Temporal logics for multi-agent systems

Nicolas Markey
LSV – ENS Cachan

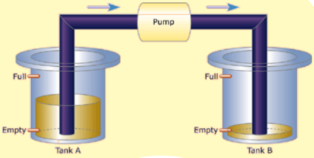
(based on joint works with Thomas Brihaye,
Arnaud Da Costa-Lopes, François Laroussinie)

« Formalisation des Activités Concurrentes »

Toulouse, 16 April 2014

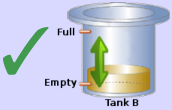
Model checking and synthesis

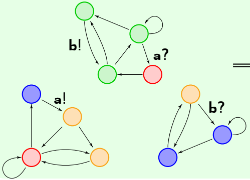
system:



[http://www.embedded.com]

propriété



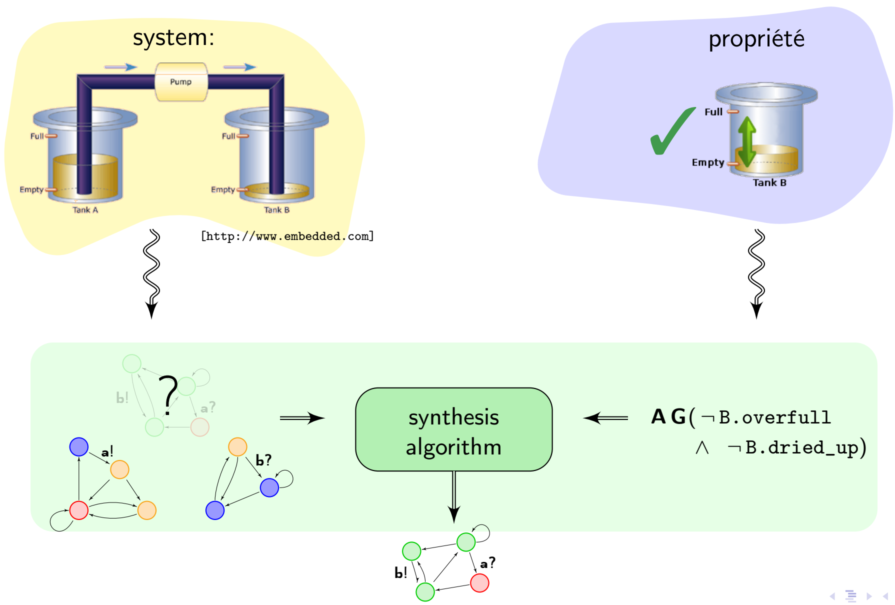


model-checking algorithm

$\mathbf{A}\,\mathbf{G}(\neg \mathtt{B.overfull} \wedge \neg \mathtt{B.dried\_up})$

yes/no

Model checking and synthesis



Outline of the presentation

1. Introduction
2. Basics of CTL and ATL
   - expressing properties of reactive systems
   - efficient verification algorithms
3. Temporal logics for multi-agent systems
   - specifying properties of complex interacting systems
   - expressive power of ATL_{sc}
   - translation into Quantified CTL (QCTL)
   - algorithms for ATL_{sc}
4. Conclusions and future works

Outline of the presentation

1. Introduction

2. Basics of CTL and ATL
   - expressing properties of reactive systems
   - efficient verification algorithms

3. Temporal logics for multi-agent systems
   - specifying properties of complex interacting systems
   - expressive power of ATL_{sc}
   - translation into Quantified CTL (QCTL)
   - algorithms for ATL_{sc}

4. Conclusions and future works

Computation-Tree Logic (CTL)

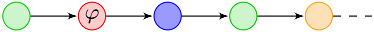
- **atomic propositions:** , , …

Computation-Tree Logic (CTL)

- **atomic propositions:** , , ...
- **boolean combinators:** $\neg\varphi$, $\varphi \vee \psi$, $\varphi \wedge \psi$, ...

Computation-Tree Logic (CTL)

- **atomic propositions:** , , ...
- **boolean combinators:** $\neg\varphi$, $\varphi \vee \psi$, $\varphi \wedge \psi$, ...
- **temporal modalities:**

X φ 

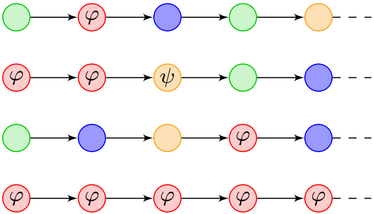
"next φ"

φ **U** ψ 

"φ until ψ"

Computation-Tree Logic (CTL)

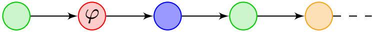
- **atomic propositions:** , ...
- **boolean combinators:** $\neg\varphi$, $\varphi \vee \psi$, $\varphi \wedge \psi$, ...
- **temporal modalities:**



$\mathbf{X}\,\varphi$	"next φ"
$\varphi\ \mathbf{U}\ \psi$	"φ until ψ"
$\mathtt{true}\ \mathbf{U}\ \varphi \equiv \mathbf{F}\,\varphi$	"eventually φ"
$\neg\ \mathbf{F}\ \neg\varphi \equiv \mathbf{G}\,\varphi$	"always φ"

Computation-Tree Logic (CTL)

- **atomic propositions:** , , ...
- **boolean combinators:** $\neg\varphi$, $\varphi \vee \psi$, $\varphi \wedge \psi$, ...
- **temporal modalities:**

$\mathbf{X}\,\varphi$ 

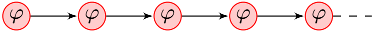
"next φ"

$\varphi\ \mathbf{U}\ \psi$ 

"φ until ψ"

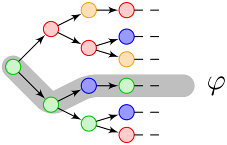
$\mathtt{true}\ \mathbf{U}\ \varphi \equiv \mathbf{F}\,\varphi$ 

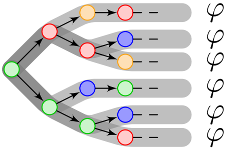
"eventually φ"

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

"always φ"

- **path quantifiers:**

$\mathbf{E}\varphi$ 

$\mathbf{A}\varphi$ 

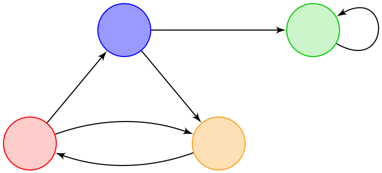
Examples of CTL and CTL* formulas

In CTL, each temporal modality is in the immediate scope of a path quantifier.

Examples of CTL and CTL* formulas

In CTL, each temporal modality is in the immediate scope of a path quantifier.

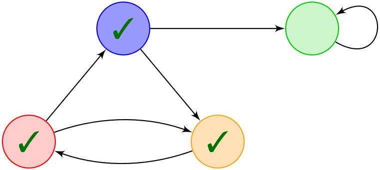
E F is reachable



Examples of CTL and CTL* formulas

In CTL, each temporal modality is in the immediate scope of a path quantifier.

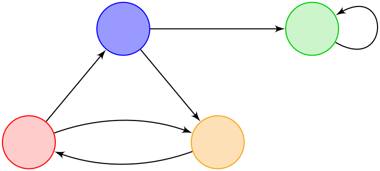
E F ●   ● is reachable



Examples of CTL and CTL* formulas

In CTL, each temporal modality is in the immediate scope of a path quantifier.

