Overview

Introduction
Modelling parallel systems
Linear Time Properties
Regular Properties

**Linear Temporal Logic (LTL)**
- syntax and semantics of LTL
- automata-based LTL model checking
- complexity of LTL model checking

Computation-Tree Logic

Equivalences and Abstraction
main steps of automata-based LTL model checking:

- construction of an NBA $\mathcal{A}$ for $\neg \varphi$
- persistence checking in the product $\mathcal{T} \otimes \mathcal{A}$
main steps of automata-based LTL model checking:

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

  $\mathcal{O}(\exp(|\varphi|))$

- 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 $\neg \varphi$ \[\mathcal{O}(\exp(|\varphi|))\]
persistence checking in the product $\mathcal{T} \otimes \mathcal{A}$ \[\mathcal{O}(\text{size}(\mathcal{T}) \cdot \text{size}(\mathcal{A}))\]
main steps of automata-based LTL model checking:

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

complexity: \(\mathcal{O}(\text{size}(\mathcal{T}) \cdot \exp(|\varphi|))\)
Complexity of LTL model checking

main steps of automata-based LTL model checking:

- Construction of an NBA $\mathcal{A}$ for $\neg \varphi$ \( \mathcal{O}(\exp(|\varphi|)) \)
- Persistence checking in the product $\mathcal{T} \otimes \mathcal{A}$ \( \mathcal{O}(\text{size}(\mathcal{T}) \cdot \text{size}(\mathcal{A})) \)

complexity: \( \mathcal{O}(\text{size}(\mathcal{T}) \cdot \exp(|\varphi|)) \)

The LTL model checking problem is \text{PSPACE}-complete
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 $P$, $NP$

$NP$-complete problems

$NPC = \text{class of } NP$-complete problems
Complexity classes $P$, $NP$

$NPC = \text{class of } NP\text{-complete problems}$

1. $L \in NP$
2. $L$ is $NP$-hard, i.e., $K \leq_{poly} L$ for all $K \in NP$
Complexity classes $P$, $NP$

$NP$-hard problems

$NPC = \text{class of } NP\text{-complete problems}$

(1) $L \in NP$

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

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

complement of $L$

$NP$-hard problems
**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 $P$, $NP$, $coNP$

- $coNP$-hard problems
- $NP$-hard problems

$coNPC = \text{class of } coNP\text{-complete problems}$

$L$ is $coNP$-hard iff $\overline{L}$ is $NP$-hard
Complexity classes $P$, $NP$, $coNP$

$coNP$ is the class of $coNP$-hard problems.

$NP$ is the class of $NP$-hard problems.

$coNP$-hard problems

$NP$-hard problems

$coNPC \quad = \quad \text{class of } coNP\text{-complete problems}$

$L$ is $coNP$-hard iff $\overline{L}$ is $NP$-hard
Complexity classes $P$, $NP$, $coNP$

c$coNP$-hard problems

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

$L$ is $coNP$-hard iff $\overline{L}$ is $NP$-hard
Complexity classes $P$, $NP$, $coNP$

$coNP$-hard problems

$LTL-MC$

$LTL-MC$

$NP$-hard problems

$coNP$  $=\quad$ class of $coNP$-complete problems

$L$ is $coNP$-hard  iff $\overline{L}$ is $NP$-hard
The **LTL** model checking problem is **coNP**-hard
The **LTL** model checking problem is **coNP**-hard proof by a polynomial reduction

\[
\text{Hamilton path problem} \leq_{\text{poly}} \text{complement of the LTL model checking problem}
\]
The **LTL** model checking problem is **coNP**-hard

proof by a polynomial reduction

Hamilton path 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?
The LTL model checking problem is \textit{coNP}-hard proof by a polynomial reduction

Hamilton path problem \[ \leq_{\text{poly}} \] complement of the LTL model checking problem

complement of the LTL model checking problem:

\textit{given}: finite transition system \( \mathcal{I} \), LTL-formula \( \varphi \)

\textit{question}: does \( \mathcal{I} \not\models \varphi \) hold?
The \textbf{LTL} model checking problem is \textit{coNP}-hard

proof by a polynomial reduction

\[ \text{HP} \leq_{\text{poly}} \text{LTL-MC} \]

complement of the \textbf{LTL} model checking problem

complement of the \textbf{LTL} model checking problem:

given: finite transition system \( \mathcal{T} \), LTL-formula \( \varphi \)

question: does \( \mathcal{T} \not\models \varphi \) hold?
Hamilton path problem *HP*

**HP** Hamilton path problem:

*given:* finite directed graph $G$

*question:* does $G$ have a Hamilton path $\exists$, i.e., a path that visits each node exactly once
Hamilton path problem *HP*

*HP* Hamilton path problem:

*given:* finite directed graph $G$

*question:* does $G$ has a Hamilton path $?$, i.e., a path that visits each node exactly once.
Hamilton path problem \( HP \)

\( HP \) Hamilton path problem:

\[ \begin{align*}
given: & \quad \text{finite directed graph } G \\
question: & \quad \text{does } G \text{ has a Hamilton path } ?, \text{ i.e., a} \\
        & \quad \text{path that visits each node exactly once}
\end{align*} \]
Hamilton path problem \( HP \)

**HP** Hamilton path problem:

*given:* finite directed graph \( G \)

*question:* does \( G \) have a Hamilton path \(?\), i.e., a path that visits each node exactly once

---

Diagram:

- A directed graph with nodes and edges that form a Hamilton path.
- A directed graph with nodes and edges that do not form a Hamilton path.

---

"has no Hamilton path"
Hamilton path problem \textit{HP}:

\textit{HP} Hamilton path problem:

\textit{given:} finite directed graph $G$

\textit{question:} does $G$ has a Hamilton path $?$, i.e., a path that visits each node exactly once

\textit{HP} is known to be \textbf{NP}-complete
Polynomial reduction

Hamilton path problem $\leq_{\text{poly}}$ complement of the LTL model checking problem
Polynomial reduction

Hamilton path problem \( \leq_{\text{poly}} \) complement of the LTL model checking problem

finite directed graph \( G \) poly time finite TS \( T \) LTL formula \( \varphi \)
Polynomial reduction

Hamilton path problem $\leq_{\text{poly}}$ complement of the LTL model checking problem

finite directed graph $G \rightarrow$ finite TS $\mathcal{T}$

$G$ has a Hamilton path iff $\mathcal{T} \notsat \varphi$
Polynomial reduction

\[ \text{finite directed graph } G \]

\[ G \text{ has a Hamilton path} \]

\[ \leq_{\text{poly}} \quad \text{poly time} \quad \rightarrow \]

\[ \text{finite TS } T \]

\[ \text{LTL formula } \varphi \]

\[ T \not\models \varphi \]
Polynomial reduction

\[ HP \leq_{poly} LTL-MC \]

- Finite directed graph \( G \) has a Hamilton path iff \( T \not\models \varphi \)
- Node-set \( V \) of \( G \) \( \cong \) states of \( T \)
Polynomial reduction

HP \ \leq_{poly} \ \text{poly time} \ \Rightarrow \ LTL-MC

finite directed graph \( G \)

\( G \) has a Hamilton path

\( \text{finite TS} \ T \)

LTL formula \( \varphi \)

iff

iff

node-set \( V \) of \( G \) \( \cong \) states of \( T \)

\( AP = V \)

35 / 187
**Polynomial reduction**

\[ HP \leq_{\text{poly}} LTL-MC \]

- **finite directed graph** $G$
  - $G$ has a Hamilton path
  - node-set $V$ of $G$ $\cong$ states of $T$
    - $AP = V$
    - additional trap state $t$

- poly time
  - $T \not\models \varphi$

\[ \text{finite TS } T \]

\[ \text{LTL formula } \varphi \]
Polynomial reduction

\[ \begin{align*}
HP & \leq_{poly} LTL-MC \\
\text{finite directed graph } G & \text{ poly time} \\
G \text{ has a Hamilton path} & \iff \\
\text{finite TS } T & \not \models \varphi \\
\text{node-set } V \text{ of } G & \cong \\
\text{states of } T \text{ AP } = V \\
\text{additional trap state } t
\end{align*} \]
Polynomial reduction

\[ HP \leq_{\text{poly}} \text{LTL-MC} \]

- **finite directed graph** \( G \)
- **poly time**
- **finite TS** \( T \)
- **LTL formula** \( \varphi \)

\( G \) has a Hamilton path iff \( T \not\models \varphi \)

**node-set** \( V \) of \( G \) \( \cong \) states of \( T \)

\( \text{AP} = V \)

additional trap state \( t \)

\[ \varphi = \bigwedge_{v \in V} (\Diamond v \land \Box (v \rightarrow \Box \Box \neg v)) \]
Polynomial reduction

\[ HP \leq_{\text{poly}} LTL-MC \]

finite directed graph \( G \)

\( G \) has a Hamilton path

\[ \text{finite TS } T \]

LTL formula \( \varphi \)

iff

\[ T \not\models \varphi \]

node-set \( V \) of \( G \)

\[ V \cong \text{states of } T \]

\[ AP = V \]

additional trap state \( t \)

\[ \varphi = \neg \bigwedge_{v \in V} (\Diamond v \land \Box(v \to \Box\Box\neg v)) \]
We just saw:

The **LTL** model checking problem is \textit{coNP}-hard
We just saw:

The **LTL** model checking problem is \( \text{coNP} \)-hard

We now prove:

The **LTL** model checking problem is \( \text{PSPACE} \)-complete
The complexity class $PSPACE$
The complexity class *PSPACE* is the class of decision problems solvable by a deterministic polynomially space-bounded algorithm.
The complexity class $PSPACE$

$PSPACE = \text{class of decision problems solvable by a deterministic polynomially space-bounded algorithm}$

- $NP \subseteq PSPACE$
The complexity class \textbf{PSPACE} = class of decision problems solvable by a deterministic polynomially space-bounded algorithm.

