Specification and Verification

Lecture 6: Model checking TSs against LTL

Guillermo A. Pérez

University of Antwerp

November 3, 2024

TL;DR: This lecture in short

What is model checking? Why study it?

Essentially checking whether a transition system satisfies some formal specification

Main references

- Christel Baier, Joost-Pieter Katoen: **Principles of Model Checking.** MIT Press 2018.
- Mickael Randour: Verification course @ UMONS.

Required and target competences

What tools do we need?

Discrete maths, formal language theory, computational models

What skills will we obtain?

- theory: the automata-theoretic tools usual for verification
- practice: algorithms used at every step of the model-checking pipeline

How will these skills be useful?

Model checking is the most-widely adopted automatic verification technique

LTL model checking: decision problem

Definition: LTL model checking problem

Given a TS $\mathcal{T}$ and an LTL formula φ, decide if $\mathcal{T} \models \varphi$ or not.

$+$ if $\mathcal{T} \not\models \varphi$ we would like a counter-example (trace witnessing it)

LTL model checking: decision problem

Definition: LTL model checking problem

Given a TS $\mathcal{T}$ and an LTL formula φ, decide if $\mathcal{T} \models \varphi$ or not.

$+$ if $\mathcal{T} \not\models \varphi$ we would like a counter-example (trace witnessing it)

$\Longrightarrow$ Model checking algorithm via **automata-based approach** (Vardi and Wolper, 1986)

Intuition.

- Represent φ as an NBA
- Use it to try to find a path π in $\mathcal{T}$ such that $\pi \not\models \varphi$
- If one is found, a prefix of it is an *error trace*; otherwise, $\mathcal{T} \models \varphi$

LTL model checking: key observation

$$\mathcal{T} \models \varphi \qquad \text{iff} \quad \mathrm{Traces}(\mathcal{T}) \subseteq \mathrm{Words}(\varphi)$$

LTL model checking: key observation

$$
\begin{aligned}
\mathcal{T} \models \varphi \quad & \text{iff} \quad \mathrm{Traces}(\mathcal{T}) \subseteq \mathrm{Words}(\varphi) \\
& \text{iff} \quad \mathrm{Traces}(\mathcal{T}) \cap ((2^{P})^{\omega} \setminus \mathrm{Words}(\varphi)) = \varnothing
\end{aligned}
$$

LTL model checking: key observation

$$\begin{aligned}
\mathcal{T} \models \varphi \qquad & \text{iff} \quad \mathrm{Traces}(\mathcal{T}) \subseteq \mathrm{Words}(\varphi) \\
& \text{iff} \quad \mathrm{Traces}(\mathcal{T}) \cap ((2^{P})^{\omega} \setminus \mathrm{Words}(\varphi)) = \varnothing \\
& \text{iff} \quad \mathrm{Traces}(\mathcal{T}) \cap \mathrm{Words}(\neg\varphi) = \varnothing
\end{aligned}$$

Line 3 uses negation for paths.

LTL model checking: key observation

$$\begin{aligned}
\mathcal{T} \models \varphi \quad & \text{iff} \quad \mathrm{Traces}(\mathcal{T}) \subseteq \mathrm{Words}(\varphi) \\
& \text{iff} \quad \mathrm{Traces}(\mathcal{T}) \cap ((2^P)^\omega \setminus \mathrm{Words}(\varphi)) = \varnothing \\
& \text{iff} \quad \mathrm{Traces}(\mathcal{T}) \cap \mathrm{Words}(\neg\varphi) = \varnothing \\
& \text{iff} \quad \mathrm{Traces}(\mathcal{T}) \cap \mathcal{L}(\mathcal{A}_{\neg\varphi}) = \varnothing
\end{aligned}$$

Line 3 uses negation for paths.

Line 4 uses the existence of an NBA for any ω-regular language and the fact that **all LTL formulas describe ω-regular languages**.

LTL model checking: key observation

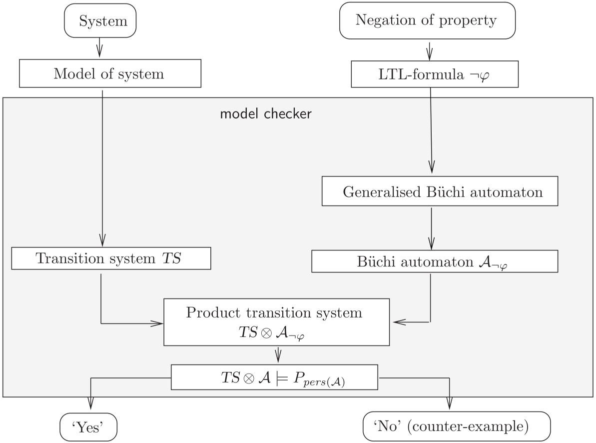
$$
\begin{array}{lll}
\mathcal{T} \models \varphi & \text{iff} & \mathrm{Traces}(\mathcal{T}) \subseteq \mathrm{Words}(\varphi) \\
 & \text{iff} & \mathrm{Traces}(\mathcal{T}) \cap ((2^P)^\omega \setminus \mathrm{Words}(\varphi)) = \varnothing \\
 & \text{iff} & \mathrm{Traces}(\mathcal{T}) \cap \mathrm{Words}(\neg\varphi) = \varnothing \\
 & \text{iff} & \mathrm{Traces}(\mathcal{T}) \cap \mathcal{L}(\mathcal{A}_{\neg\varphi}) = \varnothing \\
 & \text{iff} & \mathcal{T} \otimes \mathcal{A}_{\neg\varphi} \models \Diamond\Box\neg F
\end{array}
$$

Line 3 uses negation for paths.

Line 4 uses the existence of an NBA for any ω-regular language and the fact that **all LTL formulas describe ω-regular languages**.

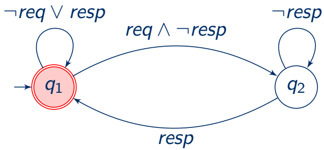
Line 5 reduces the language intersection problem to the satisfaction of a persistence property over the product TS $\mathcal{T} \otimes \mathcal{A}_{\neg\varphi}$. The idea is to **check that no trace yielded by $\mathcal{T}$ will satisfy the acceptance condition of the NBA $\mathcal{A}_{\neg\varphi}$**.

Overview of the algorithm



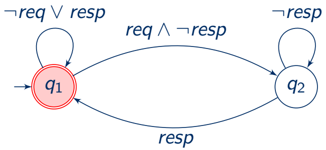
From LTL to GNBA: examples

- NBA for $\Box(req \rightarrow \Diamond resp)$

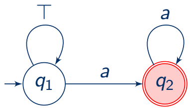


From LTL to GNBA: examples

- NBA for $\Box(req \rightarrow \Diamond resp)$

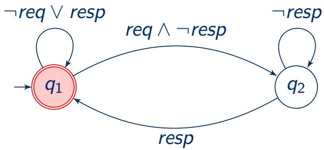


- NBA for $\Diamond\Box a$

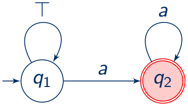


From LTL to GNBA: examples

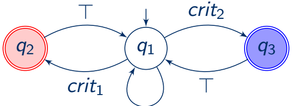
- NBA for $\square(req \rightarrow \lozenge resp)$



- NBA for $\lozenge\square a$



- GNBA for $\square\lozenge crit_1 \wedge \square\lozenge crit_2$



From LTL to GNBA: intuition (1/3)

Goal

For an LTL formula φ, build GNBA $\mathcal{G}_\varphi$ over alphabet 2^P such that $\mathcal{L}(\mathcal{G}_\varphi) = \mathrm{Words}(\varphi)$.

From LTL to GNBA: intuition (1/3)

Goal

For an LTL formula φ, build GNBA $\mathcal{G}_\varphi$ over alphabet 2^P such that $\mathcal{L}(\mathcal{G}_\varphi) = \mathrm{Words}(\varphi)$.

- Assume φ only contains core operators $\wedge$, $\neg$, $\bigcirc$, $\mathcal{U}$ (w.l.o.g., see core syntax) and $\varphi \neq \top$ (otherwise, trivial GNBA).

From LTL to GNBA: intuition (1/3)

Goal

For an LTL formula φ, build GNBA $\mathcal{G}_\varphi$ over alphabet 2^P such that $\mathcal{L}(\mathcal{G}_\varphi) = \mathrm{Words}(\varphi)$.

- Assume φ only contains core operators $\wedge$, $\neg$, $\bigcirc$, $\mathcal{U}$ (w.l.o.g., see core syntax) and $\varphi \neq \top$ (otherwise, trivial GNBA).
- **What will the states of $\mathcal{G}_\varphi$ be?**

From LTL to GNBA: intuition (1/3)

Goal

For an LTL formula φ, build GNBA $\mathcal{G}_\varphi$ over alphabet 2^P such that $\mathcal{L}(\mathcal{G}_\varphi) = \mathrm{Words}(\varphi)$.