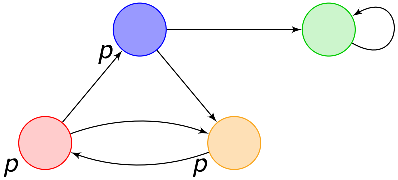
$\mathbf{E}\,\mathbf{G}(\neg \bigcirc \wedge \mathbf{E}\,\mathbf{F} \bigcirc)$ there is a path along which $\bigcirc$ is always reachable, but never reached



Examples of CTL and CTL* formulas

In CTL, each temporal modality is in the immediate scope of a path quantifier.

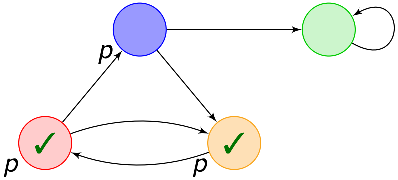
$\mathbf{E}\,\mathbf{G}(\neg \bigcirc \wedge \underbrace{\mathbf{E}\,\mathbf{F}\,\bigcirc}_{p})$ there is a path along which $\bigcirc$ is always reachable, but never reached



Examples of CTL and CTL* formulas

In CTL, each temporal modality is in the immediate scope of a path quantifier.

$\mathbf{E}\,\mathbf{G}(\neg \bigcirc \wedge \underbrace{\mathbf{E}\,\mathbf{F}\,\bigcirc}_{p})$ there is a path along which $\bigcirc$ is always reachable, but never reached



Examples of CTL and CTL* formulas

In CTL, each temporal modality is in the immediate scope of a path quantifier.

Theorem ([CE81,QS82])

CTL model checking is PTIME*-complete.*

[CE81] Clarke, Emerson. Design and Synthesis of Synchronization Skeletons using Branching-Time Temporal Logic. LOP'81.
[QS82] Queille, Sifakis. Specification and verification of concurrent systems in CESAR. SOP'82.

Examples of CTL and CTL* formulas

In CTL, each temporal modality is in the immediate scope of a path quantifier.

Theorem ([CE81,QS82])

CTL model checking is PTIME*-complete.*

Theorem ([KVW94])

CTL model checking on product structures is PSPACE-complete.

[CE81] Clarke, Emerson. Design and Synthesis of Synchronization Skeletons using Branching-Time Temporal Logic. LOP'81.
[QS82] Queille, Sifakis. Specification and verification of concurrent systems in CESAR. SOP'82.
[KVW94] Kupferman, Vardi, Wolper. An automata-theoretic approach to branching-time model checking. CAV'94.

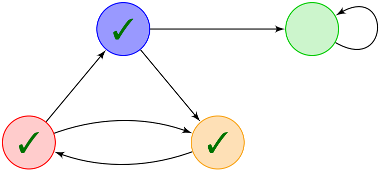
Examples of CTL and CTL* formulas

In CTL*, we have no restriction on modalities and quantifiers.

Examples of CTL and CTL* formulas

In CTL*, we have no restriction on modalities and quantifiers.

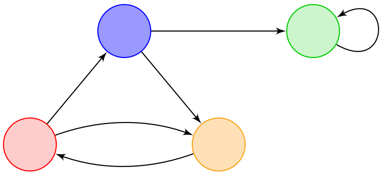
E G F  there is a path visiting  infinitely many times



Examples of CTL and CTL* formulas

In CTL*, we have no restriction on modalities and quantifiers.

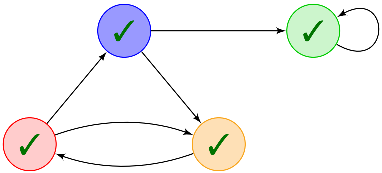
$\mathbf{A}(\mathbf{G}\,\mathbf{F}\,\bigcirc \Rightarrow \mathbf{G}(\neg \bigcirc))$ any path that visits $\bigcirc$ infinitely many times, never visits $\bigcirc$



Examples of CTL and CTL* formulas

In CTL*, we have no restriction on modalities and quantifiers.

$\mathbf{A}(\mathbf{G}\,\mathbf{F} \bigcirc \Rightarrow \mathbf{G}(\neg \bigcirc))$ any path that visits $\bigcirc$ infinitely many times, never visits $\bigcirc$



Examples of CTL and CTL* formulas

In CTL*, we have no restriction on modalities and quantifiers.

Theorem ([EH86,KVW94])

CTL model checking is* PSPACE*-complete.*

Theorem ([KVW94])

CTL model checking on product structures is* PSPACE*-complete.*

[EH86] Emerson, Halpern. "Sometimes" and "Not Never" Revisited: On Branching versus Linear Time Temporal Logic. J.ACM, 1986.
[KVW94] Kupferman, Vardi, Wolper. An automata-theoretic approach to branching-time model checking. CAV'94.

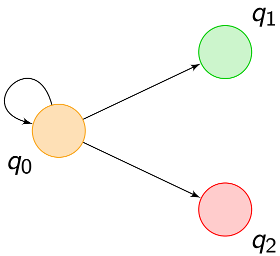
Reasoning about open systems

Reasoning about open systems

Concurrent games

A concurrent game is made of

- a transition system;

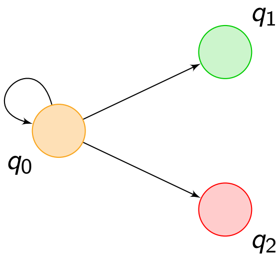


Reasoning about open systems

Concurrent games

A concurrent game is made of

- a transition system;
- a set of agents (or players);

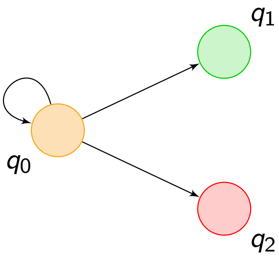


Reasoning about open systems

Concurrent games

A concurrent game is made of

- a transition system;
- a set of agents (or players);
- a table indicating the transition to be taken given the actions of the players.



		player 1		
				
player 2		q_0	q_2	q_1
		q_1	q_0	q_2
		q_2	q_1	q_0

Reasoning about open systems

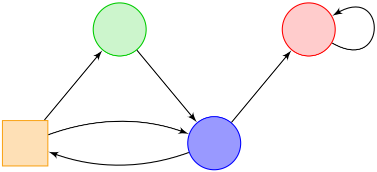
Concurrent games

A concurrent game is made of

- a transition system;
- a set of agents (or players);
- a table indicating the transition to be taken given the actions of the players.

Turn-based games

A turn-based game is a game where only one agent plays at a time.



Reasoning about open systems

Strategies

A strategy for a given player is a function telling what to play depending on what has happened previously.

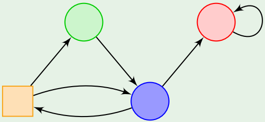
Reasoning about open systems

Strategies

A strategy for a given player is a function telling what to play depending on what has happened previously.

Example

Strategy for player :
alternately go to  and .



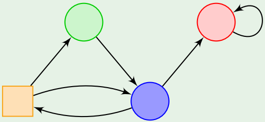
Reasoning about open systems

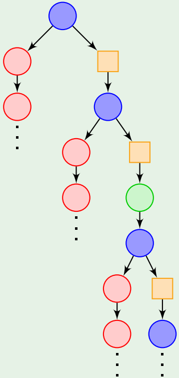
Strategies

A strategy for a given player is a function telling what to play depending on what has happened previously.

