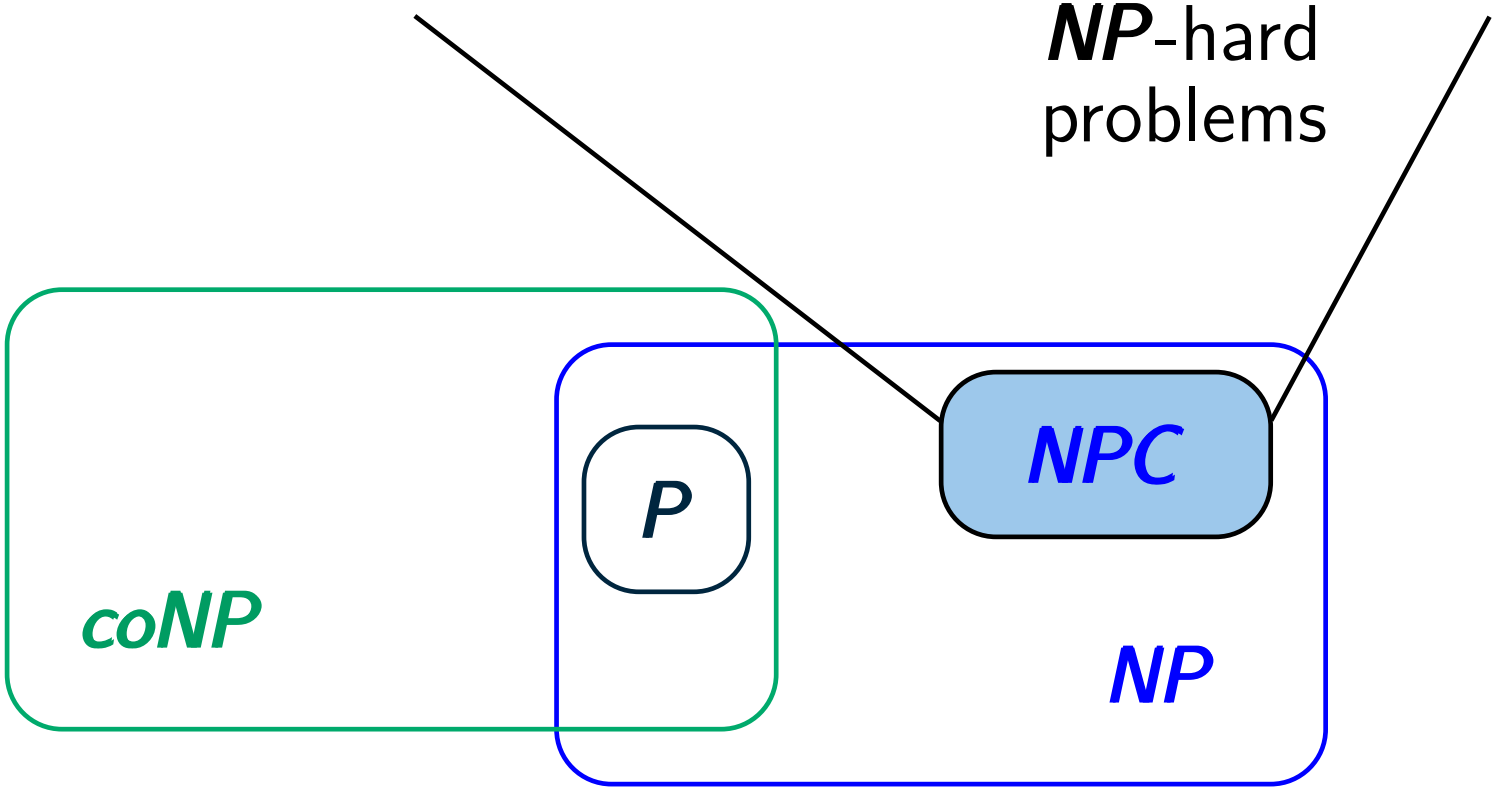

***NPC*** = class of ***NP***-complete problems

(1) $L \in NP$

(2) $L$ is ***NP***-hard, i.e., $K \leq_{poly} L$ for all $K \in NP$

# Complexity classes $P$, $NP$, $coNP$



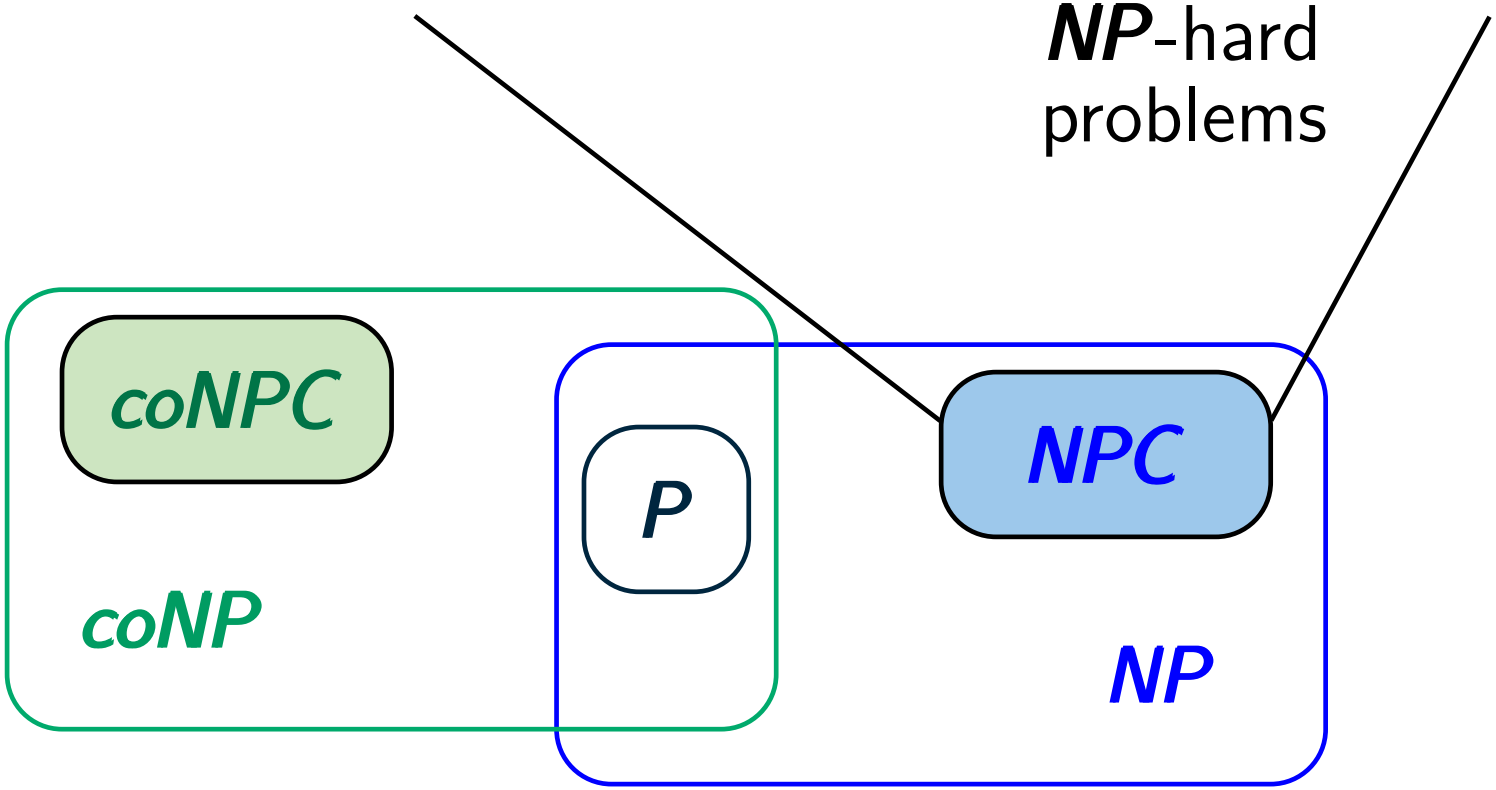



$$coNP = \{ \overline{L} : L \in NP \}$$

$\overline{L}$: complement of $L$

# Complexity classes *P*, *NP*, *coNP*





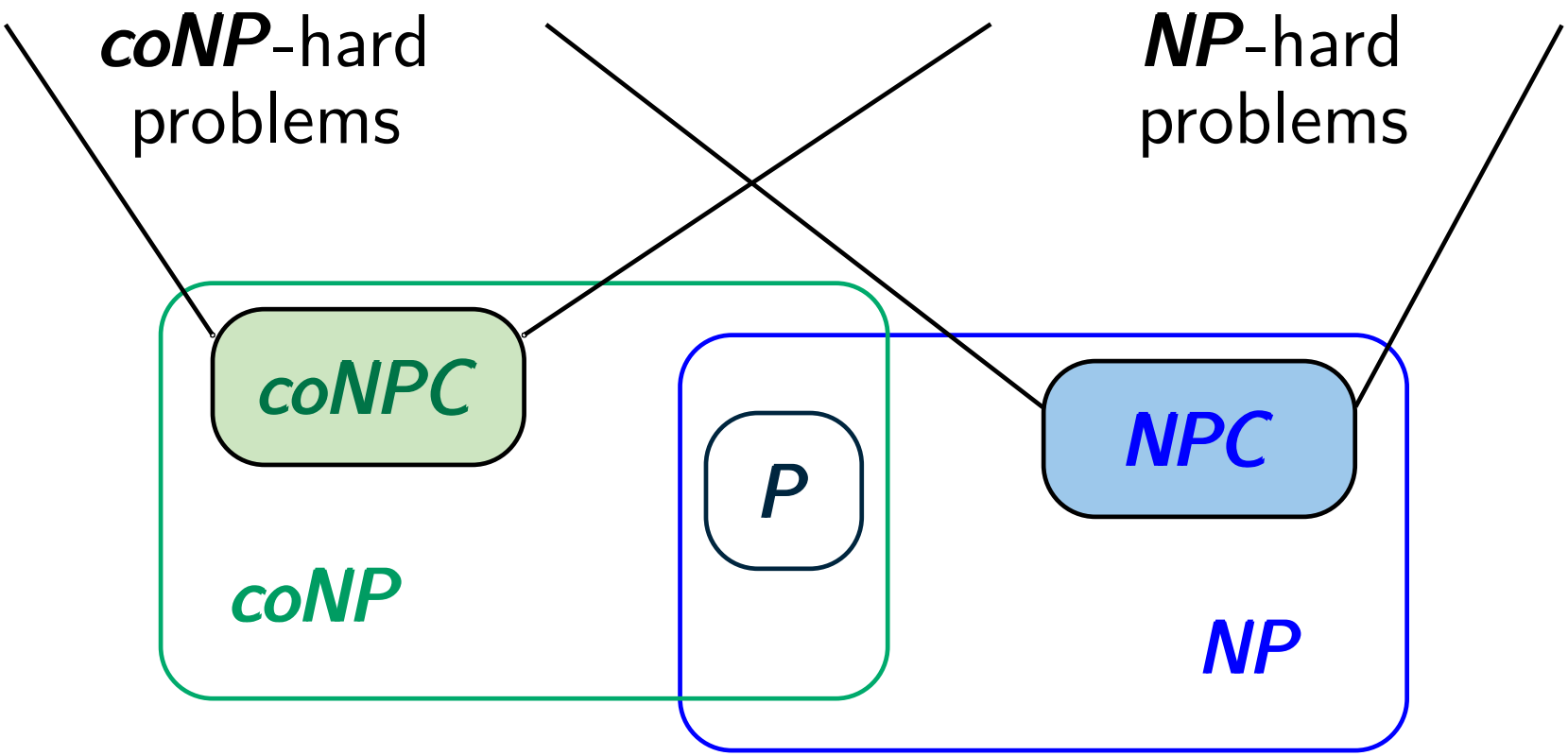

***coNPC*** = class of ***coNP***-complete problems

(1) $L \in$ ***coNP***

