Quantified-CTL model checking

Arnaud Da Costa\textsuperscript{1}, François Laroussinie\textsuperscript{2}, Nicolas Markey\textsuperscript{1}

\textsuperscript{1} LSV, CNRS & ENS Cachan, France
\textsuperscript{2} LIAFA, CNRS & Univ. Paris7-Diderot, France

September 10, 2011
Model checking

system:

⇒

property:

model-checking algorithm

G(request⇒F grant)

yes/no
Computation-Tree Logic (CTL)

Definition

\[
\text{CTL } \exists \varphi ::= \bigcirc | \varphi \lor \varphi | \neg \varphi | \text{EX}\varphi | \text{EG}\varphi | \text{E}\varphi \text{ U } \varphi
\]
Computation-Tree Logic (CTL)

Definition

\[
\text{CTL } \exists \varphi ::= \bigcirc | \varphi \lor \varphi | \neg \varphi | \mathbf{E} \mathbf{X} \varphi | \mathbf{E} \mathbf{G} \varphi | \mathbf{E} \varphi \mathbf{U} \varphi
\]

\[
\checkmark \quad \mathbf{E}(\text{true } \mathbf{U} \bigcirc) \equiv \mathbf{E} \mathbf{F} \bigcirc
\]
Computation-Tree Logic (CTL)

**Definition**

\[
\text{CTL } \exists \varphi ::= \square \varphi \mid \varphi \lor \varphi \mid \neg \varphi \mid E\ X \varphi \mid E\ G \varphi \mid E\ \varphi \ U \varphi
\]

\[
\checkmark \ E(\text{true} \ U \bigcirc) \equiv E\ F \bigcirc
\]

\[
\checkmark \ E\ G \neg \bigcirc \equiv \neg (A\ F \neg \bigcirc)
\]
Computation-Tree Logic (CTL)

**Definition**

\[
\text{CTL } \exists \phi ::= \begin{array}{c}
\bigcirc \\
\phi \lor \phi \\
\neg \phi \\
E X \phi \\
E G \phi \\
E \phi U \phi
\end{array}
\]

\[
\begin{aligned}
\checkmark & \quad E(\text{true} U \bigcirc) \equiv EF \bigcirc \\
\checkmark & \quad E G \neg \bigcirc \equiv \neg (A F \neg \bigcirc) \\
\times & \quad E(\neg \bigcirc U \bigcirc)
\end{aligned}
\]

Theorem

CTL model checking is \(\text{PTIME}\)-complete.
Computation-Tree Logic (CTL)

**Definition**

\[ \text{CTL } \exists \varphi ::= \bigcirc | \varphi \lor \varphi | \neg \varphi | \text{EX} \varphi | \text{EG} \varphi | \text{E} \varphi \text{ U } \varphi \]

- **Theorem**: CTL model checking is \( \text{PTIME} \)-complete.

\[ \checkmark \  \text{E(true U } \bigcirc \text{) } \equiv \text{EF } \bigcirc \]

\[ \checkmark \  \text{EG } \neg \bigcirc \equiv \neg (\text{AF } \neg \bigcirc) \]

\[ \times \  \text{E(} \neg \bigcirc \text{ U } \bigcirc ) \]

\[ \checkmark \  \text{EG(} \neg \bigcirc \land \text{EF } \bigcirc \) \]
Computation-Tree Logic (CTL)

**Definition**

\[
\text{CTL} \ni \varphi ::= \bigcirc | \varphi \lor \varphi | \neg \varphi | \text{EX} \varphi | \text{EG} \varphi | \text{E} \varphi \text{ U } \varphi
\]

\[
\checkmark \text{E(true U } \bigcirc ) \equiv \text{EF} \bigcirc \\
\checkmark \text{EG} \neg \bigcirc \equiv \neg (\text{AF} \neg \bigcirc )
\times \text{E} (\neg \bigcirc \text{ U } \bigcirc )
\checkmark \text{EG} (\neg \bigcirc \land \text{EF} \bigcirc )
\]

**Theorem**

*CTL* model checking is PTIME-complete.
Quantified-CTL

Definition

\[
QCTL \exists \varphi ::= \bigcirc | \varphi \lor \varphi | \neg \varphi | E X \varphi | E G \varphi | E \varphi U \varphi | \exists \bigcirc. \varphi
\]
Quantified-CTL