Example

Strategy for player :
alternately go to and .





Temporal logics for games: ATL

ATL extends CTL with strategy quantifiers

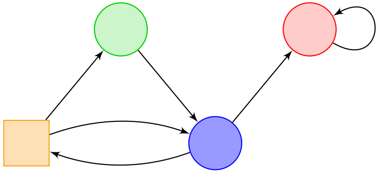
$\langle\!\langle A \rangle\!\rangle\, \varphi$ expresses that A has a strategy to enforce φ.

[AHK02] Alur, Henzinger, Kupferman. Alternating-time Temporal Logic. J. ACM, 2002.

Temporal logics for games: ATL

ATL extends CTL with strategy quantifiers

$\langle\langle A \rangle\rangle\, \varphi$ expresses that A has a strategy to enforce φ.



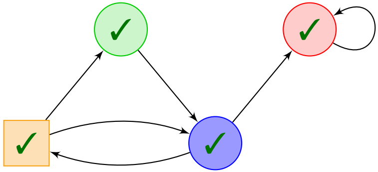
- $\langle\langle \bigcirc \rangle\rangle\ \mathbf{F}\ \bigcirc$

[AHK02] Alur, Henzinger, Kupferman. Alternating-time Temporal Logic. J. ACM, 2002.

Temporal logics for games: ATL

ATL extends CTL with strategy quantifiers

$\langle\!\langle A \rangle\!\rangle\, \varphi$ expresses that A has a strategy to enforce φ.



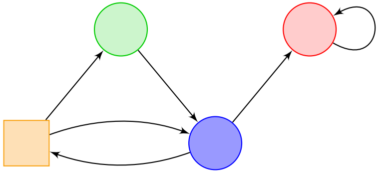
- $\langle\!\langle \bigcirc \rangle\!\rangle\ \mathbf{F}\ \bigcirc$

[AHK02] Alur, Henzinger, Kupferman. Alternating-time Temporal Logic. J. ACM, 2002.

Temporal logics for games: ATL

ATL extends CTL with strategy quantifiers

$\langle\!\langle A \rangle\!\rangle\, \varphi$ expresses that A has a strategy to enforce φ.



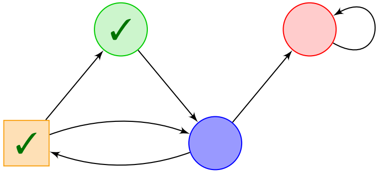
- $\langle\!\langle \bigcirc \rangle\!\rangle$ **F** $\bigcirc$
- $\langle\!\langle \square \rangle\!\rangle$ **F** $\bigcirc$

[AHK02] Alur, Henzinger, Kupferman. Alternating-time Temporal Logic. J. ACM, 2002.

Temporal logics for games: ATL

ATL extends CTL with strategy quantifiers

$\langle\!\langle A \rangle\!\rangle\, \varphi$ expresses that A has a strategy to enforce φ.



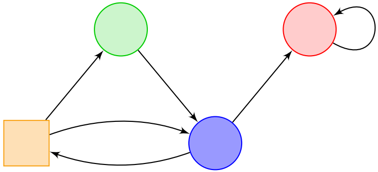
- $\langle\!\langle \bigcirc \rangle\!\rangle$ **F** ○
- $\langle\!\langle \Box \rangle\!\rangle$ **F** ○

[AHK02] Alur, Henzinger, Kupferman. Alternating-time Temporal Logic. J. ACM, 2002.

Temporal logics for games: ATL

ATL extends CTL with strategy quantifiers

$\langle\!\langle A \rangle\!\rangle\, \varphi$ expresses that A has a strategy to enforce φ.



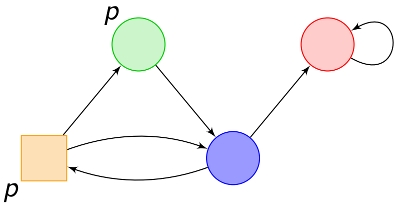
- $\langle\!\langle \bigcirc \rangle\!\rangle\ \mathbf{F}\ \bigcirc$
- $\langle\!\langle \square \rangle\!\rangle\ \mathbf{F}\ \bigcirc$
- $\langle\!\langle \bigcirc \rangle\!\rangle\ \mathbf{G}(\ \langle\!\langle \square \rangle\!\rangle\ \mathbf{F}\ \bigcirc)$

[AHK02] Alur, Henzinger, Kupferman. Alternating-time Temporal Logic. J. ACM, 2002.

Temporal logics for games: ATL

ATL extends CTL with strategy quantifiers

$\langle\!\langle A \rangle\!\rangle\, \varphi$ expresses that A has a strategy to enforce φ.



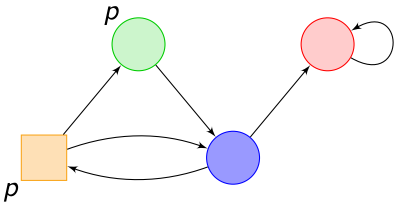
- $\langle\!\langle \bigcirc \rangle\!\rangle\ \mathbf{F}\ \bigcirc$
- $\langle\!\langle \Box \rangle\!\rangle\ \mathbf{F}\ \bigcirc$
- $\langle\!\langle \bigcirc \rangle\!\rangle\ \mathbf{G}(\underbrace{\langle\!\langle \Box \rangle\!\rangle\ \mathbf{F}\ \bigcirc}_{p}) \equiv \langle\!\langle \bigcirc \rangle\!\rangle\ \mathbf{G}\ p$

[AHK02] Alur, Henzinger, Kupferman. Alternating-time Temporal Logic. J. ACM, 2002.

Temporal logics for games: ATL

ATL extends CTL with strategy quantifiers

$\langle\!\langle A \rangle\!\rangle\, \varphi$ expresses that A has a strategy to enforce φ.



- $\langle\!\langle \bigcirc \rangle\!\rangle\ \mathbf{F}\ \bigcirc$
- $\langle\!\langle \square \rangle\!\rangle\ \mathbf{F}\ \bigcirc$
- $\langle\!\langle \bigcirc \rangle\!\rangle\ \mathbf{G}(\underbrace{\langle\!\langle \square \rangle\!\rangle\ \mathbf{F}\ \bigcirc}_{p}) \equiv \langle\!\langle \bigcirc \rangle\!\rangle\ \mathbf{G}\, p$

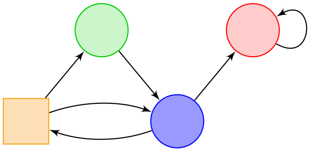
Theorem ([AHK02])

Model checking ATL is PTIME*-complete.*
Model checking ATL* is 2-EXPTIME*-complete.*

[AHK02] Alur, Henzinger, Kupferman. Alternating-time Temporal Logic. J. ACM, 2002.