(2) $L$ is ***coNP***-hard, i.e., $K \leq_{poly} L$ for all $K \in$ ***coNP***

# Complexity classes *P*, *NP*, *coNP*



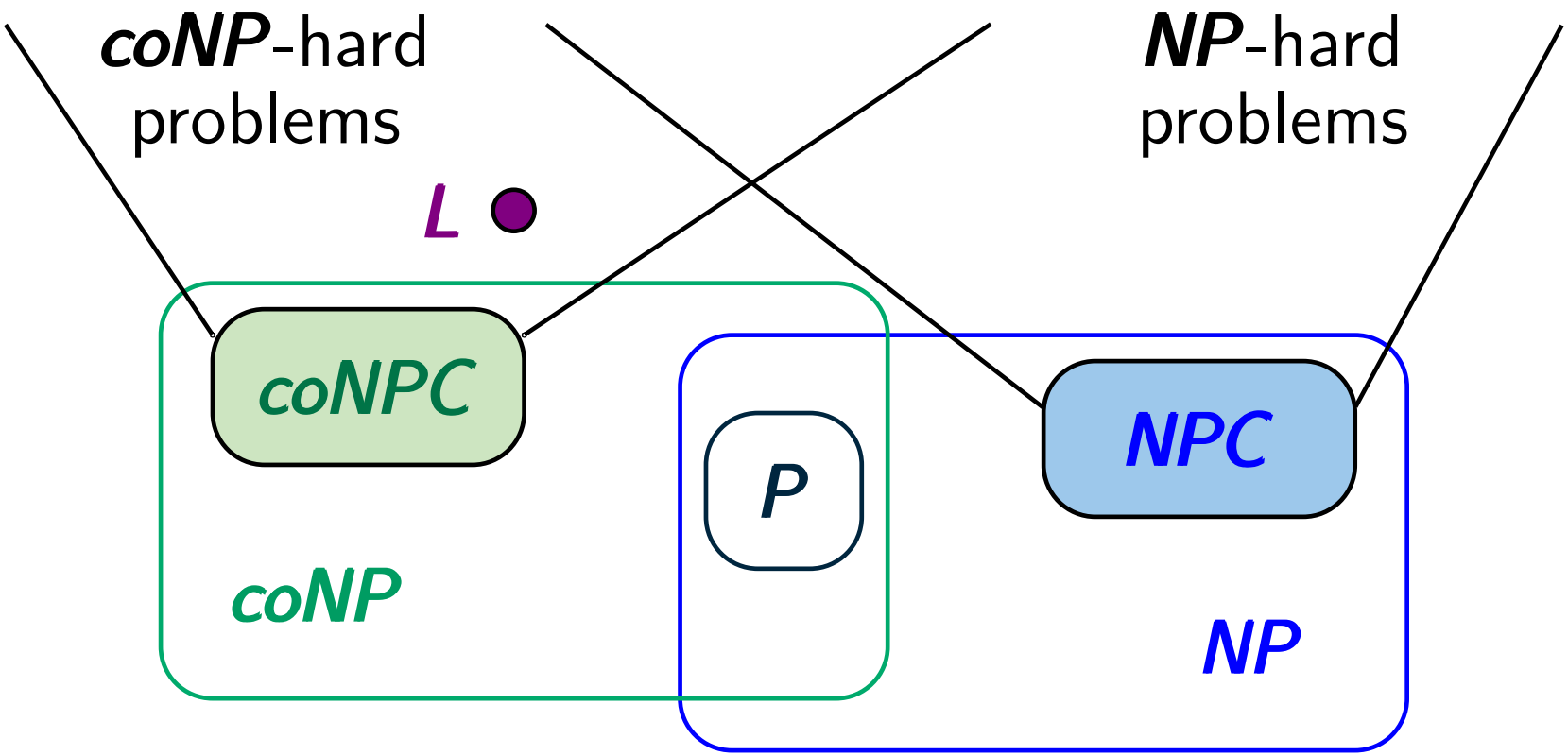



***coNPC*** = class of ***coNP***-complete problems

> $L$ is ***coNP***-hard iff $\overline{L}$ is ***NP***-hard

# Complexity classes *P*, *NP*, *coNP*



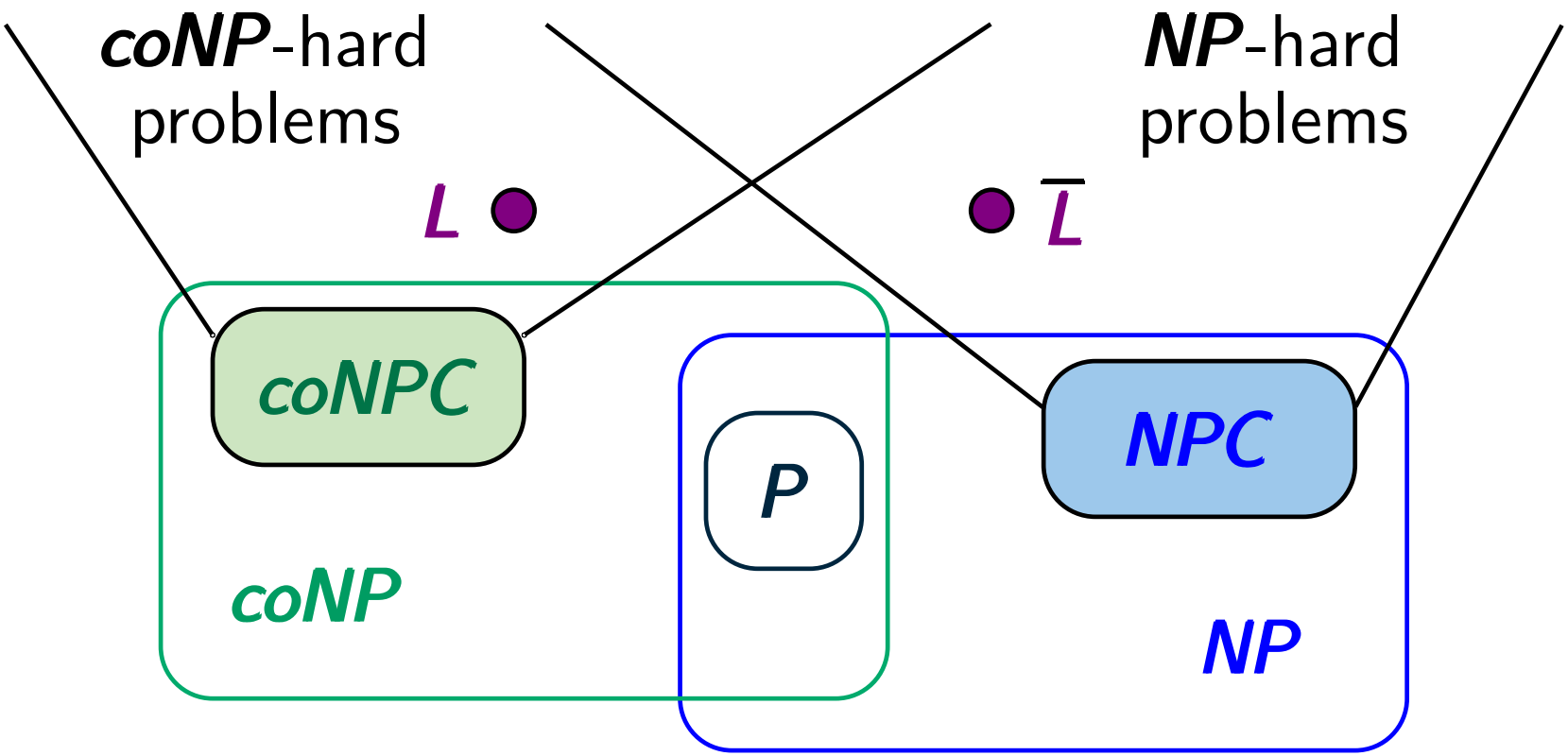



*coNPC* = class of *coNP*-complete problems

*L* is *coNP*-hard iff $\overline{L}$ is *NP*-hard