- \textbf{NP} $\subseteq$ \textbf{PSPACE}

\textbf{DFS}-based analysis of the computation tree of an \textbf{NP}-algorithm.
The complexity class \( PSPACE \)

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

- \( \mathcal{NP} \subseteq PSPACE \)

DFS-based analysis of the computation tree of an \( \mathcal{NP} \)-algorithm

space requirements:

recursion depth \( \preceq \) height of computation tree
The complexity class $PSPACE$ is the 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 \textit{PSPACE} is the class of decision problems solvable by a deterministic polynomially space-bounded algorithm.

- \textit{NP} $\subseteq$ \textit{PSPACE}
- \textit{PSPACE} = co\textit{PSPACE} (holds for any deterministic complexity class)
- \textit{PSPACE} = \textit{NPSPACE} (Savitch’s Theorem)
The complexity class \textit{PSPACE} is the class of decision problems solvable by a deterministic polynomially space-bounded algorithm.

- \textit{NP} \subseteq \textit{PSPACE}
- \textit{PSPACE} = \text{coPSPACE} (holds for any deterministic complexity class)
- \textit{PSPACE} = \textit{NPSPACE} (Savitch’s Theorem)

To prove \( L \in \text{PSPACE} \) it suffices to provide a nondeterministic polynomially space-bounded algorithm for the complement \( \overline{L} \) of \( L \).
$PSPACE = \text{class of decision problems that are decidable in deterministic polynomial space}$
**Complexity classes** $P$, $NP$, $coNP$, $PSPACE$

$PSPACE = \text{class of decision problems that are decidable in deterministic polynomial space}$
Complexity classes $P$, $NP$, $coNP$, $PSPACE$

$PSPACE = \text{class of decision problems that are decidable in deterministic polynomial space}$
**PSPACE** = class of decision problems that are decidable in deterministic polynomial space
PSPACE-completeness
decision problem $L$ is $PSPACE$-complete iff

1. $L \in PSPACE$
2. $L$ is $PSPACE$-hard

for all $K \in PSPACE$
**PSPACE-completeness**

Decision problem \( L \) is **PSPACE**-complete iff

1. \( L \in \text{PSPACE} \)
2. \( L \) is **PSPACE**-hard

for all \( K \in \text{PSPACE} \)

as \( \text{PSPACE} = \text{coPSPACE} \):

\( L \) is **PSPACE**-hard \( \iff \) \( \overline{L} \) is **PSPACE**-hard
decision problem $L$ is $PSPACE$-complete iff

1. $L \in PSPACE$
2. $L$ is $PSPACE$-hard for all $K \in PSPACE$

as $PSPACE = \text{coPSPACE} = \text{NPSPACE}$:

$L$ is $PSPACE$-hard $\iff \overline{L}$ is $PSPACE$-hard

$L \in PSPACE \iff \overline{L} \in \text{NPSPACE}$
**PSPACE-completeness of LTL-MC**

**LTL-MC** LTL model checking problem

“does $\pi \models \varphi$ hold for all paths $\pi$ of $\mathcal{T}$?”

---

$LTL-MC \neq$ complement of $LTL-MC$

“does $\pi \not\models \varphi$ hold for some path $\pi$ of $\mathcal{T}$?”
**PSPACE-completeness of LTL-MC**

\( LTL-MC \) LTL model checking problem

“does \( \pi \models \varphi \) hold for all paths \( \pi \) of \( T \)?”

\( \overline{LTL-MC} \) = complement of \( LTL-MC \)

“does \( \pi \not\models \varphi \) hold for some path \( \pi \) of \( T \)?”

\( \exists LTL-MC \) existential LTL model checking problem

for \( T \) and LTL formula \( \psi = \neg \varphi \)
**PSPACE-completeness of LTL-MC**

**LTL-MC** LTL model checking problem

“does $\pi \models \varphi$ hold for all paths $\pi$ of $T$ ?”

$\overline{\text{LTL-MC}} = \text{complement of LTL-MC}$

“does $\pi \not\models \varphi$ hold for some path $\pi$ of $T$ ?”

$\exists \text{LTL-MC}$ existential LTL model checking problem

for $T$ and LTL formula $\psi = \neg \varphi$

“does $\pi \models \psi$ hold for some path $\pi$ of $T$ ?”
PSPACE-completeness of \textit{LTL-MC}

\begin{itemize}
  \item \textbf{LTL-MC} LTL model checking problem
    \begin{itemize}
      \item “does $\pi \models \varphi$ hold for all paths $\pi$ of $\mathcal{T}$ ?”
    \end{itemize}
  \item $\overline{\text{LTL-MC}} = \text{complement of LTL-MC}$
    \begin{itemize}
      \item “does $\pi \not\models \varphi$ hold for some path $\pi$ of $\mathcal{T}$ ?”
    \end{itemize}
  \item \textbf{∃LTL-MC} existential LTL model checking problem
    \begin{itemize}
      \item for $\mathcal{T}$ and LTL formula $\psi = \neg \varphi$
      \item “does $\pi \models \psi$ hold for some path $\pi$ of $\mathcal{T}$ ?”
    \end{itemize}
\end{itemize}

show: \textbf{∃LTL-MC} \in \textit{NPSPACE}

∃LTL-MC is PSPACE-hard
PSPACE-completeness of LTL-MC

\[ \exists \text{LTL-MC} \in \text{NPSPACE} \implies \text{LTL-MC} \in \text{PSPACE} \]

\[ \exists \text{LTL-MC} \text{ is PSPACE-hard} \]

LTL-MC LTL model checking problem

“does \( \pi \models \varphi \) hold for all paths \( \pi \) of \( \mathcal{T} \)?”

\[ \overline{\text{LTL-MC}} = \text{complement of LTL-MC} \]

“does \( \pi \not\models \varphi \) hold for some path \( \pi \) of \( \mathcal{T} \)?”

\[ \text{∃LTL-MC} \text{ existential LTL model checking problem} \]

for \( \mathcal{T} \) and LTL formula \( \psi = \neg \varphi \)

“does \( \pi \models \psi \) hold for some path \( \pi \) of \( \mathcal{T} \)?”
**PSPACE-completeness of** $LTL-MC$

$LTL-MC$ LTL model checking problem

“does $\pi \models \varphi$ hold for all paths $\pi$ of $T$ ?”

$LTL-MC = \text{complement of } LTL-MC$

“does $\pi \not\models \varphi$ hold for some path $\pi$ of $T$ ?”

$\exists LTL-MC$ existential LTL model checking problem

for $T$ and LTL formula $\psi = \neg \varphi$

“does $\pi \models \psi$ hold for some path $\pi$ of $T$ ?”

show: $\exists LTL-MC \in \text{NPSPACE}$

$\exists LTL-MC$ is $\text{PSPACE}$-hard $\implies$ $LTL-MC$ is $\text{PSPACE}$-hard
Existential LTL model checking problem

**given:** \( \mathcal{T} \) be a finite transition system
\( \varphi \) an LTL formula

**question:** does there exist a path \( \pi \) in \( \mathcal{T} \) with \( \pi \models \varphi \)?
Existential LTL model checking problem

**given:** \( \mathcal{T} \) be a finite transition system

\( \varphi \) an LTL formula

**question:** does there exist a path \( \pi \) in \( \mathcal{T} \) with \( \pi \models \varphi \) ?

**goal:** find a criterion for the existence of a path \( \pi \) in \( \mathcal{T} \) with \( \pi \models \varphi \) that can be checked nondeterministically in poly-space
Existential LTL model checking problem

**given:** \( \mathcal{T} \) be a finite transition system
\( \varphi \) an LTL formula

**question:** does there exist a path \( \pi \) in \( \mathcal{T} \) with \( \pi \models \varphi \)?

**goal:** find a criterion for the existence of a path \( \pi \) in \( \mathcal{T} \) with \( \pi \models \varphi \) that can be checked nondeterministically in poly-space

