

Introduction to Model Checking

René Thiemann

Institute of Computer Science
University of Innsbruck

WS 2011/2012



Outline

- LTL - Linear Time Logic
 - Syntax and Semantics
 - Equivalences

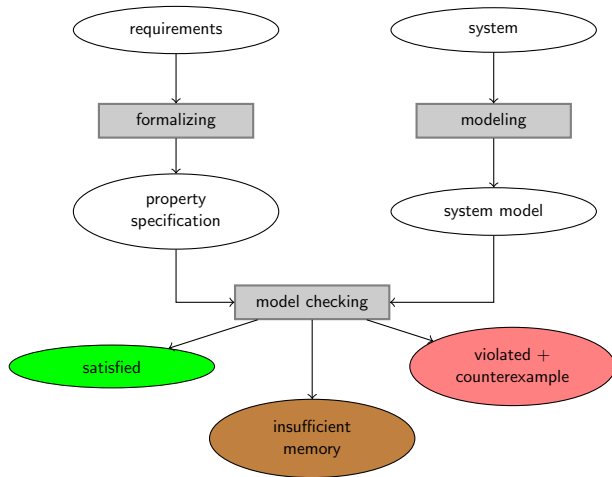
- LTL Model Checking
 - Overview
 - Transforming LTL into GNBA's
 - Complexity of LTL Model Checking

Outline

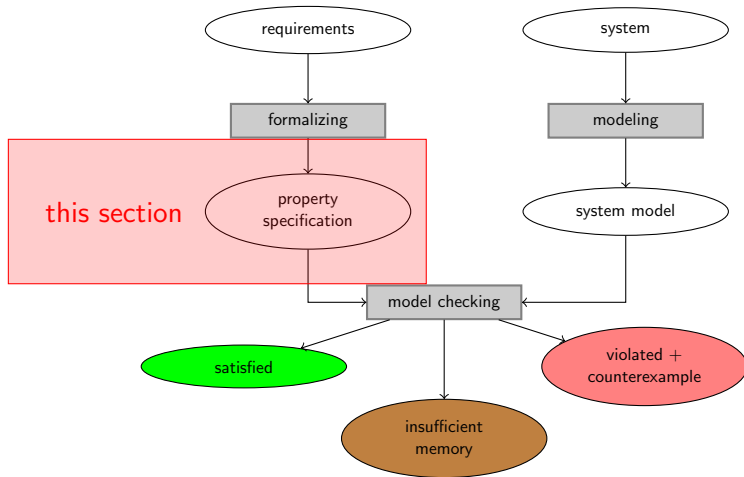
- LTL - Linear Time Logic
 - Syntax and Semantics
 - Equivalences

- LTL Model Checking
 - Overview
 - Transforming LTL into GNBA's
 - Complexity of LTL Model Checking

Model checking overview



Model checking overview



Motivation

- so far, we used GNBA as requirements specification
- ⇒ larger specification become unreadable
(think of an intersection GNBA of four properties)
- ⇒ need nicer way to precisely state requirements
- ⇒ LTL

Syntax of Linear Temporal Logic

modal logic over infinite sequences [Pnueli 1977]

Syntax of Linear Temporal Logic

modal logic over infinite sequences [Pnueli 1977]

- propositional logic

- true
- $a, \text{paid}, \text{sprite}, \dots \in AP$
- $\neg\varphi$ and $\varphi \wedge \psi$

atomic proposition
negation and conjunction

Syntax of Linear Temporal Logic

modal logic over infinite sequences [Pnueli 1977]

- **propositional logic**

- true
- $a, \text{paid}, \text{sprite}, \dots \in AP$
- $\neg\varphi$ and $\varphi \wedge \psi$

atomic proposition
negation and conjunction

- **temporal operators**

- $X\varphi$

neXt step fulfills φ

Syntax of Linear Temporal Logic

modal logic over infinite sequences [Pnueli 1977]

- **propositional logic**

- true
- $a, \text{paid}, \text{sprite}, \dots \in AP$
- $\neg\varphi$ and $\varphi \wedge \psi$

atomic proposition
negation and conjunction

- **temporal operators**

- $X\varphi$
- $F\varphi$

neXt step fulfills φ
sometimes in the Future φ will hold

Syntax of Linear Temporal Logic

modal logic over infinite sequences [Pnueli 1977]

- **propositional logic**

- true
- $a, \text{paid}, \text{sprite}, \dots \in AP$
- $\neg\varphi$ and $\varphi \wedge \psi$

atomic proposition
negation and conjunction

- **temporal operators**

- $X\varphi$
- $F\varphi$
- $G\varphi$

ne**X**t step fulfills φ
sometimes in the **F**uture φ will hold
 φ **G**lobally holds

Syntax of Linear Temporal Logic

modal logic over infinite sequences [Pnueli 1977]

- **propositional logic**

- true
- $a, \text{paid}, \text{sprite}, \dots \in AP$
- $\neg\varphi$ and $\varphi \wedge \psi$

atomic proposition
negation and conjunction

- **temporal operators**

- $X\varphi$
- $F\varphi$
- $G\varphi$
- $\varphi U \psi$

ne**X**t step fulfills φ
sometimes in the **F**uture φ will hold
 φ **G**lobally holds
 φ holds **U**ntil ψ holds

Syntax of Linear Temporal Logic

modal logic over infinite sequences [Pnueli 1977]

- **propositional logic**

- true
- $a, \text{paid}, \text{sprite}, \dots \in AP$
- $\neg\varphi$ and $\varphi \wedge \psi$

atomic proposition
negation and conjunction

- **temporal operators**

- $X\varphi$
- $F\varphi$
- $G\varphi$
- $\varphi U \psi$

ne**X**t step fulfills φ
sometimes in the **F**uture φ will hold
 φ **G**lobally holds
 φ holds **U**ntil ψ holds

linear temporal logic is a logic for describing linear time properties

Syntax of Linear Temporal Logic

modal logic over infinite sequences [Pnueli 1977]

- **propositional logic**

- true
- $a, \text{paid}, \text{sprite}, \dots \in AP$
- $\neg\varphi$ and $\varphi \wedge \psi$

atomic proposition
negation and conjunction

- **temporal operators**

- $X\varphi$
- $F\varphi$
- $G\varphi$
- $\varphi U \psi$

ne**X**t step fulfills φ
sometimes in the **F**uture φ will hold
 φ **G**lobally holds
 φ holds **U**ntil ψ holds

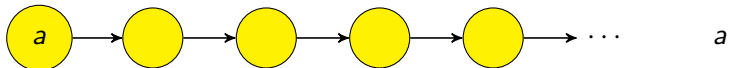
linear temporal logic is a logic for describing linear time properties

precedence order: the unary operators bind stronger than the binary ones.

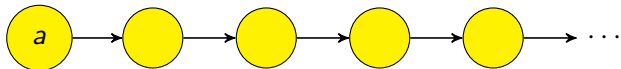
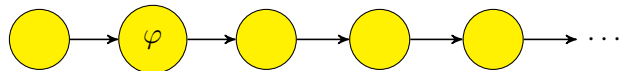
\neg and X bind equally strong. U takes precedence over \wedge , \vee , and \Rightarrow

$$\neg, X, F, G > U > \wedge, \vee, \Rightarrow, \Leftrightarrow$$

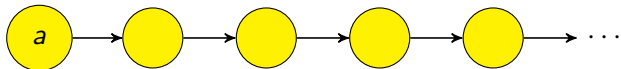
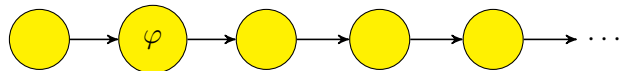
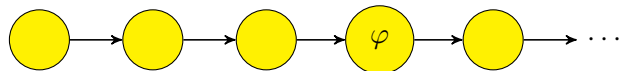
Intuitive semantics



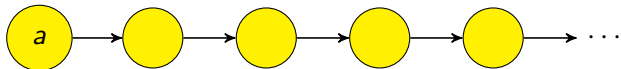
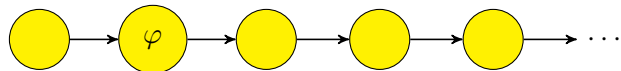
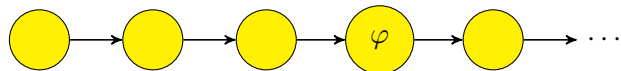
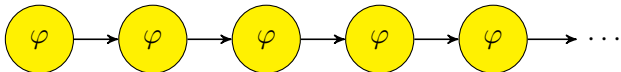
Intuitive semantics

 a  $X\varphi$

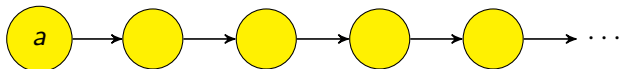
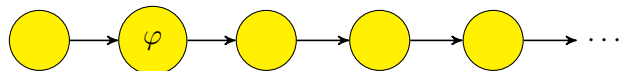
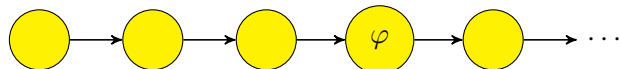
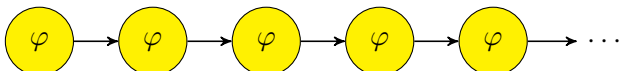
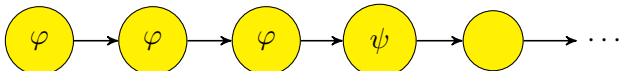
Intuitive semantics

 a  $X\varphi$  $F\varphi$

Intuitive semantics

 a  $X\varphi$  $F\varphi$  $G\varphi$

Intuitive semantics

 a  $X\varphi$  $F\varphi$  $G\varphi$  $\varphi U \psi$

Derived Boolean operators

$$\text{false} \equiv \neg \text{true}$$

$$\varphi \vee \psi \equiv \neg(\neg\varphi \wedge \neg\psi)$$

$$\varphi \Rightarrow \psi \equiv \neg\varphi \vee \psi$$

$$\varphi \Leftrightarrow \psi \equiv (\varphi \Rightarrow \psi) \wedge (\psi \Rightarrow \varphi)$$

$$\varphi \oplus \psi \equiv \neg(\varphi \Leftrightarrow \psi)$$

Practical properties in LTL

- Reachability

 $F \psi$

Practical properties in LTL

- Reachability
- Safety

$F \psi$
 $G \neg \varphi$

Practical properties in LTL

- Reachability
- Safety
- Liveness

 $F \psi$ $G \neg \varphi$ $G(\varphi \Rightarrow F \psi)$ and others

Practical properties in LTL

- Reachability
- Safety
- Liveness
- Fairness

 $F \psi$ $G \neg \varphi$ $G(\varphi \Rightarrow F \psi)$ and others $GF \varphi$ and others