### Definition

\[
\text{QCTL } \exists \varphi ::= \triangleleft \mid \varphi \lor \varphi \mid \neg \varphi \mid \text{EX} \varphi \mid \text{EG} \varphi \mid \text{EU} \varphi \mid \exists \triangleleft. \varphi
\]
Quantified-CTL

Definition

\[ QCTL \exists \varphi ::= \bigcirc | \varphi \lor \varphi | \neg \varphi | EX \varphi | EG \varphi | E\varphi U \varphi | \exists \bigcirc. \varphi \]

\[ \checkmark \exists \bigcirc. EG \bigcirc \]

\[ \checkmark \exists \bigcirc. EG \bigcirc \]

\[ \checkmark \exists \bigcirc. EG \bigcirc \]
Quantified-CTL

**Definition**

\[ \text{QCTL } \exists \varphi ::= \bigcirc | \varphi \lor \varphi | \neg \varphi | \text{EX} \varphi | \text{EG} \varphi | \text{EU} \varphi | \exists \bigcirc. \varphi \]

\[ \checkmark \exists \bigcirc. \text{EG} \bigcirc \]

\[ \text{EF}(\forall \bigcirc. (\bigcirc \Rightarrow \text{EX} \bigcirc)) \]
Quantified-CTL

Definition

\[
\text{QCTL } \exists \varphi ::= \bigcirc | \varphi \lor \varphi | \neg \varphi | \mathbf{E} X \varphi | \mathbf{E} G \varphi | \mathbf{E} \varphi \mathbf{U} \varphi | \exists \bigcirc. \varphi
\]

✓ \exists \bigcirc. \mathbf{E} G \bigcirc

✓ \mathbf{E} F(\forall \bigcirc. (\bigcirc \Rightarrow \mathbf{E} X \bigcirc))
Quantified-CTL

**Definition**

$$\text{QCTL } \exists \varphi ::= \bigcirc \mid \varphi \lor \varphi \mid \neg \varphi \mid \text{E X } \varphi \mid \text{E G } \varphi \mid \text{E } \varphi \cup \varphi \mid \exists \bigcirc \cdot \varphi$$

- $\exists \bigcirc \cdot \text{E G } \bigcirc$
- $\text{E F}(\forall \bigcirc \cdot (\bigcirc \Rightarrow \text{E X } \bigcirc))$
- $\text{E F}(\forall \bigcirc \cdot \text{E X } \bigcirc \Rightarrow \text{A X } \bigcirc)$
Quantified-CTL

Definition

\[ \text{QCTL } \exists \varphi ::= \bigcirc | \varphi \lor \varphi | \neg \varphi | \text{EX} \varphi | \text{EG} \varphi | \text{EU} \varphi | \exists \bigcirc \varphi \]

\[
\begin{align*}
&\exists \bigcirc. \text{EG} \\
&\text{EF}(\forall \bigcirc. (\bigcirc \Rightarrow \text{EX} \bigcirc)) \\
&\text{EF}(\forall \bigcirc. \text{EX} \bigcirc \Rightarrow \text{AX} \bigcirc) \\
&\exists \bigcirc. (\bigcirc \land \text{AXAG} \neg \bigcirc)
\end{align*}
\]
Quantified-CTL

**Definition**

\[
\text{QCTL } \exists \varphi ::= \bigcirc \mid \varphi \lor \varphi \mid \neg \varphi \mid \text{E X } \varphi \mid \text{E G } \varphi \mid \text{E } \varphi \mid \text{U } \varphi \mid \exists \cdot \varphi
\]

\[
\checkmark \exists \cdot \text{E G } \cdot
\]

\[
\checkmark \text{EF}(\forall \cdot. \cdot \Rightarrow \text{E X } \cdot)
\]

\[
\checkmark \text{EF}(\forall \cdot. \text{E X } \cdot \Rightarrow \text{A X } \cdot)
\]

\[
\exists \cdot. (\cdot \land \text{A X A G } \neg \cdot)
\]

\[\times\] in the *structure* semantics

\[\checkmark\] in the *tree* semantics
-equivalent structures

**Definition**

Given a Kripke structure \( S = \langle W, R, \ell \rangle \) with \( \ell : W \rightarrow 2^{\text{AP}} \) and an atomic proposition \( \circ \), we define \( S_{\circ} \) as the Kripke structure \( \langle W, R, \ell' \rangle \) with \( \ell'(w) = \ell(w) \setminus \{\circ\} \).

