QLTL Model-Checking

François Laroussinie, Loriane Leclercq[1] and Arnaud Sangnier

CSL 2024
February 23, 2024

[1]This work has been partially funded by ANR projects ProMiS ANR-19-CE25-0015

Introduction

Temporal logic are very useful to specify computer systems.

- Expressiveness
- Verification:
  - satisfiability
  - model-checking

- Extension of LTL with quantifications:
  - QLTL in tree semantics: well studied by Sistla *et al.* [Sis83; SVW87]: satisfiability and model-checking are non-elementary
  - QLTL in structure semantics: satisfiability studied by French [Fre03] is undecidable

- In this paper:
  - complexity of existential and universal QLTL model-checking in structure semantics
  - variants with restrictions on the form of the structure

LTL definition

LTL Syntax: $\varphi ::= q \mid \neg\varphi \mid \varphi \vee \varphi \mid \mathbf{X}\varphi \mid \varphi\mathbf{U}\varphi$

Semantics:

$(\rho, \lambda) \models p$ iff $p \in \lambda(\rho(0))$
$(\rho, \lambda) \models \neg\varphi$ iff $(\rho, \lambda) \not\models \varphi$
$(\rho, \lambda) \models \varphi \vee \psi$ iff $(\rho, \lambda) \models \varphi$ or $(\rho, \lambda) \models \psi$
$(\rho, \lambda) \models \mathbf{X}\varphi$ iff $(\rho_{\geq 1}, \lambda) \models \varphi$
$(\rho, \lambda) \models \varphi\mathbf{U}\psi$ iff $\exists i \geq 0$ s.t. $(\rho_{\geq i}, \lambda) \models \psi$ and $(\rho_{\geq j}, \lambda) \models \varphi$ $\forall i > j \geq 0$

Abbreviations: $\mathbf{F}\varphi = \top\mathbf{U}\varphi$ and $\mathbf{G}\varphi = \neg\mathbf{F}\neg\varphi$

Kripke structure

Definition

A *Kripke structure (KS)* is a tuple $\mathcal{K} = \langle Q, R, q_{in}, \ell \rangle$ where:

- Q is a finite set of states,
- $R \subseteq Q \times Q$ is a set of transitions,
- $q_{in} \in Q$ is the initial control state and
- $\ell \colon Q \to 2^{\mathsf{AP}}$ is a labelling function.

Kripke structure
Definition

A *Kripke structure (KS)* is a tuple $\mathcal{K} = \langle Q, R, q_{in}, \ell \rangle$ where:

- Q is a finite set of states,
- $R \subseteq Q \times Q$ is a set of transitions,
- $q_{in} \in Q$ is the initial control state and
- $\ell \colon Q \to 2^{\mathsf{AP}}$ is a labelling function.

An *execution* in $\mathcal{K}$ is an infinite sequence of states $\rho = q_0 q_1 q_2 \ldots$ s.t. $\forall i \in \mathbb{N}, (q_i, q_{i+1}) \in R$.
A *labelled execution* (ρ, λ) with an execution ρ and a labelling function $\lambda : Q \to 2^{\mathsf{AP}}$.

Kripke structure

Definition

A *Kripke structure (KS)* is a tuple $\mathcal{K} = \langle Q, R, q_{in}, \ell \rangle$ where:

- Q is a finite set of states,
- $R \subseteq Q \times Q$ is a set of transitions,
- $q_{in} \in Q$ is the initial control state and
- $\ell \colon Q \to 2^{\mathsf{AP}}$ is a labelling function.

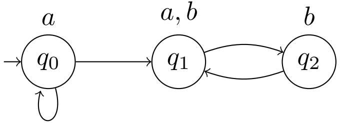
An *execution* in $\mathcal{K}$ is an infinite sequence of states $\rho = q_0 q_1 q_2 \ldots$ s.t. $\forall i \in \mathbb{N}, (q_i, q_{i+1}) \in R$.
A *labelled execution* (ρ, λ) with an execution ρ and a labelling function $\lambda : Q \to 2^{\mathsf{AP}}$.

$\rho(i) := q_i$ the i-th state and $\rho_{\geq i} := q_i q_{i+1} q_{i+2} \ldots$ the i-th suffix.
$\mathsf{Exec}^{\mathsf{lab}}{}_{\mathcal{K}}(q)$ is the set of labelled executions (ρ, ℓ) in $\mathcal{K}$ starting form q.