- Assume φ only contains core operators $\wedge$, $\neg$, $\bigcirc$, $\mathcal{U}$ (w.l.o.g., see core syntax) and $\varphi \neq \top$ (otherwise, trivial GNBA).
- **What will the states of $\mathcal{G}_\varphi$ be?**
  - Let $w = a_0 a_1 a_2 \cdots \in \mathrm{Words}(\varphi)$. Idea: **"extend" the sets $a_i \subseteq P$ with subformulas ψ of φ.**

From LTL to GNBA: intuition (1/3)

Goal

For an LTL formula φ, build GNBA $\mathcal{G}_\varphi$ over alphabet 2^P such that $\mathcal{L}(\mathcal{G}_\varphi) = \mathrm{Words}(\varphi)$.

- Assume φ only contains core operators $\wedge$, $\neg$, $\bigcirc$, $\mathcal{U}$ (w.l.o.g., see core syntax) and $\varphi \neq \top$ (otherwise, trivial GNBA).
- **What will the states of $\mathcal{G}_\varphi$ be?**
  - Let $w = a_0 a_1 a_2 \cdots \in \mathrm{Words}(\varphi)$. Idea: **"extend" the sets $a_i \subseteq P$ with subformulas ψ of φ**.
  - Obtain $\overline{w} = B_0 B_1 B_2 \ldots$ such that

$$\psi \in B_i \quad \Longleftrightarrow \quad a_i a_{i+1} a_{i+2} \ldots \models \psi.$$

  - $\overline{w}$ will be a **run for w** in the GNBA $\mathcal{G}_\varphi$.

University of Antwerp

From LTL to GNBA: intuition (2/3)

- Let $\varphi = a\,\mathcal{U}\,(\neg a \wedge b)$ and $w = \{a\}\,\{a,b\}\,\{b\}\ldots$

From LTL to GNBA: intuition (2/3)

- Let $\varphi = a\,\mathcal{U}\,(\neg a \wedge b)$ and $w = \{a\}\,\{a,b\}\,\{b\}\ldots$
  - States B_i are subsets of

$$\underbrace{\{a, \neg a, b, \neg a \wedge b, \varphi\}}_{\text{subformulas of } \varphi} \cup \underbrace{\{\neg b, \neg(\neg a \wedge b), \neg\varphi\}}_{\text{their negation}}.$$

  - Negations also considered for technical reasons

University of Antwerp

From LTL to GNBA: intuition (2/3)

- Let $\varphi = a\,\mathcal{U}\,(\neg a \wedge b)$ and $w = \{a\}\,\{a,b\}\,\{b\}\ldots$
  - States B_i are subsets of
  $$\underbrace{\{a, \neg a, b, \neg a \wedge b, \varphi\}}_{\text{subformulas of } \varphi} \cup \underbrace{\{\neg b, \neg(\neg a \wedge b), \neg\varphi\}}_{\text{their negation}}.$$
  - Negations also considered for technical reasons
- $a_0 = \{a\}$ is extended with $\neg b$, $\neg(\neg a \wedge b)$ and φ as they hold in w and no other subformula holds

University of Antwerp

From LTL to GNBA: intuition (2/3)

- Let $\varphi = a \,\mathcal{U}\, (\neg a \wedge b)$ and $w = \{a\}\,\{a,b\}\,\{b\}\ldots$
  - States B_i are subsets of
  $$\underbrace{\{a, \neg a, b, \neg a \wedge b, \varphi\}}_{\text{subformulas of } \varphi} \cup \underbrace{\{\neg b, \neg(\neg a \wedge b), \neg\varphi\}}_{\text{their negation}}.$$
  - Negations also considered for technical reasons
- $a_0 = \{a\}$ is extended with $\neg b$, $\neg(\neg a \wedge b)$ and φ as they hold in w and no other subformula holds
- $a_1 = \{a, b\}$ with $\neg(\neg a \wedge b)$ and φ as they hold in $w[1..]$

From LTL to GNBA: intuition (2/3)

- Let $\varphi = a \,\mathcal{U}\, (\neg a \wedge b)$ and $w = \{a\}\,\{a, b\}\,\{b\}\ldots$
  - States B_i are subsets of
$$\underbrace{\{a, \neg a, b, \neg a \wedge b, \varphi\}}_{\text{subformulas of } \varphi} \cup \underbrace{\{\neg b, \neg(\neg a \wedge b), \neg\varphi\}}_{\text{their negation}}.$$
  - Negations also considered for technical reasons
- $a_0 = \{a\}$ is extended with $\neg b$, $\neg(\neg a \wedge b)$ and φ as they hold in w and no other subformula holds
- $a_1 = \{a, b\}$ with $\neg(\neg a \wedge b)$ and φ as they hold in $w[1..]$
- $a_2 = \{b\}$ with $\neg a$, $\neg a \wedge b$ and φ as they hold in $w[2..]$

From LTL to GNBA: intuition (2/3)

- Let $\varphi = a\,\mathcal{U}\,(\neg a \wedge b)$ and $w = \{a\}\,\{a,b\}\,\{b\}\ldots$
  - States B_i are subsets of
  $$\underbrace{\{a, \neg a, b, \neg a \wedge b, \varphi\}}_{\text{subformulas of } \varphi} \cup \underbrace{\{\neg b, \neg(\neg a \wedge b), \neg\varphi\}}_{\text{their negation}}.$$
  - Negations also considered for technical reasons
- $a_0 = \{a\}$ is extended with $\neg b$, $\neg(\neg a \wedge b)$ and φ as they hold in w and no other subformula holds
- $a_1 = \{a, b\}$ with $\neg(\neg a \wedge b)$ and φ as they hold in $w[1..]$
- $a_2 = \{b\}$ with $\neg a$, $\neg a \wedge b$ and φ as they hold in $w[2..]$

$$\underbrace{\{a, \neg b, \neg(\neg a \wedge b), \varphi\}}_{B_0}\ \underbrace{\{a, b, \neg(\neg a \wedge b), \varphi\}}_{B_1}\ \underbrace{\{\neg a, b, \neg a \wedge b, \varphi\}}_{B_2}\ \ldots$$

$=\overline{w}$

From LTL to GNBA: intuition (3/3)

- Sets B_i will be the states of GNBA $\mathcal{G}_\varphi$

From LTL to GNBA: intuition (3/3)

- Sets B_i will be the states of GNBA $\mathcal{G}_\varphi$
- $\overline{w} = B_0 B_1 B_2 \ldots$ is a run for w in $\mathcal{G}_\varphi$ by construction

From LTL to GNBA: intuition (3/3)

- Sets B_i will be the states of GNBA $\mathcal{G}_\varphi$
- $\overline{w} = B_0 B_1 B_2 \ldots$ is a run for w in $\mathcal{G}_\varphi$ by construction
- Accepting condition chosen such that $\overline{w}$ is accepting if and only if $w \models \varphi$

From LTL to GNBA: intuition (3/3)

- Sets B_i will be the states of GNBA $\mathcal{G}_\varphi$
- $\overline{w} = B_0 B_1 B_2 \ldots$ is a run for w in $\mathcal{G}_\varphi$ by construction
- Accepting condition chosen such that $\overline{w}$ is accepting if and only if $w \models \varphi$
- **How do we encode the meaning of the logical operators?**

From LTL to GNBA: intuition (3/3)

- Sets B_i will be the states of GNBA $\mathcal{G}_\varphi$
- $\overline{w} = B_0 B_1 B_2 \ldots$ is a run for w in $\mathcal{G}_\varphi$ by construction
- Accepting condition chosen such that $\overline{w}$ is accepting if and only if $w \models \varphi$
- **How do we encode the meaning of the logical operators?**
  - $\wedge$, $\neg$ and $\top$ impose *consistent formula sets* B_i in the states (e.g., a and $\neg a$ is not possible)

From LTL to GNBA: intuition (3/3)

- Sets B_i will be the states of GNBA $\mathcal{G}_\varphi$
- $\overline{w} = B_0 B_1 B_2 \ldots$ is a run for w in $\mathcal{G}_\varphi$ by construction
- Accepting condition chosen such that $\overline{w}$ is accepting if and only if $w \models \varphi$
- **How do we encode the meaning of the logical operators?**
  - $\wedge$, $\neg$ and $\top$ impose *consistent formula sets* B_i in the states (e.g., a and $\neg a$ is not possible)
  - $\bigcirc$ encoded in the *transition relation (must be consistent)*

University of Antwerp

From LTL to GNBA: intuition (3/3)

- Sets B_i will be the states of GNBA $\mathcal{G}_\varphi$
- $\overline{w} = B_0 B_1 B_2 \ldots$ is a run for w in $\mathcal{G}_\varphi$ by construction
- Accepting condition chosen such that $\overline{w}$ is accepting if and only if $w \models \varphi$
- **How do we encode the meaning of the logical operators?**
  - $\wedge$, $\neg$ and $\top$ impose *consistent formula sets* B_i in the states (e.g., a and $\neg a$ is not possible)
  - $\bigcirc$ encoded in the *transition relation (must be consistent)*
  - $\mathcal{U}$ split according to the *expansion law* into *local condition (encoded in states)* and *next-step one (encoded in transitions)*

From LTL to GNBA: intuition (3/3)