**Definition**

Two Kripke structures \( S = \langle W, R, \ell \rangle \) and \( S' = \langle W', R', \ell' \rangle \) are \( \circ \)-equivalent if \( S_{\circ} = S'_{\circ} \).
equivalent structures

Definition
Given a Kripke structure $S = \langle W, R, \ell \rangle$ with $\ell : W \to 2^{\text{AP}}$ and an atomic proposition $\circ$, we define $S_\circ$ as the Kripke structure $\langle W, R, \ell' \rangle$ with $\ell'(w) = \ell(w) \setminus \{ \circ \}$.

Definition
Two Kripke structures $S = \langle W, R, \ell \rangle$ and $S' = \langle W', R', \ell' \rangle$ are $\circ$-equivalent if $S_\circ = S'_\circ$.

Example
Semantics of QCTL

Definition (structure semantics)

The structure semantics of QCTL extends that of CTL with

\[ S, q \models_s \exists \lozenge \varphi \iff S', q \models_s \varphi \]

for some \( S' \equiv S \).

Definition (tree semantics)

The tree semantics of QCTL is obtained from the structure semantics by

\[ S, q \models_t \varphi \iff \text{tree}(S, q), \epsilon \models_s \varphi. \]
Semantics of QCTL

**Definition (structure semantics)**
The structure semantics of QCTL extends that of CTL with

\[ S, q \models_s \exists \diamond. \varphi \iff S', q \models_s \varphi \]

for some \( S' \equiv_o S \).

**Definition (tree semantics)**
The tree semantics of QCTL is obtained from the structure semantics by

\[ S, q \models_t \varphi \iff \text{tree}(S, q), \epsilon \models_s \varphi. \]
Related work

Linear-time

- studied by Sistla, Vardi, Wolper (1987);
- model-checking is $k$-EXPSPACE-c. with quantifier alternation $k$;
Related work

**Linear-time**
- studied by Sistla, Vardi, Wolper (1987);
- model-checking is $k$-EXPSPACE-c. with quantifier alternation $k$;

**Branching-time**
  - model-checking is NP-c. (resp. EXPTIME-c.) with quantifier depth 1 for structure (resp. tree) semantics;
- Kupferman, Madhusudan, Thiagarajan, Vardi (2000):
  - model-checking is 2-EXPTIME-complete with quantifier depth 2 (tree semantics);
- French (2003):
  - expressiveness: QCTL and QCTL* are equally expressive for the tree semantics.
Alternating-time Temporal Logic (ATL)

Definition

ATL extends CTL with *strategy quantifiers*:

\[
\text{ATL } \exists \varphi ::= \bigcirc \varphi \lor \varphi \mid \neg \varphi \mid \langle A \rangle X \varphi \mid \\
\langle A \rangle \varphi U \varphi \mid \langle A \rangle \neg (\varphi U \varphi)
\]
Alternating-time Temporal Logic (ATL)

**Definition**

ATL extends CTL with *strategy quantifiers*:

\[
\begin{align*}
\text{ATL } & \exists \varphi ::= \bigcirc \mathbin{|} \varphi \lor \varphi \mathbin{|} \neg \varphi \mathbin{|} \langle A \rangle X \varphi \mathbin{|} \\
& \quad \langle A \rangle \varphi U \varphi \mathbin{|} \langle A \rangle \neg(\varphi U \varphi) \\
& \quad \langle \bigcirc \rangle F \mathbin{|} \langle \bigcirc \rangle G (\langle \bigcirc \rangle F)
\end{align*}
\]
Alternating-time Temporal Logic (ATL)