Outline of the presentation

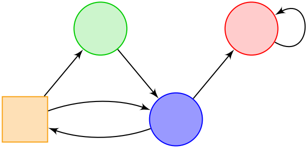
ATL with strategy contexts [BDLM09]



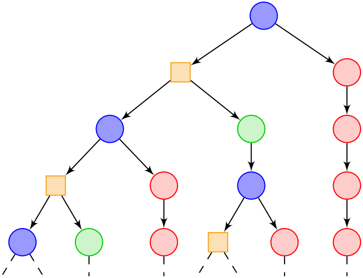
$$\langle\!\langle \bigcirc \rangle\!\rangle\ \mathbf{G}(\ \langle\!\langle \square \rangle\!\rangle\ \mathbf{F}\ \bigcirc)$$

[BDLM09] Brihaye, Da Costa, Laroussinie, M. ATL with strategy contexts. LFCS, 2009.

ATL with strategy contexts [BDLM09]



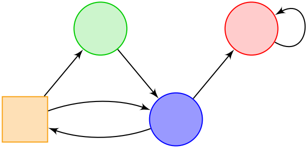
$\langle\!\langle \bigcirc \rangle\!\rangle\ \mathbf{G}(\ \langle\!\langle \square \rangle\!\rangle\ \mathbf{F}\ \bigcirc)$



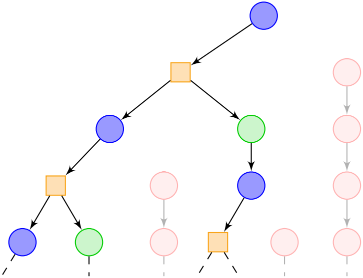
- consider the following strategy of Player ○: "always go to □";

[BDLM09] Brihaye, Da Costa, Laroussinie, M. ATL with strategy contexts. LFCS, 2009.

ATL with strategy contexts [BDLM09]



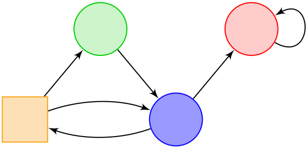
$\langle\!\langle \bigcirc \rangle\!\rangle\ \mathbf{G}(\ \langle\!\langle \square \rangle\!\rangle\ \mathbf{F}\ \bigcirc)$



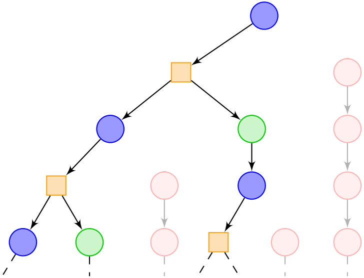
- consider the following strategy of Player ◯: "always go to □";

[BDLM09] Brihaye, Da Costa, Laroussinie, M. ATL with strategy contexts. LFCS, 2009.

ATL with strategy contexts [BDLM09]



$\langle\langle \bigcirc \rangle\rangle\ \mathbf{G}(\ \langle\langle \square \rangle\rangle\ \mathbf{F}\ \bigcirc)$



- consider the following strategy of Player ○: "always go to □";
- in the remaining tree, Player □ can always enforce a visit to ○.

[BDLM09] Brihaye, Da Costa, Laroussinie, M. ATL with strategy contexts. LFCS, 2009.

ATL with strategy contexts

Definition

ATL_{sc} has two new strategy quantifiers: $\langle\cdot A\cdot\rangle\,\varphi$ and $\langle\text{-}A\text{-}\rangle\,\varphi$.

- $\langle\cdot A\cdot\rangle$ is similar to $\langle\!\langle A\rangle\!\rangle$ but assigns the corresponding strategy to A for evaluating φ;

ATL with strategy contexts

Definition

ATL_{sc} has two new strategy quantifiers: $\langle \cdot A \cdot \rangle\, \varphi$ and $\langle \text{-} A \text{-} \rangle\, \varphi$.

- $\langle \cdot A \cdot \rangle$ is similar to $\langle\!\langle A \rangle\!\rangle$ but assigns the corresponding strategy to A for evaluating φ;
- $\langle \text{-} A \text{-} \rangle$ drops the assigned strategies for A.

ATL with strategy contexts

Definition

ATL$_{sc}$ has two new strategy quantifiers: $\langle\cdot A\cdot\rangle\,\varphi$ and $\langle\text{-}A\text{-}\rangle\,\varphi$.

- $\langle\cdot A\cdot\rangle$ is similar to $\langle\!\langle A\rangle\!\rangle$ but assigns the corresponding strategy to A for evaluating φ;
- $\langle\text{-}A\text{-}\rangle$ drops the assigned strategies for A.
- $[\cdot A\cdot]$ is dual to $\langle\cdot A\cdot\rangle$:

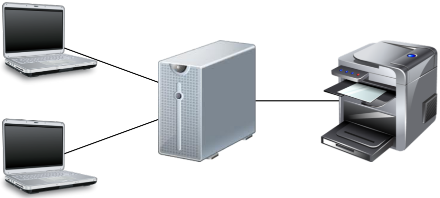
$$[\cdot A\cdot]\,\varphi \equiv \neg\,\langle\cdot A\cdot\rangle\,\neg\varphi$$

  $[\cdot A\cdot]\,\varphi$ which states that any strategy for A has an outcome along which φ holds.

What ATL_{sc} can express

- Client-server interactions for accessing a shared resource:

$$\langle\cdot \mathrm{Server}\cdot\rangle\ \mathbf{G} \left[\begin{array}{l} \displaystyle\bigwedge_{c\in \mathrm{Clients}} \langle\cdot c\cdot\rangle\ \mathbf{F}\, \mathrm{access}_c \\ \wedge \\ \displaystyle\neg \bigwedge_{c\neq c'} \mathrm{access}_c \wedge \mathrm{access}_{c'} \end{array} \right]$$



What ATL_{sc} can express

- Client-server interactions for accessing a shared resource:

$$\langle\!\cdot\text{Server}\cdot\!\rangle\ \mathbf{G}\left[\begin{array}{ll} & \bigwedge_{c\in\text{Clients}} \langle\!\cdot c\cdot\!\rangle\ \mathbf{F}\,\text{access}_c \\ \wedge & \\ & \neg \bigwedge_{c\neq c'} \text{access}_c \wedge \text{access}_{c'} \end{array}\right]$$

- Existence of Nash equilibria:

$$\langle\!\cdot A_1, \dots, A_n\cdot\!\rangle \bigwedge_i \left(\langle\!\cdot A_i\cdot\!\rangle\, \varphi_{A_i} \Rightarrow \varphi_{A_i}\right)$$

What ATL$_{sc}$ can express

- Client-server interactions for accessing a shared resource:

$$\langle\cdot \text{Server} \cdot\rangle\ \mathbf{G} \left[\begin{array}{ll} & \displaystyle\bigwedge_{c \in \text{Clients}} \langle\cdot c \cdot\rangle\ \mathbf{F}\ \text{access}_c \\ \wedge & \\ & \neg \displaystyle\bigwedge_{c \neq c'} \text{access}_c \wedge \text{access}_{c'} \end{array} \right]$$

- Existence of Nash equilibria:

$$\langle\cdot A_1, \ldots, A_n \cdot\rangle \bigwedge_i \left(\langle\cdot A_i \cdot\rangle\, \varphi_{A_i} \Rightarrow \varphi_{A_i} \right)$$

- Existence of dominating strategy:

$$\langle\cdot A \cdot\rangle\ [\cdot B \cdot] \left(\neg \varphi \Rightarrow [\cdot A \cdot]\, \neg \varphi \right)$$

More expressiveness results

Theorem

- *ATL_{sc} is strictly more expressive than ATL,*
- *The operator $\langle\!\cdot A \cdot\!\rangle$ does not add expressive power,*
- *ATL_{sc} is as expressive as ATL^*_{sc}.*

More expressiveness results

Theorem

- ***ATL_{sc} is strictly more expressive than ATL***,
- *The operator $\langle\!\text{-}A\text{-}\!\rangle$ does not add expressive power*,
- *ATL_{sc} is as expressive as ATL^*_{sc}*.

