Modeling and verifying reactive systems

Temporal logics

Nicolas Markey

Lab. Specification et Verification
ENS Cachan & CNRS, France

Outline of the course

1. Linear-time temporal logics
   - Algorithms for verifying LTL formulas (cont'd)
   - Back to expressiveness

2. Branching-time temporal logics
   - Expressiveness of branchig-time logics
   - Complexity

NP-complete fragments of LTL+Past

Definition

A word w is *ultimately periodic* if it can be written $u \cdot v^\omega$, where u and v are finite words and v is not the empty word.

NP-complete fragments of LTL+Past

Definition

A word w is *ultimately periodic* if it can be written $u \cdot v^\omega$, where u and v are finite words and v is not the empty word.

Theorem

A formula $\varphi \in$ LTL+Past is satisfiable iff it is satisfiable by an ultimately periodic word $u \cdot v^\omega$ where u and v have size exponential in $|\varphi|$.

NP-complete fragments of LTL+Past

Definition

A word w is *ultimately periodic* if it can be written $u \cdot v^\omega$, where u and v are finite words and v is not the empty word.

Theorem

A formula $\varphi \in$ LTL+Past is satisfiable iff it is satisfiable by an ultimately periodic word $u \cdot v^\omega$ where u and v have size exponential in $|\varphi|$.

Proof.

The witnessing execution of the Büchi automaton associated to φ is ultimately periodic, and has size exponential. □

An NP-complete fragment of LTL+Past

Definition

We write LTL_1 for the fragment of LTL where modalities cannot be nested.

Example

$$(p \ \mathbf{U}\ q) \wedge \mathbf{G}\, r \quad \text{is a formula of } \mathrm{LTL}_1$$

$$(p \wedge r)\ \mathbf{U}\, (q \wedge \mathbf{G}\, r) \quad \text{is not a formula of } \mathrm{LTL}_1$$

An NP-complete fragment of LTL+Past

Definition

We write LTL_1 for the fragment of LTL where modalities cannot be nested.

Theorem

Deciding the satisfiability of a formula of LTL_1 is NP*-complete.*

An NP-complete fragment of LTL+Past

Definition

We write LTL_1 for the fragment of LTL where modalities cannot be nested.

Theorem

Deciding the satisfiability of a formula of LTL_1 is NP*-complete.*

Proof.

- prove the existence of a small witness, i.e., a polynomial-size ultimately-periodic word that satisfies the formula;
- the NP algorithm consists in guessing that polynomial-size witness, and check (in polynomial time) that it satisfies φ.
- Hardness in NP follows from that of the satisfiability of a propositional logic formula.

□

An NP-complete fragment of LTL+Past

Definition

We write LTL_1 for the fragment of LTL where modalities cannot be nested.

Theorem

Model-checking for LTL_1 is NP*-complete.*

Proof.

- prove the existence of a small witness (that should now be a path of the Kripke structure);
- non-deterministically guess, then check, a polynomial-size witnessing run in the Kripke structure.

- Hardness in NP: easy encoding of 3SAT. □

Outline of the course

1. Linear-time temporal logics
   - Algorithms for verifying LTL formulas (cont'd)
   - Back to expressiveness

2. Branching-time temporal logics
   - Expressiveness of branchig-time logics
   - Complexity

Succinctness of LTL+Past

Theorem

LTL+Past can be exponentially more succinct than LTL.

Succinctness of LTL+Past

Theorem

LTL+Past can be exponentially more succinct than LTL.

Proof.

Consider the following property, built on $\mathrm{AP} = \{p_0, \ldots, p_n\}$:

($\mathcal{P}$): any two states that agree on propositions p_1 to p_n also agree on proposition p_0.

Succinctness of LTL+Past

Proof.

$(\mathcal{P})$: any two states that agree on propositions p_1 to p_n also agree on proposition p_0.

It can be expressed in LTL by enumerating the possible valuations for p_0 to p_n:

$$\bigwedge_{(b_0,\ldots,b_n)\in\{\top,\bot\}^{n+1}} \left(\mathbf{F}\Big(\bigwedge_{i\geq 0} p_i = b_i\Big) \Rightarrow \mathbf{G}\Big(\Big(\bigwedge_{i\geq 1} p_i = b_i\Big) \Rightarrow p_0 = b_0\Big)\right)$$

The size of this formula is exponential in n.