- Sets B_i will be the states of GNBA $\mathcal{G}_\varphi$
- $\overline{w} = B_0 B_1 B_2 \ldots$ is a run for w in $\mathcal{G}_\varphi$ by construction
- Accepting condition chosen such that $\overline{w}$ is accepting if and only if $w \models \varphi$
- **How do we encode the meaning of the logical operators?**
  - $\wedge$, $\neg$ and $\top$ impose *consistent formula sets* B_i in the states (e.g., a and $\neg a$ is not possible)
  - $\bigcirc$ encoded in the *transition relation (must be consistent)*
  - $\mathcal{U}$ split according to the *expansion law* into *local condition (encoded in states)* and *next-step one (encoded in transitions)*
  - Meaning of $\mathcal{U}$ is the *least solution* of the expansion law $\Longrightarrow$ reflected in the choice of *acceptance sets for $\mathcal{G}_\varphi$*

From LTL to GNBA: closure of a formula

Definition: closure of φ

The set $\mathrm{Closure}(\varphi)$ consists of all sub-formulas ψ of φ and their negation $\neg\psi$

E.g., for $\varphi = a\,\mathcal{U}\,(\neg a \wedge b)$,

$$\mathrm{Closure}(\varphi) = \{a, \neg a, b, \neg b, \neg a \wedge b, \neg(\neg a \wedge b), \varphi, \neg\varphi\}.$$

From LTL to GNBA: closure of a formula

Definition: closure of φ

The set $\mathrm{Closure}(\varphi)$ consists of all sub-formulas ψ of φ and their negation $\neg\psi$

E.g., for $\varphi = a\,\mathcal{U}\,(\neg a \wedge b)$,

$$\mathrm{Closure}(\varphi) = \{a, \neg a, b, \neg b, \neg a \wedge b, \neg(\neg a \wedge b), \varphi, \neg\varphi\}.$$

$\hookrightarrow$ $|\mathrm{Closure}(\varphi)| = \mathcal{O}(|\varphi|)$.

From LTL to GNBA: closure of a formula

Definition: closure of φ

The set $\mathrm{Closure}(\varphi)$ consists of all sub-formulas ψ of φ and their negation $\neg\psi$

E.g., for $\varphi = a \, \mathcal{U} \, (\neg a \wedge b)$,

$$\mathrm{Closure}(\varphi) = \{a, \neg a, b, \neg b, \neg a \wedge b, \neg(\neg a \wedge b), \varphi, \neg\varphi\}.$$

$\hookrightarrow$ $|\mathrm{Closure}(\varphi)| = \mathcal{O}(|\varphi|)$.

Sets B_i are elementary subsets of $\mathrm{Closure}(\varphi)$

Intuition: a set B is *elementary* if there is a path π such that B is the set of **all** formulas $\psi \in \mathrm{Closure}(\varphi)$ with $\pi \models \psi$

From LTL to GNBA: elementary sets

Definition: elementary set

A set of sub-formulas $B \subseteq \mathrm{Closure}(\varphi)$ is *elementary* if:

1. B is **logically consistent**, i.e., for all $\varphi_1 \wedge \varphi_2, \psi \in \mathrm{Closure}(\varphi)$
   - $\varphi_1 \wedge \varphi_2 \in B \iff \varphi_1 \in B \;\wedge\; \varphi_2 \in B$
   - $\psi \in B \implies \neg\psi \notin B$
   - $\top \in \mathrm{Closure}(\varphi) \implies \top \in B$

From LTL to GNBA: elementary sets

Definition: elementary set

A set of sub-formulas $B \subseteq \mathrm{Closure}(\varphi)$ is *elementary* if:

1. B is **logically consistent**, i.e., for all $\varphi_1 \wedge \varphi_2, \psi \in \mathrm{Closure}(\varphi)$
   - $\varphi_1 \wedge \varphi_2 \in B \iff \varphi_1 \in B \ \wedge \ \varphi_2 \in B$
   - $\psi \in B \implies \neg\psi \notin B$
   - $\top \in \mathrm{Closure}(\varphi) \implies \top \in B$
2. B is **locally consistent**, i.e., for all $\varphi_1 \,\mathcal{U}\, \varphi_2 \in \mathrm{Closure}(\varphi)$
   - $\varphi_2 \in B \implies \varphi_1 \,\mathcal{U}\, \varphi_2 \in B$,
   - $\varphi_1 \,\mathcal{U}\, \varphi_2 \in B \ \wedge \ \varphi_2 \notin B \implies \varphi_1 \in B$

From LTL to GNBA: elementary sets

Definition: elementary set

A set of sub-formulas $B \subseteq \mathrm{Closure}(\varphi)$ is *elementary* if:

1. B is **logically consistent**, i.e., for all $\varphi_1 \wedge \varphi_2, \psi \in \mathrm{Closure}(\varphi)$
   - $\varphi_1 \wedge \varphi_2 \in B \iff \varphi_1 \in B \ \wedge \ \varphi_2 \in B$
   - $\psi \in B \implies \neg\psi \notin B$
   - $\top \in \mathrm{Closure}(\varphi) \implies \top \in B$
2. B is **locally consistent**, i.e., for all $\varphi_1 \,\mathcal{U}\, \varphi_2 \in \mathrm{Closure}(\varphi)$
   - $\varphi_2 \in B \implies \varphi_1 \,\mathcal{U}\, \varphi_2 \in B$,
   - $\varphi_1 \,\mathcal{U}\, \varphi_2 \in B \ \wedge \ \varphi_2 \notin B \implies \varphi_1 \in B$
3. B is **maximal**, i.e., for all $\psi \in \mathrm{Closure}(\varphi)$
   - $\psi \notin B \implies \neg\psi \in B$

Elementary sets: examples (1/2)

Let $\varphi = a \, \mathcal{U} \, (\neg a \wedge b)$:

$$\mathrm{Closure}(\varphi) = \{a, \neg a, b, \neg b, \neg a \wedge b, \neg(\neg a \wedge b), \varphi, \neg \varphi\}$$

Elementary sets: examples (1/2)

Let $\varphi = a\,\mathcal{U}\,(\neg a \wedge b)$:

$$\mathrm{Closure}(\varphi) = \{a, \neg a, b, \neg b, \neg a \wedge b, \neg(\neg a \wedge b), \varphi, \neg\varphi\}$$

- Is $B = \{a, b, \varphi\} \subset \mathrm{Closure}(\varphi)$ elementary?

Elementary sets: examples (1/2)

Let $\varphi = a\,\mathcal{U}\,(\neg a \wedge b)$:

$$\mathrm{Closure}(\varphi) = \{a, \neg a, b, \neg b, \neg a \wedge b, \neg(\neg a \wedge b), \varphi, \neg\varphi\}$$

- Is $B = \{a, b, \varphi\} \subset \mathrm{Closure}(\varphi)$ elementary?

  $\hookrightarrow$ **No.** Logically and locally consistent but **not maximal** because $\neg a \wedge b \in \mathrm{Closure}(\varphi)$, yet $\neg a \wedge b \notin B$ and $\neg(\neg a \wedge b) \notin B$

Elementary sets: examples (1/2)

Let $\varphi = a\,\mathcal{U}\,(\neg a \wedge b)$:

$$\text{Closure}(\varphi) = \{a, \neg a, b, \neg b, \neg a \wedge b, \neg(\neg a \wedge b), \varphi, \neg\varphi\}$$

- Is $B = \{a, b, \varphi\} \subset \text{Closure}(\varphi)$ elementary?
  $\hookrightarrow$ **No.** Logically and locally consistent but **not maximal** because $\neg a \wedge b \in \text{Closure}(\varphi)$, yet $\neg a \wedge b \notin B$ and $\neg(\neg a \wedge b) \notin B$
- Is $B = \{a, b, \neg a \wedge b, \varphi\} \subset \text{Closure}(\varphi)$ elementary?

Elementary sets: examples (1/2)

Let $\varphi = a\,\mathcal{U}\,(\neg a \wedge b)$:

$$\mathrm{Closure}(\varphi) = \{a, \neg a, b, \neg b, \neg a \wedge b, \neg(\neg a \wedge b), \varphi, \neg\varphi\}$$

- Is $B = \{a, b, \varphi\} \subset \mathrm{Closure}(\varphi)$ elementary?
  - $\hookrightarrow$ **No.** Logically and locally consistent but **not maximal** because $\neg a \wedge b \in \mathrm{Closure}(\varphi)$, yet $\neg a \wedge b \notin B$ and $\neg(\neg a \wedge b) \notin B$
- Is $B = \{a, b, \neg a \wedge b, \varphi\} \subset \mathrm{Closure}(\varphi)$ elementary?
  - $\hookrightarrow$ **No.** It is **not logically consistent** because $a \in B$ and $\neg a \wedge b \in B$

Elementary sets: examples (1/2)

Let $\varphi = a\,\mathcal{U}\,(\neg a \wedge b)$:

$$\mathrm{Closure}(\varphi) = \{a, \neg a, b, \neg b, \neg a \wedge b, \neg(\neg a \wedge b), \varphi, \neg\varphi\}$$

- Is $B = \{a, b, \varphi\} \subset \mathrm{Closure}(\varphi)$ elementary?