**idea:** use the GNBA \( \mathcal{G} \) for \( \varphi \)
(constructed by our LTL-2-GNBA algorithm)
Existential LTL model checking

There is a path \( \pi \) in \( \mathcal{T} \) s.t. \( \pi \models \varphi \) ?

finite transition system \( \mathcal{T} \)

LTL formula \( \varphi \)

yes

no
Existential LTL model checking

finite transition system $\mathcal{T}$

$LTL$ formula $\varphi$

existential LTL model checking

check whether there is a path $\pi$ in $\mathcal{T}$ s.t. $\pi \models \varphi$

by a nondeterministic poly-space algorithm

yes

no
Existential LTL model checking

finite transition system $\mathcal{T}$

LTL formula $\varphi$

GNBA $\mathcal{G}$ for $\varphi$

existential LTL model checking
check whether there is a path $\pi$ in $\mathcal{T}$ s.t. $\pi \models \varphi$
by a nondeterministic poly-space algorithm

yes

no
Existential LTL model checking

- **finite transition system** $\mathcal{T}$

- **LTL formula** $\varphi$

- **GNBA** $\mathcal{G}$ for $\varphi$

- **nondeterministic poly-space algorithm**
  - guess an ultimately periodic path $\pi = u_0 \ldots u_{n-1} (u_n \ldots u_{n+m})^\omega$ in $\mathcal{T} \otimes \mathcal{G}$
  - check whether $\pi \models \bigwedge_{F \in \mathcal{F}} \square \diamond F$

- **yes**
- **no**
Recall: elementary formula-sets

closure $cl(\varphi)$:
- set of all subformulas of $\varphi$ and their negations
- $\psi$ and $\neg \neg \psi$ are identified

elementary formula-sets: subsets $B$ of $cl(\varphi)$
- maximal consistent w.r.t. propositional logic
- locally consistent w.r.t. U

For $\varphi = a U (\neg a \land b)$, the elementary sets are:

- $\{ a, b, \neg (\neg a \land b), \varphi \}$
- $\{ a, \neg b, \neg (\neg a \land b), \varphi \}$
- $\{ \neg a, b, \neg a \land b, \varphi \}$

- $\{ a, b, \neg (\neg a \land b), \neg \varphi \}$
- $\{ a, \neg b, \neg (\neg a \land b), \neg \varphi \}$
- $\{ \neg a, b, \neg a \land b, \neg \varphi \}$

- $\{ a, \neg b, \neg (\neg a \land b), \neg \varphi \}$
- $\{ \neg a, b, \neg a \land b, \neg \varphi \}$

- $\{ \neg a, \neg b, \neg (\neg a \land b), \neg \varphi \}$
Recall: GNBA for LTL-formula $\varphi$

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

state space: $Q = \{ B \subseteq \text{cl}(\varphi) : B \text{ is elementary} \}$

initial states: $Q_0 = \{ B \in Q : \varphi \in B \}$

transition relation: for $B \in Q$ and $A \in 2^{AP}$:

if $A \neq B \cap AP$ then $\delta(B, A) = \emptyset$

if $A = B \cap AP$ then $\delta(B, A) = \text{set of all } B' \in Q \text{ s.t.}$

<table>
<thead>
<tr>
<th>$\bigcirc \psi \in B$</th>
<th>iff</th>
<th>$\psi \in B'$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\psi_1 \cup \psi_2 \in B$</td>
<td>iff</td>
<td>$(\psi_2 \in B) \lor (\psi_1 \in B \land \psi_1 \cup \psi_2 \in B')$</td>
</tr>
</tbody>
</table>

acceptance set $F = \{ F_{\psi_1 \cup \psi_2} : \psi_1 \cup \psi_2 \in \text{cl}(\varphi) \}$

where $F_{\psi_1 \cup \psi_2} = \{ B \in Q : \psi_1 \cup \psi_2 \notin B \lor \psi_2 \in B \}$
Criterion for existential LTL properties

There exists a path $\pi$ in $T$ with $\pi \models \varphi$ iff there exist
There exists a path $\pi$ in $\mathcal{T}$ with $\pi \models \varphi$ iff there exist

- an initial finite path fragment $s_0 \ldots s_n \ldots s_{n+m}$ in $\mathcal{T}$
- a run $B_0 B_1 \ldots B_{n+1} \ldots B_{n+m+1}$ in $\mathcal{G}$ for the word $\text{trace}(s_0 s_1 \ldots s_n \ldots s_{n+m})$
Criterion for existential LTL properties

There exists a path \( \pi \) in \( \mathcal{T} \) with \( \pi \models \varphi \) iff there exist

- an initial finite path fragment \( s_0 \ldots s_n \ldots s_{n+m} \) in \( \mathcal{T} \)
- a run \( B_0 B_1 \ldots B_{n+1} \ldots B_{n+m+1} \) in \( \mathcal{G} \) for the word \( \text{trace}(s_0 s_1 \ldots s_n \ldots s_{n+m}) \)

such that

\[
(1) \quad \langle s_n, B_{n+1} \rangle = \langle s_{n+m}, B_{n+m+1} \rangle
\]
Criterion for existential LTL properties

There exists a path $\pi$ in $\mathcal{T}$ with $\pi \models \varphi$ iff there exist

- an initial finite path fragment $s_0 \ldots s_n \ldots s_{n+m}$ in $\mathcal{T}$
- a run $B_0 B_1 \ldots B_{n+1} \ldots B_{n+m+1}$ in $\mathcal{G}$ for the word $\text{trace}(s_0 s_1 \ldots s_n \ldots s_{n+m})$

such that

1. $\langle s_n, B_{n+1} \rangle = \langle s_{n+m}, B_{n+m+1} \rangle$
2. whenever $\psi_1 \cup \psi_2 \in B_{n+1} \cup \ldots \cup B_{n+m}$ then $\psi_2 \in B_{n+1} \cup \ldots \cup B_{n+m}$
Criterion for existential LTL properties

There exists a path $\pi$ in $\mathcal{T}$ with $\pi \models \varphi$ iff there exist

- an initial finite path fragment $s_0 \ldots s_n \ldots s_{n+m}$ in $\mathcal{T}$
- a run $B_0 B_1 \ldots B_{n+1} \ldots B_{n+m+1}$ in $\mathcal{G}$ for the word $\text{trace}(s_0 s_1 \ldots s_n \ldots s_{n+m})$

such that

1. $\langle s_n, B_{n+1} \rangle = \langle s_{n+m}, B_{n+m+1} \rangle$
2. whenever $\psi_1 \cup \psi_2 \in B_{n+1} \cup \ldots \cup B_{n+m}$ then $\psi_2 \in B_{n+1} \cup \ldots \cup B_{n+m}$
3. $n < |S| \cdot 2^{|\text{cl}(\varphi)|}$
There exists a path $\pi$ in $\mathcal{T}$ with $\pi \models \varphi$ iff there exist

- an initial finite path fragment $s_0 \ldots s_n \ldots s_{n+m}$ in $\mathcal{T}$
- a run $B_0 B_1 \ldots B_{n+1} \ldots B_{n+m+1}$ in $\mathcal{G}$ for the word $\text{trace}(s_0 s_1 \ldots s_n \ldots s_{n+m})$

such that

(1) $\langle s_n, B_{n+1} \rangle = \langle s_{n+m}, B_{n+m+1} \rangle$

(2) whenever $\psi_1 \cup \psi_2 \in B_{n+1} \cup \ldots \cup B_{n+m}$ then $\psi_2 \in B_{n+1} \cup \ldots \cup B_{n+m}$

(3) $n < |S| \cdot 2^{|\text{cl}(\varphi)|}$ and $m \leq |S| \cdot 2^{|\text{cl}(\varphi)|} \cdot |\varphi|$
NPSPACE-algorithm for $\exists\text{LTL-MC}$

The existential LTL model checking problem

given: finite TS $\mathcal{T}$, LTL formula $\varphi$

question: is there a path $\pi \in \text{Paths}(\mathcal{T})$ with $\pi \models \varphi$?
The existential LTL model checking problem

given: finite TS $\mathcal{I}$, LTL formula $\varphi$

question: is there a path $\pi \in \text{Paths}(\mathcal{I})$ with $\pi \models \varphi$?

is solvable by a nondeterministic polynomially space-bounded algorithm:
The existential LTL model checking problem

given: finite TS $\mathcal{T}$, LTL formula $\varphi$

question: is there a path $\pi \in \text{Paths}(\mathcal{T})$ with $\pi \models \varphi$?

is solvable by a nondeterministic polynomially space-bounded algorithm:

- guess nondeterministically an ultimatively periodic path $\pi = u_0 u_1 \ldots u_{n-1}(u_n \ldots u_{n+m})^\omega$ of $\mathcal{T} \otimes \mathcal{G}$

