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, Section 5.2.

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 φ 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 φ 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 φ over AP

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

basic idea: try to refute $\mathcal{T} \models \varphi$ by searching for a path π 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 φ over AP

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

basic idea: try to refute $\mathcal{T} \models \varphi$ by searching for a path π 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 φ 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 φ over AP

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

1. construct an **NBA** $\mathcal{A}$ for $Words(\neg\varphi)$
2. search a path π 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 φ over AP

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

1. construct an **NBA** $\mathcal{A}$ for $Words(\neg\varphi)$
2. search a path π 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 φ over AP

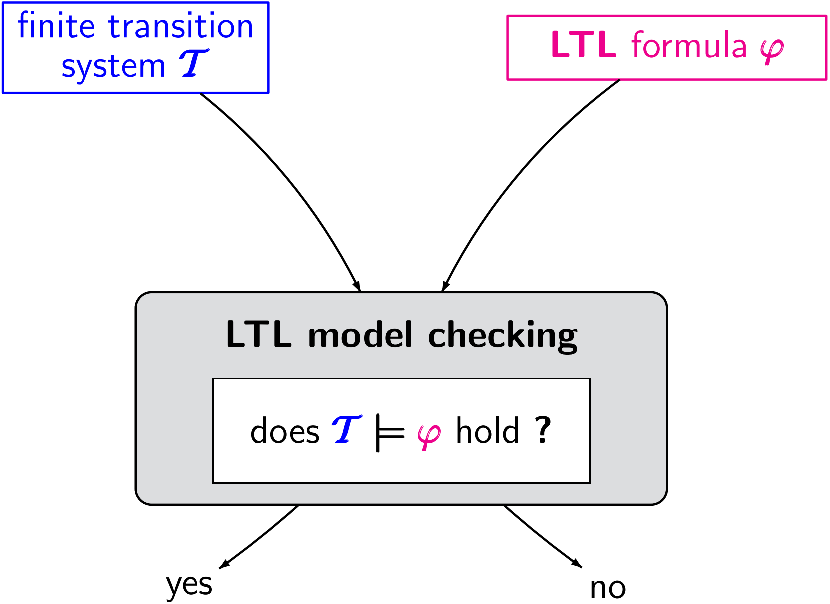
question: does $\mathcal{T} \models \varphi$ hold ?

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

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

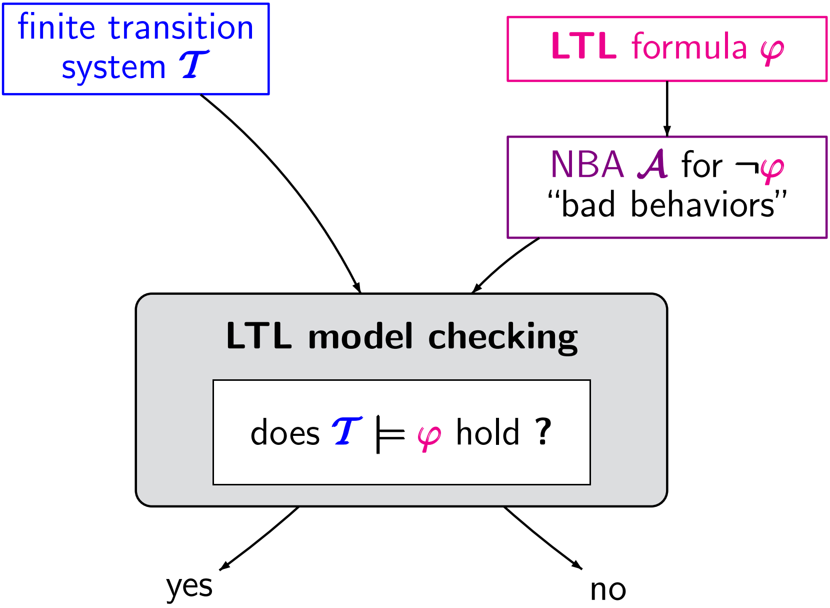
↑

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

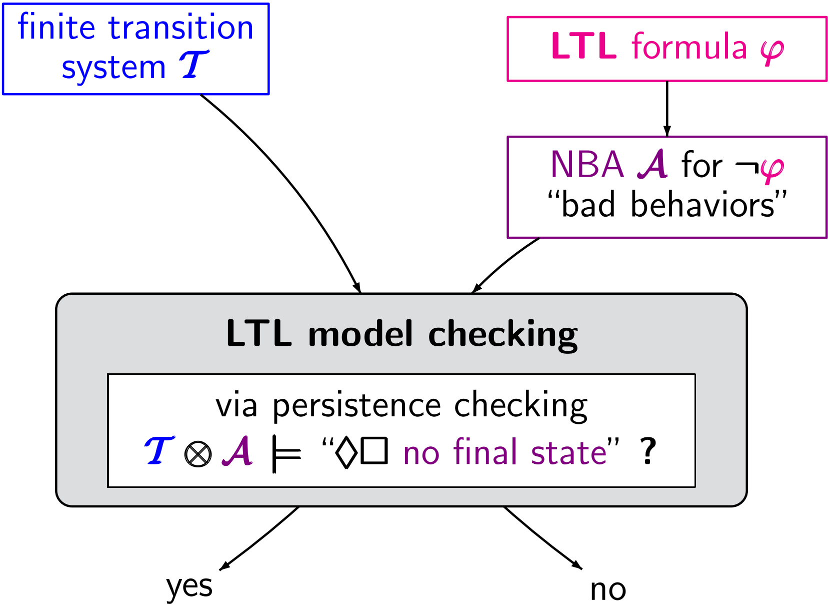


finite transition system $\mathcal{T}$
LTL formula φ
LTL model checking
does $\mathcal{T} \models \varphi$ hold ?
yes
no