  $\hookrightarrow$ **No.** Logically and locally consistent but **not maximal** because $\neg a \wedge b \in \mathrm{Closure}(\varphi)$, yet $\neg a \wedge b \notin B$ and $\neg(\neg a \wedge b) \notin B$

- Is $B = \{a, b, \neg a \wedge b, \varphi\} \subset \mathrm{Closure}(\varphi)$ elementary?

  $\hookrightarrow$ **No.** It is **not logically consistent** because $a \in B$ and $\neg a \wedge b \in B$

- Is $B = \{\neg a, \neg b, \neg(\neg a \wedge b), \varphi\} \subset \mathrm{Closure}(\varphi)$ elementary?

Elementary sets: examples (1/2)

Let $\varphi = a\,\mathcal{U}\,(\neg a \wedge b)$:

$$\mathrm{Closure}(\varphi) = \{a, \neg a, b, \neg b, \neg a \wedge b, \neg(\neg a \wedge b), \varphi, \neg\varphi\}$$

- Is $B = \{a, b, \varphi\} \subset \mathrm{Closure}(\varphi)$ elementary?
  - $\hookrightarrow$ **No.** Logically and locally consistent but **not maximal** because $\neg a \wedge b \in \mathrm{Closure}(\varphi)$, yet $\neg a \wedge b \notin B$ and $\neg(\neg a \wedge b) \notin B$
- Is $B = \{a, b, \neg a \wedge b, \varphi\} \subset \mathrm{Closure}(\varphi)$ elementary?
  - $\hookrightarrow$ **No.** It is **not logically consistent** because $a \in B$ and $\neg a \wedge b \in B$
- Is $B = \{\neg a, \neg b, \neg(\neg a \wedge b), \varphi\} \subset \mathrm{Closure}(\varphi)$ elementary?
  - $\hookrightarrow$ **No.** Logically consistent but **not locally consistent** because $\varphi = a\,\mathcal{U}\,(\neg a \wedge b) \in B$ and $\neg a \wedge b \notin B$ but $a \notin B$

Elementary sets: examples (2/2)

Let $\varphi = a \,\mathcal{U}\, (\neg a \wedge b)$:

$$\mathrm{Closure}(\varphi) = \{a, \neg a, b, \neg b, \neg a \wedge b, \neg(\neg a \wedge b), \varphi, \neg\varphi\}$$

Elementary sets: examples (2/2)

Let $\varphi = a\,\mathcal{U}\,(\neg a \wedge b)$:

$$\mathrm{Closure}(\varphi) = \{a, \neg a, b, \neg b, \neg a \wedge b, \neg(\neg a \wedge b), \varphi, \neg\varphi\}$$

All elementary sets?

Elementary sets: examples (2/2)

Let $\varphi = a\,\mathcal{U}\,(\neg a \wedge b)$:

$$\mathrm{Closure}(\varphi) = \{a, \neg a, b, \neg b, \neg a \wedge b, \neg(\neg a \wedge b), \varphi, \neg\varphi\}$$

All elementary sets?

$$\begin{aligned}
B_1 &= \{a, b, \neg(\neg a \wedge b), \varphi\} \\
B_2 &= \{a, b, \neg(\neg a \wedge b), \neg\varphi\} \\
B_3 &= \{a, \neg b, \neg(\neg a \wedge b), \varphi\} \\
B_4 &= \{a, \neg b, \neg(\neg a \wedge b), \neg\varphi\} \\
B_5 &= \{\neg a, \neg b, \neg(\neg a \wedge b), \neg\varphi\} \\
B_6 &= \{\neg a, b, \neg a \wedge b, \varphi\}
\end{aligned}$$

From LTL to GNBA: $\mathcal{G}_\varphi$ (1/2)

For formula φ over P, let $\mathcal{G}_\varphi = (Q, A = 2^P, \delta, I, \mathcal{F})$ where:

- $Q = \{B \subseteq \mathrm{Closure}(\varphi) \mid B \text{ is elementary}\}$,

University of Antwerp

From LTL to GNBA: $\mathcal{G}_\varphi$ (1/2)

For formula φ over P, let $\mathcal{G}_\varphi = (Q, A = 2^P, \delta, I, \mathcal{F})$ where:

- $Q = \{B \subseteq \mathrm{Closure}(\varphi) \mid B$ is elementary$\}$,
- $I = \{B \in Q \mid \varphi \in B\}$,

University of Antwerp

From LTL to GNBA: $\mathcal{G}_\varphi$ (1/2)

For formula φ over P, let $\mathcal{G}_\varphi = (Q, A = 2^P, \delta, I, \mathcal{F})$ where:

- $Q = \{B \subseteq \mathrm{Closure}(\varphi) \mid B \text{ is elementary}\}$,
- $I = \{B \in Q \mid \varphi \in B\}$,
- $\mathcal{F} = \{F_{\varphi_1 \mathcal{U} \varphi_2} \mid \varphi_1 \,\mathcal{U}\, \varphi_2 \in \mathrm{Closure}(\varphi)\}$ with
$$F_{\varphi_1 \mathcal{U} \varphi_2} = \{B \in Q \mid \varphi_1 \,\mathcal{U}\, \varphi_2 \notin B \vee \varphi_2 \in B\}.$$

- *Intuition: for any run $B_0 B_1 B_2 \ldots$, if $\varphi_1 \,\mathcal{U}\, \varphi_2 \in B_0$, then φ_2 must eventually become true ($\rightsquigarrow$ ensured by the acceptance condition)*

From LTL to GNBA: $\mathcal{G}_\varphi$ (1/2)

For formula φ over P, let $\mathcal{G}_\varphi = (Q, A = 2^P, \delta, I, \mathcal{F})$ where:

- $Q = \{B \subseteq \mathrm{Closure}(\varphi) \mid B \text{ is elementary}\}$,
- $I = \{B \in Q \mid \varphi \in B\}$,
- $\mathcal{F} = \{F_{\varphi_1 \mathcal{U} \varphi_2} \mid \varphi_1 \,\mathcal{U}\, \varphi_2 \in \mathrm{Closure}(\varphi)\}$ with
$$F_{\varphi_1 \mathcal{U} \varphi_2} = \{B \in Q \mid \varphi_1 \,\mathcal{U}\, \varphi_2 \notin B \vee \varphi_2 \in B\}.$$

- *Intuition: for any run $B_0 B_1 B_2 \ldots$, if $\varphi_1 \,\mathcal{U}\, \varphi_2 \in B_0$, then φ_2 must eventually become true ($\rightsquigarrow$ ensured by the acceptance condition)*

Observe that $\mathcal{F} = \varnothing$ if no until in φ.

$\Longrightarrow$ All runs are accepting in this case.

From LTL to GNBA: $\mathcal{G}_\varphi$ (2/2)

The transition relation $\delta\colon Q \times 2^P \to 2^Q$ is given by:

From LTL to GNBA: $\mathcal{G}_\varphi$ (2/2)

The transition relation $\delta\colon Q \times 2^P \to 2^Q$ is given by:

- For $a \in 2^P$ and $B \in Q$, if $a \neq B \cap P$, then $\delta(B, a) = \varnothing$
- *Intuition: transitions only exist for the set of propositions that are true in B, i.e., $B \cap P$ is the only readable letter at state B*

From LTL to GNBA: $\mathcal{G}_\varphi$ (2/2)

The transition relation $\delta\colon Q \times 2^P \to 2^Q$ is given by:

- For $a \in 2^P$ and $B \in Q$, if $a \neq B \cap P$, then $\delta(B, a) = \varnothing$
- *Intuition: transitions only exist for the set of propositions that are true in B, i.e., $B \cap P$ is the only readable letter at state B*
- If $a = B \cap P$, then $\delta(B, a)$ is the set of all elementary sets of formulas B' satisfying
  (i) for every $\bigcirc\psi \in \mathrm{Closure}(\varphi)$, $\bigcirc\psi \in B \iff \psi \in B'$, and
  (ii) for every $\varphi_1 \,\mathcal{U}\, \varphi_2 \in \mathrm{Closure}(\varphi)$,

$$\varphi_1 \,\mathcal{U}\, \varphi_2 \in B \iff (\varphi_2 \in B \vee (\varphi_1 \in B \wedge \varphi_1 \,\mathcal{U}\, \varphi_2 \in B'))$$

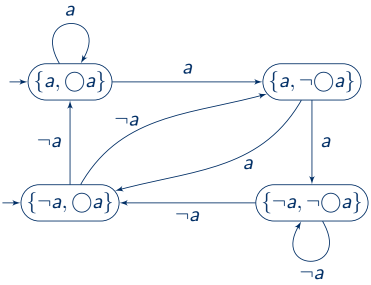
- *Intuition: (i) and (ii) reflect the semantics of $\bigcirc$ and $\mathcal{U}$ operators, (ii) is based on the expansion law*

From LTL to GNBA: e.g. $\varphi = \bigcirc a$

- $\text{Closure}(\varphi) = \{a, \neg a, \bigcirc a, \neg\bigcirc a\}$

From LTL to GNBA: e.g. $\varphi = \bigcirc a$

- $\text{Closure}(\varphi) = \{a, \neg a, \bigcirc a, \neg\bigcirc a\}$



- $Q = \{\{a, \bigcirc a\}, \{a, \neg\bigcirc a\}, \{\neg a, \bigcirc a\}, \{\neg a, \neg\bigcirc a\}\}$
- $I = \{\{a, \bigcirc a\}, \{\neg a, \bigcirc a\}\}$
- $\mathcal{F} = \varnothing$