$\text{GNBA for } \varphi$ obtained by our LTL-2-GNBA algorithm
The existential LTL model checking problem

given: finite TS \( \mathcal{T} \), LTL formula \( \varphi \)

question: is there a path \( \pi \in \text{Paths}(\mathcal{T}) \) with \( \pi \models \varphi \) ?

is solvable by a nondeterministic polynomially space-bounded algorithm:

- guess nondeterministically an ultimately periodic path \( \pi = u_0 u_1 \ldots u_{n-1}(u_n \ldots u_{n+m})^\omega \) of \( \mathcal{T} \otimes \mathcal{G} \)

  GNBA for \( \varphi \) obtained by our LTL-2-GNBA algorithm

- check whether the guessed path meets the acceptance condition of \( \mathcal{G} \)
NPSPACE-algorithm for $\exists LTL$-MC

guess two natural numbers $n, m \leq k$ s.t. $m \geq 1$
where $k = |S| \cdot 2^{cl(\varphi)} \cdot |\varphi|$
NPSPACE-algorithm for $\exists\text{LTL-MC}$

guess two natural numbers $n, m \leq k$ s.t. $m \geq 1$

where $k = |S| \cdot 2^{|\text{cl}(\varphi)|} \cdot |\varphi|$

guess $s_0 \ldots s_n \ldots s_{n+m} \in \text{Paths}_{\text{fin}}(T)$
**NPSPACE-algorithm for ∃LTL-MC**

guess two natural numbers $n, m \leq k$ s.t. $m \geq 1$

where $k = |S| \cdot 2^{cl(\varphi)} \cdot |\varphi|$

guess $s_0 \ldots s_n \ldots s_{n+m} \in \text{Paths}_{\text{fin}}(T)$

guess $n+m+2$ subsets $B_0, \ldots, B_n, \ldots, B_{n+m+1}$ of $cl(\varphi)$
**NPSPACE-algorithm for \( \exists LTL-MC \)**

guess two natural numbers \( n, m \leq k \) s.t. \( m \geq 1 \)
where \( k = |S| \cdot 2^{|cl(\varphi)|} \cdot |\varphi| \)

guess \( s_0 \ldots s_n \ldots s_{n+m} \in Paths_{\text{fin}}(T) \)

guess \( n+m+2 \) subsets \( B_0, \ldots, B_n, \ldots, B_{n+m+1} \) of \( cl(\varphi) \)
check whether the following three conditions hold:
NPSPACE-algorithm for $\exists LTL$-$MC$

guess two natural numbers $n, m \leq k$ s.t. $m \geq 1$

where $k = |S| \cdot 2^{\text{cl}(\varphi)} \cdot |\varphi|$

guess $s_0 \ldots s_n \ldots s_{n+m} \in \text{Paths}_{\text{fin}}(T)$

guess $n+m+2$ subsets $B_0, \ldots, B_n, \ldots, B_{n+m+1}$ of $\text{cl}(\varphi)$

check whether the following three conditions hold:

- $\langle s_n, B_{n+1} \rangle = \langle s_{n+m}, B_{n+m+1} \rangle$
**NPSPACE-algorithm for ∃LTL-MC**

guess two natural numbers $n, m \leq k$ s.t. $m \geq 1$

where $k = |S| \cdot 2^{|\text{cl}(\varphi)|} \cdot |\varphi|$

guess $s_0 \ldots s_n \ldots s_{n+m} \in \text{Paths}_{\text{fin}}(T)$

guess $n+m+2$ subsets $B_0, \ldots, B_n, \ldots, B_{n+m+1}$ of $\text{cl}(\varphi)$

check whether the following three conditions hold:

- $\langle s_n, B_{n+1} \rangle = \langle s_{n+m}, B_{n+m+1} \rangle$
- $B_0 \ldots B_n \ldots B_{n+m+1}$ is an initial run for $\text{trace}(s_0 \ldots s_n \ldots s_{n+m+1})$ in GNBA $G$
NPSPACE-algorithm for ∃LTL-MC

guess two natural numbers \( n, m \leq k \) s.t. \( m \geq 1 \)
where \( k = |S| \cdot 2^{cl(\varphi)} \cdot |\varphi| \)

guess \( s_0 \ldots s_n \ldots s_{n+m} \in \text{Paths}_{\text{fin}}(T) \)

guess \( n+m+2 \) subsets \( B_0, \ldots, B_n, \ldots, B_{n+m+1} \) of \( cl(\varphi) \)

check whether the following three conditions hold:

- \( \langle s_n, B_{n+1} \rangle = \langle s_{n+m}, B_{n+m+1} \rangle \)
- \( B_0 \ldots B_n \ldots B_{n+m+1} \) is an initial run for \( \text{trace}(s_0 \ldots s_n \ldots s_{n+m+1}) \) in GNBA \( G \)
- \( \{ \psi_2 : \psi_1 \cup \psi_2 \in \bigcup_{n<i \leq n+m} B_i \} \subseteq \bigcup_{n<i \leq n+m} B_i \)
NPSPACE-algorithm for $\exists\text{LTL-MC}$

guess two natural numbers $n, m \leq k$ s.t. $m \geq 1$
where $k = |S| \cdot 2^{|cl(\phi)|} \cdot |\phi|$

guess $s_0 \ldots s_n \ldots s_{n+m} \in \text{Paths}_{\text{fin}}(T)$
guess $n+m+2$ subsets $B_0, \ldots, B_n, \ldots, B_{n+m+1}$ of $cl(\phi)$

check whether the following three conditions hold:

- $\langle s_n, B_{n+1} \rangle = \langle s_{n+m}, B_{n+m+1} \rangle$
- $B_0 \ldots B_n \ldots B_{n+m+1}$ is an initial run for $\text{trace}(s_0 \ldots s_n \ldots s_{n+m+1})$ in GNBA $G$
- $\{ \psi_2 : \psi_1 \cup \psi_2 \in \bigcup_{n<i \leq n+m} B_i \} \subseteq \bigcup_{n<i \leq n+m} B_i$

If so then return “yes”. Otherwise return “no”.

We saw that:

The existential LTL model checking problem

given: finite TS $\mathcal{T}$, LTL formula $\varphi$

question: is there a path $\pi$ in $\mathcal{T}$ with $\pi \models \varphi$?

belongs to $NPSPACE$
We saw that:

The existential LTL model checking problem

given: finite TS $\mathcal{T}$, LTL formula $\varphi$

question: is there a path $\pi$ in $\mathcal{T}$ with $\pi \models \varphi$ ?

belongs to $NPSPACE = PSPACE$
PSPACE-completeness of $\exists LTL$-MC

We saw that:

The existential LTL model checking problem

given: finite TS $\mathcal{T}$, LTL formula $\varphi$

question: is there a path $\pi$ in $\mathcal{T}$ with $\pi \models \varphi$?

belongs to $NPSPACE = PSPACE$

It remains to prove the $PSPACE$-hardness
we show that for all problems $K \in \text{PSPACE}$:

$$K \leq_{\text{poly}} \exists \text{LTL-MC}$$
we show that for all problems $K \in \text{PSPACE}$:

$$K \leq_{\text{poly}} \exists\text{LTL-MC}$$

Let

- $\mathcal{M}$ be a deterministic Turing machine (DTM) that decides $K$,
- $P$ a polynomial

such that $\mathcal{M}$ started with an input word $w$ visits at most $P(|w|)$ tape cells
we show that for all problems \( K \in PSPACE \):

\[
K \leq_{\text{poly}} \exists \mathsf{LTL-MC}
\]

Given DTM \( \mathcal{M} \) that decides \( K \) with polynomial space bound \( P(n) \), provide a polynomial reduction:

| input word \( w \) for \( \mathcal{M} \) | poly time | TS \( \mathcal{T} \) | LTL-formula \( \varphi \) |
we show that for all problems $K \in PSPACE$:

$$K \leq_{\text{poly}} \exists\text{LTL-MC}$$

Given DTM $\mathcal{M}$ that decides $K$ with polynomial space bound $P(n)$, provide a polynomial reduction:

\[\text{input word } w \text{ for } \mathcal{M} \quad \xrightarrow{\text{poly time}} \quad \text{TS } \mathcal{I} \quad \text{LTL-formula } \varphi\]

$\mathcal{M}$ accepts $w$, i.e., $w \in K$ iff there is path $\pi$ of $\mathcal{I}$ with $\pi \models \varphi$
Polynomial reduction $w \mapsto (T, \varphi)$

DTM $M$ visits at the most the tape cells $1, 2, ..., P(n)$ for inputs of length $n$ (where $P$ is a polynomial)
Polynomial reduction $w \mapsto (T, \varphi)$

DTM $M$ visits at the most the tape cells $1, 2, \ldots, P(n)$ for inputs of length $n$ (where $P$ is a polynomial)

$$
\begin{array}{cccccccc}
$ & A_1 & \ldots & A_{n-1} & A_n & \square & \square & \ldots & \square & \ldots \\
1 & 2 & n & n+1 & & & & & & \\
\end{array}
$$

initial tape configuration for input $w = A_1 A_2 \ldots A_n$
Polynomial reduction $w \mapsto (\mathcal{I}, \varphi)$