Proof

$$\langle\!\langle A \rangle\!\rangle\, \varphi \equiv \langle\!\text{-}\mathrm{Agt}\text{-}\!\rangle\ \langle\cdot A \cdot\rangle\ \hat{\varphi}$$

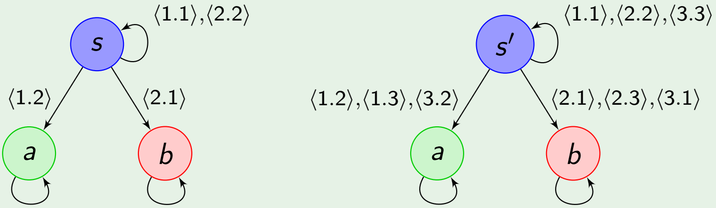
More expressiveness results

Theorem

- ***ATL_{sc} is strictly more expressive than ATL***,
- *The operator $\langle\text{-}A\text{-}\rangle$ does not add expressive power,*
- *ATL_{sc} is as expressive as ATL^*_{sc}.*

Proof

$\langle\cdot 1\cdot\rangle\,(\,\langle\cdot 2\cdot\rangle\ \mathbf{X}\ a \wedge\ \langle\cdot 2\cdot\rangle\ \mathbf{X}\ b)$ is only true in the second game.
But ATL cannot distinguish between these two games.



More expressiveness results

Theorem

- *ATL_{sc} is strictly more expressive than ATL,*
- ***The operator $\langle\!\cdot A \cdot\!\rangle$ does not add expressive power,***
- ***ATL_{sc} is as expressive as ATL^*_{sc}.***

Proof

Replace implicit quantification with explicit one:

$$\langle\!\cdot 1 \cdot\!\rangle\, \varphi \equiv \langle\!\cdot 1 \cdot\!\rangle\, [\!\cdot \mathrm{Agt} \setminus \{1\} \cdot\!]\, \langle\!\cdot \emptyset \cdot\!\rangle\, \widehat{\varphi}$$

$\leadsto$ we can always assume that the context is full.

More expressiveness results

Theorem

- *ATL_{sc} is strictly more expressive than ATL,*
- ***The operator $\langle\!\cdot A \cdot\!\rangle$ does not add expressive power,***
- ***ATL_{sc} is as expressive as ATL^*_{sc}.***

Proof

Replace implicit quantification with explicit one:

$$\langle\!\cdot 1 \cdot\!\rangle\, \varphi \equiv \langle\!\cdot 1 \cdot\!\rangle\, [\cdot \mathrm{Agt} \setminus \{1\} \cdot]\, \langle\!\cdot \emptyset \cdot\!\rangle\, \widehat{\varphi}$$

$\rightsquigarrow$ we can always assume that the context is full.

- $\langle\!\cdot A \cdot\!\rangle\, \varphi$ is then equivalent to $[\cdot A \cdot]\, \langle\!\cdot \emptyset \cdot\!\rangle\, \varphi$;
- $\langle\!\cdot \emptyset \cdot\!\rangle$ can be inserted between two temporal modalities.

Outline of the presentation

1. Introduction
2. Basics of CTL and ATL
   - expressing properties of reactive systems
   - efficient verification algorithms
3. Temporal logics for multi-agent systems
   - specifying properties of complex interacting systems
   - expressive power of ATL_{sc}
   - translation into Quantified CTL (QCTL)
   - algorithms for ATL_{sc}
4. Conclusions and future works

Quantified CTL [ES84,Kup95,Fre01]

QCTL extends CTL with propositional quantifiers

$\exists p.\ \varphi$ means that there exists a labelling of the model with p under which φ holds.

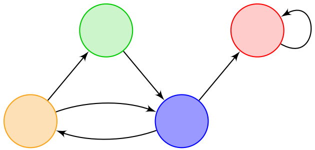
[ES84] Emerson and Sistla. Deciding Full Branching Time Logic. Information & Control, 1984.
[Kup95] Kupferman. Augmenting Branching Temporal Logics with Existential Quantification over Atomic Propositions. CAV, 1995.
[Fre01] French. Decidability of Quantifed Propositional Branching Time Logics. AJCAI, 2001.

Quantified CTL [ES84,Kup95,Fre01]

QCTL extends CTL with propositional quantifiers

$\exists p.\ \varphi$ means that there exists a labelling of the model with p under which φ holds.

- $\mathbf{E}\,\mathbf{F}\,\bigcirc \wedge \forall p.\ \big[\mathbf{E}\,\mathbf{F}(p \wedge \bigcirc) \Rightarrow \mathbf{A}\,\mathbf{G}(\bigcirc \Rightarrow p)\big]$



[ES84] Emerson and Sistla. Deciding Full Branching Time Logic. Information & Control, 1984.
[Kup95] Kupferman. Augmenting Branching Temporal Logics with Existential Quantification over Atomic Propositions. CAV, 1995.
[Fre01] French. Decidability of Quantifed Propositional Branching Time Logics. AJCAI, 2001.

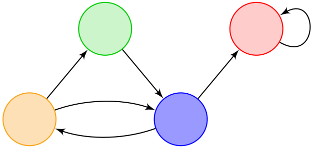
Quantified CTL

[ES84,Kup95,Fre01]

QCTL extends CTL with propositional quantifiers

$\exists p.\ \varphi$ means that there exists a labelling of the model with p under which φ holds.

- $\mathbf{E\,F}\bigcirc \wedge \forall p.\ \big[\mathbf{E\,F}(p \wedge \bigcirc) \Rightarrow \mathbf{A\,G}(\bigcirc \Rightarrow p)\big] \equiv \mathrm{uniq}(\bigcirc)$



[ES84] Emerson and Sistla. Deciding Full Branching Time Logic. Information & Control, 1984.
[Kup95] Kupferman. Augmenting Branching Temporal Logics with Existential Quantification over Atomic Propositions. CAV, 1995.
[Fre01] French. Decidability of Quantifed Propositional Branching Time Logics. AJCAI, 2001.

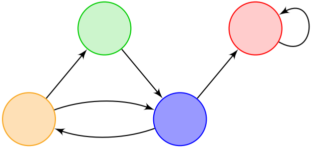
Quantified CTL

[ES84,Kup95,Fre01]

QCTL extends CTL with propositional quantifiers

$\exists p.\ \varphi$ means that there exists a labelling of the model with p under which φ holds.