Properties of a traffic light

- the light is never red and green: $\neg(\text{red} \wedge \text{green})$

Properties of a traffic light

- the light is never red and green: $\neg(\text{red} \wedge \text{green})$
- whenever the light is red, it cannot become green immediately afterwards:

$$\text{red} \Rightarrow \neg X \text{green}$$

Properties of a traffic light

- the light is never red and green: $\neg(\text{red} \wedge \text{green})$
- whenever the light is red, it cannot become green immediately afterwards:

$$\text{red} \Rightarrow \neg X \text{green}$$

- eventually, the light becomes green: $F \text{green}$

Properties of a traffic light

- the light is never red and green: $\neg(\text{red} \wedge \text{green})$
- whenever the light is red, it cannot become green immediately afterwards:

$$\text{red} \Rightarrow \neg X \text{green}$$

- eventually, the light becomes green: $F \text{green}$
- green holds until red appears problem in req.: should mention orange in between and at sometime orange appears; moreover, the red is later than the orange

$$(\text{green} \text{ U } \text{red}) \wedge F(\text{orange} \wedge X F \text{red})$$

consider {green} {red} {orange} {red} ...

$$\text{green} \text{ U } (\text{orange} \wedge X(\text{orange} \text{ U } \text{red}))$$

Properties of a traffic light

- the light is never red and green: $\neg(\text{red} \wedge \text{green})$
- whenever the light is red, it cannot become green immediately afterwards:

$$\text{red} \Rightarrow \neg X \text{green}$$

- eventually, the light becomes green: $F \text{green}$
- green holds until red appears problem in req.: should mention orange in between and at sometime orange appears; moreover, the red is later than the orange

$$(\text{green} \text{ U } \text{red}) \wedge F(\text{orange} \wedge X F \text{red})$$

consider {green} {red} {orange} {red} ...

$$\text{green} \text{ U } (\text{orange} \wedge X(\text{orange} \text{ U } \text{red}))$$

most of these requirements do not correspond to the specifications

Properties of a traffic light

- the light is never red and green: $G(\neg(\text{red} \wedge \text{green}))$
- whenever the light is red, it cannot become green immediately afterwards:

$$G(\text{red} \Rightarrow \neg X \text{green})$$

- eventually, the light becomes green: $F \text{green}$
- green holds until red appears problem in req.: should mention orange in between and at sometime orange appears; moreover, the red is later than the orange

$$(\text{green} \text{ U } \text{red}) \wedge F(\text{orange} \wedge X F \text{red})$$

consider {green} {red} {orange} {red} ...

$$\text{green} \text{ U } (\text{orange} \wedge X(\text{orange} \text{ U } \text{red}))$$

most of these requirements do not correspond to the specifications

Requirements from Chapter 2, slide 27

1. always at most one traffic light is showing green

$$G (\neg \text{green1} \vee \neg \text{green2})$$

2. green cannot be directly followed by red

$$G (\text{green} \Rightarrow X \neg \text{red})$$

3. we will see green infinitely often

$$G F \text{ green}$$

4. whenever we select sprite then later on we will get a sprite

$$G (\text{select} \Rightarrow X F \text{ get})$$

Semantics over words

the language induced by LTL formula φ over $AP = \{a_1, \dots, a_n\}$ is:

$$\mathcal{L}(\varphi) = \left\{ w \in \left(2^{AP} \right)^\omega \mid w \models \varphi \right\}, \text{ where } \models \text{ is defined as follows:}$$

Semantics over words

the language induced by LTL formula φ over $AP = \{a_1, \dots, a_n\}$ is:

$$\mathcal{L}(\varphi) = \left\{ w \in \left(2^{AP} \right)^\omega \mid w \models \varphi \right\}, \text{ where } \models \text{ is defined as follows:}$$

(let $w = A_0A_1A_2\dots$ and $w[i..] = A_iA_{i+1}A_{i+2}\dots$ is the suffix of w from index i on)

$$w \models \text{true}$$

Semantics over words

the language induced by LTL formula φ over $AP = \{a_1, \dots, a_n\}$ is:

$$\mathcal{L}(\varphi) = \left\{ w \in \left(2^{AP} \right)^\omega \mid w \models \varphi \right\}, \text{ where } \models \text{ is defined as follows:}$$

(let $w = A_0A_1A_2\dots$ and $w[i..] = A_iA_{i+1}A_{i+2}\dots$ is the suffix of w from index i on)

$$w \models \text{true}$$

$$w \models a_i \quad \text{iff} \quad a_i \in A_0$$

Semantics over words

the language induced by LTL formula φ over $AP = \{a_1, \dots, a_n\}$ is:

$$\mathcal{L}(\varphi) = \left\{ w \in \left(2^{AP} \right)^\omega \mid w \models \varphi \right\}, \text{ where } \models \text{ is defined as follows:}$$

(let $w = A_0A_1A_2\dots$ and $w[i..] = A_iA_{i+1}A_{i+2}\dots$ is the suffix of w from index i on)

$$w \models \text{true}$$

$$w \models a_i \quad \text{iff} \quad a_i \in A_0$$

$$w \models \varphi_1 \wedge \varphi_2 \quad \text{iff} \quad w \models \varphi_1 \text{ and } w \models \varphi_2$$

Semantics over words

the language induced by LTL formula φ over $AP = \{a_1, \dots, a_n\}$ is:

$$\mathcal{L}(\varphi) = \left\{ w \in \left(2^{AP} \right)^\omega \mid w \models \varphi \right\}, \text{ where } \models \text{ is defined as follows:}$$

(let $w = A_0A_1A_2\dots$ and $w[i..] = A_iA_{i+1}A_{i+2}\dots$ is the suffix of w from index i on)

$$w \models \text{true}$$

$$w \models a_i \quad \text{iff} \quad a_i \in A_0$$

$$w \models \varphi_1 \wedge \varphi_2 \quad \text{iff} \quad w \models \varphi_1 \text{ and } w \models \varphi_2$$

$$w \models \neg \varphi \quad \text{iff} \quad w \not\models \varphi$$

Semantics over words

the language induced by LTL formula φ over $AP = \{a_1, \dots, a_n\}$ is:

$$\mathcal{L}(\varphi) = \left\{ w \in \left(2^{AP} \right)^\omega \mid w \models \varphi \right\}, \text{ where } \models \text{ is defined as follows:}$$

(let $w = A_0A_1A_2\dots$ and $w[i..] = A_iA_{i+1}A_{i+2}\dots$ is the suffix of w from index i on)

$$w \models \text{true}$$

$$w \models a_i \quad \text{iff} \quad a_i \in A_0$$

$$w \models \varphi_1 \wedge \varphi_2 \quad \text{iff} \quad w \models \varphi_1 \text{ and } w \models \varphi_2$$

$$w \models \neg \varphi \quad \text{iff} \quad w \not\models \varphi$$

$$w \models X\varphi \quad \text{iff} \quad w[1..] = A_1A_2A_3\dots \models \varphi$$

Semantics over words

the language induced by LTL formula φ over $AP = \{a_1, \dots, a_n\}$ is:

$$\mathcal{L}(\varphi) = \left\{ w \in \left(2^{AP} \right)^\omega \mid w \models \varphi \right\}, \text{ where } \models \text{ is defined as follows:}$$

(let $w = A_0A_1A_2\dots$ and $w[i..] = A_iA_{i+1}A_{i+2}\dots$ is the suffix of w from index i on)

$$w \models \text{true}$$

$$w \models a_i \quad \text{iff} \quad a_i \in A_0$$

$$w \models \varphi_1 \wedge \varphi_2 \quad \text{iff} \quad w \models \varphi_1 \text{ and } w \models \varphi_2$$

$$w \models \neg \varphi \quad \text{iff} \quad w \not\models \varphi$$

$$w \models X\varphi \quad \text{iff} \quad w[1..] = A_1A_2A_3\dots \models \varphi$$

$$w \models \varphi_1 \text{ U } \varphi_2 \quad \text{iff} \quad \exists j \geq 0. w[j..] \models \varphi_2 \text{ and } \forall 0 \leq i < j : w[i..] \models \varphi_1$$

Semantics over words

the language induced by LTL formula φ over $AP = \{a_1, \dots, a_n\}$ is:

$$\mathcal{L}(\varphi) = \left\{ w \in \left(2^{AP} \right)^\omega \mid w \models \varphi \right\}, \text{ where } \models \text{ is defined as follows:}$$

(let $w = A_0A_1A_2\dots$ and $w[i..] = A_iA_{i+1}A_{i+2}\dots$ is the suffix of w from index i on)

$$w \models \text{true}$$

$$w \models a_i \quad \text{iff} \quad a_i \in A_0$$

$$w \models \varphi_1 \wedge \varphi_2 \quad \text{iff} \quad w \models \varphi_1 \text{ and } w \models \varphi_2$$

$$w \models \neg \varphi \quad \text{iff} \quad w \not\models \varphi$$

$$w \models X\varphi \quad \text{iff} \quad w[1..] = A_1A_2A_3\dots \models \varphi$$

$$w \models \varphi_1 \text{ U } \varphi_2 \quad \text{iff} \quad \exists j \geq 0. w[j..] \models \varphi_2 \text{ and } \forall 0 \leq i < j : w[i..] \models \varphi_1$$

$$w \models F\varphi \quad \text{iff} \quad \exists j \geq 0. w[j..] \models \varphi$$