Succinctness of LTL+Past

Proof.

($\mathcal{P}$): any two states that agree on propositions p_1 to p_n also agree on proposition p_0.

Let $\mathcal{A}$ be a Büchi automaton corresponding to property ($\mathcal{P}$).

Let $\Sigma = \{a_0, a_1, ..., a_{2^n-1}\}$ be the subsets of $\{p_1, ..., p_n\}$.

Succinctness of LTL+Past

Proof.

($\mathcal{P}$): any two states that agree on propositions p_1 to p_n also agree on proposition p_0.

For each $K \subseteq \{0, ..., 2^n - 1\}$, we define $w_K = b_0 ... b_{2^n-1}$ with

$$b_i = \begin{cases} a_i & \text{if } i \in K \\ a_i \cup \{p_0\} & \text{otherwise} \end{cases}$$

Succinctness of LTL+Past

Proof.

($\mathcal{P}$): any two states that agree on propositions p_1 to p_n also agree on proposition p_0.

For each $K \subseteq \{0, ..., 2^n - 1\}$, we define $w_K = b_0 ... b_{2^n-1}$ with

$$b_i = \begin{cases} a_i & \text{if } i \in K \\ a_i \cup \{p_0\} & \text{otherwise} \end{cases}$$

Lemma

There are 2^{2^n} different such words.

Succinctness of LTL+Past

Proof.

($\mathcal{P}$): any two states that agree on propositions p_1 to p_n also agree on proposition p_0.

For each $K \subseteq \{0, ..., 2^n - 1\}$, we define $w_K = b_0 ... b_{2^n-1}$ with

$$b_i = \begin{cases} a_i & \text{if } i \in K \\ a_i \cup \{p_0\} & \text{otherwise} \end{cases}$$

Lemma

For any $K \subseteq \{0, ..., 2^n - 1\}$, the word w_K^ω is accepted by $\mathcal{A}$.

Succinctness of LTL+Past

Proof.

($\mathcal{P}$): any two states that agree on propositions p_1 to p_n also agree on proposition p_0.

For each $K \subseteq \{0, ..., 2^n - 1\}$, we define $w_K = b_0 ... b_{2^n-1}$ with

$$b_i = \begin{cases} a_i & \text{if } i \in K \\ a_i \cup \{p_0\} & \text{otherwise} \end{cases}$$

Lemma

For any $K \subseteq \{0, ..., 2^n - 1\}$, the word w_K^ω is accepted by $\mathcal{A}$.

Lemma

For any $K \neq K'$, the word $w_{K'} \cdot w_K^\omega$ is not accepted by $\mathcal{A}$.

Succinctness of LTL+Past

Proof.

($\mathcal{P}$): any two states that agree on propositions p_1 to p_n also agree on proposition p_0.

Lemma

For any $K \subseteq \{0, ..., 2^n - 1\}$, the word w_K^ω is accepted by $\mathcal{A}$.

Lemma

For any $K \neq K'$, the word $w_{K'} \cdot w_K^\omega$ is not accepted by $\mathcal{A}$.

For any $K \neq K'$, the states reached after reading w_K and after reading $w_{K'}$ must be different.

Succinctness of LTL+Past

Proof.

($\mathcal{P}$): any two states that agree on propositions p_1 to p_n also agree on proposition p_0.

Lemma

For any $K \subseteq \{0, ..., 2^n - 1\}$, the word w_K^ω is accepted by $\mathcal{A}$.

Lemma

For any $K \neq K'$, the word $w_{K'} \cdot w_K^\omega$ is not accepted by $\mathcal{A}$.

Theorem

Any Büchi automaton $\mathcal{A}$ characterizing property ($\mathcal{P}$) has at least 2^{2^n} states.

Succinctness of LTL+Past

Proof.

($\mathcal{P}$): any two states that agree on propositions p_1 to p_n also agree on proposition p_0.