- $\mathbf{E\,F}\,\bigcirc \wedge \forall p.\ \big[\mathbf{E\,F}(p \wedge \bigcirc) \Rightarrow \mathbf{A\,G}(\bigcirc \Rightarrow p)\big] \equiv \mathrm{uniq}(\bigcirc)$



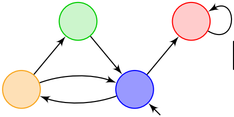
$\rightsquigarrow$ true if we label the Kripke structure;

$\rightsquigarrow$ false if we label the computation tree;

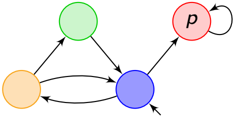
[ES84] Emerson and Sistla. Deciding Full Branching Time Logic. Information & Control, 1984.
[Kup95] Kupferman. Augmenting Branching Temporal Logics with Existential Quantification over Atomic Propositions. CAV, 1995.
[Fre01] French. Decidability of Quantifed Propositional Branching Time Logics. AJCAI, 2001.

Semantics of QCTL

- structure semantics:



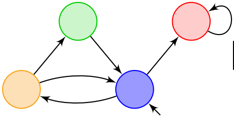
$\models_s \exists p.\varphi \quad \Leftrightarrow$



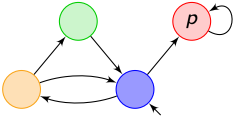
$\models \varphi$

Semantics of QCTL

- structure semantics:

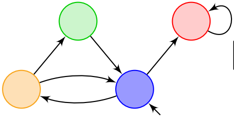


$\models_s \exists p.\varphi \quad \Leftrightarrow$

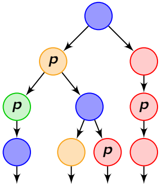


$\models \varphi$

- tree semantics:



$\models_t \exists p.\varphi \quad \Leftrightarrow$



$\models \varphi$

Expressiveness of QCTL

- QCTL can "count":

$$\mathbf{E}\,\mathbf{X}_1\,\varphi \equiv \mathbf{E}\,\mathbf{X}\,\varphi \wedge \forall p.\ [\mathbf{E}\,\mathbf{X}(p \wedge \varphi) \Rightarrow \mathbf{A}\,\mathbf{X}(\varphi \Rightarrow p)]$$
$$\mathbf{E}\,\mathbf{X}_2\,\varphi \equiv \exists q.\ [\mathbf{E}\,\mathbf{X}_1(\varphi \wedge q) \wedge \mathbf{E}\,\mathbf{X}_1(\varphi \wedge \neg q)]$$

[DLM12] Da Costa, Laroussinie, M. Quantified CTL: expressiveness and model checking. CONCUR, 2012.

Expressiveness of QCTL

- QCTL can "count":

$$\mathbf{E}\,\mathbf{X}_1\,\varphi \equiv \mathbf{E}\,\mathbf{X}\,\varphi \wedge \forall p.\ [\mathbf{E}\,\mathbf{X}(p \wedge \varphi) \Rightarrow \mathbf{A}\,\mathbf{X}(\varphi \Rightarrow p)]$$

$$\mathbf{E}\,\mathbf{X}_2\,\varphi \equiv \exists q.\ [\mathbf{E}\,\mathbf{X}_1(\varphi \wedge q) \wedge \mathbf{E}\,\mathbf{X}_1(\varphi \wedge \neg q)]$$

- QCTL can express (least or greatest) fixpoints:

$$\mu T.\varphi(T) \equiv \exists t.\ [\mathbf{A}\,\mathbf{G}(t \iff \varphi(t)) \wedge (\forall t.'(\mathbf{A}\,\mathbf{G}(t' \iff \varphi(t')) \Rightarrow \mathbf{A}\,\mathbf{G}(t \Rightarrow t')))]$$

[DLM12] Da Costa, Laroussinie, M. Quantified CTL: expressiveness and model checking. CONCUR, 2012.

Expressiveness of QCTL

- QCTL can "count":

$$\mathbf{EX}_1\,\varphi \equiv \mathbf{EX}\,\varphi \wedge \forall p.\ [\mathbf{EX}(p \wedge \varphi) \Rightarrow \mathbf{AX}(\varphi \Rightarrow p)]$$
$$\mathbf{EX}_2\,\varphi \equiv \exists q.\ [\mathbf{EX}_1(\varphi \wedge q) \wedge \mathbf{EX}_1(\varphi \wedge \neg q)]$$

- QCTL can express (least or greatest) fixpoints:

$$\mu T.\varphi(T) \equiv \exists t.\ [\mathbf{AG}(t \iff \varphi(t)) \wedge (\forall t.'(\mathbf{AG}(t' \iff \varphi(t')) \Rightarrow \mathbf{AG}(t \Rightarrow t')))]$$

Theorem

QCTL, QCTL* and MSO are equally expressive (under both semantics).

[DLM12] Da Costa, Laroussinie, M. Quantified CTL: expressiveness and model checking. CONCUR, 2012.

QCTL with structure semantics

Theorem

Model checking QCTL for the structure semantics is PSPACE*-complete.*

[DLM12] Da Costa, Laroussinie, M. Quantified CTL: expressiveness and model checking. CONCUR, 2012.

QCTL with structure semantics

Theorem

Model checking QCTL for the structure semantics is PSPACE-*complete.*

Proof

Membership:

Iteratively
- (nondeterministically) pick a labelling,
- check the subformula.

Hardness:

QBF is a special case (without even using temporal modalities).

[DLM12] Da Costa, Laroussinie, M. Quantified CTL: expressiveness and model checking. CONCUR, 2012.

QCTL with structure semantics

Theorem

Model checking QCTL for the structure semantics is PSPACE*-complete.*

Proof

Membership:

Iteratively
- (nondeterministically) pick a labelling,
- check the subformula.

Hardness:
QBF is a special case (without even using temporal modalities).

Theorem

QCTL satisfiability for the structure semantics is undecidable.

[DLM12] Da Costa, Laroussinie, M. Quantified CTL: expressiveness and model checking. CONCUR, 2012.

QCTL with tree semantics

Theorem

- *Model checking QCTL with k quantifiers in the tree semantics is* k-EXPTIME*-complete.*
- *Satisfiability of QCTL with k quantifiers in the tree semantics is* (k+1)-EXPTIME*-complete.*

[DLM12] Da Costa, Laroussinie, M. Quantified CTL: expressiveness and model checking. CONCUR, 2012.
[LM13a] Laroussinie, M. Quantified CTL: expressiveness and complexity. Submitted, 2013.

QCTL with tree semantics

Theorem

- *Model checking QCTL with k quantifiers in the tree semantics is* k-EXPTIME*-complete.*
- *Satisfiability of QCTL with k quantifiers in the tree semantics is* (k+1)-EXPTIME*-complete.*

Proof

Using (alternating) parity tree automata:

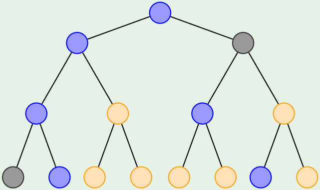
QCTL with tree semantics

Theorem

- *Model checking QCTL with k quantifiers in the tree semantics is* k-EXPTIME-*complete.*
- *Satisfiability of QCTL with k quantifiers in the tree semantics is* (k+1)-EXPTIME-*complete.*

Proof

Using (alternating) parity tree automata:



QCTL with tree semantics

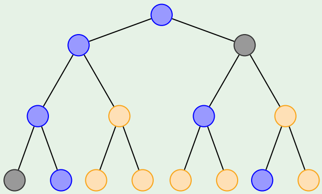
Theorem

- *Model checking QCTL with k quantifiers in the tree semantics is* k-EXPTIME*-complete.*
- *Satisfiability of QCTL with k quantifiers in the tree semantics is* (k+1)-EXPTIME*-complete.*