Semantics over words

the language induced by LTL formula φ over $AP = \{a_1, \dots, a_n\}$ is:

$$\mathcal{L}(\varphi) = \left\{ w \in \left(2^{AP} \right)^\omega \mid w \models \varphi \right\}, \text{ where } \models \text{ is defined as follows:}$$

(let $w = A_0A_1A_2\dots$ and $w[i..] = A_iA_{i+1}A_{i+2}\dots$ is the suffix of w from index i on)

$$w \models \text{true}$$

$$w \models a_i \quad \text{iff} \quad a_i \in A_0$$

$$w \models \varphi_1 \wedge \varphi_2 \quad \text{iff} \quad w \models \varphi_1 \text{ and } w \models \varphi_2$$

$$w \models \neg \varphi \quad \text{iff} \quad w \not\models \varphi$$

$$w \models X\varphi \quad \text{iff} \quad w[1..] = A_1A_2A_3\dots \models \varphi$$

$$w \models \varphi_1 \text{ U } \varphi_2 \quad \text{iff} \quad \exists j \geq 0. w[j..] \models \varphi_2 \text{ and } \forall 0 \leq i < j : w[i..] \models \varphi_1$$

$$w \models F\varphi \quad \text{iff} \quad \exists j \geq 0. w[j..] \models \varphi$$

$$w \models G\varphi \quad \text{iff} \quad \forall j \geq 0. w[j..] \models \varphi$$

Often used constructs

- $w \models \mathbf{GF}\varphi$ iff $\forall i : w[i..] \models \mathbf{F}\varphi$
 iff $\forall i \exists j : w[i..][j..] \models \varphi$
 iff $\forall i \exists j : w[i + j..] \models \varphi$
 iff $\forall i \exists j \geq i : w[j..] \models \varphi$
 iff infinitely often φ is satisfied in w
- $w \models \mathbf{FG}\varphi$ iff $\exists i \forall j \geq i : w[j..] \models \varphi$ iff
 from some point onwards φ is always satisfied in w

Semantics for Transition Systems

semantics is defined via set inclusion as indicated in previous section, so a system satisfies a formula iff all traces are allowed w.r.t. the formula:

$$TS \models \varphi \text{ iff } \text{Traces}(TS) \subseteq \mathcal{L}(\varphi)$$

A note on negations

for trace w , it holds $w \models \varphi$ iff $w \not\models \neg\varphi$ since

$$\mathcal{L}(\neg\varphi) = (2^{AP})^\omega \setminus \mathcal{L}(\varphi)$$

A note on negations

for trace w , it holds $w \models \varphi$ iff $w \not\models \neg\varphi$ since

$$\mathcal{L}(\neg\varphi) = (2^{AP})^\omega \setminus \mathcal{L}(\varphi)$$

but: $TS \not\models \varphi$ and $TS \models \neg\varphi$ are **not** equivalent in general

usually it holds: $TS \models \neg\varphi$ implies $TS \not\models \varphi$ but not always the reverse!

usually = TS must contain at least one trace

A note on negations

for trace w , it holds $w \models \varphi$ iff $w \not\models \neg\varphi$ since

$$\mathcal{L}(\neg\varphi) = (2^{AP})^\omega \setminus \mathcal{L}(\varphi)$$

but: $TS \not\models \varphi$ and $TS \models \neg\varphi$ are **not** equivalent in general

usually it holds: $TS \models \neg\varphi$ implies $TS \not\models \varphi$ but not always the reverse!

usually = TS must contain at least one trace

example: a transition system for which $TS \not\models F a$ and $TS \not\models \neg F a$



Equivalence of LTL formulas

Definition (Equivalence)

LTL formulas φ, ψ are **equivalent**, denoted $\varphi \equiv \psi$, iff

$$\mathcal{L}(\varphi) = \mathcal{L}(\psi)$$

Equivalence of LTL formulas

Definition (Equivalence)

LTL formulas φ, ψ are **equivalent**, denoted $\varphi \equiv \psi$, iff

$$\mathcal{L}(\varphi) = \mathcal{L}(\psi)$$

equivalences are useful to

- simplify formulas
- reduce the number of operators in algorithms / proofs

Equivalence of LTL formulas

Definition (Equivalence)

LTL formulas φ, ψ are **equivalent**, denoted $\varphi \equiv \psi$, iff

$$\mathcal{L}(\varphi) = \mathcal{L}(\psi)$$

equivalences are useful to

- simplify formulas
- reduce the number of operators in algorithms / proofs

proving a new equivalence

- is a proof of set-equality $\mathcal{L}(\varphi) = \mathcal{L}(\psi)$
- can be done by showing for all words w that $w \in \mathcal{L}(\varphi)$ iff $w \in \mathcal{L}(\psi)$

Deriving F and G

- $F\varphi \equiv \text{true} \cup \varphi$

$$w \models F\varphi$$

$$\text{iff } \exists j \geq 0. w[j..] \models \varphi$$

$$\text{iff } \exists j \geq 0. w[j..] \models \varphi \text{ and } \forall 0 \leq i < j : w[i..] \models \text{true}$$

$$\text{iff } w \models \text{true} \cup \varphi$$

- $G\varphi \equiv \neg(F\neg\varphi) \equiv \neg(\text{true} \cup \neg\varphi)$

$$w \models G\varphi$$

$$\text{iff } \forall j \geq 0. w[j..] \models \varphi$$

$$\text{iff } \text{not } (\text{not } \forall j \geq 0. w[j..] \models \varphi)$$

$$\text{iff } \text{not } \exists j \geq 0. w[j..] \not\models \varphi$$

$$\text{iff } \text{not } w[j..] \models F\neg\varphi$$

$$\text{iff } w \models \neg F\neg\varphi$$

\Rightarrow the temporal operators X and U suffice to express every LTL formula

Duality and idempotence laws

duality: $\neg G\varphi \equiv F\neg\varphi$

Duality and idempotence laws

duality:

$$\neg \mathbf{G} \varphi \equiv \mathbf{F} \neg \varphi$$

$$\neg \mathbf{F} \varphi \equiv \mathbf{G} \neg \varphi$$

Duality and idempotence laws

duality:

$$\neg G \varphi \equiv F \neg \varphi$$

$$\neg F \varphi \equiv G \neg \varphi$$

$$\neg X \varphi \equiv X \neg \varphi$$

Duality and idempotence laws

duality:

$$\neg G \varphi \equiv F \neg \varphi$$

$$\neg F \varphi \equiv G \neg \varphi$$

$$\neg X \varphi \equiv X \neg \varphi$$

idempotency:

$$G G \varphi \equiv G \varphi$$

$$F F \varphi \equiv F \varphi$$

Duality and idempotence laws

duality:

$$\neg G \varphi \equiv F \neg \varphi$$
$$\neg F \varphi \equiv G \neg \varphi$$
$$\neg X \varphi \equiv X \neg \varphi$$

idempotency:

$$G G \varphi \equiv G \varphi$$
$$F F \varphi \equiv F \varphi$$
$$\varphi U (\varphi U \psi) \equiv \varphi U \psi$$
$$(\varphi U \psi) U \psi \equiv \varphi U \psi$$

Absorption and distributive laws

absorption: $FGF\varphi \equiv GF\varphi$
 $GFG\varphi \equiv FG\varphi$

Absorption and distributive laws

absorption:

$$\begin{aligned}FGF\varphi &\equiv GF\varphi \\GFG\varphi &\equiv FG\varphi\end{aligned}$$

distribution:

$$\begin{aligned}X(\varphi \cup \psi) &\equiv X\varphi \cup X\psi \\F(\varphi \vee \psi) &\equiv F\varphi \vee F\psi \\G(\varphi \wedge \psi) &\equiv G\varphi \wedge G\psi\end{aligned}$$

Absorption and distributive laws

absorption:

$$\begin{aligned}FGF\varphi &\equiv GF\varphi \\GFG\varphi &\equiv FG\varphi\end{aligned}$$

distribution:

$$\begin{aligned}X(\varphi \cup \psi) &\equiv X\varphi \cup X\psi \\F(\varphi \vee \psi) &\equiv F\varphi \vee F\psi \\G(\varphi \wedge \psi) &\equiv G\varphi \wedge G\psi\end{aligned}$$

but:

$$\begin{aligned}F(\varphi \wedge \psi) &\not\equiv F\varphi \wedge F\psi \\G(\varphi \vee \psi) &\not\equiv G\varphi \vee G\psi\end{aligned}$$

Expansion laws

expansion:

$$F\varphi \equiv \varphi \vee XF\varphi$$
$$G\varphi \equiv \varphi \wedge XG\varphi$$

Expansion laws

expansion:
$$F\varphi \equiv \varphi \vee XF\varphi$$

$$G\varphi \equiv \varphi \wedge XG\varphi$$

$$\varphi U \psi \equiv \psi \vee (\varphi \wedge X(\varphi U \psi))$$

$$w \models F\varphi$$

$$\text{iff } \exists i \geq 0 : w[i..] \models \varphi$$

$$\text{iff } w[0..] \models \varphi \text{ or } \exists i > 0 : w[i..] \models \varphi$$

$$\text{iff } w \models \varphi \text{ or } \exists j \geq 0 : w[1+j..] \models \varphi$$

$$\text{iff } w \models \varphi \text{ or } \exists j \geq 0 : w[1..][j..] \models \varphi$$

$$\text{iff } w \models \varphi \text{ or } w[1..] \models F\varphi$$

$$\text{iff } w \models \varphi \text{ or } w \models XF\varphi$$

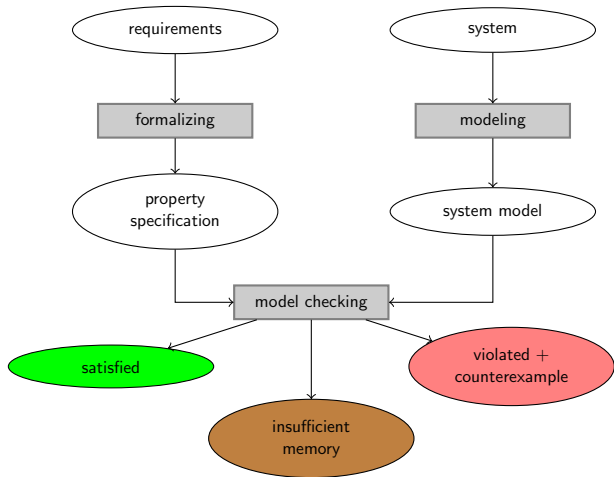
$$\text{iff } w \models \varphi \vee XF\varphi$$