Kripke structure

Example

A Kripke structure $\mathcal{K}$:

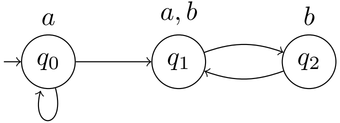


with $\ell(q_0) = \{a\}$, $\ell(q_1) = \{a, b\}$ and $\ell(q_2) = \{b\}$

Kripke structure

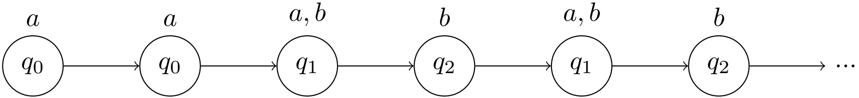
Example

A Kripke structure $\mathcal{K}$:



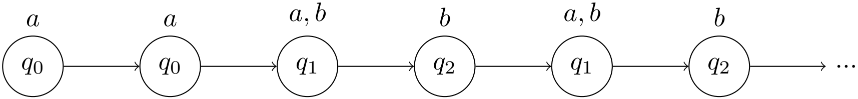
with $\ell(q_0) = \{a\}$, $\ell(q_1) = \{a, b\}$ and $\ell(q_2) = \{b\}$

A execution $\rho = q_0 q_0 (q_1 q_2)^\omega$ of $\mathcal{K}$:



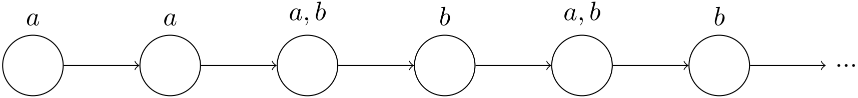
Execution in structure semantics vs tree semantics

Labelled execution (ρ, λ) with $\rho \in S^\omega$ for the structure semantics:



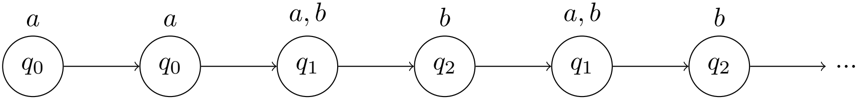
with $\lambda(q_0) = \{a\}$, $\lambda(q_1) = \{a, b\}$ and $\lambda(q_2) = \{b\}$

Execution $\pi \in (2^{\mathsf{AP}})^\omega$ for LTL and the tree semantics:



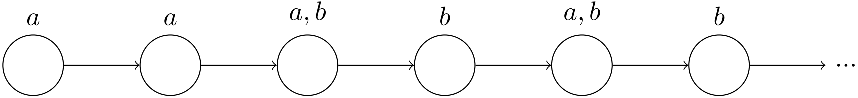
Execution in structure semantics vs tree semantics

Labelled execution (ρ, λ) with $\rho \in S^\omega$ for the structure semantics:



with $\lambda(q_0) = \{a\}$, $\lambda(q_1) = \{a, b\}$ and $\lambda(q_2) = \{b\}$
Notation : $(\rho, \lambda) \models \varphi$

Execution $\pi \in (2^{\mathsf{AP}})^\omega$ for LTL and the tree semantics:



QLTL definition

Structure semantics

QLTL syntax: $\varphi ::= q \mid \neg\varphi \mid \varphi \vee \varphi \mid \mathbf{X}\varphi \mid \varphi\mathbf{U}\varphi \mid \exists p.\ \varphi$

Semantics:

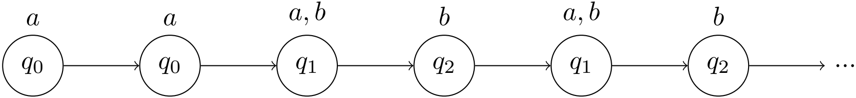
$(\rho, \lambda) \models p$ iff $p \in \lambda(\rho(0))$
$(\rho, \lambda) \models \neg\varphi$ iff $(\rho, \lambda) \not\models \varphi$
$(\rho, \lambda) \models \varphi \vee \psi$ iff $(\rho, \lambda) \models \varphi$ or $(\rho, \lambda) \models \psi$
$(\rho, \lambda) \models \mathbf{X}\varphi$ iff $(\rho_{\geq 1}, \lambda) \models \varphi$
$(\rho, \lambda) \models \varphi\mathbf{U}\psi$ iff $\exists i \geq 0$ s.t. $(\rho_{\geq i}, \lambda) \models \psi$ and $(\rho_{\geq j}, \lambda) \models \varphi$ $\forall i > j \geq 0$
$(\rho, \lambda) \models \exists p.\ \varphi$ iff there exists a labelling λ' s.t. $\lambda' \equiv_{\mathsf{AP}\setminus\{p\}} \lambda$ and $(\rho, \lambda') \models \varphi$