Proof

Using (alternating) parity tree automata:

$$\delta(q_0, \textcolor{blue}{\bullet}) = (q_0, q_1) \vee (q_1, q_0)$$
$$\delta(q_0, \textcolor{orange}{\bullet}) = (q_1, q_1)$$
$$\delta(q_0, \textcolor{gray}{\bullet}) = (q_2, q_2)$$
$$\delta(q_1, \circledast) = (q_1, q_1)$$
$$\delta(q_2, \circledast) = (q_2, q_2)$$



QCTL with tree semantics

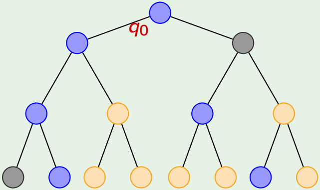
Theorem

- *Model checking QCTL with k quantifiers in the tree semantics is* k-EXPTIME-*complete.*
- *Satisfiability of QCTL with k quantifiers in the tree semantics is* (k+1)-EXPTIME-*complete.*

Proof

Using (alternating) parity tree automata:

$$\delta(q_0, {\color{blue}\bullet}) = (q_0, q_1) \vee (q_1, q_0)$$
$$\delta(q_0, {\color{orange}\bullet}) = (q_1, q_1)$$
$$\delta(q_0, {\color{gray}\bullet}) = (q_2, q_2)$$
$$\delta(q_1, \circledast) = (q_1, q_1)$$
$$\delta(q_2, \circledast) = (q_2, q_2)$$



QCTL with tree semantics

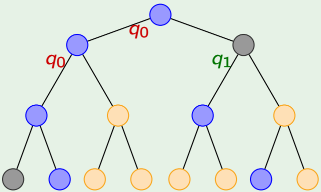
Theorem

- *Model checking QCTL with k quantifiers in the tree semantics is* k-EXPTIME-*complete.*
- *Satisfiability of QCTL with k quantifiers in the tree semantics is* (k+1)-EXPTIME-*complete.*

Proof

Using (alternating) parity tree automata:

$$\delta(q_0, \bigcirc) = (q_0, q_1) \vee (q_1, q_0)$$
$$\delta(q_0, \bigcirc) = (q_1, q_1)$$
$$\delta(q_0, \bigcirc) = (q_2, q_2)$$
$$\delta(q_1, \circledast) = (q_1, q_1)$$
$$\delta(q_2, \circledast) = (q_2, q_2)$$



QCTL with tree semantics

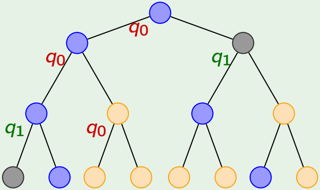
Theorem

- *Model checking QCTL with k quantifiers in the tree semantics is* k-EXPTIME-*complete.*
- *Satisfiability of QCTL with k quantifiers in the tree semantics is* (k+1)-EXPTIME-*complete.*

Proof

Using (alternating) parity tree automata:

$$\delta(q_0, \bigcirc) = (q_0, q_1) \vee (q_1, q_0)$$
$$\delta(q_0, \bigcirc) = (q_1, q_1)$$
$$\delta(q_0, \bigcirc) = (q_2, q_2)$$
$$\delta(q_1, \circledast) = (q_1, q_1)$$
$$\delta(q_2, \circledast) = (q_2, q_2)$$



QCTL with tree semantics

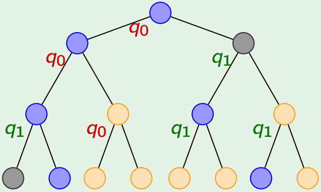
Theorem

- *Model checking QCTL with k quantifiers in the tree semantics is* k-EXPTIME-*complete.*
- *Satisfiability of QCTL with k quantifiers in the tree semantics is* (k+1)-EXPTIME-*complete.*

Proof

Using (alternating) parity tree automata:

$$\delta(q_0, {\color{blue}\bullet}) = (q_0, q_1) \vee (q_1, q_0)$$
$$\delta(q_0, {\color{orange}\bullet}) = (q_1, q_1)$$
$$\delta(q_0, {\color{gray}\bullet}) = (q_2, q_2)$$
$$\delta(q_1, \circledast) = (q_1, q_1)$$
$$\delta(q_2, \circledast) = (q_2, q_2)$$



QCTL with tree semantics

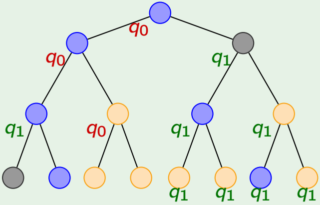
Theorem

- *Model checking QCTL with k quantifiers in the tree semantics is* k-EXPTIME*-complete.*
- *Satisfiability of QCTL with k quantifiers in the tree semantics is* (k+1)-EXPTIME*-complete.*

Proof

Using (alternating) parity tree automata:

$$\delta(q_0, \color{blue}{\bullet}) = (q_0, q_1) \vee (q_1, q_0)$$
$$\delta(q_0, \color{orange}{\bullet}) = (q_1, q_1)$$
$$\delta(q_0, \color{gray}{\bullet}) = (q_2, q_2)$$
$$\delta(q_1, \circledast) = (q_1, q_1)$$
$$\delta(q_2, \circledast) = (q_2, q_2)$$



QCTL with tree semantics

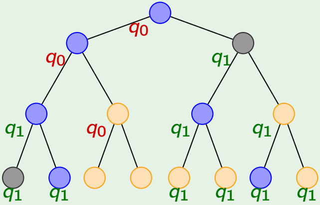
Theorem

- *Model checking QCTL with k quantifiers in the tree semantics is* k-EXPTIME-*complete.*
- *Satisfiability of QCTL with k quantifiers in the tree semantics is* (k+1)-EXPTIME-*complete.*

Proof

Using (alternating) parity tree automata:

$$\delta(q_0, \textcolor{blue}{\bullet}) = (q_0, q_1) \vee (q_1, q_0)$$
$$\delta(q_0, \textcolor{orange}{\bullet}) = (q_1, q_1)$$
$$\delta(q_0, \textcolor{gray}{\bullet}) = (q_2, q_2)$$
$$\delta(q_1, \circledast) = (q_1, q_1)$$
$$\delta(q_2, \circledast) = (q_2, q_2)$$



QCTL with tree semantics

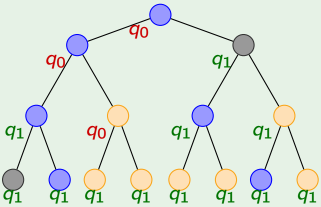
Theorem

- *Model checking QCTL with k quantifiers in the tree semantics is* k-EXPTIME*-complete.*
- *Satisfiability of QCTL with k quantifiers in the tree semantics is* (k+1)-EXPTIME*-complete.*

Proof

Using (alternating) parity tree automata:

$$\delta(q_0, \bigcirc) = (q_0, q_1) \vee (q_1, q_0)$$
$$\delta(q_0, \bigcirc) = (q_1, q_1)$$
$$\delta(q_0, \bigcirc) = (q_2, q_2)$$
$$\delta(q_1, \circledast) = (q_1, q_1)$$
$$\delta(q_2, \circledast) = (q_2, q_2)$$