From LTL to GNBA: $\varphi = a\,\mathcal{U}\,b$ (1/3)

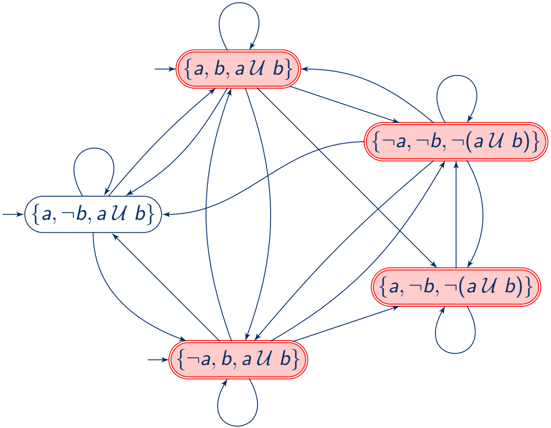
- $\mathrm{Closure}(\varphi) = \{a, \neg a, b, \neg b, a\,\mathcal{U}\,b, \neg(a\,\mathcal{U}\,b)\}$

$\Longrightarrow$ **Blackboard construction of the GNBA**

From LTL to GNBA: $\varphi = a\,\mathcal{U}\,b$ (1/3)

- $\text{Closure}(\varphi) = \{a, \neg a, b, \neg b, a\,\mathcal{U}\,b, \neg(a\,\mathcal{U}\,b)\}$

$\Longrightarrow$ **Blackboard construction of the GNBA**



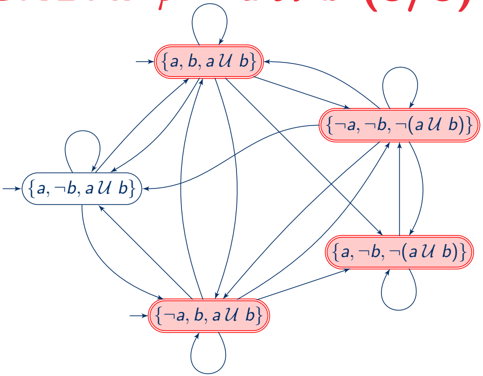
From LTL to GNBA: $\varphi = a \,\mathcal{U}\, b$ (2/3)

Some explanations

Let $B_1 = \{a, b, a \,\mathcal{U}\, b\}$, $B_2 = \{\neg a, b, a \,\mathcal{U}\, b\}$, $B_3 = \{a, \neg b, a \,\mathcal{U}\, b\}$, $B_4 = \{\neg a, \neg b, \neg(a \,\mathcal{U}\, b)\}$ and $B_5 = \{a, \neg b, \neg(a \,\mathcal{U}\, b)\}$.

- $Q = \{B_1, B_2, B_3, B_4, B_5\}$, $I = \{B_1, B_2, B_3\}$
- $\mathcal{F} = \{F_{a\mathcal{U}b}\} = \{\{B_1, B_2, B_4, B_5\}\}$.
  $\hookrightarrow$ $\mathcal{G}_\varphi$ is actually a **simple NBA**
- Labels omitted for readability (recall label is $B \cap P$)
- From B_1 (resp. B_2), we can go anywhere because $a \,\mathcal{U}\, b$ is already fulfilled by $b \in B_1$ (resp. B_2)
- From B_3, we need to go where $a \,\mathcal{U}\, b$ holds: B_1, B_2 or B_3
- From B_4, we can go anywhere because $\neg(a \,\mathcal{U}\, b)$ is already fulfilled by $\neg a, \neg b \in B_4$
- From B_5, we need to go where $\neg(a \,\mathcal{U}\, b)$ holds: B_4 or B_5

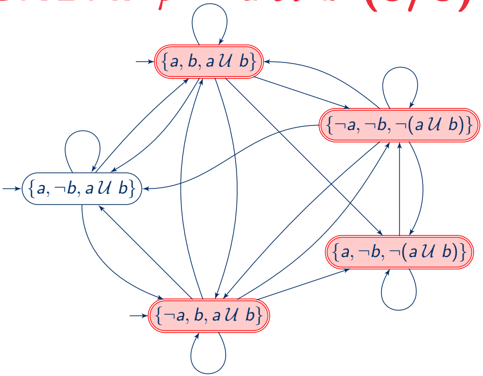
From LTL to GNBA: $\varphi = a \,\mathcal{U}\, b$ (3/3)



Sample words/runs:

- $\{a\}\,\{a\}\,\{b\}^\omega \in \mathrm{Words}(\varphi)$ has accepting run $B_3 B_3 B_2^\omega$ in $\mathcal{G}_\varphi$

From LTL to GNBA: $\varphi = a \,\mathcal{U}\, b$ (3/3)



Sample words/runs:

- $\{a\}\,\{a\}\,\{b\}^\omega \in \mathrm{Words}(\varphi)$ has accepting run $B_3 B_3 B_2^\omega$ in $\mathcal{G}_\varphi$
- $\{a\}^\omega \notin \mathrm{Words}(\varphi)$ has only one run B_3^ω in $\mathcal{G}_\varphi$ and it is not accepting since $B_3 \notin F_{a\mathcal{U}b}$

From LTL to. . . NBA: construction

Idea: LTL $\rightsquigarrow$ GNBA $\rightsquigarrow$ NBA

From LTL to... NBA: construction

Idea: LTL $\rightsquigarrow$ GNBA $\rightsquigarrow$ NBA

Theorem: LTL to NBA

For any LTL formula φ over propositions P, there exists an NBA $\mathcal{A}_\varphi$ with $\mathrm{Words}(\varphi) = \mathcal{L}(\mathcal{A}_\varphi)$ which can be constructed in time and space $2^{\mathcal{O}(|\varphi|)}$.

From LTL to... NBA: construction

Idea: LTL $\rightsquigarrow$ GNBA $\rightsquigarrow$ NBA

Theorem: LTL to NBA

For any LTL formula φ over propositions P, there exists an NBA $\mathcal{A}_\varphi$ with $\mathrm{Words}(\varphi) = \mathcal{L}(\mathcal{A}_\varphi)$ which can be constructed in time and space $2^{\mathcal{O}(|\varphi|)}$.

Sketch

1. Construct the GNBA $\mathcal{G}_\varphi$
   - $|\mathrm{Closure}(\varphi)| = \mathcal{O}(|\varphi|)$ and $|Q| \leq 2^{|\mathrm{Closure}(\varphi)|} = 2^{\mathcal{O}(|\varphi|)}$
   - $\#$ accepting sets of $\mathcal{G}_\varphi = \#$ until-operators in $\varphi \leq \mathcal{O}(|\varphi|)$

From LTL to...NBA: construction

Idea: LTL $\rightsquigarrow$ GNBA $\rightsquigarrow$ NBA

Theorem: LTL to NBA

For any LTL formula φ over propositions P, there exists an NBA $\mathcal{A}_\varphi$ with $\mathrm{Words}(\varphi) = \mathcal{L}(\mathcal{A}_\varphi)$ which can be constructed in time and space $2^{\mathcal{O}(|\varphi|)}$.

Sketch

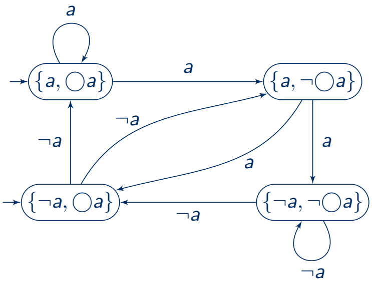
1. Construct the GNBA $\mathcal{G}_\varphi$
   - $|\mathrm{Closure}(\varphi)| = \mathcal{O}(|\varphi|)$ and $|Q| \leq 2^{|\mathrm{Closure}(\varphi)|} = 2^{\mathcal{O}(|\varphi|)}$
   - $\#$ accepting sets of $\mathcal{G}_\varphi$ $=$ $\#$ until-operators in $\varphi \leq \mathcal{O}(|\varphi|)$
2. Construct the NBA $\mathcal{A}_\varphi$
   - $\#$ states of $\mathcal{A}_\varphi = |Q| \times \#$ accepting sets of $\mathcal{G}_\varphi$
   - $\#$ states of $\mathcal{A}_\varphi \leq 2^{\mathcal{O}(|\varphi|)} \cdot \mathcal{O}(|\varphi|) = 2^{\mathcal{O}(|\varphi|)} \cdot 2^{\log(\mathcal{O}(|\varphi|))} = 2^{\mathcal{O}(|\varphi|)}$

From LTL to...NBA: better? (1/3)

The algorithm presented here is conceptually simple but may lead to unnecessary large GNBAs (and thus NBAs)