Abbreviation: $\forall p.\varphi = \neg\exists p.\neg\varphi$

Difference between tree and structure semantics

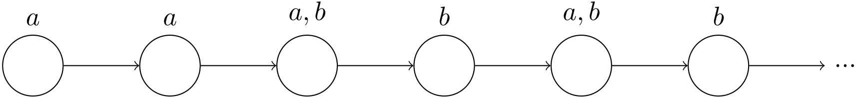
Let $\varphi_1 = \exists p.\, \mathbf{F}\,(p \wedge a \wedge b)$ and $\varphi_2 = \exists p.\, \mathbf{F}\,(p \wedge a \wedge b \wedge \mathbf{XFG}\neg p)$

Labelled execution (ρ, λ) with $\rho \in S^\omega$ for the structure semantics:



with $\lambda(q_0) = \{a\}$, $\lambda(q_1) = \{a, b\}$ and $\lambda(q_2) = \{b\}$

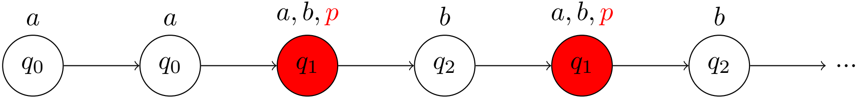
Execution $\pi \in (2^{\mathsf{AP}})^\omega$ for LTL and the tree semantics:



Difference between tree and structure semantics

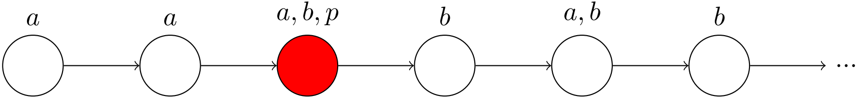
Let $\varphi_1 = \exists p.\, \mathbf{F}\, (p \wedge a \wedge b)$ and $\varphi_2 = \exists p.\, \mathbf{F}\, (p \wedge a \wedge b \wedge \mathbf{XFG}\neg p)$

Labelled execution (ρ, λ) with $\rho \in S^\omega$ for the structure semantics:



with $\lambda(q_0) = \{a\}$, $\lambda(q_1) = \{a, b\}$ and $\lambda(q_2) = \{b\}$
$\lambda'(q_0) = \{a\}$, $\lambda'(q_1) = \{a, b, p\}$ and $\lambda'(q_2) = \{b\}$

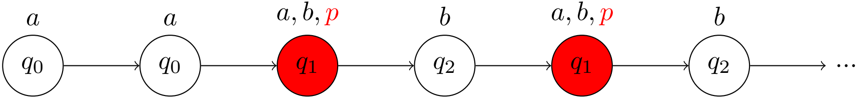
Execution $\pi \in (2^{\mathsf{AP}})^\omega$ for LTL and the tree semantics:



Difference between tree and structure semantics

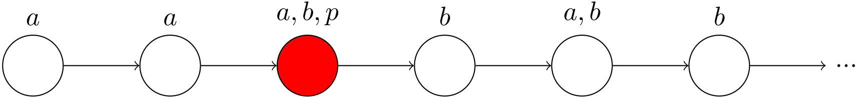
Let $\varphi_1 = \exists p.\, \mathbf{F}\,(p \wedge a \wedge b)$ and $\varphi_2 = \exists p.\, \mathbf{F}\,(p \wedge a \wedge b \wedge \mathbf{XFG}\neg p)$

Labelled execution (ρ, λ) with $\rho \in S^\omega$ for the structure semantics:



with $\lambda(q_0) = \{a\}$, $\lambda(q_1) = \{a, b\}$ and $\lambda(q_2) = \{b\}$
$\lambda'(q_0) = \{a\}$, $\lambda'(q_1) = \{a, b, p\}$ and $\lambda'(q_2) = \{b\}$
$(\rho, \lambda) \models \varphi_1$