Theorem

Any Büchi automaton $\mathcal{A}$ characterizing property ($\mathcal{P}$) has at least 2^{2^n} states.

Corollary

Any LTL formula expressing property ($\mathcal{P}$) has size at least 2^{n-1}.

Succinctness of LTL+Past

Proof.

Consider now the following property, slightly different:

($\mathcal{P}'$): any state that agrees on propositions p_1 to p_n with the initial state also agrees on proposition p_0.

Succinctness of LTL+Past

Proof.

Consider now the following property, slightly different:

($\mathcal{P}'$): any state that agrees on propositions p_1 to p_n with the initial state also agrees on proposition p_0.

This can be expressed in LTL+Past by the following (polynomial-size) formula:

$$\mathbf{G}\Big(\big(\bigwedge_{i \geq 1} p_i \Leftrightarrow \mathbf{F}^{-1}\,\mathbf{G}^{-1}\,p_i\big) \Rightarrow \big(p_0 \Leftrightarrow \mathbf{F}^{-1}\,\mathbf{G}^{-1}\,p_0\big)\Big).$$

Succinctness of LTL+Past

Proof.

Consider now the following property, slightly different:

($\mathcal{P}'$): any state that agrees on propositions p_1 to p_n with the initial state also agrees on proposition p_0.

This can be expressed in LTL+Past by the following (polynomial-size) formula:

$$\mathbf{G}\Big(\big(\bigwedge_{i\geq 1} p_i \Leftrightarrow \mathbf{F}^{-1}\,\mathbf{G}^{-1}\,p_i\big) \Rightarrow \big(p_0 \Leftrightarrow \mathbf{F}^{-1}\,\mathbf{G}^{-1}\,p_0\big)\Big).$$

Let φ be an LTL formula expressing property ($\mathcal{P}'$). Then $\mathbf{G}\,\varphi$ precisely expresses property ($\mathcal{P}$), and thus has size at least 2^{n-1}. □

Outline of the course

1. Linear-time temporal logics
   - Algorithms for verifying LTL formulas (cont'd)
   - Back to expressiveness

2. Branching-time temporal logics
   - Expressiveness of branchig-time logics
   - Complexity

CTL, CTL$^+$ and CTL*

Definition

$$\text{CTL} = \mathcal{B}(\mathbf{X}, \mathbf{U}) \ni \varphi_b ::= p \mid \neg \varphi_b \mid \varphi_b \vee \varphi_b \mid \mathbf{E}\varphi_l \mid \mathbf{A}\varphi_l$$
$$\varphi_l ::= \mathbf{X}\, \varphi_b \mid \varphi_b \,\mathbf{U}\, \varphi_b$$

$$\text{CTL}^+ = \mathcal{B}^+(\mathbf{X}, \mathbf{U}) \ni \varphi_b ::= p \mid \neg \varphi_b \mid \varphi_b \vee \varphi_b \mid \mathbf{E}\varphi_l \mid \mathbf{A}\varphi_l$$
$$\varphi_l ::= \neg \varphi_l \mid \varphi_l \vee \varphi_l \mid \mathbf{X}\, \varphi_b \mid \varphi_b \,\mathbf{U}\, \varphi_b$$

$$\text{CTL}^* = \mathcal{B}^*(\mathbf{X}, \mathbf{U}) \ni \varphi_b ::= p \mid \neg \varphi_b \mid \varphi_b \vee \varphi_b \mid \mathbf{E}\varphi_l \mid \mathbf{A}\varphi_l$$
$$\varphi_l ::= \varphi_b \mid \neg \varphi_l \mid \varphi_l \vee \varphi_l \mid \mathbf{X}\, \varphi_l \mid \varphi_l \,\mathbf{U}\, \varphi_l$$

CTL and CTL$^+$ are equally expressive

Theorem

CTL$^+$ can be translated in CTL.