DTM $\mathcal{M}$ visits at the most the tape cells $1, 2, \ldots, P(n)$ for inputs of length $n$ (where $P$ is a polynomial)

<table>
<thead>
<tr>
<th>$$$</th>
<th>$A_1$</th>
<th>$\ldots$</th>
<th>$A_{n-1}$</th>
<th>$A_n$</th>
<th>$\square$</th>
<th>$\square$</th>
<th>$\ldots$</th>
<th>$\square$</th>
<th>$\ldots$</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>2</td>
<td>$n$</td>
<td>$n+1$</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

initial tape configuration for input $w = A_1 A_2 \ldots A_n$

$\square \equiv$ blank symbol of $\mathcal{M}$

$\$$ \equiv$ symbol for the left border of the tape
Polynomial reduction $w \mapsto (T, \varphi)$

DTM $M$ visits at the most the tape cells $1, 2, \ldots, P(n)$ for inputs of length $n$ (where $P$ is a polynomial)

| $|$ | $A_1$ | $\ldots$ | $A_{n-1}$ | $A_n$ | $\blacksquare$ | $\blacksquare$ | $\ldots$ | $\blacksquare$ | $\blacksquare$ | $\ldots$ |
|-----|-------|-----------|-----------|-------|----------------|----------------|-----------|----------------|----------------|-----------|
| 1   | 2     | $n$       | $n+1$     | $P(n)$|

Initial tape configuration for input $w = A_1 A_2 \ldots A_n$

$\blacksquare \equiv$ blank symbol of $M$

$\$ \equiv symbol for the left border of the tape

w.l.o.g. $P(n) > n$
Polynomial reduction $w \mapsto (I, \varphi)$

DTM $M$ visits at the most the tape cells $1, 2, \ldots, P(n)$ for inputs of length $n$ (where $P$ is a polynomial)

<table>
<thead>
<tr>
<th>$$</th>
<th>$A_1$</th>
<th>$\ldots$</th>
<th>$A_{n-1}$</th>
<th>$A_n$</th>
<th>$\square$</th>
<th>$\square$</th>
<th>$\ldots$</th>
<th>$\square$</th>
<th>$\ldots$</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>2</td>
<td>$n$</td>
<td>$n+1$</td>
<td>$P(n)$</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

initial tape configuration for input $w = A_1 A_2 \ldots A_n$

$\square \triangleq$ blank symbol of $M$

$\$ \triangleq$ symbol for the left border of the tape

w.l.o.g. $P(n) > n$
Polynomial reduction $w \mapsto (\mathcal{T}, \varphi)$

DTM $\mathcal{M}$ visits at most the tape cells $1, 2, \ldots, P(n)$ for inputs of length $n$ (where $P$ is a polynomial)

\[
\begin{array}{cccccccc}
\$ & A_1 & \ldots & A_{n-1} & A_n & \| & \| & \ldots & \| & \ldots \\
1 & 2 & n & n+1 & P(n) & \uparrow & \text{not visited}
\end{array}
\]

TS $\mathcal{T}$:
Polynomial reduction $w \mapsto (\mathcal{T}, \varphi)$

DTM $\mathcal{M}$ visits at the most the tape cells $1, 2, \ldots, P(n)$ for inputs of length $n$ (where $P$ is a polynomial)

$\begin{array}{cccccccc}
\$ & A_1 & \ldots & A_{n-1} & A_n & \_ & \_ & \ldots & \_ & \ldots \\
1 & 2 & n & n+1 & P(n) & \uparrow & \\
\end{array}$

TS $\mathcal{T}$:

states of $\mathcal{T}$: $0, 1, \ldots, P(n)$,
Polynomial reduction $w \mapsto (\mathcal{T}, \varphi)$

DTM $M$ visits at the most the tape cells $1, 2, \ldots, P(n)$ for inputs of length $n$ (where $P$ is a polynomial)

| $|$ | $A_1$ | $\ldots$ | $A_{n-1}$ | $A_n$ | $\sqcup$ | $\sqcup$ | $\ldots$ | $\sqcup$ | $\ldots$ |
|---|---|---|---|---|---|---|---|---|---|
| 1 | 2 | $n$ | $n+1$ | $P(n)$ | $\uparrow$ |

TS $T$:

states of $T$: $0, 1, \ldots, P(n), \langle q, A, i \rangle, \langle *, A, i \rangle$
Polynomial reduction \( w \mapsto (\mathcal{I}, \varphi) \)

DTM \( \mathcal{M} \) visits at the most the tape cells \( 1, 2, \ldots, P(n) \) for inputs of length \( n \) (where \( P \) is a polynomial)

<table>
<thead>
<tr>
<th>$</th>
<th>A_1</th>
<th>\ldots</th>
<th>A_{n-1}</th>
<th>A_n</th>
<th>\square</th>
<th>\square</th>
<th>\ldots</th>
<th>\square</th>
<th>\ldots</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>2</td>
<td>( n )</td>
<td>( n+1 )</td>
<td>( \ldots )</td>
<td>( P(n) )</td>
<td>( \uparrow )</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

TS \( \mathcal{I} \):

states of \( \mathcal{I} \): \( 0, 1, \ldots, P(n), \langle q, A, i \rangle, \langle *, A, i \rangle \)

where \( q \) is a state of \( \mathcal{M} \), \( A \) a tape symbol, \( 1 \leq i \leq P(n) \)
Polynomial reduction $w \mapsto (T, \varphi)$

DTM $M$ visits at the most the tape cells $1, 2, \ldots, P(n)$ for inputs of length $n$ (where $P$ is a polynomial)

<table>
<thead>
<tr>
<th>$$</th>
<th>$A_1$</th>
<th>$\ldots$</th>
<th>$A_{n-1}$</th>
<th>$A_n$</th>
<th>$\downarrow$</th>
<th>$\downarrow$</th>
<th>$\ldots$</th>
<th>$\downarrow$</th>
<th>$\ldots$</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>2</td>
<td>$n$</td>
<td>$n+1$</td>
<td>$P(n)$</td>
<td>↑</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

idea: TS $T$ encodes each configuration of $M$ by a path fragment from state $0$ to state $P(n)$
Polynomial reduction $w \mapsto (T, \varphi)$

<table>
<thead>
<tr>
<th>A</th>
<th>B</th>
<th>C</th>
<th>...</th>
<th>D</th>
<th>E</th>
<th>...</th>
<th>C</th>
<th>...</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>2</td>
<td>3</td>
<td>(P(n))</td>
<td>↑</td>
<td>not visited</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

TS $\mathcal{T}$:
Polynomial reduction $w \mapsto (\mathcal{T}, \varphi)$

<table>
<thead>
<tr>
<th>A</th>
<th>B</th>
<th>C</th>
<th>...</th>
<th>D</th>
<th>E</th>
<th>...</th>
<th>C</th>
<th>...</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>2</td>
<td>3</td>
<td>$i$</td>
<td>$P(n)$</td>
<td>↑</td>
<td>not visited</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

TS $\mathcal{T}$:

State $q$
Polynomial reduction $w \mapsto (\mathcal{I}, \varphi)$

### Table

<table>
<thead>
<tr>
<th>A</th>
<th>B</th>
<th>C</th>
<th>...</th>
<th>D</th>
<th>E</th>
<th>...</th>
<th>C</th>
<th>...</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>2</td>
<td>3</td>
<td></td>
<td>$i$</td>
<td></td>
<td>$P(n)$</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**TS $\mathcal{I}$:**

Suppose $\delta(q, D) = (p, B, +1)$
Polynomial reduction \( w \mapsto (T, \varphi) \)

**TS** \( T \): 

```
A B C ... D E ... C
1 2 3 i P(n)
```

Path fragment for the configuration \( ABC \ldots q D \ldots C \)
Polynomial reduction \( w \mapsto (\mathcal{T}, \varphi) \)

<table>
<thead>
<tr>
<th>A</th>
<th>B</th>
<th>C</th>
<th>...</th>
<th>D</th>
<th>E</th>
<th>...</th>
<th>C</th>
<th>...</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>2</td>
<td>3</td>
<td>i</td>
<td>P(n)</td>
<td></td>
<td>↑</td>
<td></td>
<td>not visited</td>
</tr>
</tbody>
</table>

**TS \( \mathcal{T} \):**

0 \( \langle *, A, 1 \rangle \)

Path fragment for the configuration \( ABC \ldots q D \ldots C \)
Polynomial reduction $w \mapsto (\mathcal{T}, \varphi)$