Definition
ATL extends CTL with *strategy quantifiers*:

\[
\text{ATL } \exists \varphi ::= \bigcirc \varphi \lor \varphi \lor \neg \varphi \lor \langle A \rangle X \varphi \lor \langle A \rangle \varphi U \varphi \lor \langle A \rangle \neg (\varphi U \varphi)
\]
Alternating-time Temporal Logic (ATL)

Definition

ATL extends CTL with *strategy quantifiers*:

\[
\text{ATL } \exists \varphi ::= \bigcirc \mid \varphi \lor \varphi \mid \neg \varphi \mid \langle A \rangle X \varphi \mid \langle A \rangle \varphi U \varphi \mid \langle A \rangle \neg(\varphi U \varphi)
\]
 Alternating-time Temporal Logic (ATL)

Definition

ATL extends CTL with strategy quantifiers:

\[
\text{ATL} \ni \varphi ::= \bigcirc \mid \varphi \lor \varphi \mid \neg \varphi \mid \langle A \rangle X \varphi \mid \\
\langle A \rangle \varphi \lor \varphi \mid \langle A \rangle \neg (\varphi \lor \varphi)
\]

\[
\checkmark \langle \bigcirc \rangle F \circ \\
\times \langle \blacksquare \rangle F \circ \\
\times \langle \bigcirc \rangle G(\langle \blacksquare \rangle F \circ)
\]
Alternating-time Temporal Logic (ATL)

Definition

ATL extends CTL with strategy quantifiers:

\[ \text{ATL} \ni \varphi ::= \circ \mid \varphi \lor \varphi \mid \neg \varphi \mid \langle\langle A \rangle\rangle X \varphi \mid \langle\langle A \rangle\rangle \varphi U \varphi \mid \langle\langle A \rangle\rangle \neg (\varphi U \varphi) \]

Theorem

ATL model checking is PTIME-complete.
ATL with strategy contexts

ATL_{sc} has the same syntax as ATL, but different semantics:

\[ \langle \circ \rangle G (\langle \Box \rangle F \circ) \]
**ATL with strategy contexts**

**ATL_{sc}** has the same syntax as ATL, but different semantics:

\[
\langle \circ \rangle G (\langle \Box \rangle F \circ)
\]

Evaluate the formula on the execution tree:
ATL with strategy contexts

ATL_{sc} has the same syntax as ATL, but different semantics:

\[ \langle \bigcirc \rangle \mathbf{G} (\langle \Box \rangle \mathbf{F} \bigcirc) \]

Evaluate the formula on the execution tree:

- apply a strategy of Player \( \bigcirc \);
ATL with strategy contexts

$\text{ATL}_{sc}$ has the same syntax as ATL, but different semantics:

Evaluate the formula on the execution tree:

- apply a strategy of Player $\bigcirc$;
- in the remaining tree, check that Player $\square$ can always enforce a visit to $\bigcirc$. 

$\langle\bigcirc\rangle G(\langle\square\rangle \mathbf{F} \bigcirc)$
ATL with strategy contexts

\( \text{ATL}_{sc} \) has the same syntax as ATL, but different semantics:

\[ \langle \bigcirc \rangle \ G ( \langle \Box \rangle \ F \ C) \]

Evaluate the formula on the execution tree:

- apply a strategy of Player \( \bigcirc \);
- in the remaining tree, check that Player \( \Box \) can always enforce a visit to \( C \).
From ATL$_{sc}$ to QCTL

ATL$_{sc}$ and QCTL are very tightly connected:

strategy = labelling of configurations with the action to be played.

$\langle\langle A \rangle\rangle \varphi$
From ATL\textsubscript{sc} to QCTL

ATL\textsubscript{sc} and QCTL are very tightly connected:

strategy = labelling of configurations with the action to be played.

\[ \langle\langle A \rangle\rangle \varphi \]
From ATL\textsubscript{sc} to QCTL

ATL\textsubscript{sc} and QCTL are very tightly connected:

strategy = labelling of configurations with the action to be played.

\[ \sigma : A \mapsto m_1 \]

\[ A \mapsto m_3 \]

\[ A \mapsto m_2 \]

\[ \langle \langle A \rangle \varphi \]

\[ \neg m_1 \neg m_2 \neg m_3 \]

\[ \exists m_A i. A \mathbin{G} (\text{one} (m_A 1, \ldots, m_A k)) \land A \mathbin{G} (\text{outcome} \Rightarrow \hat{\varphi}) \]