Example

$$\mathbf{E}(p\ \mathbf{U}\ q \wedge p'\ \mathbf{U}\ q') \equiv \mathbf{E}(p \wedge p')\ \mathbf{U}\,(q \wedge \mathbf{E}p'\ \mathbf{U}\ q') \vee \mathbf{E}(p \wedge p')\ \mathbf{U}\,(q' \wedge \mathbf{E}p\ \mathbf{U}\ q)$$

CTL and CTL$^+$ are equally expressive

Theorem

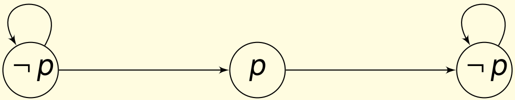
CTL$^+$ can be translated in CTL.

Theorem

$\mathbf{E\,G\,F}\,p$ *cannot be expressed in CTL.*

Example

The tentative formula $\mathbf{E\,G\,E\,F}\,p$ is not equivalent:



CTL and CTL$^+$ are equally expressive

Theorem

CTL$^+$ can be translated in CTL.

Theorem

$\mathbf{E\,G\,F}\,p$ *cannot be expressed in CTL.*

Definition

$$\mathrm{ECTL} = \mathcal{B}(\mathbf{X}, \mathbf{U}, \overset{\infty}{\mathbf{F}})$$
$$\mathrm{ECTL}^+ = \mathcal{B}^+(\mathbf{X}, \mathbf{U}, \overset{\infty}{\mathbf{F}})$$

CTL and CTL$^+$ are equally expressive

Theorem

CTL$^+$ can be translated in CTL.

Theorem

$\mathbf{E}\,\mathbf{G}\,\mathbf{F}\,p$ *cannot be expressed in CTL.*

Theorem

$\mathbf{E}(\overset{\infty}{\boldsymbol{F}} p \wedge \overset{\infty}{\boldsymbol{F}} q)$ *cannot be expressed in ECTL.*

CTL and CTL$^+$ are equally expressive

Theorem

CTL$^+$ can be translated in CTL.

Theorem

$\mathbf{E}\,\mathbf{G}\,\mathbf{F}\,p$ *cannot be expressed in CTL.*

Theorem

$\mathbf{E}(\overset{\infty}{\mathbf{F}} p \wedge \overset{\infty}{\mathbf{F}} q)$ *cannot be expressed in ECTL.*

Theorem

$\mathbf{E}(p \;\mathbf{U}\; q \vee p' \;\mathbf{U}\; q')\;\mathbf{U}\; r$ *cannot be expressed in ECTL$^+$.*

CTL and CTL$^+$ are equally expressive

Theorem

CTL$^+$ can be translated in CTL.

Theorem

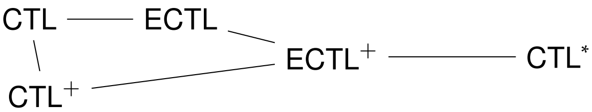
$\mathbf{E}\,\mathbf{G}\,\mathbf{F}\,p$ *cannot be expressed in CTL.*

Theorem

$\mathbf{E}(\overset{\infty}{\mathbf{F}}p \wedge \overset{\infty}{\mathbf{F}}q)$ *cannot be expressed in ECTL.*

Theorem

$\mathbf{E}(p\ \mathbf{U}\ q \vee p'\ \mathbf{U}\ q')\ \mathbf{U}\ r$ *cannot be expressed in ECTL$^+$.*



Outline of the course

1. Linear-time temporal logics
   - Algorithms for verifying LTL formulas (cont'd)
   - Back to expressiveness
2. Branching-time temporal logics
   - Expressiveness of branchig-time logics
   - Complexity

Complexity of branching-time logic verification

Theorem

	Model-checking	*Satisfiability*
CTL	PTIME-*complete*	EXPTIME-*complete*
CTL^+	Δ_2^P-*complete*	2EXPTIME-*complete*
ECTL	PTIME-*complete*	EXPTIME-*complete*
$ECTL^+$	Δ_2^P-*complete*	2EXPTIME-*complete*
CTL^*	PSPACE-*complete*	2EXPTIME-*complete*

ECTL model-checking

Theorem

Model-checking ECTL is PTIME*-complete.*

ECTL model-checking

Theorem

Model-checking ECTL is PTIME*-complete.*

Proof.

- hardness in PTIME: encode CIRCUIT-VALUE as a CTL model-checking problem.
- membership in PTIME: recursively label each state with the set of subformulas it satisfies.

□