Outline

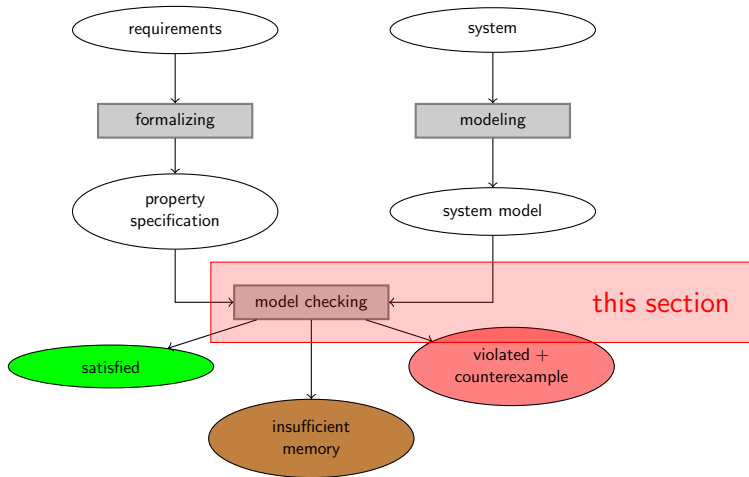
- LTL - Linear Time Logic
 - Syntax and Semantics
 - Equivalences

- LTL Model Checking
 - Overview
 - Transforming LTL into GNBA's
 - Complexity of LTL Model Checking

Model checking overview



Model checking overview



LTL Model Checking using GNBA's

transition system TS and LTL formula φ given

$$TS \models \varphi$$

$$\text{iff } \text{Traces}(TS) \subseteq \mathcal{L}(\varphi)$$

$$\text{iff } \text{Traces}(TS) \setminus \mathcal{L}(\varphi) = \text{Traces}(TS) \cap \mathcal{L}(\neg\varphi) = \emptyset$$

LTL Model Checking using GNBA's

transition system TS and LTL formula φ given

$$TS \models \varphi$$

$$\text{iff } \text{Traces}(TS) \subseteq \mathcal{L}(\varphi)$$

$$\text{iff } \text{Traces}(TS) \setminus \mathcal{L}(\varphi) = \text{Traces}(TS) \cap \mathcal{L}(\neg\varphi) = \emptyset$$

\Rightarrow LTL model checking can be done in four steps

LTL Model Checking using GNBA

transition system TS and LTL formula φ given

$$TS \models \varphi$$

$$\text{iff } \text{Traces}(TS) \subseteq \mathcal{L}(\varphi)$$

$$\text{iff } \text{Traces}(TS) \setminus \mathcal{L}(\varphi) = \text{Traces}(TS) \cap \mathcal{L}(\neg\varphi) = \emptyset$$

\Rightarrow LTL model checking can be done in four steps

1. calculate GNBA \mathcal{A}_{TS} with $\text{Traces}(TS) = \mathcal{L}(\mathcal{A}_{TS})$ (Chapter 2)

LTL Model Checking using GNBA

transition system TS and LTL formula φ given

$$TS \models \varphi$$

$$\text{iff } \text{Traces}(TS) \subseteq \mathcal{L}(\varphi)$$

$$\text{iff } \text{Traces}(TS) \setminus \mathcal{L}(\varphi) = \text{Traces}(TS) \cap \mathcal{L}(\neg\varphi) = \emptyset$$

\Rightarrow LTL model checking can be done in four steps

1. calculate GNBA \mathcal{A}_{TS} with $\text{Traces}(TS) = \mathcal{L}(\mathcal{A}_{TS})$ (Chapter 2)
2. calculate GNBA $\mathcal{A}_{\neg\varphi}$ with $\mathcal{L}(\neg\varphi) = \mathcal{L}(\mathcal{A}_{\neg\varphi})$ (this section)

LTL Model Checking using GNBA

transition system TS and LTL formula φ given

$$TS \models \varphi$$

$$\text{iff } \text{Traces}(TS) \subseteq \mathcal{L}(\varphi)$$

$$\text{iff } \text{Traces}(TS) \setminus \mathcal{L}(\varphi) = \text{Traces}(TS) \cap \mathcal{L}(\neg\varphi) = \emptyset$$

\Rightarrow LTL model checking can be done in four steps

1. calculate GNBA \mathcal{A}_{TS} with $\text{Traces}(TS) = \mathcal{L}(\mathcal{A}_{TS})$ (Chapter 2)
2. calculate GNBA $\mathcal{A}_{\neg\varphi}$ with $\mathcal{L}(\neg\varphi) = \mathcal{L}(\mathcal{A}_{\neg\varphi})$ (this section)
3. calculate GNBA \mathcal{A} for intersection of \mathcal{A}_{TS} and $\mathcal{A}_{\neg\varphi}$ (Chapter 2)

LTL Model Checking using GNBA

transition system TS and LTL formula φ given

$$TS \models \varphi$$

$$\text{iff } \text{Traces}(TS) \subseteq \mathcal{L}(\varphi)$$

$$\text{iff } \text{Traces}(TS) \setminus \mathcal{L}(\varphi) = \text{Traces}(TS) \cap \mathcal{L}(\neg\varphi) = \emptyset$$

\Rightarrow LTL model checking can be done in four steps

1. calculate GNBA \mathcal{A}_{TS} with $\text{Traces}(TS) = \mathcal{L}(\mathcal{A}_{TS})$ (Chapter 2)
2. calculate GNBA $\mathcal{A}_{\neg\varphi}$ with $\mathcal{L}(\neg\varphi) = \mathcal{L}(\mathcal{A}_{\neg\varphi})$ (this section)
3. calculate GNBA \mathcal{A} for intersection of \mathcal{A}_{TS} and $\mathcal{A}_{\neg\varphi}$ (Chapter 2)
4. perform non-emptiness test for $\mathcal{L}(\mathcal{A})$ (Chapter 2)

Fischer Ladner Closure

let φ be an LTL formula over $AP = \{a_1, \dots, a_n\}$.

Definition

the **Fischer Ladner closure** $cl(\varphi)$ is the list of sub-formulas of φ (starting from small formulas and ending with φ):

$$a_1, \dots, a_n, \dots, \varphi$$

Fischer Ladner Closure

let φ be an LTL formula over $AP = \{a_1, \dots, a_n\}$.

Definition

the **Fischer Ladner closure** $cl(\varphi)$ is the list of sub-formulas of φ (starting from small formulas and ending with φ):

$$a_1, \dots, a_n, \dots, \varphi$$

Example

$$cl(\neg b \wedge (\text{X } a \text{ U } b)) = a, b, \neg b, \text{X } a, \text{X } a \text{ U } b, \neg b \wedge (\text{X } a \text{ U } b)$$

φ -Expansion

idea:

- expand word by new row for each $\psi \in cl(\varphi)$ which is not an atomic proposition
- write truth-values of ψ in i -th column for subword $w[i..]$

φ -Expansion

idea:

- expand word by new row for each $\psi \in cl(\varphi)$ which is not an atomic proposition
- write truth-values of ψ in i -th column for subword $w[i..]$

Definition

for $w \in (2^n)^\omega$ and LTL-formula φ with $cl(\varphi) = \varphi_1, \dots, \varphi_m$ define the φ -**expansion** as word $v \in (2^m)^\omega$:

$$v[i]^j = 1 \text{ iff } w[i..] \models \varphi_j$$

($v[i]$ is the i -th letter of the infinite word v ,
and $v[i]^j$ is the j -th component of the vector $v[i]$)

Example

φ -expansion for $\varphi = \neg b \wedge (X a U b)$

explain that first two line define word and are given

a	0 1 1 0 1 0 0 1 1 1 ...
b	0 0 1 1 0 1 0 1 0 1 ...
$\neg b$	1 1 0 0 1 0 1 0 1 0 ...
$X a$	1 1 0 1 0 0 1 1 1 1 ...
$X a U b$	1 1 1 1 0 1 1 1 1 1 ...
$\neg b \wedge (X a U b)$	1 1 0 0 0 0 1 0 1 0 ...

Idea of LTL to GNBA Translation

- GNBA guesses the φ -expansion of w

Idea of LTL to GNBA Translation

- GNBA guesses the φ -expansion of w
- ... and **checks that guesses are correct** (mostly done locally!)