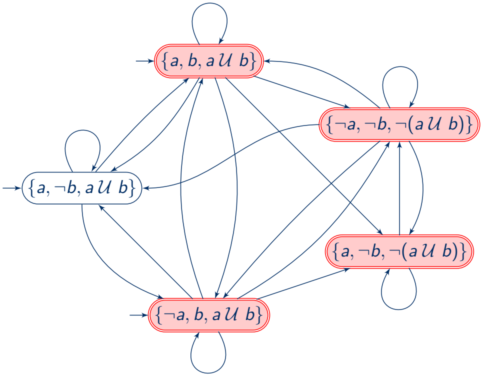
From LTL to. . . NBA: better? (1/3)

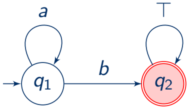
The algorithm presented here is conceptually simple but may lead to unnecessary large GNBAs (and thus NBAs)



Example: the right NBA also recognizes $\bigcirc a$ but is *smaller*

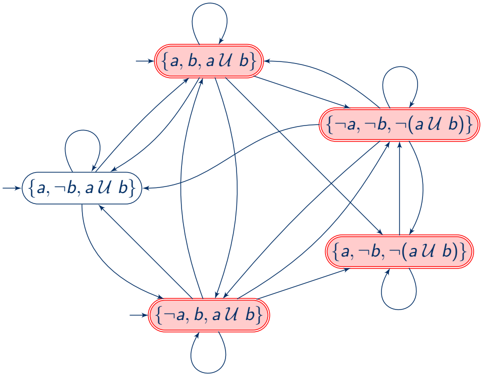
From LTL to... NBA: better? (2/3)

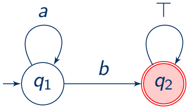




Example: the right NBA also recognizes $a\,\mathcal{U}\,b$ but is *much smaller*

From LTL to. . . NBA: better? (2/3)





Example: the right NBA also recognizes $a\,\mathcal{U}\,b$ but is *much smaller*

Can we always do better?

From LTL to…NBA: better? (3/3)

In practice, there exist more efficient (but more complex) algorithms in the literature

From LTL to. . . NBA: better? (3/3)

In practice, there exist more efficient (but more complex) algorithms in the literature

Still, **the exponential blowup cannot be avoided** in the worst-case!

Theorem: lower bound for NBA from LTL formula

There exists a family of LTL formulas φ_n with $|\varphi_n| = \mathcal{O}(poly(n))$ such that every NBA $\mathcal{A}_{\varphi_n}$ for φ_n has at least 2^n states.

From LTL to... NBA: better? (3/3)

In practice, there exist more efficient (but more complex) algorithms in the literature

Still, **the exponential blowup cannot be avoided** in the worst-case!

Theorem: lower bound for NBA from LTL formula

There exists a family of LTL formulas φ_n with $|\varphi_n| = \mathcal{O}(poly(n))$ such that every NBA $\mathcal{A}_{\varphi_n}$ for φ_n has at least 2^n states.

$\Longrightarrow$ **Proof in the next slides**

From LTL to...NBA: lower bound (1/2)

Let P be arbitrary and *non-empty*, i.e., $2^{|P|} \geq 2$. Let

$$\mathcal{L}_n = \left\{ a_1 \ldots a_n a_1 \ldots a_n \tau \;\middle|\; a_i \subseteq P \wedge \tau \in (2^P)^\omega \right\} \quad \text{for } n \geq 0.$$

University of Antwerp

From LTL to...NBA: lower bound (1/2)

Let P be arbitrary and *non-empty*, i.e., $2^{|P|} \geq 2$. Let

$$\mathcal{L}_n = \left\{ a_1 \ldots a_n a_1 \ldots a_n \tau \;\middle|\; a_i \subseteq P \;\wedge\; \tau \in (2^P)^\omega \right\} \quad \text{for } n \geq 0.$$

This language is expressible in LTL, i.e., $\mathcal{L}_n = \mathrm{Words}(\varphi_n)$ for

$$\varphi_n = \bigwedge_{a \in P} \; \bigwedge_{0 \leq i < n} (\bigcirc^i a \longleftrightarrow \bigcirc^{n+i} a).$$

Polynomial length: $|\varphi_n| = \mathcal{O}(|P| \cdot n^2)$.

From LTL to... NBA: lower bound (1/2)

Let P be arbitrary and *non-empty*, i.e., $2^{|P|} \geq 2$. Let

$$\mathcal{L}_n = \left\{ a_1 \ldots a_n a_1 \ldots a_n \tau \;\middle|\; a_i \subseteq P \;\wedge\; \tau \in (2^P)^\omega \right\} \quad \text{for } n \geq 0.$$

This language is expressible in LTL, i.e., $\mathcal{L}_n = \mathrm{Words}(\varphi_n)$ for

$$\varphi_n = \bigwedge_{a \in P} \; \bigwedge_{0 \leq i < n} (\bigcirc^i a \longleftrightarrow \bigcirc^{n+i} a).$$

Polynomial length: $|\varphi_n| = \mathcal{O}(|P| \cdot n^2)$.

Claim: any NBA $\mathcal{A}$ with $\mathcal{L}(\mathcal{A}) = \mathcal{L}_n$ has at least 2^n states.

From LTL to. . . NBA: lower bound (2/2)

Assume $\mathcal{A}$ is such an automaton. Words $a_1 \ldots a_n a_1 \ldots a_n \varnothing^\omega$ belong to $\mathcal{L}_n$, hence are accepted by $\mathcal{A}$.

From LTL to...NBA: lower bound (2/2)

Assume $\mathcal{A}$ is such an automaton. Words $a_1 \ldots a_n a_1 \ldots a_n \varnothing^\omega$ belong to $\mathcal{L}_n$, hence are accepted by $\mathcal{A}$.

- For every word $a_1 \ldots a_n$ of length n, $\mathcal{A}$ has a state $q(a_1 \ldots a_n)$ which can be reached after consuming $a_1 \ldots a_n$.

From LTL to... NBA: lower bound (2/2)

Assume $\mathcal{A}$ is such an automaton. Words $a_1 \dots a_n a_1 \dots a_n \varnothing^\omega$ belong to $\mathcal{L}_n$, hence are accepted by $\mathcal{A}$.

- For every word $a_1 \dots a_n$ of length n, $\mathcal{A}$ has a state $q(a_1 \dots a_n)$ which can be reached after consuming $a_1 \dots a_n$.
- From $q(a_1 \dots a_n)$, it is possible to visit an accepting state infinitely often by reading the suffix $a_1 \dots a_n \varnothing^\omega$.

From LTL to...NBA: lower bound (2/2)

Assume $\mathcal{A}$ is such an automaton. Words $a_1 \ldots a_n a_1 \ldots a_n \varnothing^\omega$ belong to $\mathcal{L}_n$, hence are accepted by $\mathcal{A}$.

- For every word $a_1 \ldots a_n$ of length n, $\mathcal{A}$ has a state $q(a_1 \ldots a_n)$ which can be reached after consuming $a_1 \ldots a_n$.
- From $q(a_1 \ldots a_n)$, it is possible to visit an accepting state infinitely often by reading the suffix $a_1 \ldots a_n \varnothing^\omega$.
- If $a_1 \ldots a_n \neq a'_1 \ldots a'_n$, then $a_1 \ldots a_n a'_1 \ldots a'_n \varnothing^\omega \notin \mathcal{L}_n = \mathcal{L}(\mathcal{A})$.

From LTL to... NBA: lower bound (2/2)

Assume $\mathcal{A}$ is such an automaton. Words $a_1 \dots a_n a_1 \dots a_n \varnothing^\omega$ belong to $\mathcal{L}_n$, hence are accepted by $\mathcal{A}$.

- For every word $a_1 \dots a_n$ of length n, $\mathcal{A}$ has a state $q(a_1 \dots a_n)$ which can be reached after consuming $a_1 \dots a_n$.
- From $q(a_1 \dots a_n)$, it is possible to visit an accepting state infinitely often by reading the suffix $a_1 \dots a_n \varnothing^\omega$.
- If $a_1 \dots a_n \neq a'_1 \dots a'_n$, then $a_1 \dots a_n a'_1 \dots a'_n \varnothing^\omega \notin \mathcal{L}_n = \mathcal{L}(\mathcal{A})$.
- Therefore, **states $q(a_1 \dots a_n)$ are all pairwise different**.

From LTL to... NBA: lower bound (2/2)

Assume $\mathcal{A}$ is such an automaton. Words $a_1 \ldots a_n a_1 \ldots a_n \varnothing^\omega$ belong to $\mathcal{L}_n$, hence are accepted by $\mathcal{A}$.

- For every word $a_1 \ldots a_n$ of length n, $\mathcal{A}$ has a state $q(a_1 \ldots a_n)$ which can be reached after consuming $a_1 \ldots a_n$.
- From $q(a_1 \ldots a_n)$, it is possible to visit an accepting state infinitely often by reading the suffix $a_1 \ldots a_n \varnothing^\omega$.
- If $a_1 \ldots a_n \neq a'_1 \ldots a'_n$, then $a_1 \ldots a_n a'_1 \ldots a'_n \varnothing^\omega \notin \mathcal{L}_n = \mathcal{L}(\mathcal{A})$.
- Therefore, **states $q(a_1 \ldots a_n)$ are all pairwise different**.
- Since each a_i can take $2^{|P|}$ different values, the number of different sequences $a_1 \ldots a_n$ of length n is $(2^{|P|})^n \geq 2^n$ (by non-emptiness of P).