<table>
<thead>
<tr>
<th>A</th>
<th>B</th>
<th>C</th>
<th>...</th>
<th>D</th>
<th>E</th>
<th>...</th>
<th>C</th>
<th>...</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>2</td>
<td>3</td>
<td></td>
<td>i</td>
<td></td>
<td></td>
<td>P(n)</td>
<td>↑</td>
</tr>
</tbody>
</table>

not visited

state $q$

TS $\mathcal{T}$:

0 ⟨∗, A, 1⟩ 1

path fragment for the configuration $ABC\ldots q D\ldots C$
Polynomial reduction $w \mapsto (\mathcal{I}, \varphi)$

TS $\mathcal{I}$:

$$0 \langle *, A, 1 \rangle 1 \langle *, B, 2 \rangle$$

path fragment for the configuration $ABC \ldots q D \ldots C$
Polynomial reduction $w \mapsto (\mathcal{T}, \varphi)$

**TS $\mathcal{T}$:**

0 ⟨∗, $A$, 1⟩ 1 ⟨∗, $B$, 2⟩ 2

path fragment for the configuration $ABCD\ldots qD\ldots C$
Polynomial reduction \( w \mapsto (\mathcal{T}, \varphi) \)

\[
\begin{array}{ccccccc}
A & B & C & \ldots & D & E & \ldots & C & \ldots \\
1 & 2 & 3 & i & P(n) & \uparrow & \text{not visited} \\
\end{array}
\]

TS \( \mathcal{T} \):

0 \langle *, A, 1 \rangle 1 \langle *, B, 2 \rangle 2 \ldots (i-1)

path fragment for the configuration \( ABC \ldots q D \ldots C \)
Polynomial reduction $w \mapsto (\mathcal{T}, \varphi)$

<table>
<thead>
<tr>
<th>A</th>
<th>B</th>
<th>C</th>
<th>...</th>
<th>D</th>
<th>E</th>
<th>...</th>
<th>C</th>
<th>...</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>2</td>
<td>3</td>
<td></td>
<td>i</td>
<td></td>
<td></td>
<td>P(n)</td>
<td>↑</td>
</tr>
</tbody>
</table>

TS $\mathcal{T}$:

state $q$

0 $\langle \ast, A, 1 \rangle$ 1 $\langle \ast, B, 2 \rangle$ 2 $\ldots$ $(i-1) \langle q, D, i \rangle$

path fragment for the configuration $ABC \ldots q D \ldots C$
Polynomial reduction $w \mapsto (\mathcal{T}, \varphi)$

<table>
<thead>
<tr>
<th>A</th>
<th>B</th>
<th>C</th>
<th>...</th>
<th>D</th>
<th>E</th>
<th>...</th>
<th>C</th>
<th>...</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>2</td>
<td>3</td>
<td></td>
<td>$i$</td>
<td></td>
<td></td>
<td>$P(n)$</td>
<td></td>
</tr>
</tbody>
</table>

not visited

state $q$

TS $\mathcal{T}$:

path fragment for the configuration $ABC ... q D ... C$
**Polynomial reduction** \( w \mapsto (\mathcal{I}, \varphi) \)

\[
\begin{array}{ccccccc}
A & B & C & \ldots & D & E & \ldots & C & \ldots \\
1 & 2 & 3 & & i & P(n) & & & \\
\end{array}
\]

**TS \( \mathcal{I} \):**

0 \( \langle *, A, 1 \rangle \) 1 \( \langle *, B, 2 \rangle \) 2 \( \ldots \) \( (i-1) \langle q, D, i \rangle \) \( i \) \( \ldots \) \( P(n) \)

suppose \( \delta(q, D) = (p, B, +1) \)
Polynomial reduction $w \mapsto (\mathcal{T}, \varphi)$

<table>
<thead>
<tr>
<th></th>
<th>A</th>
<th>B</th>
<th>C</th>
<th>...</th>
<th>B</th>
<th>E</th>
<th>...</th>
<th>C</th>
<th>...</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>2</td>
<td>3</td>
<td>i</td>
<td>i+1</td>
<td>P(n)</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

not visited

**TS \( \mathcal{T} \):**

\[
\begin{align*}
0 \langle *, A, 1 \rangle & \quad 1 \langle *, B, 2 \rangle & \quad 2 \cdots & \quad \langle q, D, i \rangle & \quad i \langle *, E, i+1 \rangle & \quad \cdots & \quad P(n) \\
0 & & & & & & \\
\end{align*}
\]

suppose $\delta(q, D) = (p, B, +1)$
Polynomial reduction $w \mapsto (\mathcal{T}, \varphi)$

<table>
<thead>
<tr>
<th>A</th>
<th>B</th>
<th>C</th>
<th>...</th>
<th>B</th>
<th>E</th>
<th>...</th>
<th>C</th>
<th>...</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>2</td>
<td>3</td>
<td></td>
<td>i</td>
<td>i+1</td>
<td></td>
<td>P(n)</td>
<td>↑</td>
</tr>
</tbody>
</table>

not visited

TS $\mathcal{T}$:

0 $\langle *, A, 1 \rangle$
1 $\langle *, B, 2 \rangle$
2 $\langle q, D, i \rangle$
i $\langle *, E, i+1 \rangle$
... $P(n)$

0 $\langle *, A, 1 \rangle$
Polynomial reduction $w \mapsto (\mathcal{I}, \varphi)$

TS $\mathcal{I}$:

$0 \langle *, A, 1 \rangle 1 \langle *, B, 2 \rangle 2 \ldots \langle q, D, i \rangle i \langle *, E, i+1 \rangle \ldots P(n)$

0 $\langle *, A, 1 \rangle$ 1
Polynomial reduction $w \mapsto (\mathcal{T}, \varphi)$

<table>
<thead>
<tr>
<th>A</th>
<th>B</th>
<th>C</th>
<th>...</th>
<th>B</th>
<th>E</th>
<th>...</th>
<th>C</th>
<th>...</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>2</td>
<td>3</td>
<td></td>
<td>$i$</td>
<td>$i+1$</td>
<td></td>
<td>$P(n)$</td>
<td></td>
</tr>
</tbody>
</table>

not visited

**TS $\mathcal{T}$:**

0 \langle*, A, 1\rangle 1 \langle*, B, 2\rangle 2 \ldots \langle q, D, i\rangle i \langle*, E, i+1\rangle \ldots P(n)

0 \langle*, A, 1\rangle 1 \langle*, B, 2\rangle
Polynomial reduction $w \mapsto (\mathcal{T}, \varphi)$

<table>
<thead>
<tr>
<th>A</th>
<th>B</th>
<th>C</th>
<th>...</th>
<th>B</th>
<th>E</th>
<th>...</th>
<th>C</th>
<th>...</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>2</td>
<td>3</td>
<td></td>
<td>$i$</td>
<td>$i+1$</td>
<td></td>
<td>$P(n)$</td>
<td></td>
</tr>
</tbody>
</table>

TS $\mathcal{T}$:

$0 \langle *, A, 1 \rangle 1 \langle *, B, 2 \rangle 2 \ldots \langle q, D, i \rangle i \langle *, E, i+1 \rangle \ldots P(n)

0 \langle *, A, 1 \rangle 1 \langle *, B, 2 \rangle 2$
Polynomial reduction $w \mapsto (\mathcal{T}, \varphi)$

<table>
<thead>
<tr>
<th>A</th>
<th>B</th>
<th>C</th>
<th>...</th>
<th>B</th>
<th>E</th>
<th>...</th>
<th>C</th>
<th>...</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>2</td>
<td>3</td>
<td></td>
<td>$i$</td>
<td>$i+1$</td>
<td></td>
<td>$P(n)$</td>
<td></td>
</tr>
</tbody>
</table>

TS $\mathcal{T}$:

0 $\langle *, A, 1 \rangle$ 1 $\langle *, B, 2 \rangle$ 2 ... $\langle q, D, i \rangle$ $i$ $\langle *, E, i+1 \rangle$ ... $P(n)$

0 $\langle *, A, 1 \rangle$ 1 $\langle *, B, 2 \rangle$ 2 ... $\langle *, B, i \rangle$ $i$
Polynomial reduction $w \mapsto (\mathcal{T}, \varphi)$

\[
\begin{array}{cccccccc}
A & B & C & \ldots & B & E & \ldots & C & \ldots \\
1 & 2 & 3 & i & i+1 & P(n) & \uparrow & \text{not visited} \\
\end{array}
\]

TS $\mathcal{T}$:

0 $\langle *, A, 1 \rangle$ 1 $\langle *, B, 2 \rangle$ 2 $\ldots$ $\langle q, D, i \rangle$ $i$ $\langle *, E, i+1 \rangle$ $\ldots$ $P(n)$

0 $\langle *, A, 1 \rangle$ 1 $\langle *, B, 2 \rangle$ 2 $\ldots$ $\langle *, B, i \rangle$ $i$ $\langle p, E, i+1 \rangle$
Polynomial reduction \( w \mapsto (\mathcal{T}, \varphi) \)