Idea of LTL to GNBA Translation

- GNBA guesses the φ -expansion of w
- ... and **checks that guesses are correct** (mostly done locally!)
- ... and demands that value for whole formula is 1 for whole word w

Idea of LTL to GNBA Translation

- GNBA guesses the φ -expansion of w
- ... and **checks that guesses are correct** (mostly done locally!)
- ... and demands that value for whole formula is 1 for whole word w

Definition (Consistency Checks)

let $cl(\varphi) = \varphi_1, \dots, \varphi_m$; a sequence of two successive vectors
 $(b_1, \dots, b_m)^T (c_1, \dots, c_m)^T$ is **consistent w.r.t. $cl(\varphi)$** iff whenever

Idea of LTL to GNBA Translation

- GNBA guesses the φ -expansion of w
- ... and **checks that guesses are correct** (mostly done locally!)
- ... and demands that value for whole formula is 1 for whole word w

Definition (Consistency Checks)

let $cl(\varphi) = \varphi_1, \dots, \varphi_m$; a sequence of two successive vectors
 $(b_1, \dots, b_m)^T (c_1, \dots, c_m)^T$ is **consistent w.r.t. $cl(\varphi)$** iff whenever

$$\varphi_j = \text{true} \quad \text{then } b_j$$

Idea of LTL to GNBA Translation

- GNBA guesses the φ -expansion of w
- ... and **checks that guesses are correct** (mostly done locally!)
- ... and demands that value for whole formula is 1 for whole word w

Definition (Consistency Checks)

let $cl(\varphi) = \varphi_1, \dots, \varphi_m$; a sequence of two successive vectors
 $(b_1, \dots, b_m)^T$ $(c_1, \dots, c_m)^T$ is **consistent w.r.t. $cl(\varphi)$** iff whenever

$$\begin{array}{ll} \varphi_j = \text{true} & \text{then } b_j \\ \varphi_j = \neg\varphi_{j_1} & \text{then } b_j \Leftrightarrow \neg b_{j_1} \end{array}$$

Idea of LTL to GNBA Translation

- GNBA guesses the φ -expansion of w
- ... and **checks that guesses are correct** (mostly done locally!)
- ... and demands that value for whole formula is 1 for whole word w

Definition (Consistency Checks)

let $cl(\varphi) = \varphi_1, \dots, \varphi_m$; a sequence of two successive vectors $(b_1, \dots, b_m)^T$ $(c_1, \dots, c_m)^T$ is **consistent w.r.t. $cl(\varphi)$** iff whenever

$$\begin{array}{ll} \varphi_j = \text{true} & \text{then } b_j \\ \varphi_j = \neg\varphi_{j_1} & \text{then } b_j \Leftrightarrow \neg b_{j_1} \\ \varphi_j = \varphi_{j_1} \wedge \varphi_{j_2} & \text{then } b_j \Leftrightarrow (b_{j_1} \wedge b_{j_2}) \end{array}$$

Idea of LTL to GNBA Translation

- GNBA guesses the φ -expansion of w
- ... and **checks that guesses are correct** (mostly done locally!)
- ... and demands that value for whole formula is 1 for whole word w

Definition (Consistency Checks)

let $cl(\varphi) = \varphi_1, \dots, \varphi_m$; a sequence of two successive vectors $(b_1, \dots, b_m)^T$ $(c_1, \dots, c_m)^T$ is **consistent w.r.t. $cl(\varphi)$** iff whenever

$\varphi_j = \text{true}$	then b_j
$\varphi_j = \neg\varphi_{j_1}$	then $b_j \Leftrightarrow \neg b_{j_1}$
$\varphi_j = \varphi_{j_1} \wedge \varphi_{j_2}$	then $b_j \Leftrightarrow (b_{j_1} \wedge b_{j_2})$
$\varphi_j = \text{X} \varphi_{j_1}$	then $b_j \Leftrightarrow c_{j_1}$

Idea of LTL to GNBA Translation

- GNBA guesses the φ -expansion of w
- ... and **checks that guesses are correct** (mostly done locally!)
- ... and demands that value for whole formula is 1 for whole word w

Definition (Consistency Checks)

let $cl(\varphi) = \varphi_1, \dots, \varphi_m$; a sequence of two successive vectors
 $(b_1, \dots, b_m)^T (c_1, \dots, c_m)^T$ is **consistent w.r.t. $cl(\varphi)$** iff whenever

$\varphi_j = \text{true}$	then b_j
$\varphi_j = \neg\varphi_{j_1}$	then $b_j \Leftrightarrow \neg b_{j_1}$
$\varphi_j = \varphi_{j_1} \wedge \varphi_{j_2}$	then $b_j \Leftrightarrow (b_{j_1} \wedge b_{j_2})$
$\varphi_j = \mathbf{X} \varphi_{j_1}$	then $b_j \Leftrightarrow c_{j_1}$
$\varphi_j = \varphi_{j_1} \mathbf{U} \varphi_{j_2}$	then $b_j \Leftrightarrow (b_{j_2} \vee (b_{j_1} \wedge c_j))$

(for last check recall expansion law: $\varphi_{j_1} \mathbf{U} \varphi_{j_2} \equiv \varphi_{j_2} \vee (\varphi_{j_1} \wedge \mathbf{X}(\varphi_{j_1} \mathbf{U} \varphi_{j_2}))$)

Consistency Checks and LTL-Models

Lemma

$w \models \varphi$ iff there exists an expansion $v \in (2^m)^\omega$ of w such that

Consistency Checks and LTL-Models

Lemma

$w \models \varphi$ iff there exists an expansion $v \in (2^m)^\omega$ of w such that

1. $v[i] \vee v[i + 1]$ is consistent for all i

Consistency Checks and LTL-Models

Lemma

$w \models \varphi$ iff there exists an expansion $v \in (2^m)^\omega$ of w such that

1. $v[i] \vee v[i + 1]$ is consistent for all i
2. $v[0]^m = 1$

Consistency Checks and LTL-Models

Lemma

$w \models \varphi$ iff there exists an expansion $v \in (2^m)^\omega$ of w such that

1. $v[i] \vee v[i + 1]$ is consistent for all i
2. $v[0]^m = 1$
3. whenever $\varphi_j = \varphi_{j_1} \cup \varphi_{j_2}$ and $v[i]^{j_1} = 1$ then there exists $i' \geq i$ such that $v[i']^{j_2} = 1$

Example (3. is necessary NOT IN HANDOUT)

Let $w = \begin{array}{c|cccccccc} a & 1 & 1 & 1 & 1 & 1 & 1 & \dots \\ b & 0 & 0 & 0 & 0 & 0 & 0 & \dots \end{array}$ and $\varphi = a \cup b$.

Choose $v = \begin{array}{c|cccccccc} a & 1 & 1 & 1 & 1 & 1 & 1 & \dots \\ b & 0 & 0 & 0 & 0 & 0 & 0 & \dots \\ a \cup b & 1 & 1 & 1 & 1 & 1 & 1 & \dots \end{array}$

v satisfies 1 and 2, but $v \not\models a \cup b$.

Translating LTL to GNBA

Definition (GNBA for an LTL formula φ)

let $cl(\varphi) = a_1, \dots, a_n, \varphi_{n+1}, \dots, \varphi_m$ where $\varphi_m = \varphi$
define $\mathcal{A}_\varphi = (2^m \uplus \{q_0\}, 2^n, q_0, \delta, F_1, \dots, F_k)$ where

Translating LTL to GNBA

Definition (GNBA for an LTL formula φ)

let $cl(\varphi) = a_1, \dots, a_n, \varphi_{n+1}, \dots, \varphi_m$ where $\varphi_m = \varphi$

define $\mathcal{A}_\varphi = (2^m \uplus \{q_0\}, 2^n, q_0, \delta, F_1, \dots, F_k)$ where

- $(c_1, \dots, c_m)^T \in \delta((b_1, \dots, b_m)^T, (d_1, \dots, d_n)^T)$ iff

Translating LTL to GNBA

Definition (GNBA for an LTL formula φ)

let $c/(\varphi) = a_1, \dots, a_n, \varphi_{n+1}, \dots, \varphi_m$ where $\varphi_m = \varphi$

define $\mathcal{A}_\varphi = (2^m \uplus \{q_0\}, 2^n, q_0, \delta, F_1, \dots, F_k)$ where

- $(c_1, \dots, c_m)^T \in \delta((b_1, \dots, b_m)^T, (d_1, \dots, d_n)^T)$ iff
 1. $c_j \Leftrightarrow d_j$ for all $j \leq n$

(expansion)

Translating LTL to GNBA

Definition (GNBA for an LTL formula φ)

let $c/(\varphi) = a_1, \dots, a_n, \varphi_{n+1}, \dots, \varphi_m$ where $\varphi_m = \varphi$

define $\mathcal{A}_\varphi = (2^m \uplus \{q_0\}, 2^n, q_0, \delta, F_1, \dots, F_k)$ where

- $(c_1, \dots, c_m)^T \in \delta((b_1, \dots, b_m)^T, (d_1, \dots, d_n)^T)$ iff
 1. $c_j \Leftrightarrow d_j$ for all $j \leq n$ (expansion)
 2. $(b_1, \dots, b_m)^T (c_1, \dots, c_m)^T$ is consistent (consistent expansion)

Translating LTL to GNBA

Definition (GNBA for an LTL formula φ)

let $c/(\varphi) = a_1, \dots, a_n, \varphi_{n+1}, \dots, \varphi_m$ where $\varphi_m = \varphi$

define $\mathcal{A}_\varphi = (2^m \uplus \{q_0\}, 2^n, q_0, \delta, F_1, \dots, F_k)$ where

- $(c_1, \dots, c_m)^T \in \delta((b_1, \dots, b_m)^T, (d_1, \dots, d_n)^T)$ iff
 1. $c_j \Leftrightarrow d_j$ for all $j \leq n$ (expansion)
 2. $(b_1, \dots, b_m)^T (c_1, \dots, c_m)^T$ is consistent (consistent expansion)
- $(c_1, \dots, c_m)^T \in \delta(q_0, (d_1, \dots, d_n)^T)$ iff

Translating LTL to GNBA

Definition (GNBA for an LTL formula φ)

let $c/(\varphi) = a_1, \dots, a_n, \varphi_{n+1}, \dots, \varphi_m$ where $\varphi_m = \varphi$

define $\mathcal{A}_\varphi = (2^m \uplus \{q_0\}, 2^n, q_0, \delta, F_1, \dots, F_k)$ where

- $(c_1, \dots, c_m)^T \in \delta((b_1, \dots, b_m)^T, (d_1, \dots, d_n)^T)$ iff
 1. $c_j \Leftrightarrow d_j$ for all $j \leq n$ (expansion)
 2. $(b_1, \dots, b_m)^T (c_1, \dots, c_m)^T$ is consistent (consistent expansion)
- $(c_1, \dots, c_m)^T \in \delta(q_0, (d_1, \dots, d_n)^T)$ iff
 1. $c_j \Leftrightarrow d_j$ for all $j \leq n$ (expansion)