Tree semantics corresponds to ATL\textsubscript{sc}; structure semantics represents memoryless-strategy quantification.
From ATL\textsubscript{sc} to QCTL

ATL\textsubscript{sc} and QCTL are very tightly connected:

strategy = labelling of configurations with the action to be played.

\[
\sigma: A \mapsto m_1
\]

\[
A \mapsto m_3
\]

\[
A \mapsto m_2
\]

\[
\langle\langle A \rangle\rangle \varphi
\]
From ATL\textsubscript{sc} to QCTL

ATL\textsubscript{sc} and QCTL are very tightly connected: strategy = labelling of configurations with the action to be played.

\[ \sigma : A \mapsto m_1 \]

\[ A \mapsto m_3 \]

\[ A \mapsto m_2 \]

\[ \langle \langle A \rangle \rangle \varphi \]

\[ m_1^A \]

\[ \neg m_2^A \]

\[ \neg m_3^A \]

\[ \neg m_1^A \]

\[ \neg m_2^A \]

\[ m_3^A \]

\[ \neg m_1^A \]

\[ \neg m_2^A \]

\[ \neg m_3^A \]

\[ \ldots \]

\[ \ldots \]

\[ \ldots \]
From $\text{ATL}_{sc}$ to QCTL

$\text{ATL}_{sc}$ and QCTL are very tightly connected:

**strategy** $=$ **labelling** of configurations with the action to be played.

$$\sigma : A \mapsto m_1$$

$$A \mapsto m_3$$

$$A \mapsto m_2$$

$$\varphi$$

$$\varphi$$

$$\langle \langle A \rangle \rangle \varphi$$

$$\exists m_i^A \cdot A \mathbf{G}(\text{one}(m_1^A, \ldots, m_k^A)) \land A(\text{outcome} \Rightarrow \hat{\varphi})$$
From ATL\textsubscript{sc} to QCTL

ATL\textsubscript{sc} and QCTL are very tightly connected:

\textbf{strategy} = \textit{labelling} of configurations with the action to be played.

- \sigma: A \mapsto m_1
- A \mapsto m_3
- A \mapsto m_2

\[ \varphi \quad \varphi \]

\[ \neg m_1^A \quad \neg m_2^A \quad m_3^A \]

\[ \neg m_1^A \quad \neg m_2^A \quad \neg m_3^A \]

\[ \exists m_i^A. ~ A G (\text{one}(m_1^A, \ldots, m_k^A)) \land A (\text{outcome} \Rightarrow \hat{\varphi}) \]

- tree semantics corresponds to ATL\textsubscript{sc};
- structure semantics represents memoryless-strategy quantification.
Outline of the talk

1 Introduction
   • Semantics of QCTL
   • Motivations

2 Expressiveness results

3 Model-checking complexity

4 Conclusions
Outline of the talk

1. Introduction
   - Semantics of QCTL
   - Motivations

2. Expressiveness results

3. Model-checking complexity

4. Conclusions
Theorem

*QCTL and QCTL* are equally expressive.

Proof

QCTL can express μ-calculus, hence CTL*, hence QCTL*.

\[ q \models \mu T.\varphi(T) \iff q \models \exists T. \left[ T \land \mathsf{AG}(T \leftrightarrow \varphi(T)) \land \mathsf{AG}(T \Rightarrow U) \Rightarrow \mathsf{AG}(T \Rightarrow U) \right] \]
Expressiveness of QCTL and QCTL*

**Theorem**

QCTL and QCTL* are equally expressive.

**Proof**

QCTL can express $\mu$-calculus, hence CTL*, hence QCTL*.

\[
q \models \mu T.\varphi(T) \iff q \models \exists T. \left[ T \land AG(T \leftrightarrow \varphi(T)) \land \\
\forall U. (AG(U \leftrightarrow \varphi(U)) \Rightarrow AG(T \Rightarrow U)) \right]
\]
Expressiveness of QCTL

Theorem

QCTL and QCTL* are as expressive as MSO.