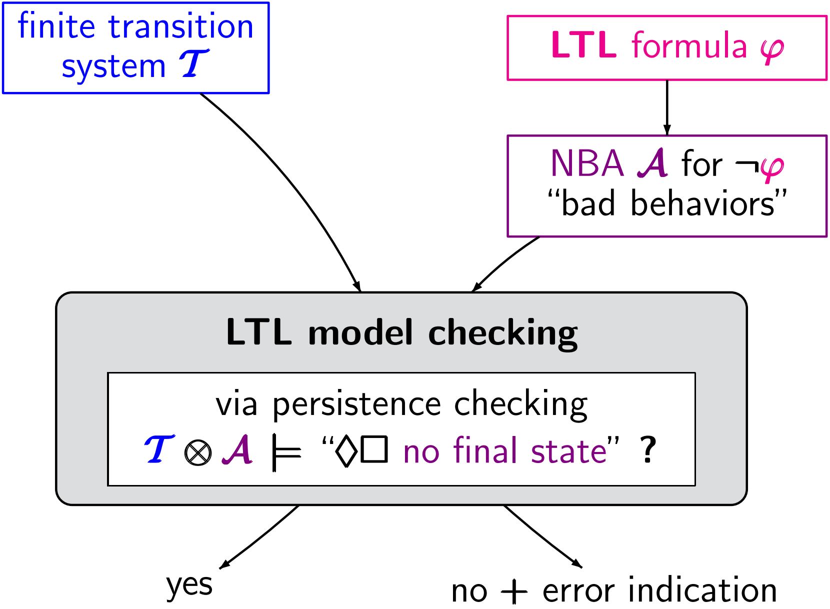
Automata-based LTL model checking



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 φ

Safety and LTL model checking

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

Safety and LTL model checking

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

Safety and LTL model checking

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

Safety and LTL model checking

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

Safety and LTL model checking

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

Safety and LTL model checking

safety property E	LTL-formula φ
NFA for the bad prefixes for E $\mathcal{L}(\mathcal{A}) \subseteq (2^{AP})^+$	**NBA** for the "bad behaviors" $\mathcal{L}_\omega(\mathcal{A}) = Words(\neg\varphi)$
$Traces_{fin}(\mathcal{T}) \cap \mathcal{L}(\mathcal{A}) = \varnothing$	$Traces(\mathcal{T}) \cap \mathcal{L}_\omega(\mathcal{A}) = \varnothing$
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 property E	LTL-formula φ
NFA for the bad prefixes for E $\mathcal{L}(\mathcal{A}) \subseteq (2^{AP})^+$	**NBA** for the "bad behaviors" $\mathcal{L}_\omega(\mathcal{A}) = Words(\neg\varphi)$
$Traces_{fin}(\mathcal{T}) \cap \mathcal{L}(\mathcal{A}) = \varnothing$	$Traces(\mathcal{T}) \cap \mathcal{L}_\omega(\mathcal{A}) = \varnothing$
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: $\widehat{\pi} \in Paths_{fin}(\mathcal{T})$ s.t. $trace(\widehat{\pi}) \in \mathcal{L}(\mathcal{A})$	

Safety and LTL model checking

safety property E	LTL-formula φ
NFA for the bad prefixes for E $\mathcal{L}(\mathcal{A}) \subseteq (2^{AP})^+$	**NBA** for the "bad behaviors" $\mathcal{L}_\omega(\mathcal{A}) = Words(\neg\varphi)$
$Traces_{fin}(\mathcal{T}) \cap \mathcal{L}(\mathcal{A}) = \varnothing$	$Traces(\mathcal{T}) \cap \mathcal{L}_\omega(\mathcal{A}) = \varnothing$
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: $\widehat{\pi} \in Paths_{fin}(\mathcal{T})$ s.t. $trace(\widehat{\pi}) \in \mathcal{L}(\mathcal{A})$	error indication: prefix of a path π s.t. $trace(\pi) \in \mathcal{L}_\omega(\mathcal{A})$

Safety vs LTL model checking

Safety vs LTL model checking

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

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

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

$\mathcal{T} \models$ LTL-formula φ

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

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

Safety vs LTL model checking

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

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

iff there is <u>no</u> 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 φ

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

iff there is <u>no</u> 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 $\mathit{Traces_{fin}}(\mathcal{T}) \cap \mathcal{L}(\mathcal{A}) = \varnothing$

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 φ

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

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 φ

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

Recall: nondeterministic Büchi automata

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

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

run for a word $A_0 A_1 A_2 \ldots \in \Sigma^\omega$:

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

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

$\mathcal{L}_\omega(\mathcal{A}) \overset{\text{def}}{=}$ set of infinite words over Σ 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
- Σ alphabet $\longleftarrow$ here: $\Sigma = 2^{AP}$
- $\delta : Q \times \Sigma \rightarrow 2^Q$ transition relation
- $Q_0 \subseteq Q$ set of initial states
- $F \subseteq Q$ set of final states, also called accept states

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

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

From LTL to NBA

From LTL to NBA

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

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

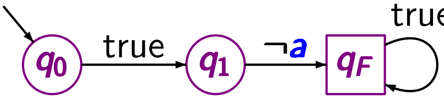
From LTL to NBA

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

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

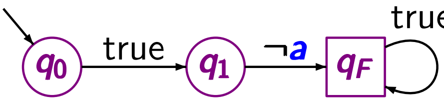
proof: ... later ...

NBA for LTL formulas



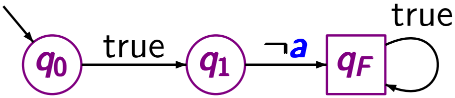
$\mathcal{L}_\omega(\mathcal{A}) = \textbf{?}$