Execution $\pi \in (2^{\mathsf{AP}})^\omega$ for LTL and the tree semantics:

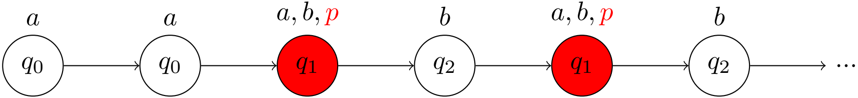


$\pi \models \varphi_1$ and $\pi \models \varphi_2$

Difference between tree and structure semantics

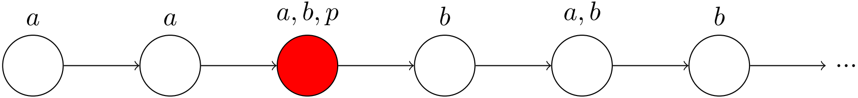
Let $\varphi_1 = \exists p.\, \mathbf{F}\,(p \wedge a \wedge b)$ and $\varphi_2 = \exists p.\, \mathbf{F}\,(p \wedge a \wedge b \wedge \mathbf{XFG}\neg p)$

Labelled execution (ρ, λ) with $\rho \in S^\omega$ for the structure semantics:



with $\lambda(q_0) = \{a\}$, $\lambda(q_1) = \{a, b\}$ and $\lambda(q_2) = \{b\}$
$\lambda'(q_0) = \{a\}$, $\lambda'(q_1) = \{a, b, p\}$ and $\lambda'(q_2) = \{b\}$
$(\rho, \lambda) \models \varphi_1$ but $(\rho, \lambda) \not\models \varphi_2$

Execution $\pi \in (2^{\mathsf{AP}})^\omega$ for LTL and the tree semantics:



$\pi \models \varphi_1$ and $\pi \models \varphi_2$

Existentially or universally satisfy a formula

We write:

- $\mathcal{K} \models_\exists \varphi$ when φ is satisfied by a labelled execution (ρ, λ) in $\mathcal{K}$ rooted at the initial state q_{in}

- $\mathcal{K} \models_\forall \varphi$ when every such labelled execution in $\mathcal{K}$ satisfy φ

Existentially or universally satisfy a formula

We write:

- $\mathcal{K} \models_\exists \varphi$ when φ is satisfied by a labelled execution (ρ, λ) in $\mathcal{K}$ rooted at the initial state q_{in}
- $\mathcal{K} \models_\forall \varphi$ when every such labelled execution in $\mathcal{K}$ satisfy φ

We have: $\mathcal{K} \models_\exists \varphi \Leftrightarrow \mathcal{K} \not\models_\forall \neg\varphi$.

The satisfiability problem: for a given formula φ, does there exists a KS $\mathcal{K}$ such that $\mathcal{K} \models_\exists \varphi$? is undecidable [Fre03].

What can be expressed using QLTL?

We can label a single control state by p using:

$$\exists^1 p.\varphi \quad = \quad \exists p.\Big(\mathbf{F}p \wedge \big(\forall p'.(\mathbf{F}(p \wedge p') \;\Rightarrow\; \mathbf{G}(p \Rightarrow p'))\big) \wedge \varphi\Big)$$

We ensure that we loop forever around the current state:

$$\Phi_{\mathsf{inf\_loop}} \quad = \quad \exists^1 p.\big(p \wedge \mathbf{GF}p\big)$$

We can specify that any *position* satisfying the proposition a is always followed by the same control state with:

$$\Phi_{\mathsf{succ}} \quad = \quad \exists^1 q.\Big(\mathbf{G}\;(a \Rightarrow \mathbf{X}\;q)\Big)$$

We ensure that an execution is determinist with:

$$\Phi_{\mathsf{deter}} \quad = \quad \forall^1 q.\forall^1 q'.\Big(\mathbf{F}(q \;\wedge\; \mathbf{X}q') \;\Rightarrow\; \mathbf{G}(q \;\Rightarrow\; \mathbf{X}q')\Big)$$

Model-checking problems

Definition

The model-checking (MC) problem consists in verifying that a given formula is satisfied by a given model.