Translating LTL to GNBA

Definition (GNBA for an LTL formula φ)

let $c/(\varphi) = a_1, \dots, a_n, \varphi_{n+1}, \dots, \varphi_m$ where $\varphi_m = \varphi$

define $\mathcal{A}_\varphi = (2^m \uplus \{q_0\}, 2^n, q_0, \delta, F_1, \dots, F_k)$ where

- $(c_1, \dots, c_m)^T \in \delta((b_1, \dots, b_m)^T, (d_1, \dots, d_n)^T)$ iff
 1. $c_j \Leftrightarrow d_j$ for all $j \leq n$ (expansion)
 2. $(b_1, \dots, b_m)^T (c_1, \dots, c_m)^T$ is consistent (consistent expansion)
- $(c_1, \dots, c_m)^T \in \delta(q_0, (d_1, \dots, d_n)^T)$ iff
 1. $c_j \Leftrightarrow d_j$ for all $j \leq n$ (expansion)
 2. c_m (φ is satisfied)

Translating LTL to GNBA

Definition (GNBA for an LTL formula φ)

let $cl(\varphi) = a_1, \dots, a_n, \varphi_{n+1}, \dots, \varphi_m$ where $\varphi_m = \varphi$

define $\mathcal{A}_\varphi = (2^m \uplus \{q_0\}, 2^n, q_0, \delta, F_1, \dots, F_k)$ where

- $(c_1, \dots, c_m)^T \in \delta((b_1, \dots, b_m)^T, (d_1, \dots, d_n)^T)$ iff
 1. $c_j \Leftrightarrow d_j$ for all $j \leq n$ (expansion)
 2. $(b_1, \dots, b_m)^T (c_1, \dots, c_m)^T$ is consistent (consistent expansion)
- $(c_1, \dots, c_m)^T \in \delta(q_0, (d_1, \dots, d_n)^T)$ iff
 1. $c_j \Leftrightarrow d_j$ for all $j \leq n$ (expansion)
 2. c_m (φ is satisfied)
- if $\varphi_j = \varphi_{j_1} \cup \varphi_{j_2}$ is i -th \cup -subformula in $cl(\varphi)$ then

$$F_i = \{(b_1, \dots, b_m)^T \mid \neg b_{j_1} \vee b_{j_2}\}$$

Example

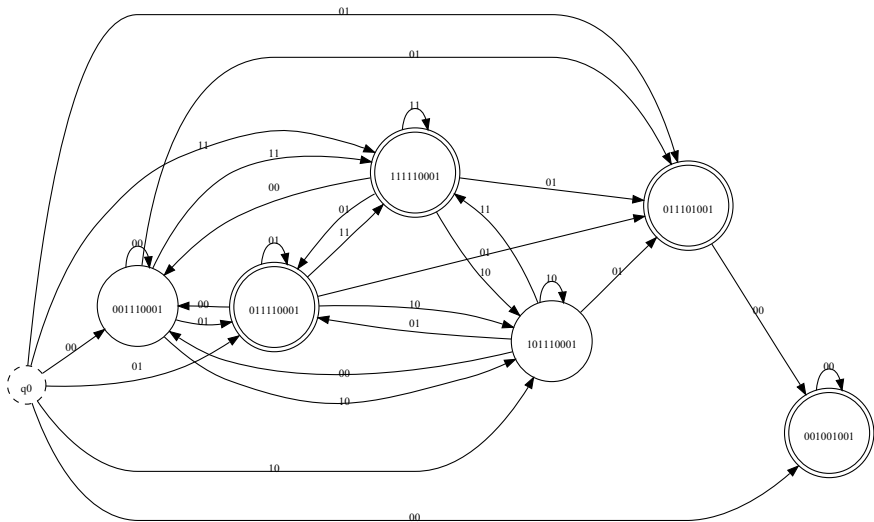
$$\varphi = G(\text{req} \Rightarrow X F \text{resp})$$

$$\equiv \neg(\text{true} \cup (\text{req} \wedge \neg X(\text{true} \cup \text{resp}))) = \psi$$

$cl(\psi)$	expansion	consistency
1 : <i>req</i>	$d_1 \Leftrightarrow c_1$	b_3
2 : <i>resp</i>	$d_2 \Leftrightarrow c_2$	$b_4 \Leftrightarrow (b_2 \vee (b_3 \wedge c_4))$
3 : <i>true</i>		$b_5 \Leftrightarrow c_4$
4 : <i>true</i> U <i>resp</i>		$b_6 \Leftrightarrow \neg b_5$
5 : X (<i>true</i> U <i>resp</i>)		$b_7 \Leftrightarrow (b_1 \wedge b_6)$
6 : $\neg X(\text{true} \cup \text{resp})$		$b_8 \Leftrightarrow (b_7 \vee (b_3 \wedge c_8))$
7 : <i>req</i> \wedge $\neg X(\text{true} \cup \text{resp})$		$b_9 \Leftrightarrow \neg b_8$
8 : <i>true</i> U (<i>req</i> \wedge $\neg X(\text{true} \cup \text{resp})$)		(513st/2304tr)
9 : $\neg(\text{true} \cup (\text{req} \wedge \neg X(\text{true} \cup \text{resp})))$		(7/28compressed)

- $(c_1, \dots, c_9)^T \in \delta((b_1, \dots, b_9)^T, (d_1, d_2)^T)$ iff expansion and consistent
- $(c_1, \dots, c_9)^T \in \delta(q_0, (d_1, d_2)^T)$ iff expansion and c_9
- $F_1 = \{(b_1, \dots, b_9) \mid \neg b_4 \vee b_2\}$ $F_2 = \{(b_1, \dots, b_9) \mid \neg b_8 \vee b_7\}$

Example (parts of GNBA)



Soundness of Translation

Theorem

for every LTL formula φ

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

Soundness of Translation

Theorem

for every LTL formula φ

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

Proof of Lemma.

by induction on φ using the consistency checks



Soundness of Translation

Theorem

for every LTL formula φ

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

Proof of Lemma.

by induction on φ using the consistency checks ■

Proof of Theorem.

- construction of \mathcal{A}_φ directly corresponds to requirements 1 and 2 in lemma
- remaining difficulty:
show that visiting F_i infinitely often is the same as requirement 3 in lemma for i -th U-subformula $\varphi_j = \varphi_{j_1} \cup \varphi_{j_2}$ ■

Optimizing the Translation

- observation: many states do not have outgoing transitions

Optimizing the Translation

- observation: many states do not have outgoing transitions
- reason: several inconsistencies due to Boolean conditions
example: if $\varphi_j = \varphi_{j_1} \wedge \varphi_{j_2}$ then b_j cannot be freely chosen;
value of b_j is determined by b_{j_1} and b_{j_2}

Optimizing the Translation

- observation: many states do not have outgoing transitions
- reason: several inconsistencies due to Boolean conditions
example: if $\varphi_j = \varphi_{j_1} \wedge \varphi_{j_2}$ then b_j cannot be freely chosen;
value of b_j is determined by b_{j_1} and b_{j_2}
- idea: take **reduced Fischer-Ladner closure** $cl'(\varphi)$ which only contains
 - atomic propositions
 - X-formulas
 - U-formulas
 - ... and no other formula

Optimizing the Translation

- observation: many states do not have outgoing transitions
- reason: several inconsistencies due to Boolean conditions
example: if $\varphi_j = \varphi_{j_1} \wedge \varphi_{j_2}$ then b_j cannot be freely chosen;
value of b_j is determined by b_{j_1} and b_{j_2}
- idea: take **reduced Fischer-Ladner closure** $cl'(\varphi)$ which only contains
 - atomic propositions
 - X-formulas
 - U-formulas
 - ... and no other formula

and then incorporate the Boolean connectives directly into consistency, final states, ...

Towards an Improved Translation

let $cl'(\varphi) = \varphi_1, \dots, \varphi_m$ over $AP = \{a_1, \dots, a_n\}$

Definition (Unwinding)

the **unwinding** of a subformula ψ of φ w.r.t. a vector $B = (b_1, \dots, b_m)$ is defined as $\mathcal{U}_B(\psi)$ where

Towards an Improved Translation

let $cl'(\varphi) = \varphi_1, \dots, \varphi_m$ over $AP = \{a_1, \dots, a_n\}$

Definition (Unwinding)

the **unwinding** of a subformula ψ of φ w.r.t. a vector $B = (b_1, \dots, b_m)$ is defined as $\mathcal{U}_B(\psi)$ where

$$\mathcal{U}_B(a_i) = b_i \text{ for all atomic propositions } a_i$$

Towards an Improved Translation

let $cl'(\varphi) = \varphi_1, \dots, \varphi_m$ over $AP = \{a_1, \dots, a_n\}$

Definition (Unwinding)

the **unwinding** of a subformula ψ of φ w.r.t. a vector $B = (b_1, \dots, b_m)$ is defined as $\mathcal{U}_B(\psi)$ where

$$\mathcal{U}_B(a_i) = b_i \text{ for all atomic propositions } a_i$$

$$\mathcal{U}_B(\text{true}) = \text{true}$$

Towards an Improved Translation

let $cl'(\varphi) = \varphi_1, \dots, \varphi_m$ over $AP = \{a_1, \dots, a_n\}$

Definition (Unwinding)

the **unwinding** of a subformula ψ of φ w.r.t. a vector $B = (b_1, \dots, b_m)$ is defined as $\mathcal{U}_B(\psi)$ where

$$\mathcal{U}_B(a_i) = b_i \text{ for all atomic propositions } a_i$$

$$\mathcal{U}_B(\text{true}) = \text{true}$$

$$\mathcal{U}_B(\neg\psi) = \neg\mathcal{U}_B(\psi)$$

Towards an Improved Translation

let $cl'(\varphi) = \varphi_1, \dots, \varphi_m$ over $AP = \{a_1, \dots, a_n\}$

Definition (Unwinding)

the **unwinding** of a subformula ψ of φ w.r.t. a vector $B = (b_1, \dots, b_m)$ is defined as $\mathcal{U}_B(\psi)$ where

$$\mathcal{U}_B(a_i) = b_i \text{ for all atomic propositions } a_i$$

$$\mathcal{U}_B(\text{true}) = \text{true}$$

$$\mathcal{U}_B(\neg\psi) = \neg\mathcal{U}_B(\psi)$$

$$\mathcal{U}_B(\psi_1 \wedge \psi_2) = \mathcal{U}_B(\psi_1) \wedge \mathcal{U}_B(\psi_2)$$