NBA for LTL formulas

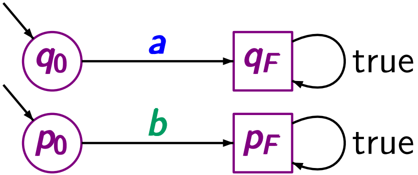


$$\mathcal{L}_\omega(\mathcal{A}) = Words(\bigcirc \neg a)$$

NBA for LTL formulas

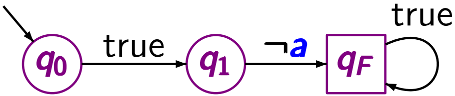


$\mathcal{L}_\omega(\mathcal{A}) = Words(\bigcirc \neg a)$

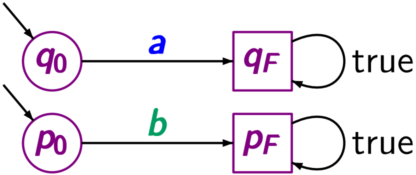


$\mathcal{L}_\omega(\mathcal{A}) = \textbf{?}$

NBA for LTL formulas

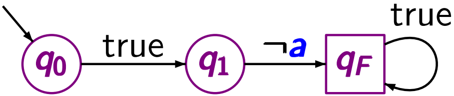


$$\mathcal{L}_\omega(\mathcal{A}) = Words(\bigcirc \neg a)$$

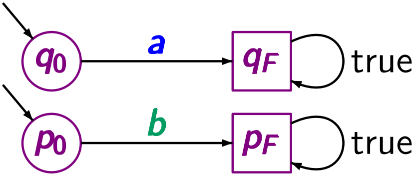


$$\mathcal{L}_\omega(\mathcal{A}) = Words(a \vee b)$$

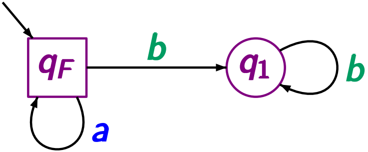
NBA for LTL formulas



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

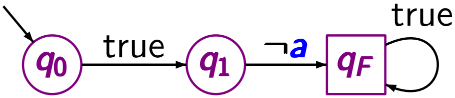


$$\mathcal{L}_\omega(\mathcal{A}) = \mathit{Words}(a \vee b)$$

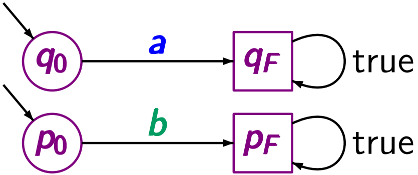


$$\mathcal{L}_\omega(\mathcal{A}) = \,?$$

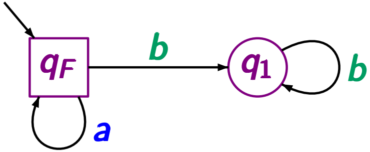
NBA for LTL formulas



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

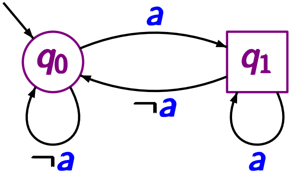


$$\mathcal{L}_\omega(\mathcal{A}) = Words(a \vee b)$$



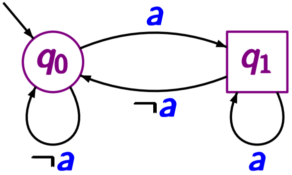
$$\mathcal{L}_\omega(\mathcal{A}) = Words(\Box a)$$

NBA for LTL formulas



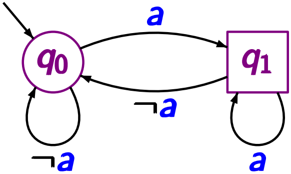
$$\mathcal{L}_\omega(\mathcal{A}) = \textbf{?}$$

NBA for LTL formulas

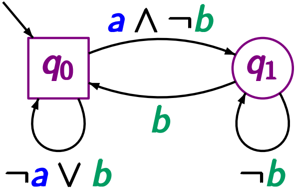


$$\mathcal{L}_\omega(\mathcal{A}) = Words(\Box\Diamond a)$$

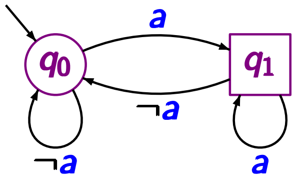
NBA for LTL formulas



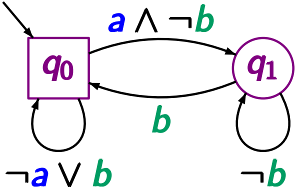
$$\mathcal{L}_\omega(\mathcal{A}) = Words(\Box\Diamond a)$$



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

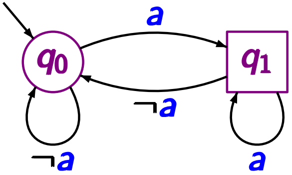


$$\mathcal{L}_\omega(\mathcal{A}) = Words(\Box\Diamond a)$$

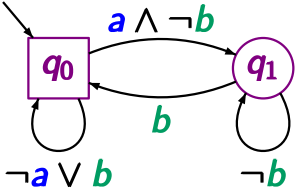


$$\mathcal{L}_\omega(\mathcal{A}) = \textbf{?}$$

e.g., $\varnothing\,\varnothing\,\varnothing\,\varnothing \ldots = \varnothing^\omega$ and $(\{a\}\,\{b\})^\omega$ are accepted by $\mathcal{A}$



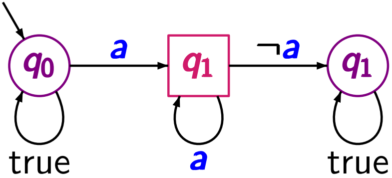
$$\mathcal{L}_\omega(\mathcal{A}) = \mathit{Words}(\Box\Diamond a)$$



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

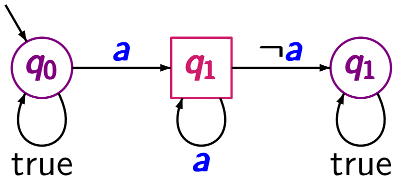
e.g., $\varnothing\,\varnothing\,\varnothing\,\varnothing \ldots = \varnothing^\omega$ and $(\{a\}\,\{b\})^\omega$ are accepted by $\mathcal{A}$

NBA for LTL formula



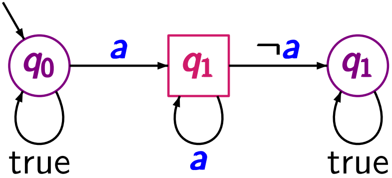
$$\mathcal{L}_\omega(\mathcal{A}) = \textbf{?}$$

NBA for LTL formula



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

NBA for LTL formula



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

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:

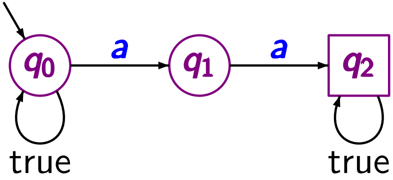
$$\mathcal{L}_\omega(\mathcal{A}) = \overline{E} = \left(2^{AP}\right)^\omega \setminus E$$

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:

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

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

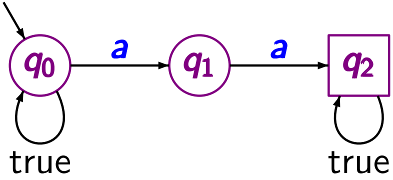


NFA and NBA for safety properties

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

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

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



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

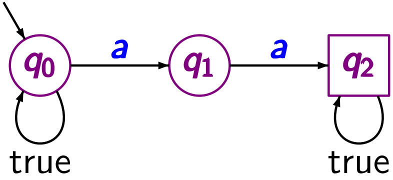
NFA and NBA for safety properties

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

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

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

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



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

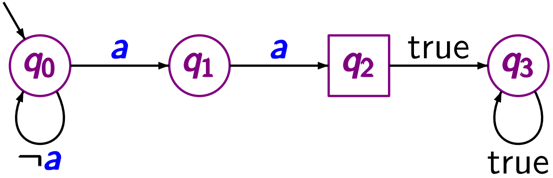
NFA and NBA for safety properties

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

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

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

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



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

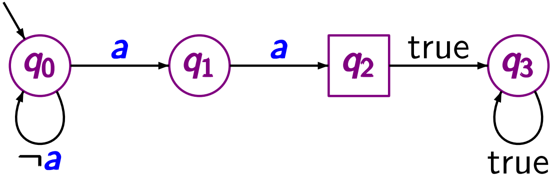
NFA and NBA for safety properties

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

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

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"