From LTL to... NBA: lower bound (2/2)

Assume $\mathcal{A}$ is such an automaton. Words $a_1 \ldots a_n a_1 \ldots a_n \varnothing^\omega$ belong to $\mathcal{L}_n$, hence are accepted by $\mathcal{A}$.

- For every word $a_1 \ldots a_n$ of length n, $\mathcal{A}$ has a state $q(a_1 \ldots a_n)$ which can be reached after consuming $a_1 \ldots a_n$.
- From $q(a_1 \ldots a_n)$, it is possible to visit an accepting state infinitely often by reading the suffix $a_1 \ldots a_n \varnothing^\omega$.
- If $a_1 \ldots a_n \neq a'_1 \ldots a'_n$, then $a_1 \ldots a_n a'_1 \ldots a'_n \varnothing^\omega \notin \mathcal{L}_n = \mathcal{L}(\mathcal{A})$.
- Therefore, **states $q(a_1 \ldots a_n)$ are all pairwise different**.
- Since each a_i can take $2^{|P|}$ different values, the number of different sequences $a_1 \ldots a_n$ of length n is $(2^{|P|})^n \geq 2^n$ (by non-emptiness of P).
- Hence, **the NBA has at least 2^n states**.

LTL vs. NBAs

What have we learned?

LTL vs. NBAs

What have we learned?

Corollary

Every LTL formula expresses an ω-regular property, i.e., for all LTL formula φ, $\mathrm{Words}(\varphi)$ is an ω-regular language.

Why? Because LTL can be transformed to NBA and NBAs coincide with ω-regular languages.

LTL vs. NBAs

What have we learned?

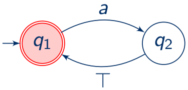
Corollary

Every LTL formula expresses an ω-regular property, i.e., for all LTL formula φ, $\mathrm{Words}(\varphi)$ is an ω-regular language.

Why? Because LTL can be transformed to NBA and NBAs coincide with ω-regular languages.

The converse is false!

Recall $\mathcal{L} = \left\{ a_0 a_1 a_2 \cdots \in (2^{\{a\}})^\omega \;\middle|\; \forall\, i \geq 0,\; a \in a_{2i} \right\}$.



LTL vs. NBAs

What have we learned?

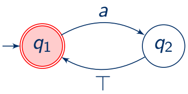
Corollary

Every LTL formula expresses an ω-regular property, i.e., for all LTL formula φ, $\mathrm{Words}(\varphi)$ is an ω-regular language.

Why? Because LTL can be transformed to NBA and NBAs coincide with ω-regular languages.

The converse is false!

Recall $\mathcal{L} = \left\{ a_0 a_1 a_2 \cdots \in (2^{\{a\}})^\omega \;\middle|\; \forall\, i \geq 0,\ a \in a_{2i} \right\}$.



$\Longrightarrow$ **There are ω-regular properties not expressible in LTL.**

Model checking algorithm for LTL

$$
\begin{aligned}
\mathcal{T} \models \varphi \quad & \text{iff} \quad \mathrm{Traces}(\mathcal{T}) \subseteq \mathrm{Words}(\varphi) \\
& \text{iff} \quad \mathrm{Traces}(\mathcal{T}) \cap ((2^{P})^{\omega} \setminus \mathrm{Words}(\varphi)) = \varnothing \\
& \text{iff} \quad \mathrm{Traces}(\mathcal{T}) \cap \mathrm{Words}(\neg\varphi) = \varnothing \\
& \text{iff} \quad \mathrm{Traces}(\mathcal{T}) \cap \mathcal{L}(\mathcal{A}_{\neg\varphi}) = \varnothing \\
& \text{iff} \quad \mathcal{T} \otimes \mathcal{A}_{\neg\varphi} \models \Diamond\Box\neg F
\end{aligned}
$$

Model checking algorithm for LTL

$$
\begin{aligned}
\mathcal{T} \models \varphi \qquad & \text{iff} \quad \mathrm{Traces}(\mathcal{T}) \subseteq \mathrm{Words}(\varphi) \\
& \text{iff} \quad \mathrm{Traces}(\mathcal{T}) \cap ((2^{P})^{\omega} \setminus \mathrm{Words}(\varphi)) = \varnothing \\
& \text{iff} \quad \mathrm{Traces}(\mathcal{T}) \cap \mathrm{Words}(\neg\varphi) = \varnothing \\
& \text{iff} \quad \mathrm{Traces}(\mathcal{T}) \cap \mathcal{L}(\mathcal{A}_{\neg\varphi}) = \varnothing \\
& \text{iff} \quad \mathcal{T} \otimes \mathcal{A}_{\neg\varphi} \models \Diamond\Box\neg F
\end{aligned}
$$

It remains to consider the last line

Two remaining questions:

1. How to compute the product TS $\mathcal{T} \otimes \mathcal{A}_{\neg\varphi}$?
2. How to check persistence, i.e., $\mathcal{T} \otimes \mathcal{A}_{\neg\varphi} \models \Diamond\Box\neg F$?

Product of TS and NBA

Definition: product of TS and NBA

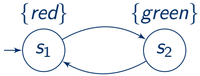
Let $\mathcal{T} = (S, A, \longrightarrow, I, P, L)$ be a TS without terminal states and $\mathcal{A} = (Q, A = 2^P, \delta, I_\mathcal{A}, F)$ a non-blocking NBA. Then, $\mathcal{T} \otimes \mathcal{A}$ is the following TS:

$$\mathcal{T} \otimes \mathcal{A} = (S', A, \longrightarrow', I', P', L') \text{ where}$$

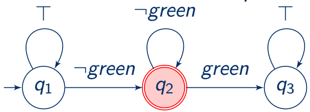
- $S' = S \times Q$, $P' = Q$ and $L'(\langle s, q\rangle) = \{q\}$,
- $\longrightarrow'$ is the smallest relation such that if $s \xrightarrow{a} t$ and $q \xrightarrow{L(t)} p$, then $\langle s, q\rangle \xrightarrow{a}' \langle t, p\rangle$,
- $I' = \{\langle s_0, q\rangle \mid s_0 \in I \wedge \exists\, q_0 \in I_\mathcal{A},\ q_0 \xrightarrow{L(s_0)} q\}$.

Product of TS and NBA: example

Simple traffic light with two modes: *red* and *green*. LTL formula to check $\varphi = \Box\Diamond green$.



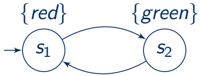
TS $\mathcal{T}$ for the traffic light



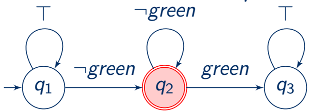
NBA $\mathcal{A}_{\neg\varphi}$ for $\neg\varphi = \Diamond\Box\neg green$

Product of TS and NBA: example

Simple traffic light with two modes: *red* and *green*. LTL formula to check $\varphi = \Box\Diamond green$.



TS $\mathcal{T}$ for the traffic light

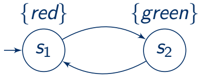


NBA $\mathcal{A}_{\neg\varphi}$ for $\neg\varphi = \Diamond\Box\neg green$

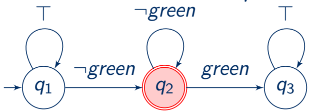
$\Longrightarrow$ **Blackboard construction of $\mathcal{T} \otimes \mathcal{A}_{\neg\varphi}$**

Product of TS and NBA: example

Simple traffic light with two modes: *red* and *green*. LTL formula to check $\varphi = \Box\Diamond green$.

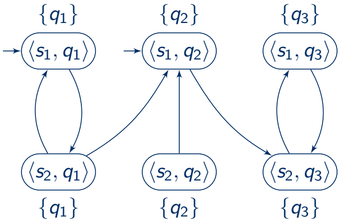


TS $\mathcal{T}$ for the traffic light



NBA $\mathcal{A}_{\neg\varphi}$ for $\neg\varphi = \Diamond\Box\neg green$

$\Longrightarrow$ **Blackboard construction of $\mathcal{T} \otimes \mathcal{A}_{\neg\varphi}$**



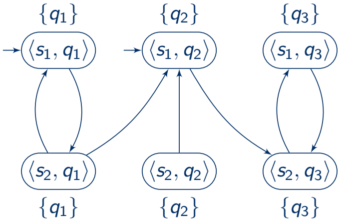
Persistence checking: illustration (1/2)

It remains to check $\mathcal{T} \otimes \mathcal{A}_{\neg\varphi} \models \Diamond\Box\neg F$ to see that $\mathcal{T} \models \varphi$.

University of Antwerp

Persistence checking: illustration (1/2)

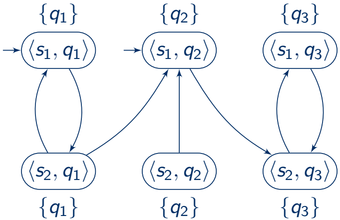
It remains to check $\mathcal{T} \otimes \mathcal{A}_{\neg\varphi} \models \Diamond\Box\neg F$ to see that $\mathcal{T} \models \varphi$.