We distinguish between the existential and the universal MC problem:

- $\mathsf{MC}_\exists(\mathsf{QLTL})$:
  Input: a KS $\mathcal{K}$ and a formula $\varphi \in \mathsf{QLTL}$
  Output: Yes iff $\mathcal{K} \models_\exists \varphi$

- $\mathsf{MC}_\forall(\mathsf{QLTL})$:
  Input: a KS $\mathcal{K}$ and a formula $\varphi \in \mathsf{QLTL}$
  Output: Yes iff $\mathcal{K} \models_\forall \varphi$

Model-checking problems

Complexity

Theorem

$\mathsf{MC}_\exists(\mathsf{QLTL})$ *and* $\mathsf{MC}_\forall(\mathsf{QLTL})$ *are* ***EXPSPACE****-complete.*

- Membership:
  - build an alternating Büchi atomaton (ABA) $\mathcal{A}_{\varphi,\mathcal{K}}$ that recognises the executions of $\mathcal{K}$ satisfying the formula φ
  - $|\mathcal{A}_{\varphi,\mathcal{K}}|$ is in $O(|Q| + |\varphi| \cdot 2^{|\mathsf{Prop}(\varphi)| \cdot |Q|})$
  - $\mathcal{K} \models_\exists \varphi \;\; \Leftrightarrow \;\; \mathcal{L}(\mathcal{A}_{\varphi,\mathcal{K}}) \neq \emptyset$
  - checking emptiness of an ABA is PSPACE-complete

- Hardness: for the lower bound, we use a domino tiling problem for a $n \times 2^m$-grid.

Model-checking problems

Complexity

Theorem

$\mathsf{MC}_\exists(\mathsf{QLTL})$ *and* $\mathsf{MC}_\forall(\mathsf{QLTL})$ *are* ***EXPSPACE**-complete.*

- Membership:
  - build an alternating Büchi atomaton (ABA) $\mathcal{A}_{\varphi,\mathcal{K}}$ that recognises the executions of $\mathcal{K}$ satisfying the formula φ
  - $|\mathcal{A}_{\varphi,\mathcal{K}}|$ is in $O(|Q| + |\varphi| \cdot 2^{|\mathsf{Prop}(\varphi)|\cdot|Q|})$
  - $\mathcal{K} \models_\exists \varphi \;\; \Leftrightarrow \;\; \mathcal{L}(\mathcal{A}_{\varphi,\mathcal{K}}) \neq \emptyset$
  - checking emptiness of an ABA is PSPACE-complete

- Hardness: for the lower bound, we use a domino tiling problem for a $n \times 2^m$-grid.

This construction also gives the satifiability of a formula by a KS with a fixed number of states.

Paths and flat structures

Definition

Labelled path: $(\rho_1 \cdot \rho_2^\omega, \ell)$

Flat Kripke structure: each node in the underlying graph belongs to at most one simple cycle (a cycle where each edge appears at most once).

Flat structure

Example

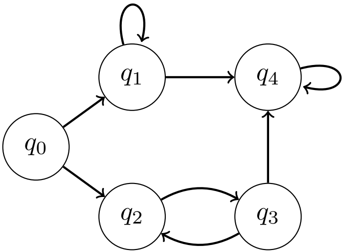


Figure: A flat Kripke structure

Flat structure

Example

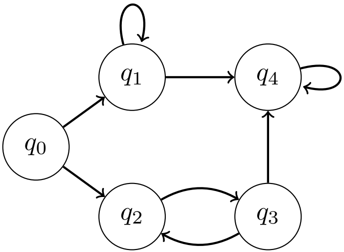


Figure: A flat Kripke structure

Each state is in at most one loop: there is no nested loops.

Paths and flat structures

Model-checking problems

$\mathsf{MC_p}(\mathsf{QLTL})$: Does $(\rho_1 \cdot \rho_2^\omega, \ell) \models \varphi$?

$\mathsf{MC_{f,\exists}}(\mathsf{QLTL})$:
Input: a flat KS $\mathcal{K}$ and a formula $\varphi \in \mathsf{QLTL}$
Output: Yes iff $\mathcal{K} \models_\exists \varphi$

$\mathsf{MC_{f,\forall}}(\mathsf{QLTL})$:
Input: a flat KS $\mathcal{K}$ and a formula $\varphi \in \mathsf{QLTL}$
Output: Yes iff $\mathcal{K} \models_\forall \varphi$

Paths and flat structures

Model-checking problems

$\mathsf{MC_p}(\mathsf{QLTL})$: Does $(\rho_1 \cdot \rho_2^\omega, \ell) \models \varphi$?