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

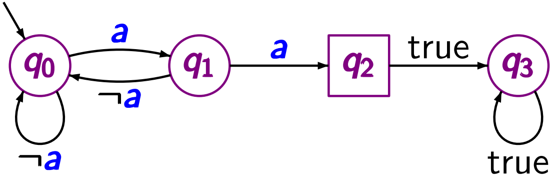
NFA and NBA for safety properties

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

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

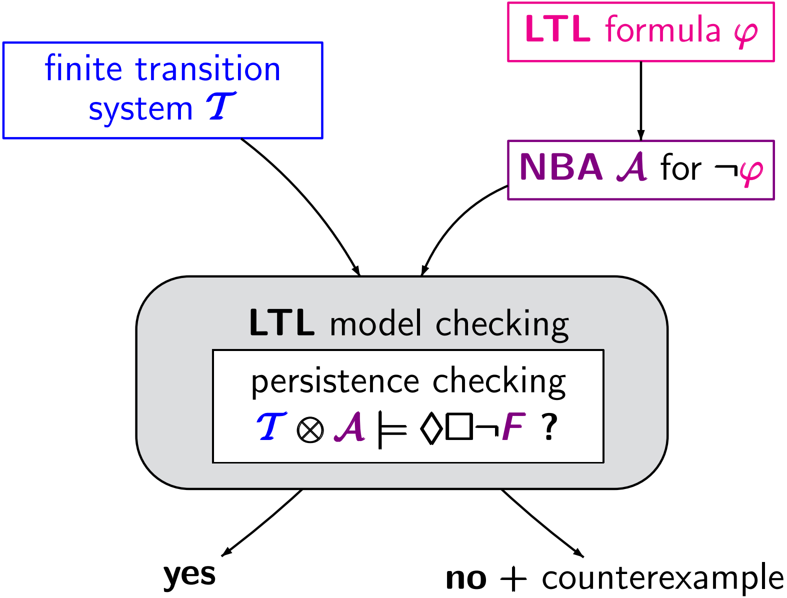
wrong, if $\mathcal{L}(\mathcal{A})$ = language of minimal bad prefixes even if $\mathcal{A}$ is a non-blocking DFA

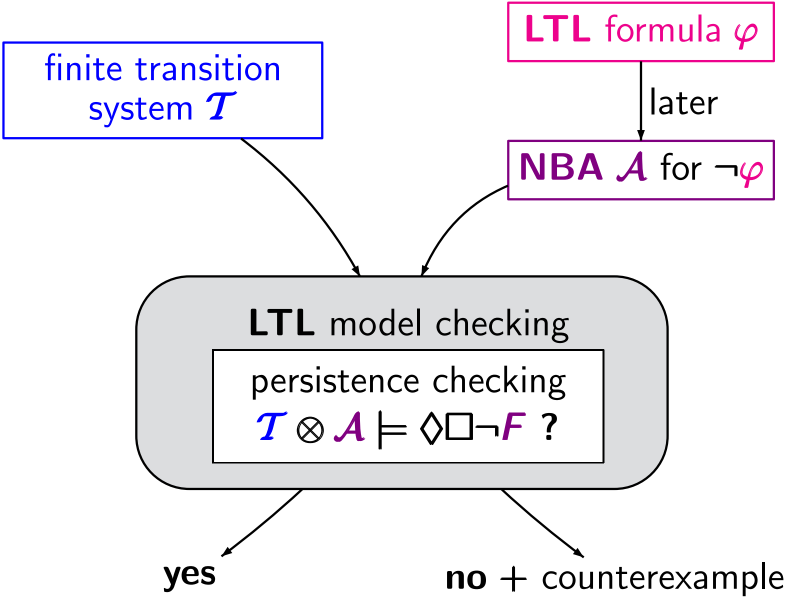
Example: $E \mathrel{\widehat{=}}$ "never a twice in a row"



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

LTL model checking




finite transition system $\mathcal{T}$
LTL formula φ
later
NBA $\mathcal{A}$ for $\neg\varphi$
LTL model checking
persistence checking
$\mathcal{T} \otimes \mathcal{A} \models \Diamond\Box\neg F$?
yes
no + counterexample

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

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

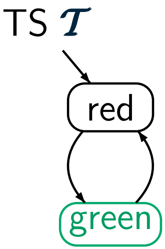
initial states: $S_0' = \{\langle s_0, q\rangle : s_0 \in S_0, q \in \delta(Q_0, L(s_0))\}$

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

transition relation:

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

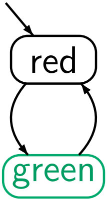
Example: LTL model checking



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

Example: LTL model checking

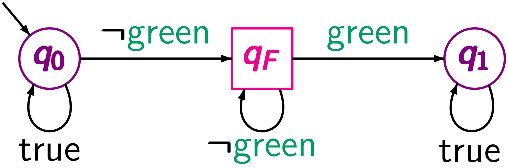
TS $\mathcal{T}$



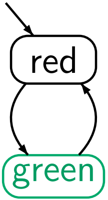
LTL formula $\varphi = \Box\Diamond \text{green}$

NBA $\mathcal{A}$ for the complement

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



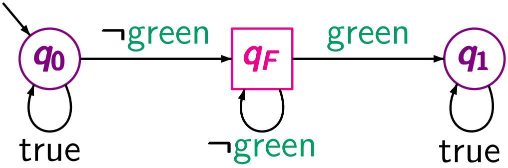
TS $\mathcal{T}$

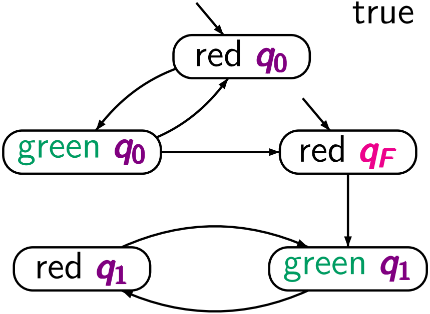


LTL formula $\varphi = \Box\Diamond green$

NBA $\mathcal{A}$ for the complement

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

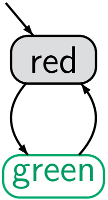




reachable fragment of the product TS $\mathcal{T} \otimes \mathcal{A}$

Example: LTL model checking

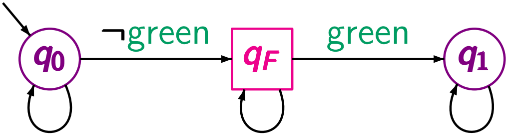
TS $\mathcal{T}$

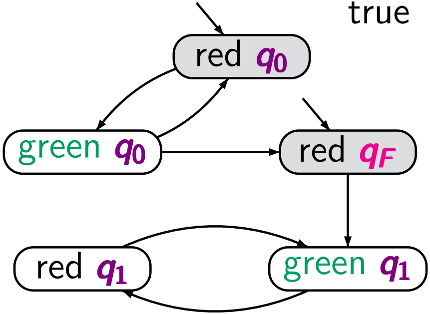


LTL formula $\varphi = \Box\Diamond green$

NBA $\mathcal{A}$ for the complement

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



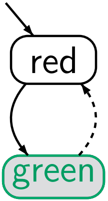


initial states:

$\langle red, q\rangle$ where

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

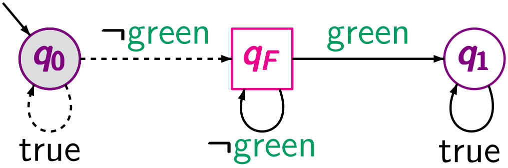
TS $\mathcal{T}$

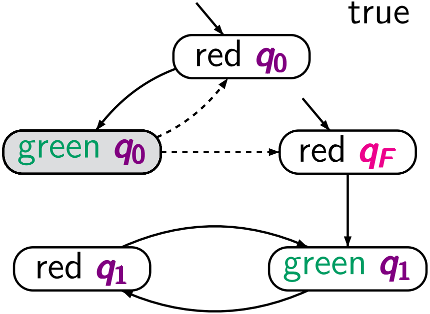


LTL formula $\varphi = \Box\Diamond \text{green}$

NBA $\mathcal{A}$ for the complement

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





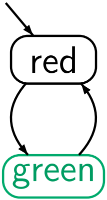
transition

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

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

Example: LTL model checking

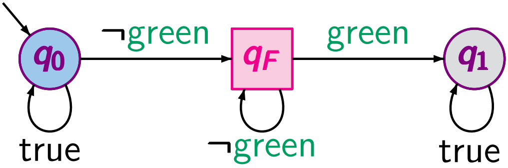
TS $\mathcal{T}$

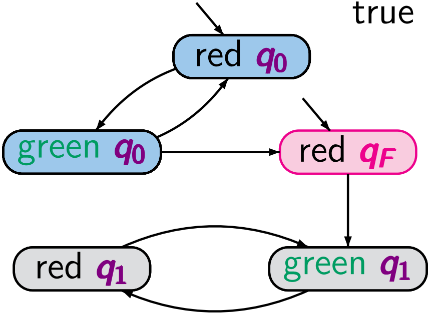


LTL formula $\varphi = \Box\Diamond \text{green}$

NBA $\mathcal{A}$ for the complement

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





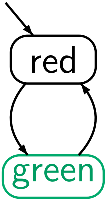
atomic propositions

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

obvious labeling function

Example: LTL model checking

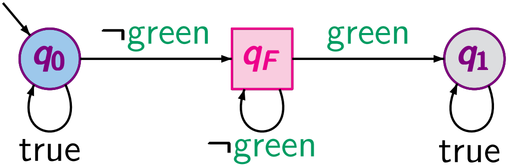
TS $\mathcal{T}$

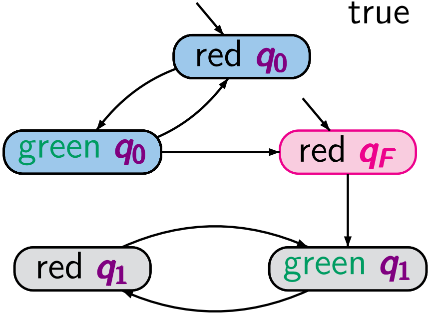


LTL formula $\varphi = \Box\Diamond \text{green}$

NBA $\mathcal{A}$ for the complement

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

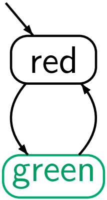




$\mathcal{T} \otimes \mathcal{A} \models \Diamond\Box\neg F$

Example: LTL model checking

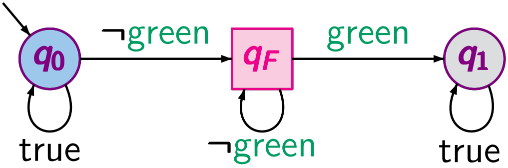
TS $\mathcal{T}$

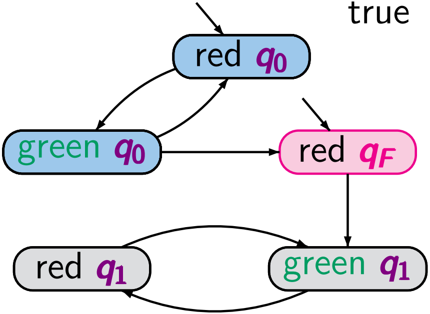


LTL formula $\varphi = \Box\Diamond green$

NBA $\mathcal{A}$ for the complement

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

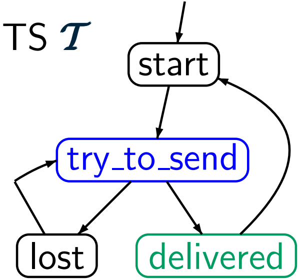




$\mathcal{T} \otimes \mathcal{A} \models \Diamond\Box\neg F$

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

Example: LTL model checking

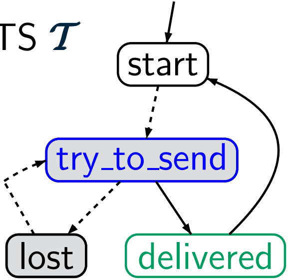


LTL formula $\varphi = \Box(\mathit{try} \rightarrow \Diamond \mathit{del})$

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

Example: LTL model checking

TS $\mathcal{T}$



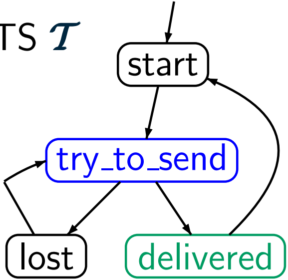
LTL formula $\varphi = \Box(\textit{try} \rightarrow \Diamond \textit{del})$