Here, $\mathcal{T} \otimes \mathcal{A}_{\neg\varphi} \overset{?}{\models} \Diamond\Box\neg F$ with $F = \{q_2\}$.

Persistence checking: illustration (1/2)

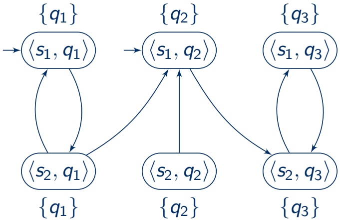
It remains to check $\mathcal{T} \otimes \mathcal{A}_{\neg\varphi} \models \Diamond\Box\neg F$ to see that $\mathcal{T} \models \varphi$.



Here, $\mathcal{T} \otimes \mathcal{A}_{\neg\varphi} \overset{?}{\models} \Diamond\Box\neg F$ with $F = \{q_2\}$. **Yes! State $\langle s_1, q_2\rangle$ can be seen at most once, and state $\langle s_2, q_2\rangle$ is not reachable.**

Persistence checking: illustration (1/2)

It remains to check $\mathcal{T} \otimes \mathcal{A}_{\neg\varphi} \models \lozenge\square\neg F$ to see that $\mathcal{T} \models \varphi$.

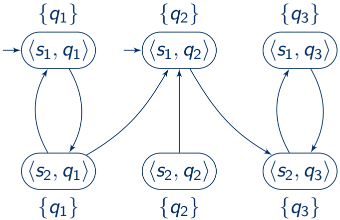


Here, $\mathcal{T} \otimes \mathcal{A}_{\neg\varphi} \overset{?}{\models} \lozenge\square\neg F$ with $F = \{q_2\}$. **Yes! State $\langle s_1, q_2 \rangle$ can be seen at most once, and state $\langle s_2, q_2 \rangle$ is not reachable.**

$\Longrightarrow$ **There is no common trace between $\mathcal{T}$ and $\mathcal{A}_{\neg\varphi}$.**

Persistence checking: illustration (1/2)

It remains to check $\mathcal{T} \otimes \mathcal{A}_{\neg\varphi} \models \Diamond\Box\neg F$ to see that $\mathcal{T} \models \varphi$.



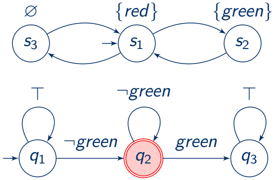
Here, $\mathcal{T} \otimes \mathcal{A}_{\neg\varphi} \overset{?}{\models} \Diamond\Box\neg F$ with $F = \{q_2\}$. **Yes! State $\langle s_1, q_2\rangle$ can be seen at most once, and state $\langle s_2, q_2\rangle$ is not reachable.**

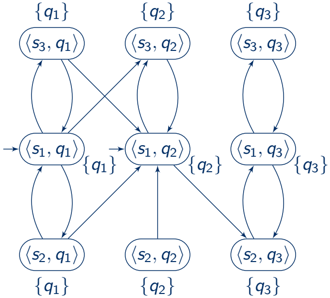
$\Longrightarrow$ **There is no common trace between $\mathcal{T}$ and $\mathcal{A}_{\neg\varphi}$.**

$\Longrightarrow \mathcal{T} \models \varphi$.

Persistence checking: illustration (2/2)

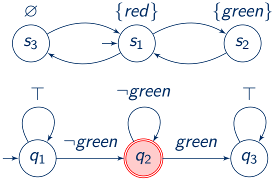
Slightly revised traffic light: can switch off to save energy. Same formula φ (hence same NBA $\mathcal{A}_{\neg\varphi}$).

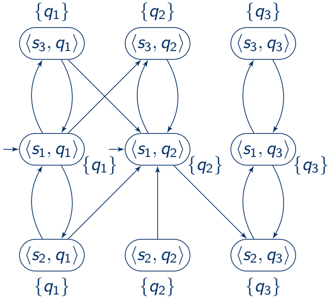




Persistence checking: illustration (2/2)

Slightly revised traffic light: can switch off to save energy. Same formula φ (hence same NBA $\mathcal{A}_{\neg\varphi}$).





Here, $\mathcal{T} \otimes \mathcal{A}_{\neg\varphi} \not\models \Diamond\Box\neg F$ with $F = \{q_2\}$. See for example path $\langle s_1, q_1\rangle\,(\langle s_3, q_2\rangle\,\langle s_1, q_2\rangle)^{\omega}$ that visits q_2 infinitely often.

Persistence checking: cycle detection

As for checking language non-emptiness of NBA, we reduce the problem to a cycle detection problem.

Persistence checking and cycle detection

Let $\mathcal{T}$ be a TS without terminal states over P and φ a *propositional* formula over P, then

$$\mathcal{T} \not\models \Diamond\Box\varphi$$
$$\Updownarrow$$
$$\exists\, s \in \mathrm{Reach}(\mathcal{T}),\ s \not\models \varphi \text{ and } s \text{ is on a cycle in the graph of } \mathcal{T}.$$

Persistence checking: cycle detection

As for checking language non-emptiness of NBA, we reduce the problem to a cycle detection problem.

Persistence checking and cycle detection

Let $\mathcal{T}$ be a TS without terminal states over P and φ a *propositional* formula over P, then

$$\mathcal{T} \not\models \Diamond\Box\varphi$$
$$\Updownarrow$$
$$\exists\, s \in \mathrm{Reach}(\mathcal{T}),\ s \not\models \varphi \text{ and } s \text{ is on a cycle in the graph of } \mathcal{T}.$$

In particular, it holds for $\varphi = \neg F$ as needed for LTL model checking (with F the acceptance set of the NBA $\mathcal{A}_{\neg\varphi}$).

Persistence checking: cycle detection

1. Compute the reachable SCCs and check if one contains a state satisfying $\neg\varphi$.
   $\hookrightarrow$ Linear time but requires to construct entirely the product TS $\mathcal{T} \otimes \mathcal{A}_{\neg\varphi}$ which may be very large (exponential)

Persistence checking: cycle detection

1. Compute the reachable SCCs and check if one contains a state satisfying $\neg\varphi$.
   ↪ Linear time but requires to construct entirely the product TS $\mathcal{T} \otimes \mathcal{A}_{\neg\varphi}$ which may be very large (exponential)
2. Another solution: **on-the-fly algorithms**
   - Construct $\mathcal{T}$ and $\mathcal{A}_{\neg\varphi}$ in parallel and simultaneously construct the reachable fragment of $\mathcal{T} \otimes \mathcal{A}_{\neg\varphi}$ via nested depth-first search.

   ↪ Construction of the product "on demand".
   ↪ **More efficient in practice** (used in software solutions such as Spin).

Persistence checking: cycle detection

1. Compute the reachable SCCs and check if one contains a state satisfying $\neg\varphi$.
   $\hookrightarrow$ Linear time but requires to construct entirely the product TS $\mathcal{T} \otimes \mathcal{A}_{\neg\varphi}$ which may be very large (exponential)
2. Another solution: **on-the-fly algorithms**
   - Construct $\mathcal{T}$ and $\mathcal{A}_{\neg\varphi}$ in parallel and simultaneously construct the reachable fragment of $\mathcal{T} \otimes \mathcal{A}_{\neg\varphi}$ via nested depth-first search.

   $\hookrightarrow$ Construction of the product "on demand".
   $\hookrightarrow$ **More efficient in practice** (used in software solutions such as Spin).

Still, the complexity of LTL model checking remains high!

Wrap-up of the automata-based approach

$$
\begin{aligned}
\mathcal{T} \models \varphi \quad & \text{iff} \quad \mathrm{Traces}(\mathcal{T}) \subseteq \mathrm{Words}(\varphi) \\
& \text{iff} \quad \mathrm{Traces}(\mathcal{T}) \cap ((2^P)^\omega \setminus \mathrm{Words}(\varphi)) = \varnothing \\
& \text{iff} \quad \mathrm{Traces}(\mathcal{T}) \cap \mathrm{Words}(\neg\varphi) = \varnothing \\
& \text{iff} \quad \mathrm{Traces}(\mathcal{T}) \cap \mathcal{L}(\mathcal{A}_{\neg\varphi}) = \varnothing \\
& \text{iff} \quad \mathcal{T} \otimes \mathcal{A}_{\neg\varphi} \models \Diamond\Box\neg F
\end{aligned}
$$

Complexity of this approach

The time and space complexity is $\mathcal{O}(|\mathcal{T}|) \cdot 2^{\mathcal{O}(|\varphi|)}$.

Complexity of LTL model checking

Complexity of the model checking problem for LTL

The LTL model checking problem is PSPACE-complete.

$\Longrightarrow$ **See the book for a proof by reduction from the membership problem for polynomial-space deterministic Turing machines.**

Summary and conclusions

Automata, languages, expressions, and logic

- We have introduced an automata-based framework that allows us to check whether a system satisfies an LTL specification.
- Our main tool is a translation from LTL to NBAs.

Model checking

A TS can be model checked against an LTL formula in PSPACE using an on-the-fly algorithm.

- The sizes of the state-sets are huge in the automaton, thus huge in the product!
- How does one implement some of these algorithms?