Theorem

$\mathsf{MC_p}(\mathsf{QLTL})$ *is PSPACE-complete.*

$\mathsf{MC}_{\mathsf{f},\exists}(\mathsf{QLTL})$:
Input: a flat KS $\mathcal{K}$ and a formula $\varphi \in \mathsf{QLTL}$
Output: Yes iff $\mathcal{K} \models_\exists \varphi$

$\mathsf{MC}_{\mathsf{f},\forall}(\mathsf{QLTL})$:
Input: a flat KS $\mathcal{K}$ and a formula $\varphi \in \mathsf{QLTL}$
Output: Yes iff $\mathcal{K} \models_\forall \varphi$

Theorem

$\mathsf{MC}_{\mathsf{f},\exists}(\mathsf{QLTL})$ *and* $\mathsf{MC}_{\mathsf{f},\forall}(\mathsf{QLTL})$ *are PSPACE-complete.*

Paths and flat structures

Model-checking problems

$\mathsf{MC_p(QLTL)}$: Does $(\rho_1 \cdot \rho_2^\omega, \ell) \models \varphi$?

Theorem

$\mathsf{MC_p(QLTL)}$ *is* *PSPACE-complete.*

$\mathsf{MC_{f,\exists}(QLTL)}$:
Input: a flat KS $\mathcal{K}$ and a formula $\varphi \in \mathsf{QLTL}$
Output: Yes iff $\mathcal{K} \models_\exists \varphi$

$\mathsf{MC_{f,\forall}(QLTL)}$:
Input: a flat KS $\mathcal{K}$ and a formula $\varphi \in \mathsf{QLTL}$
Output: Yes iff $\mathcal{K} \models_\forall \varphi$

Theorem

$\mathsf{MC_{f,\exists}(QLTL)}$ *and* $\mathsf{MC_{f,\forall}(QLTL)}$ *are* *PSPACE-complete.*

For LTL: $\mathsf{MC_p(LTL)}$ is polynomial and $\mathsf{MC_{f,\exists}(LTL)}$ and $\mathsf{MC_{f,\forall}(LTL)}$ are NP-complete.

Paths model-checking
Complexity

Theorem
$\mathsf{MC_p}(\mathsf{QLTL})$ *is PSPACE-complete.*

- PSPACE-membership comes from a translation to QCTL in structure semantics: a QLTL formula have the same truth value on a path as a QCTL formula with any path modality.
- Hardness comes from a QBF problem that clearly is a QLTL formula as well, and that can be checked over a path composed of a self loop q^ω.

Theorem

$\mathsf{MC}_{\mathsf{f},\exists}(\mathsf{QLTL})$ *and* $\mathsf{MC}_{\mathsf{f},\forall}(\mathsf{QLTL})$ *are* PSPACE*-complete.*

In order to obtain a PSPACE algorithm, we use;

- an extension to QLTL of a stuttering theorem for LTL with past
- a finite representation of the set of executions of a flat structure using path schemas $\rho = p_1 l_1 p_2 l_2 \ldots p_k l_k$

Conclusion

- interesting properties expressible with QLTL in structure semantics
- study of the model-checking problem for this logic and some variants with a restriction on the form of the structure

Complexity of QLTL model-checking in structure semantics:

Problem:	Complexity:
$\mathsf{MC}_{\exists}(\mathsf{QLTL})$ $\mathsf{MC}_{\forall}(\mathsf{QLTL})$	EXPSPACE-complete
$\mathsf{MC}_{\mathsf{p}}(\mathsf{QLTL})$ $\mathsf{MC}_{\mathsf{f},\exists}(\mathsf{QLTL})$ $\mathsf{MC}_{\mathsf{f},\forall}(\mathsf{QLTL})$	PSPACE-complete

Conclusion

- interesting properties expressible with QLTL in structure semantics
- study of the model-checking problem for this logic and some variants with a restriction on the form of the structure

Complexity of QLTL model-checking in structure semantics:

Problem:	Complexity:
$\mathsf{MC}_{\exists}(\mathsf{QLTL})$ $\mathsf{MC}_{\forall}(\mathsf{QLTL})$	EXPSPACE-complete
$\mathsf{MC}_{\mathsf{p}}(\mathsf{QLTL})$ $\mathsf{MC}_{\mathsf{f},\exists}(\mathsf{QLTL})$ $\mathsf{MC}_{\mathsf{f},\forall}(\mathsf{QLTL})$	PSPACE-complete

Perspectives:

- satisfiability of fragments of QLTL
- model-checking of more complex structures
- expressivity of prenex formulas

Thank you!