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

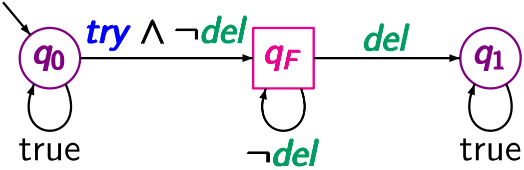
$\mathcal{T} \not\models \varphi$

Example: LTL model checking

TS $\mathcal{T}$



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

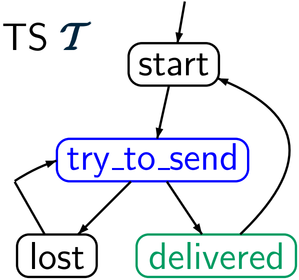


LTL formula $\varphi = \Box(try \to \Diamond del)$

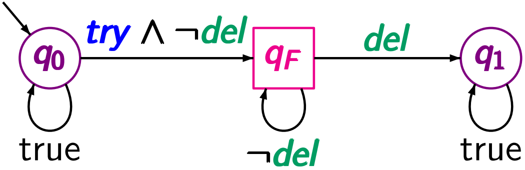
"each (repeatedly) sent message will eventually be delivered"

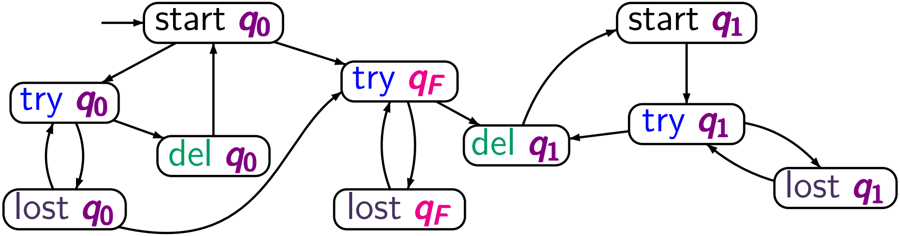
$\mathcal{T} \not\models \varphi$

Example: LTL model checking



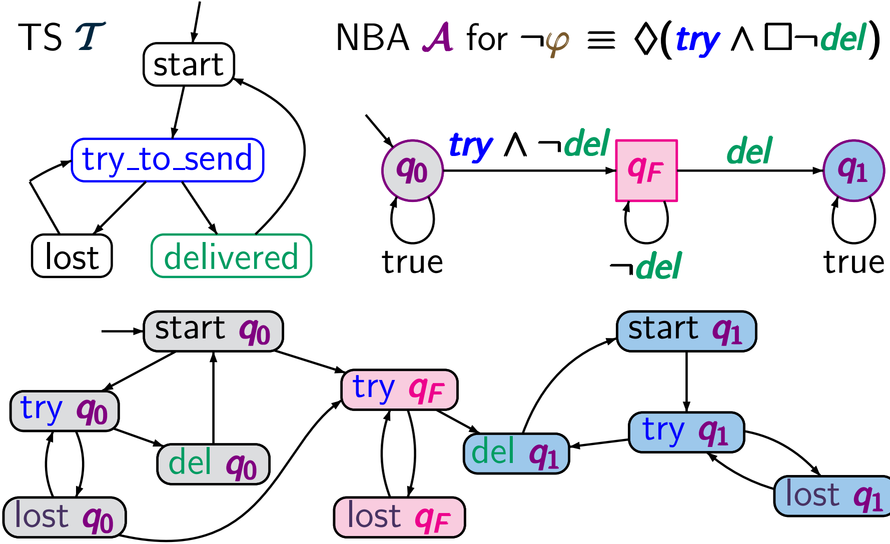
NBA $\mathcal{A}$ for $\neg\varphi \equiv \Diamond(\textit{try} \wedge \Box\neg\textit{del})$





reachable fragment of the product-TS

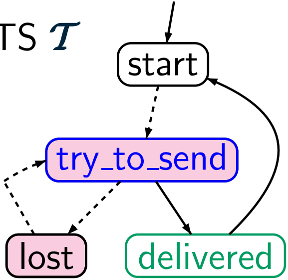
Example: LTL model checking



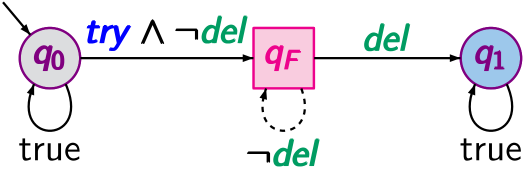
set of atomic propositions $AP' = \{q_0, q_1, q_F\}$

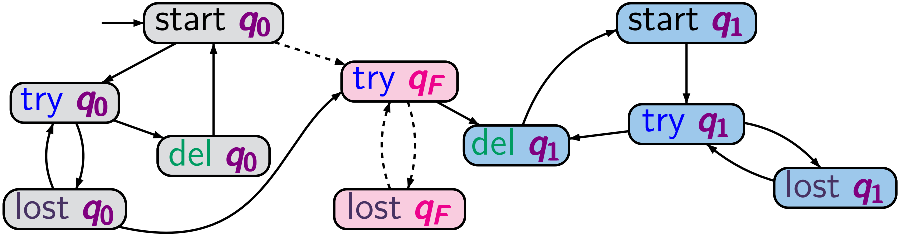
Example: LTL model checking

TS $\mathcal{T}$



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

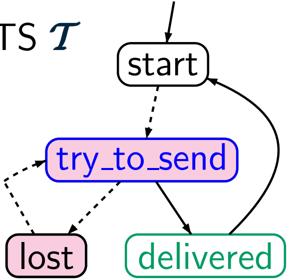




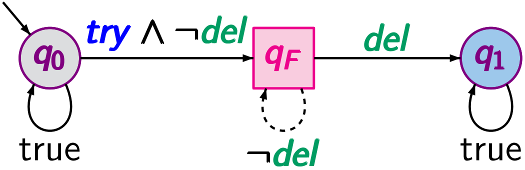
$\mathcal{T} \otimes \mathcal{A} \not\models \Diamond\Box\neg F$

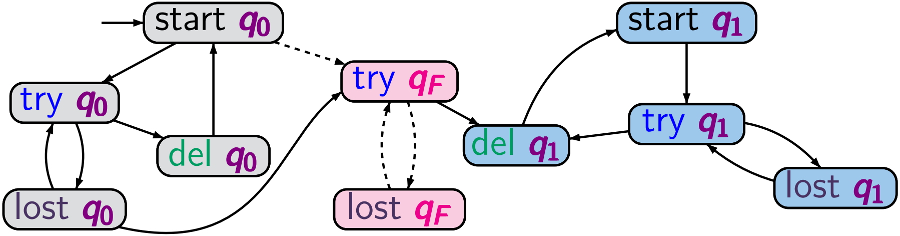
Example: LTL model checking

TS $\mathcal{T}$



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





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

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 $\mathcal{T} \otimes \mathcal{A}$	$\longleftarrow$	$\mathcal{O}(size(\mathcal{T}) \cdot size(\mathcal{A}))$