<table>
<thead>
<tr>
<th>A</th>
<th>B</th>
<th>C</th>
<th>...</th>
<th>B</th>
<th>E</th>
<th>...</th>
<th>C</th>
<th>...</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>2</td>
<td>3</td>
<td>( i )</td>
<td>( i+1 )</td>
<td>( P(n) )</td>
<td>( \uparrow )</td>
<td>not visited</td>
<td></td>
</tr>
</tbody>
</table>

**TS \( \mathcal{T} \):**

0 \( \langle *, A, 1 \rangle \) 1 \( \langle *, B, 2 \rangle \) 2 ... \( \langle q, D, i \rangle \) \( i \) \( \langle *, E, i+1 \rangle \) ... \( P(n) \)

0 \( \langle *, A, 1 \rangle \) 1 \( \langle *, B, 2 \rangle \) 2 ... \( \langle *, B, i \rangle \) \( i \) \( \langle p, E, i+1 \rangle \) ...
Polynomial reduction $w \mapsto (\mathcal{T}, \varphi)$

<table>
<thead>
<tr>
<th>A</th>
<th>B</th>
<th>C</th>
<th>...</th>
<th>B</th>
<th>E</th>
<th>...</th>
<th>C</th>
<th>...</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>2</td>
<td>3</td>
<td>$i$</td>
<td>$i+1$</td>
<td>$P(n)$</td>
<td>↑</td>
<td>not visited</td>
<td></td>
</tr>
</tbody>
</table>

**TS $\mathcal{T}$:**

0 $\langle *, A, 1 \rangle$ 1 $\langle *, B, 2 \rangle$ 2 $\ldots$ $\langle q, D, i \rangle$ $i$ $\langle *, E, i+1 \rangle$ $\ldots$ $P(n)$

0 $\langle *, A, 1 \rangle$ 1 $\langle *, B, 2 \rangle$ 2 $\ldots$ $\langle *, B, i \rangle$ $i$ $\langle p, E, i+1 \rangle$ $\ldots$ $P(n)$
Polynomial reduction \( w \mapsto (\mathcal{T}, \varphi) \)

Let \( M \) be a DTM with polynomial space bound \( P(n) \)

- state space \( Q \)
- initial state \( q_0 \)
- set of accept states \( F \)
- tape alphabet \( \Gamma \)
- input alphabet \( \Sigma \subseteq \Gamma \)
- blank symbol \( \sqcup \)

transition function \( \delta : Q \times \Gamma \rightarrow Q \times \Gamma \times \{-1, 0, +1\} \)
Polynomial reduction $w \mapsto (T, \varphi)$

Let $\mathcal{M}$ be a DTM with polynomial space bound $P(n)$

- state space $Q$
- initial state $q_0$
- set of accept states $F$
- tape alphabet $\Gamma$
- input alphabet $\Sigma \subseteq \Gamma$
- blank symbol $\sqcup$

transition function $\delta : Q \times \Gamma \rightarrow Q \times \Gamma \times \{-1, 0, +1\}$

input word $w$ for $\mathcal{M}$

poly time

TS $T$

LTL-formula $\varphi$

$\mathcal{M}$ accepts $w$, i.e., $w \in K$

iff

there is path $\pi$ of $T$

with $\pi \models \varphi$
Polynomial reduction \( w \mapsto (\mathcal{T}, \varphi) \)

Let \( \mathcal{M} = (Q, \Sigma, \Gamma, \delta, q_0, \cup, F) \) be a DTM with polynomial space bound \( P(n) \), and \( w \in \Sigma^* \), \(|w| = n\). Transition system \( \mathcal{T} \overset{\text{def}}{=} (S, \text{Act}, \rightarrow, S_0, \text{AP}, L) \) where
Polynomial reduction $\mathcal{M} = (Q, \Sigma, \Gamma, \delta, q_0, \sqcup, F)$ be a DTM with polynomial space bound $P(n)$, and $w \in \Sigma^*$, $|w| = n$.

Transition system $\mathcal{T} \overset{\text{def}}{=} (S, \text{Act}, \rightarrow, S_0, \text{AP}, L)$ where

$$S = \{0, 1, \ldots, P(n)\} \cup \{\langle q, A, i \rangle, \langle *, A, i \rangle : q \in Q, A \in \Gamma, 1 \leq i \leq P(n)\}$$
Polynomial reduction $w \mapsto (\mathcal{T}, \varphi)$

Let $\mathcal{M} = (Q, \Sigma, \Gamma, \delta, q_0, \sqcup, F)$ be a DTM with polynomial space bound $P(n)$, and $w \in \Sigma^*$, $|w| = n$.

Transition system $\mathcal{T} \overset{\text{def}}{=} (S, Act, \rightarrow, S_0, AP, L)$ where

$S = \{0, 1, \ldots, P(n)\} \cup \{\langle q, A, i \rangle, \langle *, A, i \rangle : q \in Q, A \in \Gamma, 1 \leq i \leq P(n)\}$

$S_0 = \{0\}$
Polynomial reduction $w \mapsto (\mathcal{T}, \varphi)$

Let $\mathcal{M} = (Q, \Sigma, \Gamma, \delta, q_0, \sqcup, F)$ be a DTM with polynomial space bound $P(n)$, and $w \in \Sigma^*$, $|w| = n$.

Transition system $\mathcal{T} \overset{\text{def}}{=} (S, \text{Act}, \rightarrow, S_0, \text{AP}, L)$ where

$S = \{0, 1, \ldots, P(n)\} \cup \{\langle q, A, i \rangle, \langle *, A, i \rangle : q \in Q, A \in \Gamma, 1 \leq i \leq P(n)\}$

$S_0 = \{0\}$

$\text{AP} = S$ with obvious labeling function
Polynomial reduction $w \mapsto (T, \varphi)$

Let $\mathcal{M} = (Q, \Sigma, \Gamma, \delta, q_0, \sqcup, F)$ be a DTM with polynomial space bound $P(n)$, and $w \in \Sigma^*$, $|w| = n$.

Transition system $T \overset{\text{def}}{=} (S, \text{Act}, \rightarrow, S_0, \text{AP}, L)$ where

$S = \{0, 1, \ldots, P(n)\} \cup \{\langle q, A, i\rangle, \langle *, A, i\rangle : q \in Q, A \in \Gamma, 1 \leq i \leq P(n)\}$

$S_0 = \{0\}$

$\text{AP} = S$ with obvious labeling function

transitions: $i - 1 \rightarrow \langle q, A, i\rangle$ for $1 \leq i \leq P(n)$

$\langle q, A, i\rangle \rightarrow i$ and $q \in Q \cup \{\ast\}$
Polynomial reduction $w \mapsto (\mathcal{T}, \varphi)$

Let $\mathcal{M} = (Q, \Sigma, \Gamma, \delta, q_0, \sqcup, F)$ be a DTM with polynomial space bound $P(n)$, and $w \in \Sigma^*$, $|w| = n$.

Transition system $\mathcal{T} \overset{\text{def}}{=} (S, \text{Act}, \rightarrow, S_0, A\text{P}, L)$ where

$S = \{0, 1, \ldots, P(n)\} \cup \{\langle q, A, i \rangle, \langle *, A, i \rangle : q \in Q, A \in \Gamma, 1 \leq i \leq P(n)\}$

$S_0 = \{0\}$

$A\text{P} = S$ with obvious labeling function

transitions: $i - 1 \rightarrow \langle q, A, i \rangle$ \quad $P(n) \rightarrow 0$

$\langle q, A, i \rangle \rightarrow i$
Polynomial reduction \( w \mapsto (\mathcal{T}, \varphi) \)

Let \( \mathcal{M} = (Q, \Sigma, \Gamma, \delta, q_0, \sqcup, F) \) be a DTM with polynomial space bound \( P(n) \), and \( w \in \Sigma^*, |w| = n \).

Transition system \( \mathcal{T} \overset{\text{def}}{=} (S, \text{Act}, \rightarrow, S_0, AP, L) \) where

\[
S = \{0, 1, \ldots, P(n)\} \cup \{(q, A, i), (\ast, A, i) : q \in Q, A \in \Gamma, 1 \leq i \leq P(n)\}
\]

\( S_0 = \{0\} \)

\( AP = S \) with obvious labeling function

transitions:

\[
i - 1 \rightarrow \langle q, A, i \rangle \quad P(n) \rightarrow 0
\]

\[
\langle q, A, i \rangle \rightarrow i
\]

LTL formula \( \varphi \overset{\text{def}}{=} \varphi_{\text{start}}^w \land \varphi_\delta \land \varphi_{\text{conf}} \land \varphi_{\text{accept}} \)
Complexity of LTL model checking problem
We saw that:

The **existential LTL** model checking problem

given: finite TS $\mathcal{T}$, LTL formula $\varphi$

question: is there a path $\pi$ in $\mathcal{T}$ with $\pi \models \varphi$?

is **PSPACE**-complete.
Complexity of LTL model checking problem

We saw that:

The existential LTL model checking problem
given: finite TS $\mathcal{T}$, LTL formula $\varphi$ 
question: is there a path $\pi$ in $\mathcal{T}$ with $\pi \models \varphi$ ?
is $PSPACE$-complete.