Towards an Improved Translation

let $cl'(\varphi) = \varphi_1, \dots, \varphi_m$ over $AP = \{a_1, \dots, a_n\}$

Definition (Unwinding)

the **unwinding** of a subformula ψ of φ w.r.t. a vector $B = (b_1, \dots, b_m)$ is defined as $\mathcal{U}_B(\psi)$ where

$$\mathcal{U}_B(a_i) = b_i \text{ for all atomic propositions } a_i$$

$$\mathcal{U}_B(\text{true}) = \text{true}$$

$$\mathcal{U}_B(\neg\psi) = \neg\mathcal{U}_B(\psi)$$

$$\mathcal{U}_B(\psi_1 \wedge \psi_2) = \mathcal{U}_B(\psi_1) \wedge \mathcal{U}_B(\psi_2)$$

$$\mathcal{U}_B(X\psi) = b_j \text{ where } n < j \leq m \text{ is the index s.t. } \varphi_j = X\psi$$

Towards an Improved Translation

let $cl'(\varphi) = \varphi_1, \dots, \varphi_m$ over $AP = \{a_1, \dots, a_n\}$

Definition (Unwinding)

the **unwinding** of a subformula ψ of φ w.r.t. a vector $B = (b_1, \dots, b_m)$ is defined as $\mathcal{U}_B(\psi)$ where

$$\mathcal{U}_B(a_i) = b_i \text{ for all atomic propositions } a_i$$

$$\mathcal{U}_B(\text{true}) = \text{true}$$

$$\mathcal{U}_B(\neg\psi) = \neg\mathcal{U}_B(\psi)$$

$$\mathcal{U}_B(\psi_1 \wedge \psi_2) = \mathcal{U}_B(\psi_1) \wedge \mathcal{U}_B(\psi_2)$$

$$\mathcal{U}_B(\mathbf{X}\psi) = b_j \text{ where } n < j \leq m \text{ is the index s.t. } \varphi_j = \mathbf{X}\psi$$

$$\mathcal{U}_B(\psi \mathbf{U} \chi) = b_j \text{ where } n < j \leq m \text{ is the index s.t. } \varphi_j = \psi \mathbf{U} \chi$$

Towards an Improved Translation (2)

Definition (Compressed Consistency Checks)

let $cl'(\varphi) = \varphi_1, \dots, \varphi_m$; a sequence of two successive vectors
 $B = (b_1, \dots, b_m)^T$ and $C = (c_1, \dots, c_m)^T$ is **consistent w.r.t. $cl'(\varphi)$** iff
whenever

Towards an Improved Translation (2)

Definition (Compressed Consistency Checks)

let $c'(\varphi) = \varphi_1, \dots, \varphi_m$; a sequence of two successive vectors
 $B = (b_1, \dots, b_m)^T$ and $C = (c_1, \dots, c_m)^T$ is **consistent w.r.t. $c'(\varphi)$** iff
whenever

$$\varphi_j = X\psi \quad \text{then } b_j \Leftrightarrow \mathcal{U}_C(\psi)$$

Towards an Improved Translation (2)

Definition (Compressed Consistency Checks)

let $cl'(\varphi) = \varphi_1, \dots, \varphi_m$; a sequence of two successive vectors
 $B = (b_1, \dots, b_m)^T$ and $C = (c_1, \dots, c_m)^T$ is **consistent w.r.t. $cl'(\varphi)$** iff
 whenever

$$\varphi_j = X\psi \quad \text{then } b_j \Leftrightarrow \mathcal{U}_C(\psi)$$

$$\varphi_j = \psi \cup \chi \quad \text{then } b_j \Leftrightarrow (\mathcal{U}_B(\chi) \vee (\mathcal{U}_B(\psi) \wedge c_j))$$

Improved Translation

Definition

let $cl'(\varphi) = a_1, \dots, a_n, \varphi_{n+1}, \dots, \varphi_m$ define

$\mathcal{A}_\varphi = (2^m \uplus \{q_0\}, 2^n, q_0, \delta, F_1, \dots, F_k)$ where

- $(c_1, \dots, c_m)^T \in \delta((b_1, \dots, b_m)^T, (d_1, \dots, d_n)^T)$ iff
 1. $c_j \Leftrightarrow d_j$ for all $j \leq n$ (expansion)
 2. $(b_1, \dots, b_m)^T (c_1, \dots, c_m)^T$ is **consistent** (consistent expansion)
- $C = (c_1, \dots, c_m)^T \in \delta(q_0, (d_1, \dots, d_n)^T)$ iff
 1. $c_j \Leftrightarrow d_j$ for all $j \leq n$ (expansion)
 2. $\mathcal{U}_C(\varphi)$ (φ is satisfied)
- if $\varphi_j = \psi \text{ U } \chi$ is i -th U-subformula in $cl'(\varphi)$ then

$$F_i = \{B = (b_1, \dots, b_m)^T \mid \neg b_j \vee \mathcal{U}_B(\chi)\}$$

Example NOT IN HANDOUT

$$\varphi = G(\text{req} \Rightarrow XF \text{resp})$$

$$\equiv \neg(\text{true} U (\text{req} \wedge \neg X(\text{true} U \text{resp}))) = \psi$$

 $cl'(\psi)$

expansion

consistency

1 : req

$d_1 \Leftrightarrow c_1$

$b_3 \Leftrightarrow (b_2 \vee (\text{true} \wedge c_3))$

2 : resp

$d_2 \Leftrightarrow c_2$

$b_4 \Leftrightarrow c_3$

3 : true U resp

$b_5 \Leftrightarrow ((b_1 \wedge (\neg b_4)) \vee (\text{true} \wedge c_5))$

4 : X (true U resp)

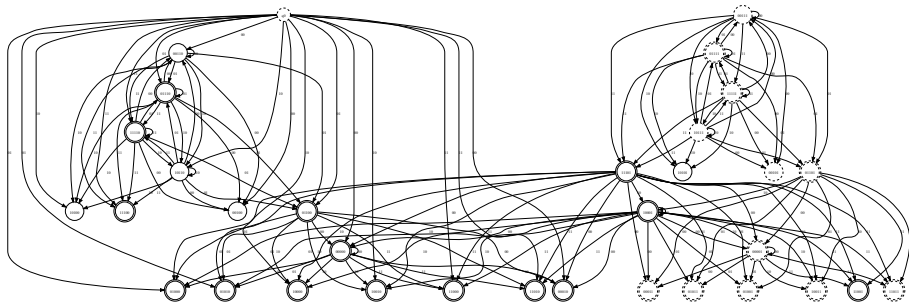
(33st/144tr)

5 : true U (req \wedge \neg X (true U resp))

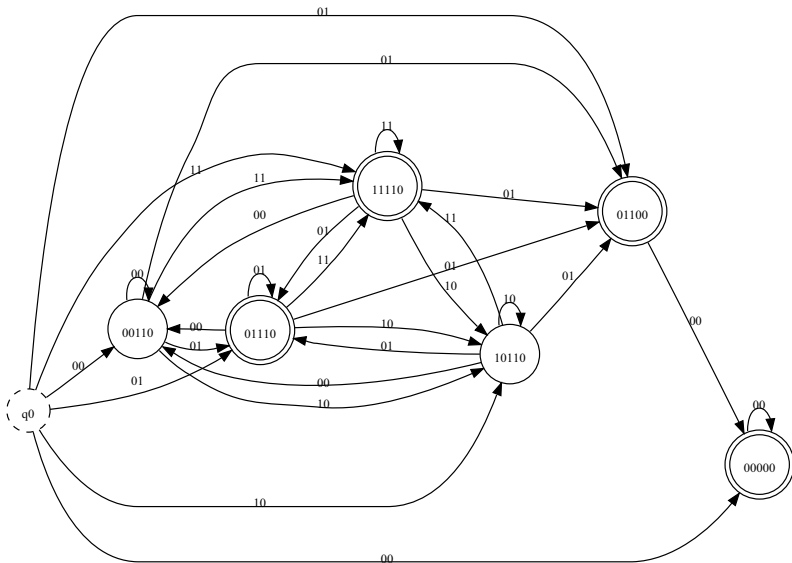
(7/28compressed)

- $(c_1, \dots, c_5)^T \in \delta((b_1, \dots, b_5)^T, (d_1, d_2)^T)$ iff expansion and consistent
- $(c_1, \dots, c_5)^T \in \delta(q_0, (d_1, d_2)^T)$ iff expansion and $\neg c_5$
- $F_1 = \{(b_1, \dots, b_5) \mid \neg b_3 \vee b_2\}$ $F_2 = \{(b_1, \dots, b_5) \mid \neg b_5 \vee (b_1 \wedge \neg b_4)\}$

Example GNBA



Example GNBA



Soundness of the Improved Transformation

Theorem

both transformations yield essentially the same automata; the only difference is that

Soundness of the Improved Transformation

Theorem

both transformations yield essentially the same automata; the only difference is that

- whenever $\varphi_i = \varphi_{i_1} \wedge \varphi_{i_2}$ then the i -th component is missing in the improved transformation; the state without the i -th component of the improved transformation is corresponding to the state $(\dots, b_{i_1}, \dots, b_{i_2}, \dots, b_i, \dots)$ where $b_i = b_{i_1} \wedge b_{i_2}$;*

Soundness of the Improved Transformation

Theorem

both transformations yield essentially the same automata; the only difference is that