# Complexity classes *P*, *NP*, *coNP*



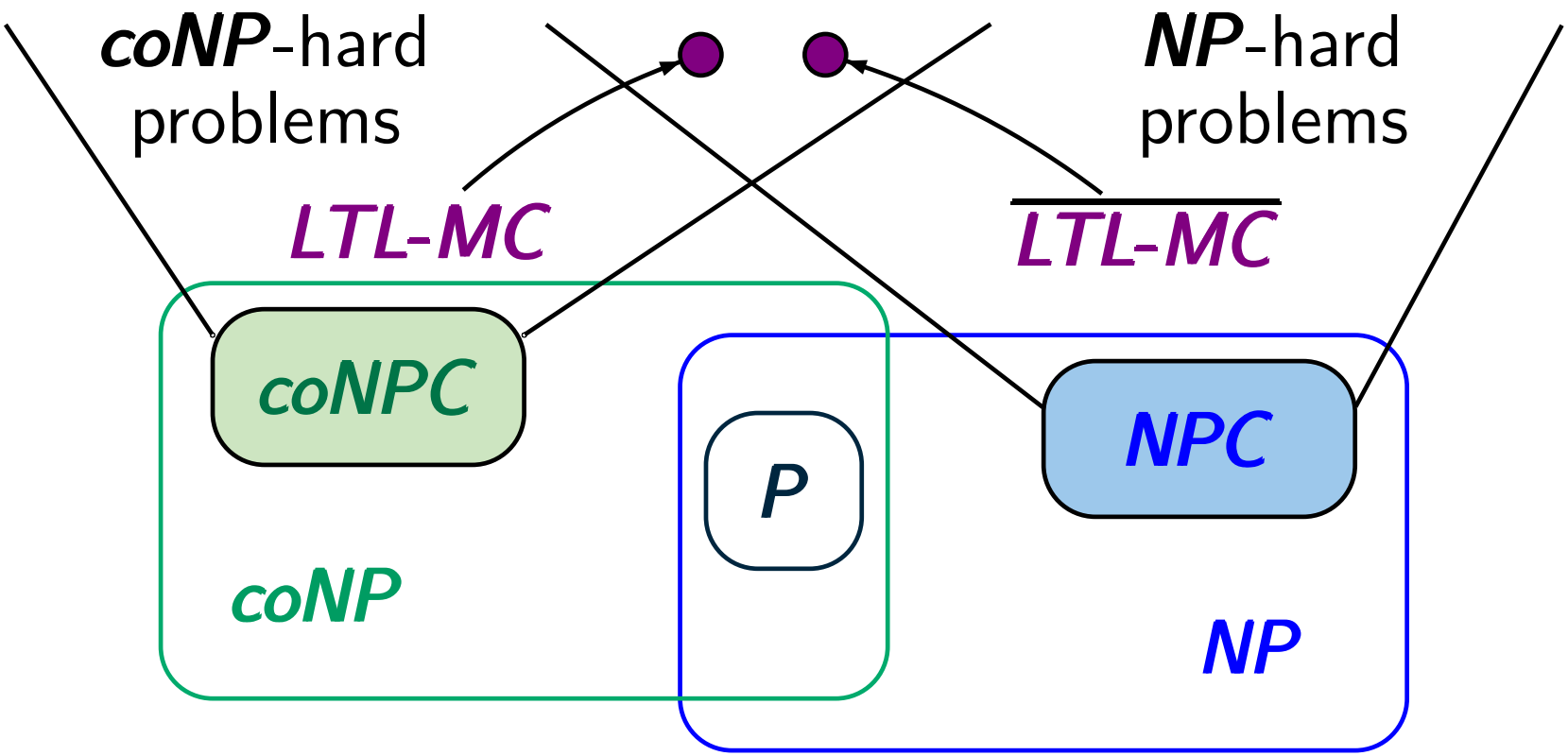



*coNPC* = class of *coNP*-complete problems

*L* is *coNP*-hard iff $\overline{L}$ is *NP*-hard

# Complexity classes *P*, *NP*, *coNP*






*coNPC* = class of *coNP*-complete problems

*L* is *coNP*-hard iff $\overline{L}$ is *NP*-hard

# *coNP*-hardness

The **LTL** model checking problem is ***coNP***-hard

# *coNP*-hardness

The **LTL** model checking problem is ***coNP***-hard

proof by a polynomial reduction

$$HP \leq_{poly} \overline{LTL\text{-}MC}$$

Hamilton path problem

complement of the **LTL** model checking problem

# *coNP*-hardness

The **LTL** model checking problem is ***coNP***-hard

proof by a polynomial reduction

$$\mathit{HP} \leq_{\mathit{poly}} \overline{\mathit{LTL\text{-}MC}}$$

Hamilton path problem

complement of the **LTL** model checking problem

complement of the **LTL** model checking problem:

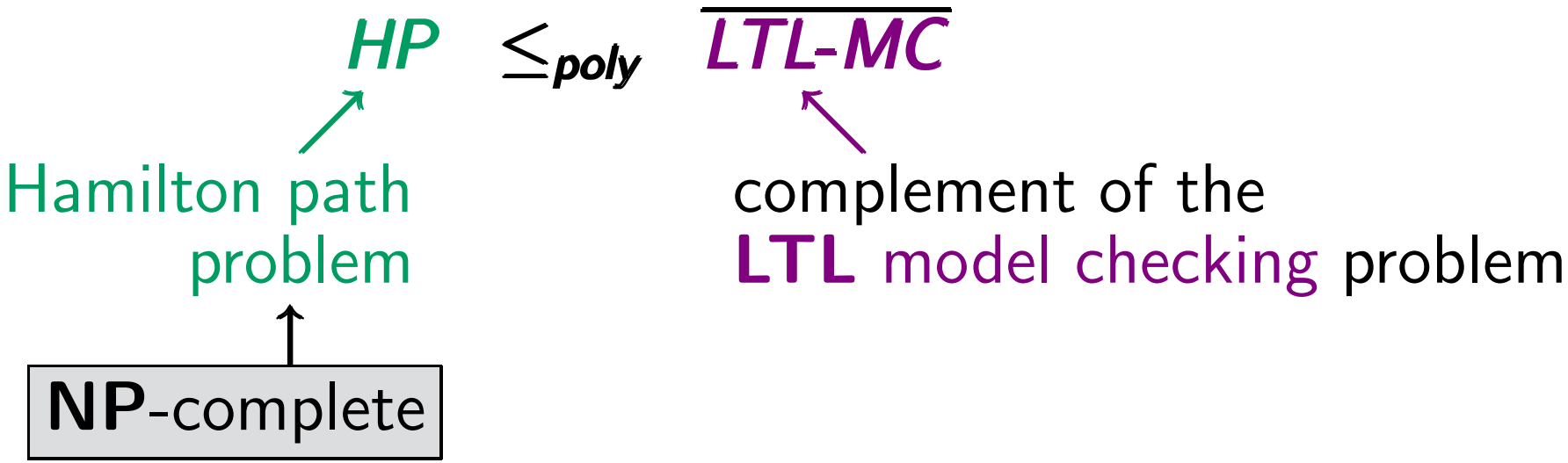
*given*: finite transition system $\mathcal{T}$, LTL-formula $\varphi$

*question*: does $\mathcal{T} \not\models \varphi$ hold **?**

# *coNP*-hardness

The **LTL** model checking problem is ***coNP***-hard

proof by a polynomial reduction




complement of the **LTL** model checking problem:

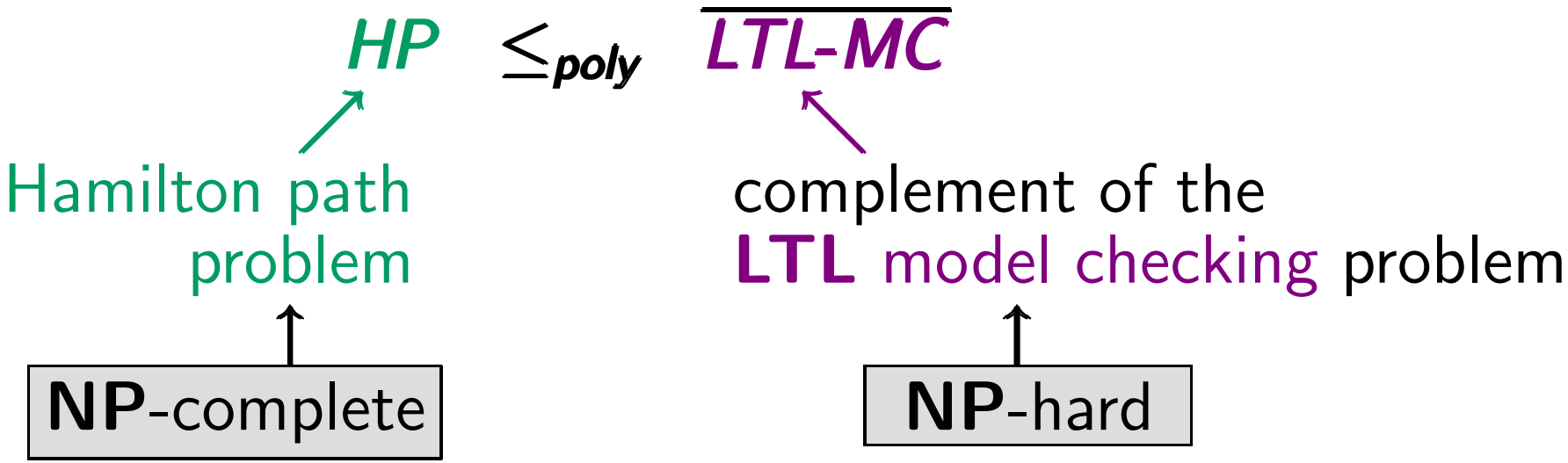
*given*: finite transition system $\mathcal{T}$, LTL-formula $\varphi$

*question*: does $\mathcal{T} \not\models \varphi$ hold **?**

# *coNP*-hardness

The **LTL** model checking problem is ***coNP***-hard

proof by a polynomial reduction




complement of the **LTL** model checking problem:

*given*: finite transition system $\mathcal{T}$, LTL-formula $\varphi$

*question*: does $\mathcal{T} \not\models \varphi$ hold **?**

# Complexity of LTL model checking



We just saw:

> The **LTL** model checking problem is ***coNP***-hard

We now prove:

> The **LTL** model checking problem is ***PSPACE***-complete

# The complexity class *PSPACE*

# The complexity class *PSPACE*



***PSPACE*** = class of decision problems solvable by a deterministic polynomially space-bounded algorithm

# The complexity class *PSPACE*



***PSPACE*** = class of decision problems solvable by a deterministic polynomially space-bounded algorithm

- ***NP*** $\subseteq$ ***PSPACE***

# The complexity class *PSPACE*



***PSPACE*** = class of decision problems solvable by a deterministic polynomially space-bounded algorithm

- ***NP*** $\subseteq$ ***PSPACE***

  ↑

  **DFS**-based analysis of the computation tree of an ***NP***-algorithm

# The complexity class *PSPACE*



***PSPACE*** = class of decision problems solvable by a deterministic polynomially space-bounded algorithm

- ***NP*** $\subseteq$ ***PSPACE***

  $\uparrow$

  **DFS**-based analysis of the computation tree of an ***NP***-algorithm

  *space requirements:*

  recursion depth $\widehat{=}$ height of computation tree

# The complexity class *PSPACE*



***PSPACE*** = class of decision problems solvable by a deterministic polynomially space-bounded algorithm

- ***NP*** $\subseteq$ ***PSPACE***
- ***PSPACE*** = ***coPSPACE***

  (holds for any deterministic complexity class)

# The complexity class *PSPACE*



***PSPACE*** = class of decision problems solvable by a deterministic polynomially space-bounded algorithm

- ***NP*** $\subseteq$ ***PSPACE***
- ***PSPACE*** = ***coPSPACE***
  (holds for any deterministic complexity class)
- ***PSPACE*** = ***NPSPACE*** (Savitch's Theorem)

# The complexity class *PSPACE*



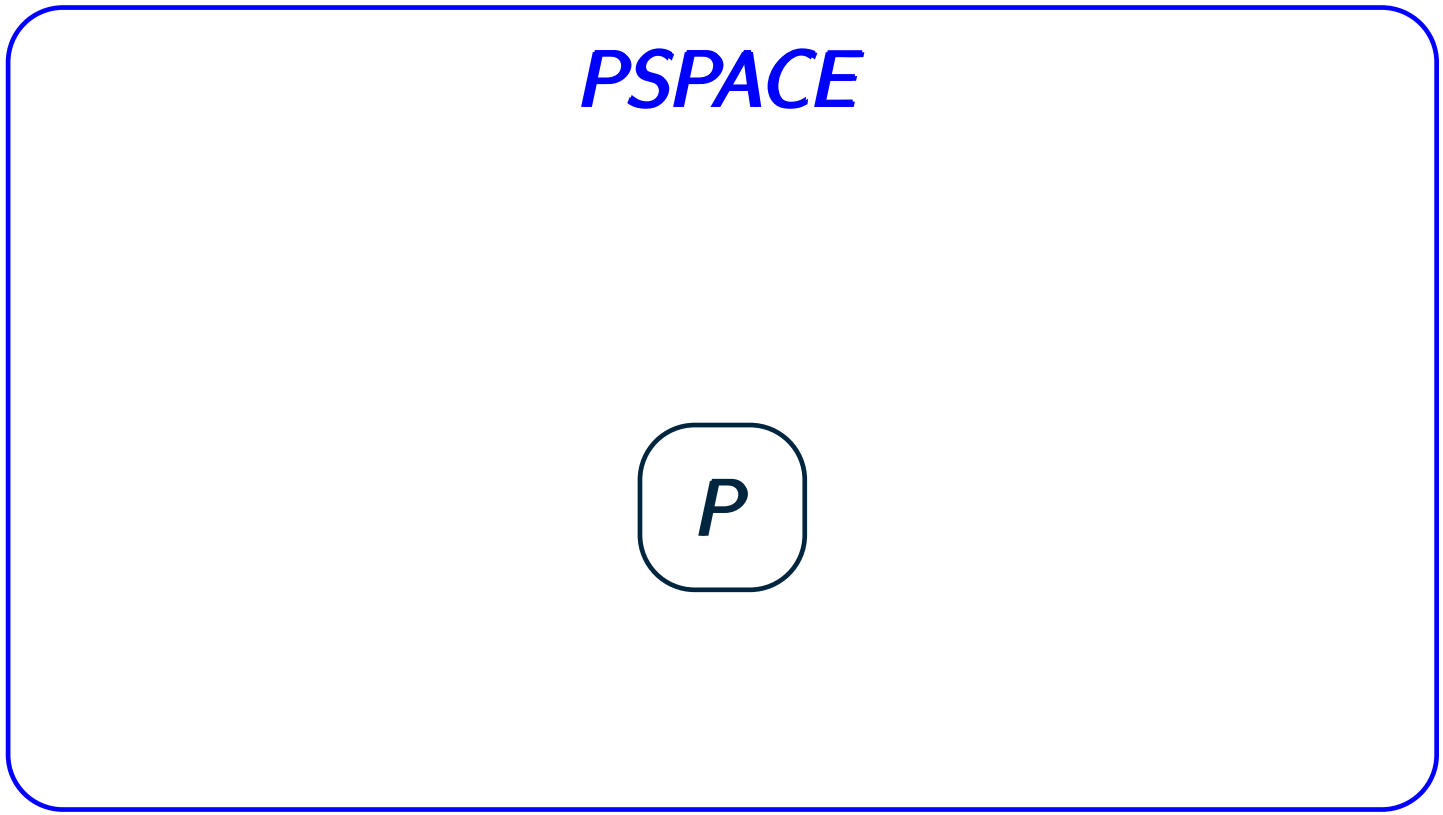
***PSPACE*** = class of decision problems solvable by a deterministic polynomially space-bounded algorithm

- ***NP*** $\subseteq$ ***PSPACE***
- ***PSPACE*** = ***coPSPACE***
  (holds for any deterministic complexity class)
- ***PSPACE*** = ***NPSPACE*** (Savitch's Theorem)

To prove $L \in$ ***PSPACE*** it suffices to provide a nondeterministic polynomially space-bounded algorithm for the complement $\overline{L}$ of $L$