References

[Fre03] T. French. “Quantified Propositional Temporal Logic with Repeating States”. In: *TIME-ICTL’03*. IEEE Comp. Soc. Press, July 2003, pp. 155–165.

[Sis83] A. P. Sistla. “Theoretical Issues in the Design and Verification of Distributed Systems”. PhD thesis. Cambridge, Massachussets, USA: Harvard University, 1983.

[SVW87] A. P. Sistla, M. Y. Vardi, and P. Wolper. “The Complementation Problem for Büchi Automata with Applications to Temporal Logics”. In: *Theoretical Computer Science* 49 (1987), pp. 217–237.

Appendix

QLTL semantics

$(\rho, \lambda) \models p$ iff $p \in \lambda(\rho(0))$
$(\rho, \lambda) \models \neg\varphi$ iff $(\rho, \lambda) \not\models \varphi$
$(\rho, \lambda) \models \varphi \vee \psi$ iff $(\rho, \lambda) \models \varphi$ or $(\rho, \lambda) \models \psi$
$(\rho, \lambda) \models \mathbf{X}\varphi$ iff $(\rho_{\geq 1}, \lambda) \models \varphi$
$(\rho, \lambda) \models \varphi \mathbf{U} \psi$ iff $\exists i \geq 0$ s.t. $(\rho_{\geq i}, \lambda) \models \psi$ and $\forall i > j \geq 0,\ (\rho_{\geq j}, \lambda) \models \varphi$
$(\rho, \lambda) \models \exists p.\ \varphi$ iff there exists a labelling λ' s.t. $\lambda' \equiv_{\mathsf{AP} \setminus \{p\}} \lambda$ and $(\rho, \lambda') \models \varphi$

Model-checking

Upper bound: transition function δ for ABA $\mathcal{A}_{\varphi,\mathcal{K}}$

$$\delta(s_{in}, q) = \delta((\varphi, \ell), q) \wedge \delta(q_{in}, q) \qquad \delta(q, q) = \bigvee_{(q,q') \in R} q'$$

$$\delta(q, q') = \bot \text{ if } q \neq q' \qquad \delta((\top, \lambda), q) = \top \qquad \delta((\bot, \lambda), q) = \bot$$

$$\delta((p, \lambda), q) = \begin{cases} \top & \text{if } p \in \lambda(q) \\ \bot & \text{otherwise} \end{cases} \qquad \delta((\neg p, \lambda), q) = \begin{cases} \bot & \text{if } p \in \lambda(q) \\ \top & \text{otherwise} \end{cases}$$

$$\delta((\psi_1 \vee \psi_2, \lambda), q) = \delta((\psi_1, \lambda), q) \vee \delta((\psi_2, \lambda), q)$$

$$\delta((\psi_1 \wedge \psi_2, \lambda), q) = \delta((\psi_1, \lambda), q) \wedge \delta((\psi_2, \lambda), q) \qquad \delta((\mathbf{X}\psi, \lambda), q) = (\psi, \lambda)$$

$$\delta((\psi_1 \mathbf{U} \psi_2, \lambda), q) = \delta((\psi_2, \lambda), q) \vee (\delta((\psi_1, \lambda), q) \wedge (\psi_1 \mathbf{U} \psi_2, \lambda))$$