- whenever $\varphi_i = \varphi_{i_1} \wedge \varphi_{i_2}$ then the i -th component is missing in the improved transformation; the state without the i -th component of the improved transformation is corresponding to the state $(\dots, b_{i_1}, \dots, b_{i_2}, \dots, b_i, \dots)$ where $b_i = b_{i_1} \wedge b_{i_2}$; moreover, all other states in the non-improved translation where $b_i \neq b_{i_1} \wedge b_{i_2}$ have no outgoing states*

Soundness of the Improved Transformation

Theorem

both transformations yield essentially the same automata; the only difference is that

- *whenever $\varphi_i = \varphi_{i_1} \wedge \varphi_{i_2}$ then the i -th component is missing in the improved transformation; the state without the i -th component of the improved transformation is corresponding to the state $(\dots, b_{i_1}, \dots, b_{i_2}, \dots, b_i, \dots)$ where $b_i = b_{i_1} \wedge b_{i_2}$; moreover, all other states in the non-improved translation where $b_i \neq b_{i_1} \wedge b_{i_2}$ have no outgoing states*
- *a similar property is true for negations*

Soundness of the Improved Transformation

Theorem

both transformations yield essentially the same automata; the only difference is that

- *whenever $\varphi_i = \varphi_{i_1} \wedge \varphi_{i_2}$ then the i -th component is missing in the improved transformation; the state without the i -th component of the improved transformation is corresponding to the state $(\dots, b_{i_1}, \dots, b_{i_2}, \dots, b_i, \dots)$ where $b_i = b_{i_1} \wedge b_{i_2}$; moreover, all other states in the non-improved translation where $b_i \neq b_{i_1} \wedge b_{i_2}$ have no outgoing states*

- *a similar property is true for negations*

\Rightarrow *soundness of the non-improved translation implies soundness of the improved one*

Complexity of LTL Model Checking

LTL model checking: given TS and φ check

$$\mathcal{L}(\mathcal{A}_3) = \emptyset$$

where

$$\mathcal{A}_1 = \mathcal{A}_{TS}$$

$$\mathcal{A}_2 = \mathcal{A}_{\neg\varphi}$$

$$\mathcal{A}_3 = \mathcal{A}_{\mathcal{A}_1 \cap \mathcal{A}_2}$$

number of states:

linear in TS , exp. in φ

Complexity of LTL Model Checking

LTL model checking: given TS and φ check

$$\mathcal{L}(\mathcal{A}_3) = \emptyset$$

where

$$\mathcal{A}_1 = \mathcal{A}_{TS}$$

$$\mathcal{A}_2 = \mathcal{A}_{\neg\varphi}$$

$$\mathcal{A}_3 = \mathcal{A}_{\mathcal{A}_1 \cap \mathcal{A}_2}$$

number of states:

$$\mathcal{A}_1 : |TS| + 1$$

linear in TS , exp. in φ

Complexity of LTL Model Checking

LTL model checking: given TS and φ check

$$\mathcal{L}(\mathcal{A}_3) = \emptyset$$

where

$$\mathcal{A}_1 = \mathcal{A}_{TS}$$

$$\mathcal{A}_2 = \mathcal{A}_{\neg\varphi}$$

$$\mathcal{A}_3 = \mathcal{A}_{\mathcal{A}_1 \cap \mathcal{A}_2}$$

number of states:

$$\mathcal{A}_1 : |TS| + 1$$

$$\mathcal{A}_2 : 2^{|\mathcal{AP}| + |\varphi|} + 1 \quad \text{where } |\varphi| \text{ is number of temporal operators in } \varphi$$

linear in TS , exp. in φ

Complexity of LTL Model Checking

LTL model checking: given TS and φ check

$$\mathcal{L}(\mathcal{A}_3) = \emptyset$$

where

$$\mathcal{A}_1 = \mathcal{A}_{TS}$$

$$\mathcal{A}_2 = \mathcal{A}_{\neg\varphi}$$

$$\mathcal{A}_3 = \mathcal{A}_{\mathcal{A}_1 \cap \mathcal{A}_2}$$

number of states:

$$\mathcal{A}_1 : |TS| + 1$$

$$\mathcal{A}_2 : 2^{|AP|+|\varphi|} + 1 \quad \text{where } |\varphi| \text{ is number of temporal operators in } \varphi$$

$$\mathcal{A}_3 : (|TS| + 1) \cdot (2^{|AP|+|\varphi|} + 1)$$

linear in TS , exp. in φ

Complexity of LTL Model Checking

LTL model checking: given TS and φ check

$$\mathcal{L}(\mathcal{A}_3) = \emptyset$$

where

$$\mathcal{A}_1 = \mathcal{A}_{TS}$$

$$\mathcal{A}_2 = \mathcal{A}_{\neg\varphi}$$

$$\mathcal{A}_3 = \mathcal{A}_{\mathcal{A}_1 \cap \mathcal{A}_2}$$

number of states:

$$\mathcal{A}_1 : |TS| + 1$$

$$\mathcal{A}_2 : 2^{|\mathcal{AP}| + |\varphi|} + 1 \quad \text{where } |\varphi| \text{ is number of temporal operators in } \varphi$$

$$\mathcal{A}_3 : (|TS| + 1) \cdot (2^{|\mathcal{AP}| + |\varphi|} + 1)$$

\Rightarrow total complexity of $\mathcal{O}(|TS| \cdot 2^{|\mathcal{AP}| + |\varphi|})$ linear in TS , exp. in φ

Lower bound

Theorem

there exists a family of LTL formulas φ_n over $AP = \{a\}$ with $|\varphi_n| = \mathcal{O}(\text{poly}(n))$ such that every NBA \mathcal{A}_n with $\mathcal{L}(\mathcal{A}_n) = \mathcal{L}(\varphi_n)$ has at least 2^n states

Proof (1)

let $AP = \{a\}$ and:

$$\mathcal{L}_n = \{ b_1 \dots b_n b_1 \dots b_n w \mid b_i \in \{0, 1\}, w \in \{0, 1\}^\omega \}, \quad \text{for } n \geq 0$$

Proof (1)

let $AP = \{a\}$ and:

$$\mathcal{L}_n = \{ b_1 \dots b_n b_1 \dots b_n w \mid b_i \in \{0, 1\}, w \in \{0, 1\}^\omega \}, \quad \text{for } n \geq 0$$

it follows $\mathcal{L}_n = \mathcal{L}(\varphi_n)$ where $\varphi_n = \bigwedge_{0 \leq i < n} (X^i a \Leftrightarrow X^{n+i} a)$

Proof (1)

let $AP = \{a\}$ and:

$$\mathcal{L}_n = \{ b_1 \dots b_n b_1 \dots b_n w \mid b_i \in \{0, 1\}, w \in \{0, 1\}^\omega \}, \quad \text{for } n \geq 0$$

it follows $\mathcal{L}_n = \mathcal{L}(\varphi_n)$ where $\varphi_n = \bigwedge_{0 \leq i < n} (X^i a \Leftrightarrow X^{n+i} a)$

φ_n is an LTL formula of polynomial length: $|\varphi_n| \in \mathcal{O}(n^2)$

Proof (1)

let $AP = \{a\}$ and:

$$\mathcal{L}_n = \{ b_1 \dots b_n b_1 \dots b_n w \mid b_i \in \{0, 1\}, w \in \{0, 1\}^\omega \}, \quad \text{for } n \geq 0$$

it follows $\mathcal{L}_n = \mathcal{L}(\varphi_n)$ where $\varphi_n = \bigwedge_{0 \leq i < n} (X^i a \Leftrightarrow X^{n+i} a)$

φ_n is an LTL formula of polynomial length: $|\varphi_n| \in \mathcal{O}(n^2)$

however, any NBA \mathcal{A} with $\mathcal{L}_\omega(\mathcal{A}) = \mathcal{L}_n$ has at least 2^n states

Proof (2)

claim: any NBA \mathcal{A} for $\bigwedge_{0 \leq i < n} (X^i a \Leftrightarrow X^{n+i} a)$ has at least 2^n states

words of the form $b_1 \dots b_n b_1 \dots b_n 000 \dots$ are accepted by \mathcal{A} .

thus, for every word $w = b_1 \dots b_n$ of length n there is a state $q(w)$ in \mathcal{A} which can be reached from the initial state by consuming $b_1 \dots b_n$.

Moreover, from $q(w)$, it is possible to visit an accept state infinitely often by reading the word $w 000 \dots$.

if for every two different words w and w' of length n , the states $q(w)$ and $q(w')$ are different, then there are at least 2^n states. Otherwise, there are two words w and w' where $w \neq w'$, but $q(w) = q(w')$. But then \mathcal{A} cannot accept the language since $w w' 000 \dots$ does not satisfy the formula, but has an accepting run (go from the initial state to $q(w) = q(w')$ by reading w , and then continue from $q(w')$ with the accepting run for $w' 000 \dots$).

Is Bad Complexity a Result of Automata Approach?

Is Bad Complexity a Result of Automata Approach? **No!**

Theorem (Sistla, Clarke)

the LTL model checking problem $TS \models \varphi$ is PSPACE-hard

Is Bad Complexity a Result of Automata Approach?

Theorem (Sistla, Clarke)

the LTL model checking problem $TS \models \varphi$ is PSPACE-hard

Theorem

the LTL model checking problem $TS \models \varphi$ is PSPACE-complete

Summary

- LTL is a logic for specifying the allowed traces of a system; it extends propositional logic by temporal operators like X and U

Summary

- LTL is a logic for specifying the allowed traces of a system; it extends propositional logic by temporal operators like X and U
- LTL model checking can be done by constructing a GNBA and then checking whether this GNBA accepts at least one word, the counterexample

Summary

- LTL is a logic for specifying the allowed traces of a system; it extends propositional logic by temporal operators like X and U
- LTL model checking can be done by constructing a GNBA and then checking whether this GNBA accepts at least one word, the counterexample
- LTL model checking is PSPACE-complete