# Complexity classes *P*, *NP*, *coNP*, *PSPACE*

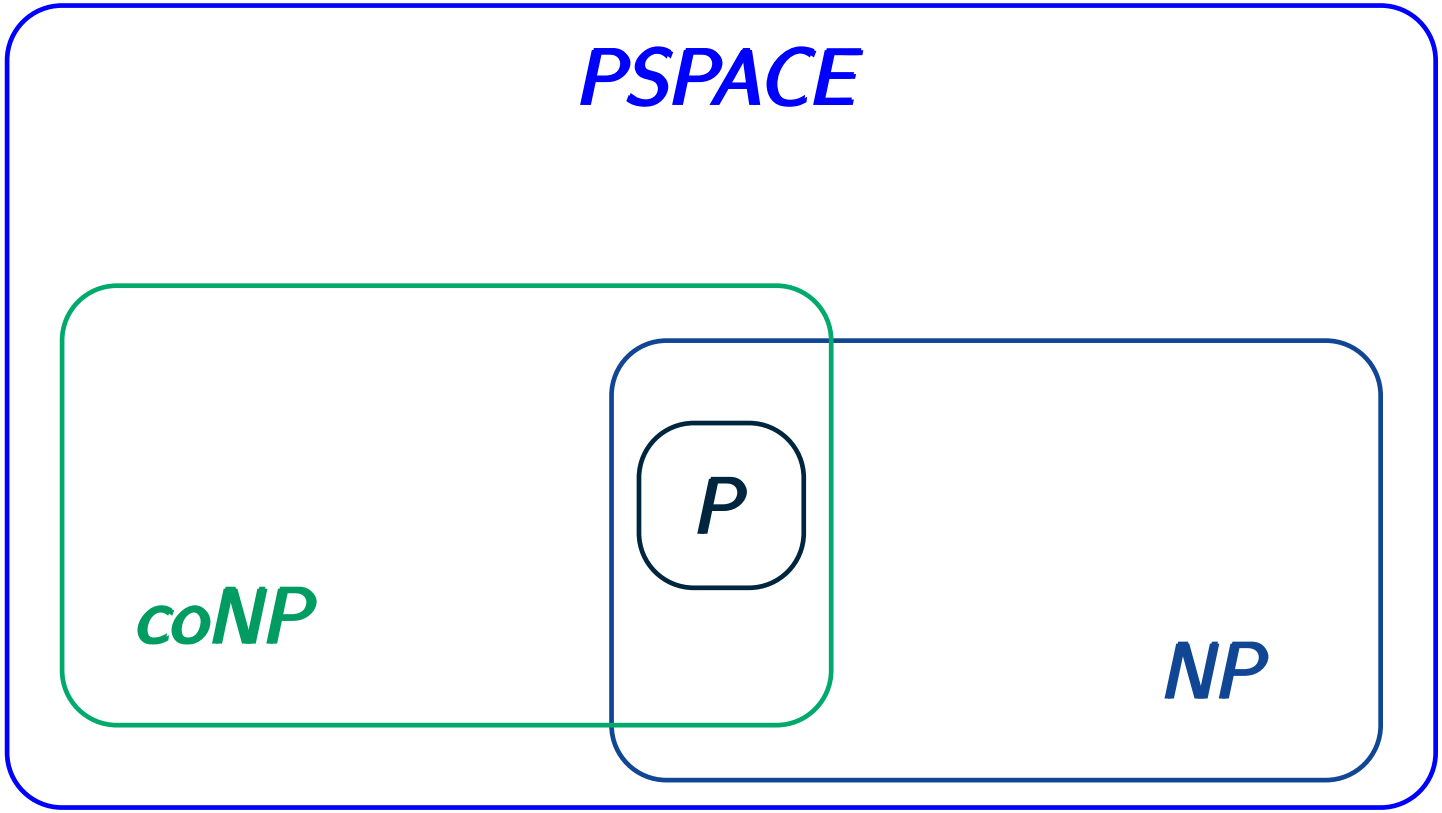





***PSPACE*** = class of decision problems that are decidable in deterministic polynomial space

# Complexity classes *P*, *NP*, *coNP*, *PSPACE*

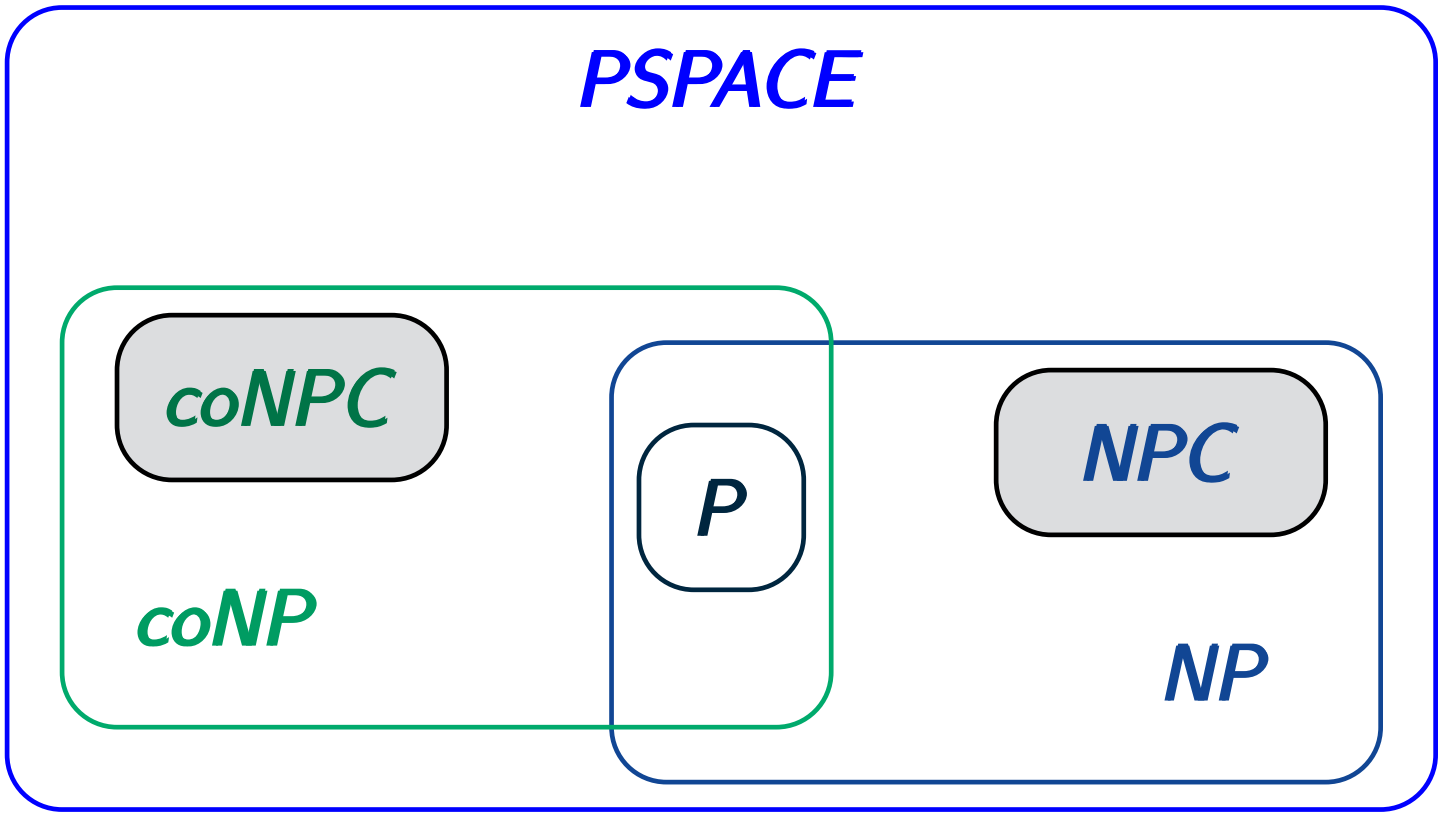





***PSPACE*** = class of decision problems that are decidable in