As $PSPACE = coPSPACE$ we get:

The LTL model checking problem
given: finite TS $\mathcal{T}$, LTL formula $\varphi$ 
question: does $\pi \models \varphi$ hold for all paths $\pi$ in $\mathcal{T}$ ?
is $PSPACE$-complete.
Summary: LTL model checking problem
The LTL model checking problem is

- solvable by an automata-based approach
  complexity: $O(size(T) \cdot \exp(|\varphi|))$

- \textit{PSPACE}\textsuperscript{-complete}
The LTL model checking problem is

- solvable by an automata-based approach
  complexity: $O(size(T) \cdot \exp(|\varphi|))$

- \textbf{PSPACE}-complete

  proof of the lower bound:
  generic reduction from poly-space bounded DTM

  proof of the upper bound:
  uses the LTL-2-GNBA algorithm
The LTL model checking problem is

- solvable by an automata-based approach
  complexity: $O(\text{size}(T) \cdot \exp(|\varphi|))$

- \textsf{PSPACE}-complete

  \textit{proof} of the lower bound:
  generic reduction from poly-space bounded DTM

  \textit{proof} of the upper bound:
  uses the LTL-2-GNBA algorithm

\textit{additionally} we proved \textsf{coNP}-hardness

using an LTL-encoding of the Hamilton-path problem
NBA are more powerful than LTL
There is no LTL formula $\varphi$ over $AP = \{a\}$ s.t.

\[
\text{Words}(\varphi) = \text{set of words } A_0A_1A_2\ldots \in (2^{AP})^\omega \text{ s.t. } a \in A_{2i} \text{ for all } i \in \mathbb{N}
\]
There is no LTL formula $\varphi$ over $AP = \{a\}$ s.t.

$$Words(\varphi) = \text{set of words } A_0A_1A_2... \in (2^{AP})^\omega \text{ s.t. } a \in A_{2i} \text{ for all } i \in \mathbb{N}$$

NBA $\mathcal{A}$:

```
NBA are more powerful than LTL

There is no LTL formula $\varphi$ over $AP = \{a\}$ s.t.

$$Words(\varphi) = \text{set of words } A_0A_1A_2... \in (2^{AP})^\omega \text{ s.t. } a \in A_{2i} \text{ for all } i \in \mathbb{N}$$

NBA $\mathcal{A}$:
```

(without proof)
NBA are more powerful than LTL

There is no LTL formula $\varphi$ over $AP = \{a\}$ s.t.

$$Words(\varphi) = \text{set of words } A_0A_1A_2... \in (2^{AP})^\omega \text{ s.t.}$$

$$a \in A_{2i} \text{ for all } i \in \mathbb{N}$$

NBA $A$:

```
LTL formula $\varphi = a \land \Box(a \rightarrow \\
\bigcirc \bigcirc \bigcirc a)$ ?
```
NBA are more powerful than LTL

There is no LTL formula $\varphi$ over $AP = \{a\}$ s.t.

$$\text{Words}(\varphi) = \text{set of words } A_0A_1A_2\ldots \in (2^{AP})^\omega \text{ s.t. } a \in A_{2i} \text{ for all } i \in \mathbb{N}$$

NBA $A$:

LTL formula $\varphi = a \land \Box(a \rightarrow 
\Box\Box\Box a)$ ?

$\sigma = \{a\} \{a\} \{a\} \emptyset \{a\}^\omega \not\models \varphi$, but $\sigma \in \mathcal{L}_\omega(A)$
LTL satisfiability problem
LTL satisfiability problem

given: \( \text{LTL formula } \varphi \text{ over } AP \)

question: is \( \varphi \) satisfiable?
LTL satisfiability problem

**given:** LTL formula $\varphi$ over $AP$

**question:** is $\varphi$ satisfiable, i.e., is $\text{Words}(\varphi) \neq \emptyset$?
LTL satisfiability problem

given: \textbf{LTL} formula \( \varphi \) over \( AP \)

question: is \( \varphi \) satisfiable, i.e., is \( \text{Words}(\varphi) \neq \emptyset \)?

examples: \( \Diamond \Box a \land \Box \Diamond \neg a \) unsatisfiable
\( a \cup b \land \Box \neg b \) unsatisfiable
\( \Diamond \Box a \land a \cup (\Box b) \) satisfiable
LTL satisfiability problem

*given:* LTL formula $\varphi$ over $AP$

*question:* is $\varphi$ satisfiable, i.e., is $\text{Words}(\varphi) \neq \emptyset$?

automata-based satisfiability checking algorithm:

construct an NBA $\mathcal{A} = (Q, 2^{AP}, \delta, Q_0, F)$ for $\varphi$
LTL satisfiability problem

*given:* LTL formula $\varphi$ over $AP$

*question:* is $\varphi$ satisfiable, i.e., is $\text{Words}(\varphi) \neq \emptyset$?

automata-based satisfiability checking algorithm:

construct an NBA $A = (Q, 2^{AP}, \delta, Q_0, F)$ for $\varphi$

check whether $L_\omega(A) \neq \emptyset$
LTL satisfiability problem

given: \( \text{LTL} \) formula \( \varphi \) over \( AP \)

question: is \( \varphi \) satisfiable, i.e., is \( \text{Words}(\varphi) \neq \emptyset \) ?

automata-based satisfiability checking algorithm:

construct an NBA \( A = (Q, 2^{AP}, \delta, Q_0, F) \) for \( \varphi \)

check whether \( \mathcal{L}_\omega(A) \neq \emptyset \)

\[ \text{nested DFS: check whether } A \not\models \Diamond \Box \neg F \]
LTL satisfiability problem

given: LTL formula $\varphi$ over $AP$

question: is $\varphi$ satisfiable, i.e., is $\text{Words}(\varphi) \neq \emptyset$?

automata-based satisfiability checking algorithm:

construct an NBA $A = (Q, 2^{AP}, \delta, Q_0, F)$ for $\varphi$

check whether $\mathcal{L}_\omega(A) \neq \emptyset$

nested DFS: check whether $A \not\models \Diamond \Box \neg F$

if yes, return “yes”, otherwise “no”
LTL satisfiability problem

given: LTL formula $\varphi$ over $AP$

question: is $\varphi$ satisfiable, i.e., is $\text{Words}(\varphi) \neq \emptyset$?

automata-based satisfiability checking algorithm:

- construct an NBA $A = (Q, 2^{AP}, \delta, Q_0, F)$ for $\varphi$
- check whether $L_\omega(A) \neq \emptyset$
- nested DFS: check whether $A \not\models \Diamond \Box \neg F$
- if yes, return “yes”, otherwise “no”

complexity: $O(\exp(|\varphi|))$
LTL satisfiability problem

given: LTL formula \( \varphi \) over \( AP \)

question: is \( \varphi \) satisfiable, i.e., is \( \text{Words}(\varphi) \neq \emptyset \)?

automata-based satisfiability checking algorithm:

- construct an NBA \( A = (Q, 2^{AP}, \delta, Q_0, F) \) for \( \varphi \)
- check whether \( L_\omega(A) \neq \emptyset \)
  - nested DFS: check whether \( A \not\models \Diamond \Box \neg F \)

  if yes, return “yes”, otherwise “no”

complexity: \( \mathcal{O}(\exp(|\varphi|)) \) ... and \( \text{PSPACE-} \)-complete
LTL validity problem
LTL validity problem

given: LTL formula $\varphi$ over $AP$

question: is $\varphi$ valid, i.e. is $Words(\varphi) = (2^{AP})^\omega$?
LTL validity problem

given: \textbf{LTL} formula \( \varphi \) over \( AP \)

question: is \( \varphi \) valid, i.e. is \( \text{Words}(\varphi) = (2^{AP})^\omega \) ?

is solvable by a \textbf{LTL} satisfiability checker as

\( \varphi \) is valid iff \( \neg \varphi \) is not satisfiable
LTL validity problem

given: LTL formula $\varphi$ over $AP$
question: is $\varphi$ valid, i.e. is $\text{Words}(\varphi) = (2^{|AP|})^\omega$ ?

is solvable by a LTL satisfiability checker as

$\varphi$ is valid iff $\neg\varphi$ is not satisfiable

complexity: $O(\exp(|\varphi|))$
LTL validity problem

given: \textbf{LTL} formula $\varphi$ over $AP$

question: is $\varphi$ valid, i.e. is $\text{Words}(\varphi) = (2^{|AP|})^\omega$?

is solvable by a \textbf{LTL} satisfiability checker as

$\varphi$ is valid \iff $\neg\varphi$ is not satisfiable

complexity: $\mathcal{O}(\exp(|\varphi|))$ ... and $PSPACE$-complete