QCTL with tree semantics

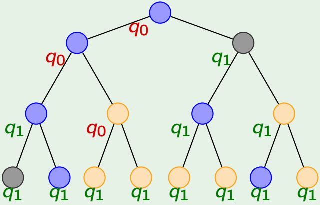
Theorem

- *Model checking QCTL with k quantifiers in the tree semantics is* k-EXPTIME-*complete.*
- *Satisfiability of QCTL with k quantifiers in the tree semantics is* (k+1)-EXPTIME-*complete.*

Proof

Using (alternating) parity tree automata:

$$\delta(q_0, \textcolor{blue}{\bullet}) = (q_0, q_1) \vee (q_1, q_0)$$
$$\delta(q_0, \textcolor{orange}{\bullet}) = (q_1, q_1)$$
$$\delta(q_0, \textcolor{gray}{\bullet}) = (q_2, q_2)$$
$$\delta(q_1, \circledast) = (q_1, q_1)$$
$$\delta(q_2, \circledast) = (q_2, q_2)$$



This automaton corresponds to **E**$\textcolor{blue}{\bullet}$ **U** $\textcolor{orange}{\bullet}$

QCTL with tree semantics

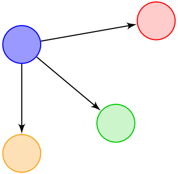
Theorem

- *Model checking QCTL with k quantifiers in the tree semantics is* k-EXPTIME*-complete.*
- *Satisfiability of QCTL with k quantifiers in the tree semantics is* (k+1)-EXPTIME*-complete.*

Proof

- polynomial-size automata for CTL;
- quantification is handled by projection, which first requires removing alternation (exponential blowup);
- an automaton equivalent to a QCTL formula can be built inductively;
- emptiness of an alternating parity tree automaton can be decided in exponential time.

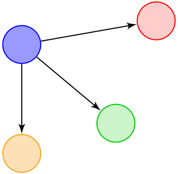
Translating ATL$_{sc}$ into QCTL



- player A has moves m_1^A, ..., m_n^A;
- from the transition table, we can compute the set Next(, A, m_i^A) of states that can be reached from  when player A plays m_i^A.

[DLM12] Da Costa, Laroussinie, M. Quantified CTL: expressiveness and model checking. CONCUR, 2012.

Translating ATL$_{sc}$ into QCTL



- player A has moves m_1^A, ..., m_n^A;
- from the transition table, we can compute the set Next(, A, m_i^A) of states that can be reached from when player A plays m_i^A.

$\langle\cdot A\cdot\rangle\,\varphi$ can be encoded as follows:

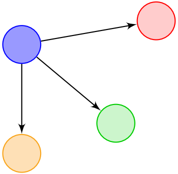
$\exists m_1^A.\ \exists m_2^A \ldots \exists m_n^A.$

- this corresponds to a strategy: $\quad \mathbf{A}\,\mathbf{G}(m_i^A \Leftrightarrow \bigwedge \neg\, m_j^A)$;
- the outcomes all satisfy φ:

$$\mathbf{A}\big[\mathbf{G}(q \wedge m_i^A \Rightarrow \mathbf{X}\,\mathrm{Next}(q, A, m_i^A)) \Rightarrow \varphi\big].$$

[DLM12] Da Costa, Laroussinie, M. Quantified CTL: expressiveness and model checking. CONCUR, 2012.

Translating ATL$_{sc}$ into QCTL



- player A has moves m_1^A, ..., m_n^A;
- from the transition table, we can compute the set Next(, A, m_i^A) of states that can be reached from when player A plays m_i^A.

Corollary

*ATL$_{sc}$ model checking is decidable, with non-elementary complexity (*TOWER*-complete).*

Corollary

ATL$_{sc}^0$ (quantification restricted to memoryless strategies) model checking is PSPACE*-complete.*

[DLM12] Da Costa, Laroussinie, M. Quantified CTL: expressiveness and model checking. CONCUR, 2012.

What about satisfiability?

Theorem

QCTL satisfiability is decidable (for the tree semantics).

What about satisfiability?

Theorem

QCTL satisfiability is decidable (for the tree semantics).

But

Theorem ([TW12])

ATL_{sc} satisfiability is undecidable.

[TW12] Troquard, Walther. On Satisfiability in ATL with Strategy Contexts. JELIA, 2012.

What about satisfiability?

Theorem

QCTL satisfiability is decidable (for the tree semantics).

But

Theorem ([TW12])

ATL_{sc} satisfiability is undecidable.

Why?

The translation from ATL_{sc} to QCTL assumes that the game structure is given!

[TW12] Troquard, Walther. On Satisfiability in ATL with Strategy Contexts. JELIA, 2012.

Satisfiability for turn-based games

Theorem (LM13b)

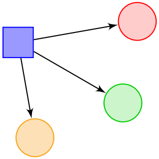
When restricted to turn-based games, ATL_{sc} satisfiability is decidable.

[LM13b] Laroussinie, M. Satisfiability of ATL with strategy contexts. Gandalf, 2013.

Satisfiability for turn-based games

Theorem (LM13b)

When restricted to turn-based games, ATL_{sc} satisfiability is decidable.



- player $\square$ has moves green, orange and red.
- a strategy can be encoded by marking some of the nodes of the tree with proposition mov_A.

$\langle\cdot A\cdot\rangle\ \varphi$ can be encoded as follows:

$\exists \mathsf{mov}_A.$

- it corresponds to a strategy: $\mathbf{A}\,\mathbf{G}(\mathsf{turn}_A \Rightarrow \mathbf{E}\,\mathbf{X}_1\, \mathsf{mov}_A)$;
- the outcomes all satisfy φ: $\mathbf{A}\big[\mathbf{G}(\mathsf{turn}_A \wedge \mathbf{X}\,\mathsf{mov}_A) \Rightarrow \varphi\big]$.

[LM13b] Laroussinie, M. Satisfiability of ATL with strategy contexts. Gandalf, 2013.

What about Strategy Logic? [CHP07,MMV10]

Strategy logic

Explicit quantification over strategies + strategy assignement

Example

$$\langle\cdot A\cdot\rangle\,\varphi \equiv \exists\sigma_1.\mathrm{assign}(\sigma_1, A).\varphi$$

Strategy logic can also be translated into QCTL.

Theorem

- *Strategy-logic model-checking is decidable.*
- *Strategy-logic satisfiability is decidable when restricted to turn-based games.*

[CHP07] Chatterjee, Henzinger, Piterman. Strategy Logic. CONCUR, 2007.
[MMV10] Mogavero, Murano, Vardi. Reasoning about strategies. FSTTCS, 2010.

Conclusions and future works

Conclusions

- QCTL is a powerful extension of CTL;
- it is equivalent to MSO over finite graphs and regular trees;
- it is a nice tool to understand temporal logics for games (ATL with strategy contexts, Strategy Logic, ...);

Conclusions and future works

Conclusions

- QCTL is a powerful extension of CTL;
- it is equivalent to MSO over finite graphs and regular trees;
- it is a nice tool to understand temporal logics for games (ATL with strategy contexts, Strategy Logic, ...);

Future directions

- Defining interesting (expressive yet tractable) fragments of those logics;
- Obtaining practicable algorithms.
- Considering randomised strategies.