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

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

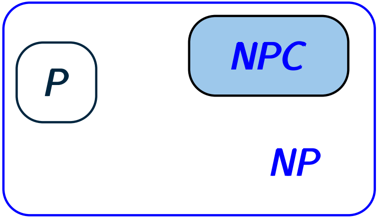
Recall: complexity classes

Complexity classes P, NP

P = class of decision problem solvable in deterministic polynomial time

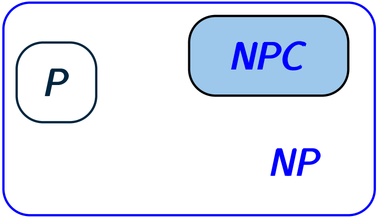
NP = class of decision problem solvable in nondeterministic polynomial time

Complexity classes *P*, *NP*



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

Complexity classes *P*, *NP*

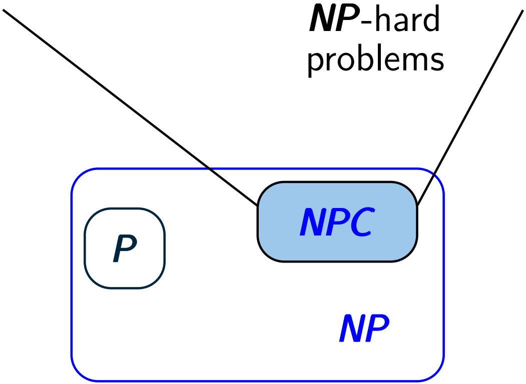


NPC = class of ***NP***-complete problems

(1) $L \in NP$

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

Complexity classes P, NP

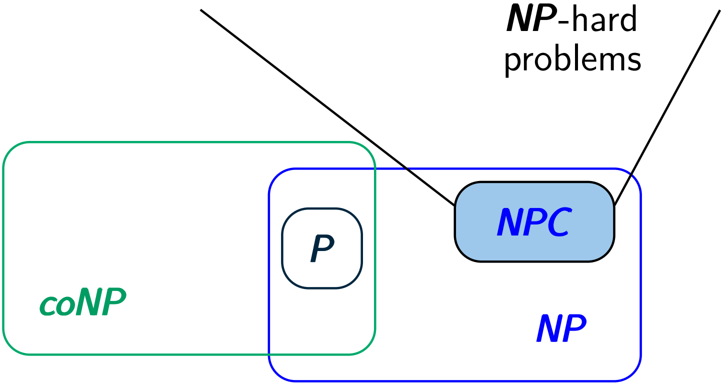


NPC = class of NP-complete problems

(1) $L \in NP$

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

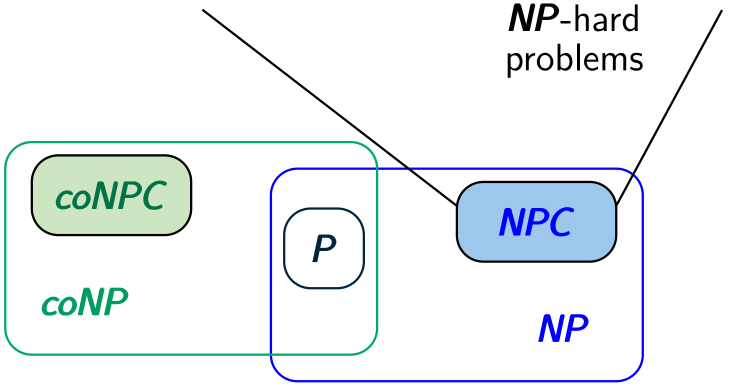
Complexity classes *P*, *NP*, *coNP*



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

↑ complement of L

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

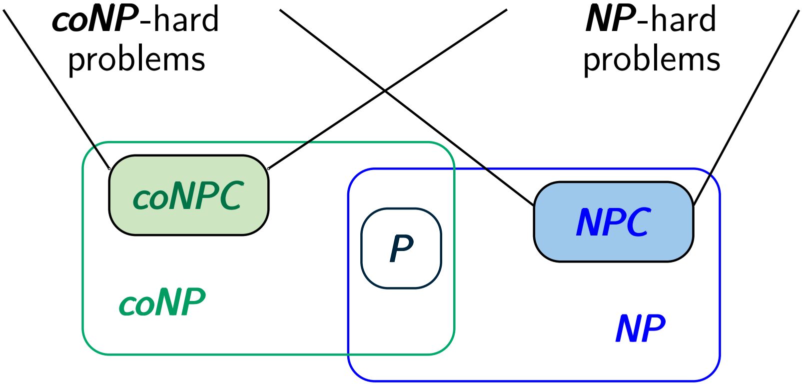


coNPC = class of ***coNP***-complete problems

(1) $L \in \mathit{coNP}$

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

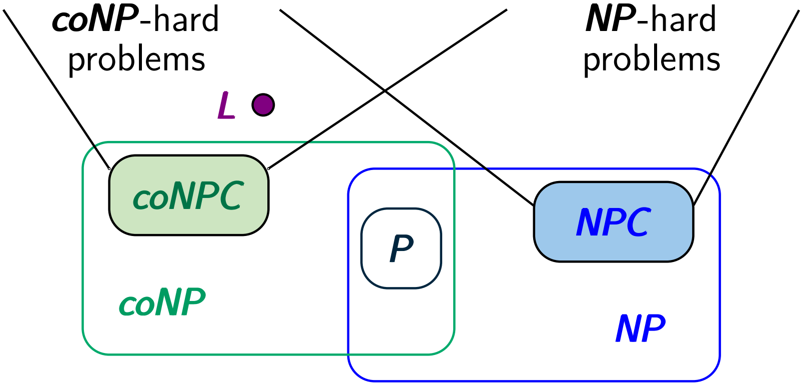
Complexity classes *P*, *NP*, *coNP*



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

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

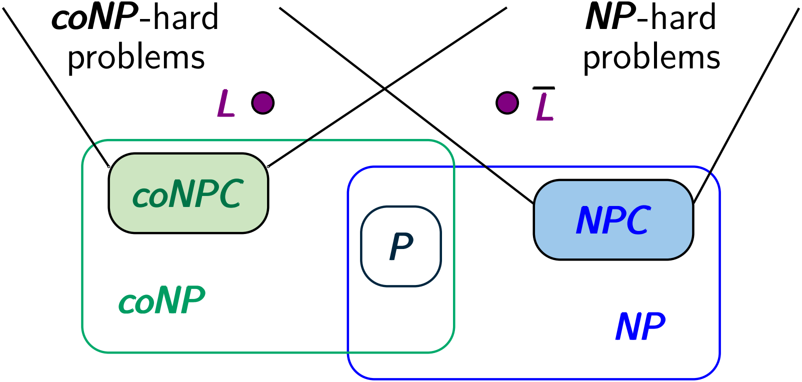
Complexity classes *P*, *NP*, *coNP*



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

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

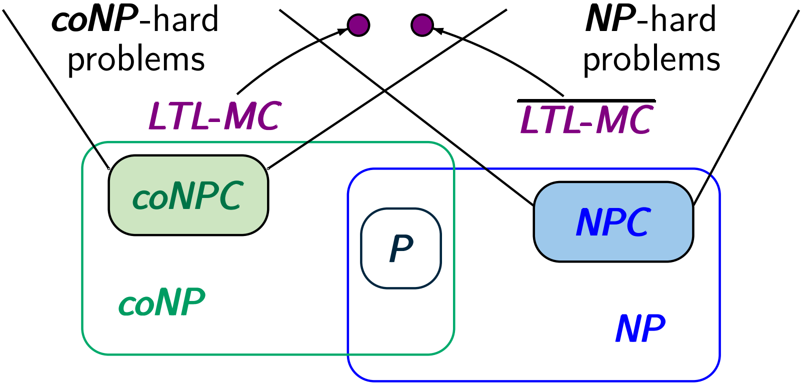
Complexity classes *P*, *NP*, *coNP*



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

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

proof by a polynomial reduction

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

HP: Hamilton path problem

$\overline{LTL\text{-}MC}$: complement of the **LTL** model checking problem

The LTL model checking problem is *coNP*-hard

proof by a polynomial reduction