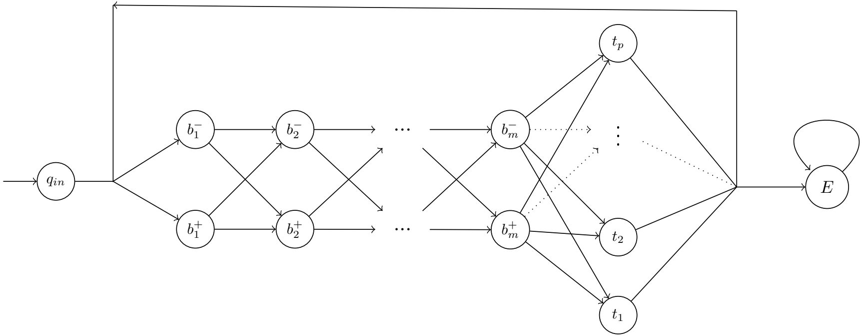
$$\delta((\psi_1 \mathbf{R} \psi_2, \lambda), q) = \delta((\psi_2, \lambda), q) \wedge (\delta((\psi_1, \lambda), q) \vee (\psi_1 \mathbf{R} \psi_2, \lambda))$$

$$\delta((\exists p.\psi, \lambda), q) = \bigvee_{P \subseteq Q} \delta((\psi, \lambda[p \rightsquigarrow P]), q)$$

$$\delta((\forall p.\psi, \lambda), q) = \bigwedge_{P \subseteq Q} \delta((\psi, \lambda[p \rightsquigarrow P]), q)$$

Model-checking

Lower bound: $n \times 2^m$-grid tiling problem



$$\mathsf{StoreNb}(p) = (\bigwedge_{j=1}^{m} \mathbf{X}^{j-1} p) \wedge \mathbf{G}(p \Rightarrow \bigvee_{j=1}^{m} b_j^{\pm}) \wedge \bigwedge_{j=1}^{m} \mathbf{F}((b_j^+ \wedge \neg p) \vee (b_j^- \wedge \neg p))$$

$$\exists p. \Big[\mathsf{StoreNb}(p) \wedge \ \mathbf{X}\Big(\ (\neg \mathsf{Nb}(p))\ \mathbf{U}\ (\mathsf{Nb}(p) \wedge \bigvee_{\substack{t' \in T \text{ s.t.} \\ t_{up} = t'_{down}}} \mathbf{X}^{m+1} t')\Big)\Big]$$

Flat structure

Path schema

Flat structure can be represented by a finite number of strict alternation of paths and loops called path schemas.

A *path* p is a finite sequence of control states $q_0, \dots, q_k \in Q^+$ s.t. $\forall i \in [0, k-1].(q_i, q_{i+1}) \in R$.
A *loop* is a path such that $\mathit{first}(p) = \mathit{last}(p)$, with $\mathit{first}(p) = q_0$ and $\mathit{last}(p) = q_k$.

A *path schema* P is an expression of the form $p_1 l_1 p_2 l_2 \dots p_k l_k$ s.t.

- p_i is a path for all $i \in [1, k]$,
- l_i is a loop for all $i \in [1, k]$ and
- the path schema is a correct execution in $\mathcal{K}$: $\mathit{first}(p_1) = q_{in}$ and $\forall i \in [1, k-1], \mathit{first}(l_i) = \mathit{last}(p_i) = \mathit{first}(p_{i+1})$ and $\mathit{first}(l_k) = \mathit{last}(p_k)$.

PSPACE algorithm for $\mathsf{MC}_{\mathsf{f},\exists}(\mathsf{QLTL})$ and $\mathsf{MC}_{\mathsf{f},\forall}(\mathsf{QLTL})$

For a formula φ with $\mathsf{th}(\varphi) \geq N$ and a flat structure $\mathcal{K}$.

Sketch of Proof.

- For each execution ρ of $\mathcal{K}$, there is a path schema of size smaller than $3 * |Q|$, for example: $\rho = p_1 l_1^{n_1} p_2 l_2^{n_2} \dots p_{k-1} l_{k-1}^{n_{k-1}} p_k l_k^{\omega}$.
- Then we take $\rho' = p_1 l_1^{n'_1} p_2 l_2^{n'_2} \dots p_{k-1} l_{k-1}^{n'_{k-1}} p_k l_k^{\omega}$ with $\forall 1 \leq j < k . n'_j = \min(n_j, 2N+5)$ using stuttering result,
- So for every labelling ℓ, we have $(\rho, \ell) \models \varphi \Leftrightarrow (\rho', \ell) \models \varphi$.

Since there is a finite number of such path schema, we could just non-deterministically check one of them and Savitch's theorem concludes.

$\square$