deterministic polynomial space

# Complexity classes *P*, *NP*, *coNP*, *PSPACE*

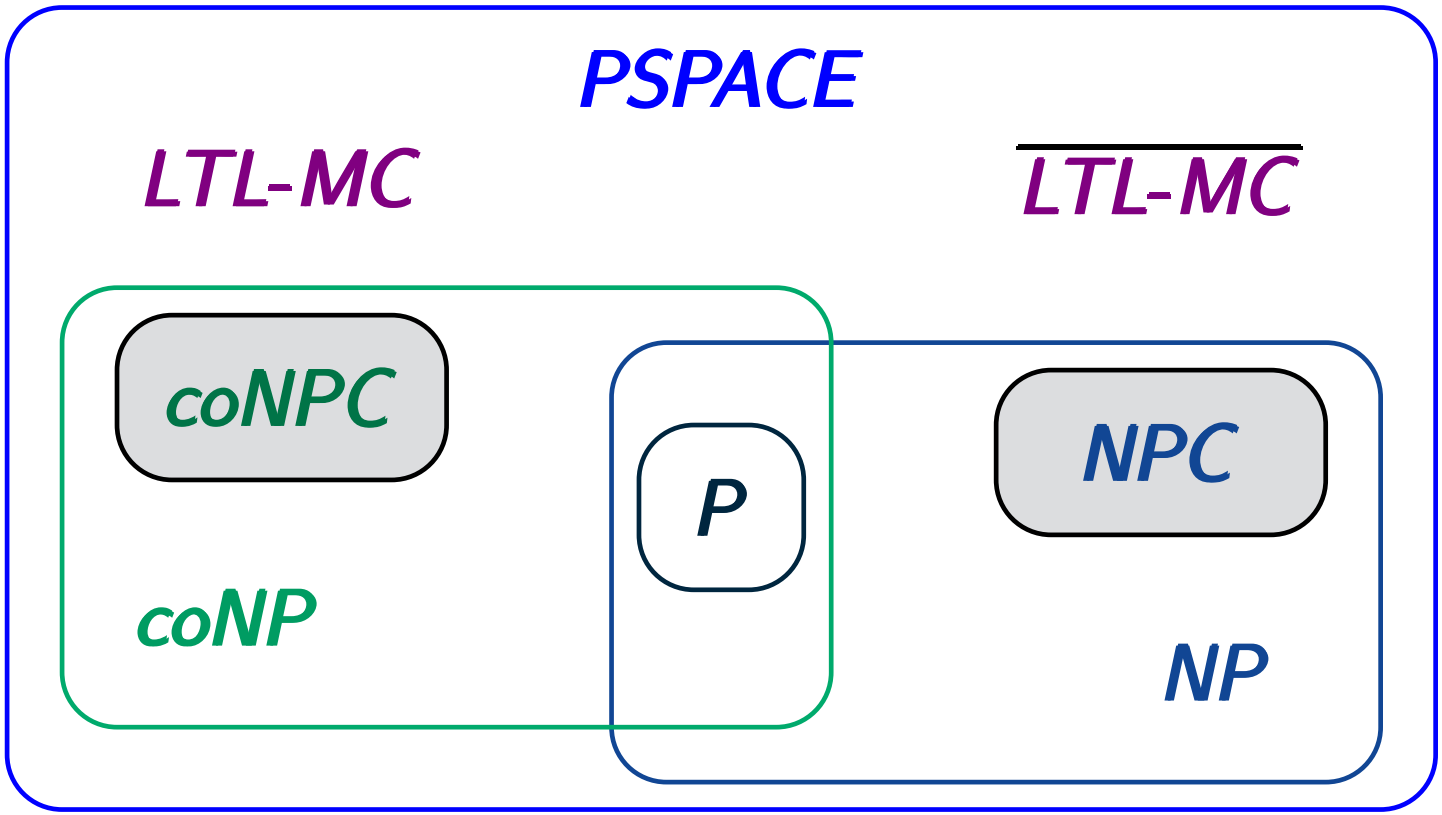





*PSPACE* = class of decision problems that are decidable in deterministic polynomial space

# Complexity classes *P*, *NP*, *coNP*, *PSPACE*






*PSPACE* = class of decision problems that are decidable in deterministic polynomial space