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

HP: Hamilton path problem

$\overline{LTL\text{-}MC}$: complement of the **LTL** model checking problem

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

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

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

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

proof by a polynomial reduction

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

Hamilton path problem ↗ *HP*

NP-complete ↑ Hamilton path problem

complement of the **LTL** model checking problem ↖ $\overline{LTL\text{-}MC}$

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

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

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

coNP-hardness

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

proof by a polynomial reduction

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

HP: Hamilton path problem — **NP**-complete

$\overline{LTL\text{-}MC}$: complement of the **LTL** model checking problem — **NP**-hard

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

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

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

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)

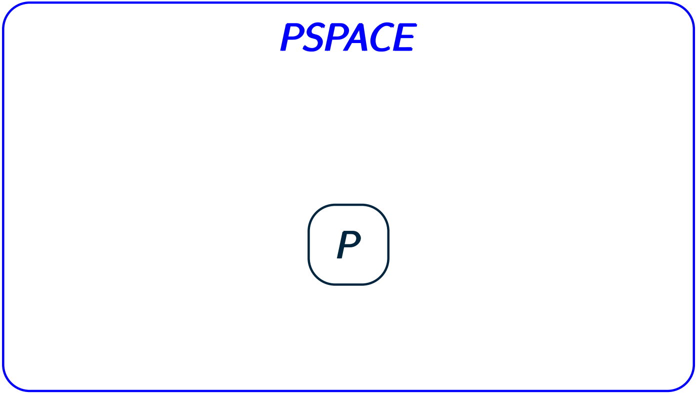
PSPACE = class of decision problems solvable by a deterministic polynomially space-bounded algorithm

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

↑

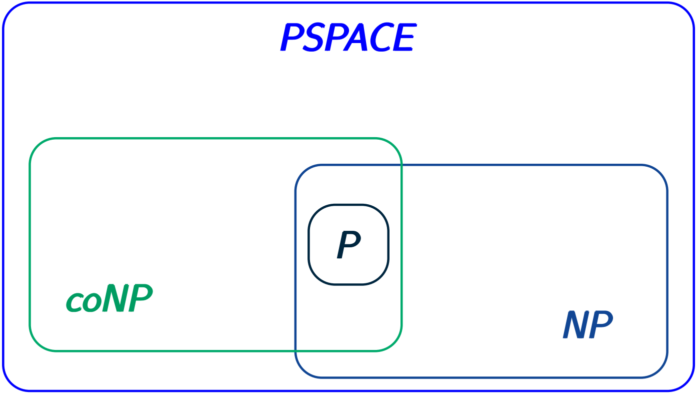
To prove $L \in$ *PSPACE* it suffices to provide a nondeterministic polynomially space-bounded algorithm for the complement $\overline{L}$ of L

Complexity classes P, NP, $coNP$, $PSPACE$



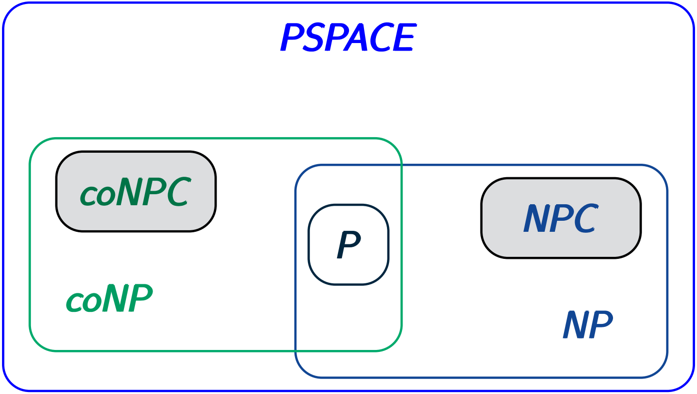
$PSPACE$ = class of decision problems that are decidable in deterministic polynomial space

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



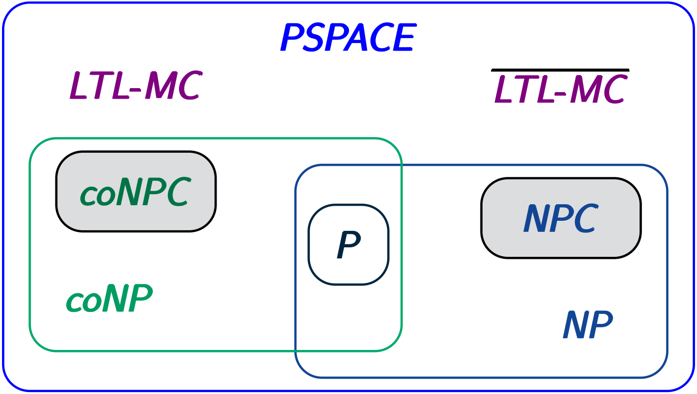
PSPACE = class of decision problems that are decidable in deterministic polynomial space

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