Remark

The following formula expresses Hamiltonicity (under structure semantics):

\[ \mathbf{E} \mathbf{G} (\exists z. \forall z'. [\text{state}(z) \land \text{state}(z') \land z \land \neg z'] \Rightarrow \mathbf{X}(\neg z \mathbf{U} z')) \]

Using similar ideas, it can express Eulerianity, which cannot be expressed in MSO.

But these are not QCTL* formulas: in QCTL*, propositional quantifiers must be followed by path quantifiers.
Expressiveness of QCTL

Theorem

QCTL and QCTL* are as expressive as MSO.

Remark

The following formula expresses Hamiltonicity (under structure semantics):

$$\mathbf{E}\,\mathbf{G}(\exists z. \forall z'. [\text{state}(z) \land \text{state}(z') \land z \land \neg z'] \Rightarrow \mathbf{X}(\neg z \mathbf{U} z'))$$

Using similar ideas, it can express Eulerianity, which cannot be expressed in MSO.

But these are not QCTL* formulas: in QCTL*, propositional quantifiers must be followed by path quantifiers.
Expressiveness of QCTL

**Theorem**

*Any QCTL formula can be turned in prenex normal form.*
Expressiveness of QCTL

Theorem

Any QCTL formula can be turned in prenex normal form.

Example

\[ \text{E} \; \text{X} (\forall x. \varphi) \equiv \exists z. \forall x. \text{E} \; \text{X} (z \land \varphi) \]
Expressiveness of QCTL

Theorem

Any QCTL formula can be turned in prenex normal form.

Example

\[ \mathbf{E} X (\forall x . \varphi) \equiv \exists z . \forall x . \mathbf{E} X (z \land \varphi) \]

Proof

Transform path quantification into propositional quantification.

- DAG-size linear in the DAG-size of the original formula.
- quantifier alternation linear in the quantifier depth of the original formula.
Outline of the talk

1. Introduction
   - Semantics of QCTL
   - Motivations

2. Expressiveness results

3. Model-checking complexity

4. Conclusions
Model checking under the structure semantics

Theorem

*QCTL and QCTL* model checking is PSPACE-complete.*

Proof

- PSPACE-hardness from QBF;
- in PSPACE by enumerating the possible labellings.
Model checking under the structure semantics

**Theorem**

QCTL and QCTL* model checking is PSPACE-complete.

**Proof**

- PSPACE-hardness from QBF;
- in PSPACE by enumerating the possible labellings.

**Theorem**

- $EQ^k CTL$ model checking is $\Sigma_{k+1}^P$-complete.
  
  $EQ^k CTL$ is the fragment in prenex normal form, with at most $k$ quantifier alternations, starting with existential quantification.

- $Q^k CTL$ model checking is $\Delta_{k+1}^P$-complete.
  
  $Q^k CTL$ is the fragment with at most $k$ quantifiers.
Theorem

- $EQ^k \text{CTL}$ and $Q^k \text{CTL}$ model checking are $k$-EXPTIME-complete.
- $EQ^k \text{CTL}^*$ and $Q^k \text{CTL}^*$ model checking are $(k+1)$-EXPTIME-complete.

Proof

**Algorithm for $EQ^k \text{CTL}$:**
- build (exponential-size) tree automaton for CTL formula;
- use projection for existential quantification (universal quantification by complementing, with exponential blowup).
- build product with the Kripke structure, and check emptiness.

**Hardness:** encoding of $k - 1$-exponential-space alternating Turing machine (Sistla, Vardi, Wolper (1987)).
Conclusions and future work

Conclusions

- QCTL is a nice and natural extension of CTL:
  - same expressiveness as MSO;
  - high complexity.
- understanding QCTL helps us understanding ATL_{sc} (and Strategy Logic).

Future works

- other semantics for modelling finite-memory strategies?
- can QCTL help us find the right bisimulation notion that corresponds to ATL_{sc}?
Conclusions and future work

Conclusions
- QCTL is a nice and natural extension of CTL:
  - same expressiveness as MSO;
  - high complexity.
- understanding QCTL helps us understanding ATL_{sc} (and Strategy Logic).

Future works
- other semantics for modelling finite-memory strategies?
- can QCTL help us find the right bisimulation notion that corresponds to ATL_{sc}?