Quantified CTL

Nicolas Markey
LSV – ENS Cachan

(joint work with François Laroussinie)

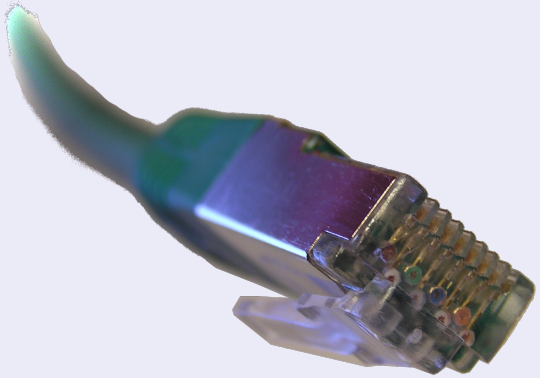
Nord-Pas-de-Calais – Belgium congress of mathematics

Valenciennes, 28 October 2013

Verification of computerised systems

- Computers are everywhere



Verification of computerised systems

- Computers are everywhere

- Bugs are everywhere...


News

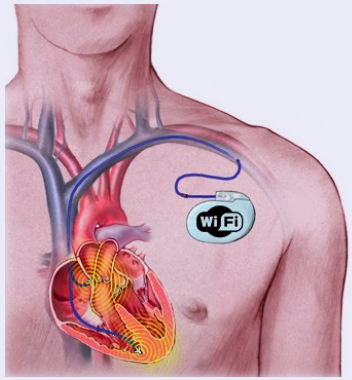
Toyota to recall Prius hybrids over ABS software

See video, below

By Martyn Williams

February 9, 2010 04:59 AM ET Comments (6) Recommended (15) Like

IDG News Service - Toyota plans to recall around 400,000 of its Prius hybrid cars to replace software that controls the antilock braking system (ABS), the auto maker said Tuesday.



Verification of computerised systems

- Computers are everywhere

- Bugs are everywhere...


News

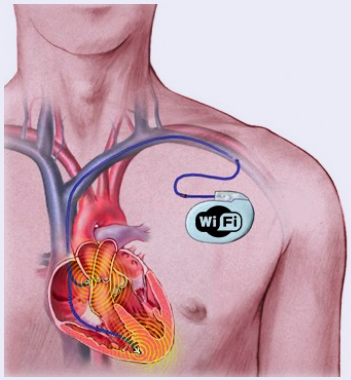
Toyota to recall Prius hybrids over ABS software

See video, below

By Martyn Williams

February 9, 2010 04:59 AM ET Comments (6) Recommended (15) Like

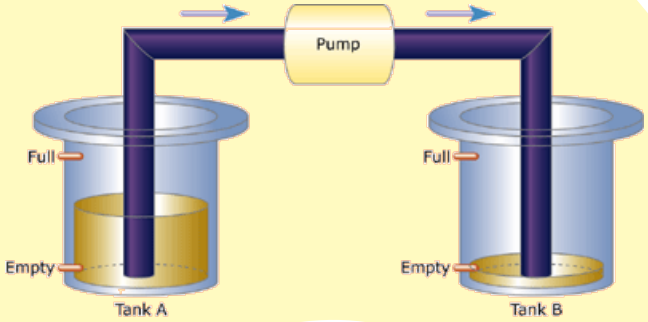
IDG News Service - Toyota plans to recall around 400,000 of its Prius hybrid cars to replace software that controls the antilock braking system (ABS), the auto maker said Tuesday.



- Verification should be everywhere!

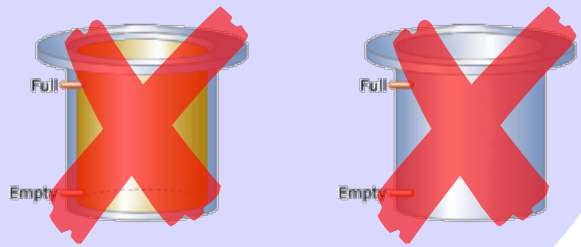
Model checking and synthesis

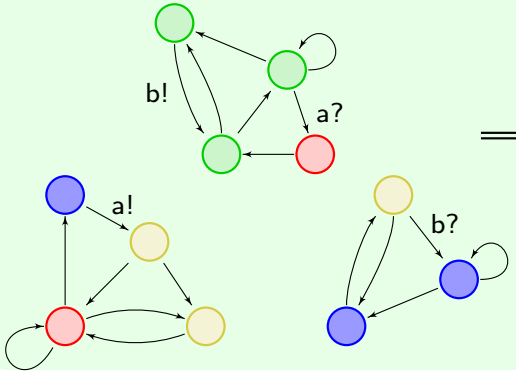
system:



[http://www.embedded.com]

property:



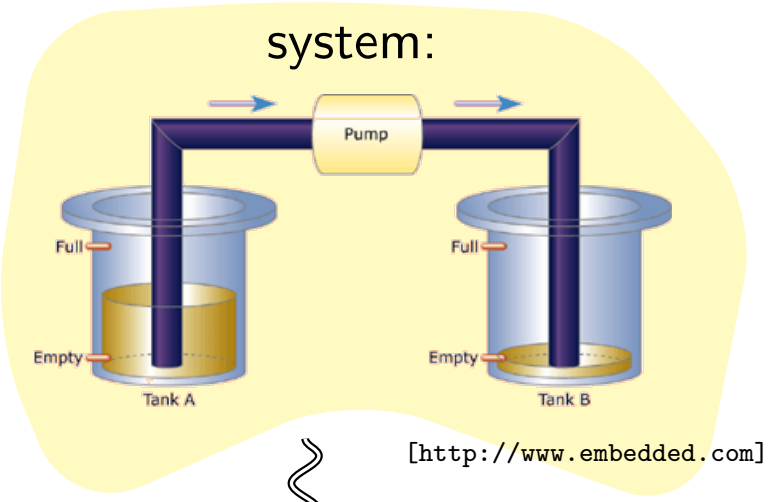


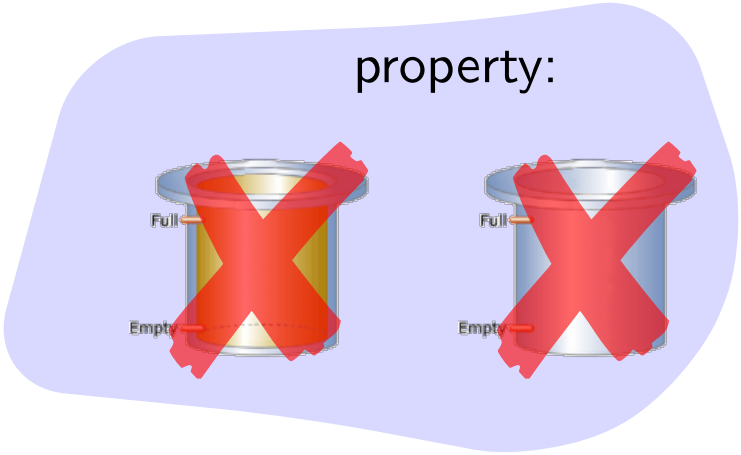
model-checking algorithm

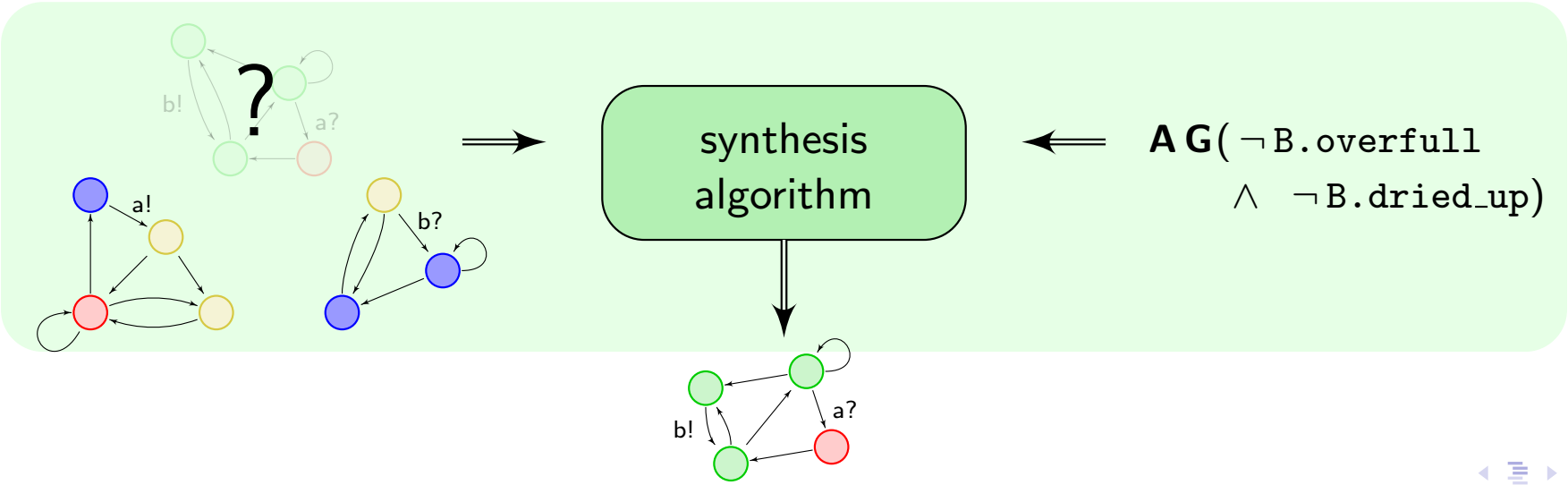
$\mathbf{A}\,\mathbf{G}(\neg\,\texttt{B.overfull} \wedge \neg\,\texttt{B.dried\_up})$

yes/no

Model checking and synthesis







Outline of the presentation

1. Basics about CTL
   - expressing properties of reactive systems
   - efficient verification algorithms
2. Quantified CTL
   - CTL with quantification over atomic propositions
   - model checking and satisfiability are mostly decidable
3. Temporal logics for games: ATL and extensions
   - expressing properties of complex interacting systems
   - QCTL-based decision procedures for ATL_{sc}

Computation-Tree Logic (CTL)

- **atomic propositions:**  ,  , ...

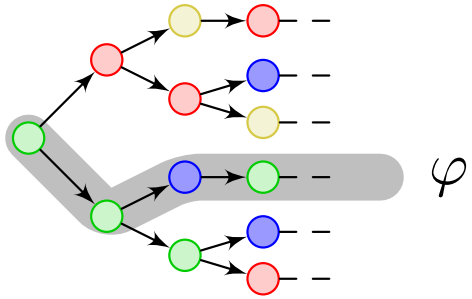
Computation-Tree Logic (CTL)

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

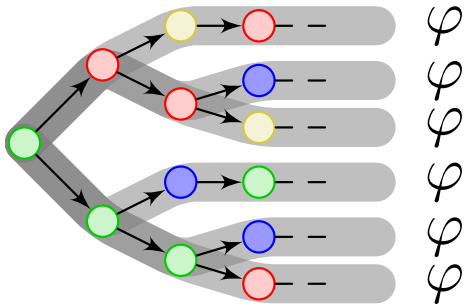
Computation-Tree Logic (CTL)

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

Eφ



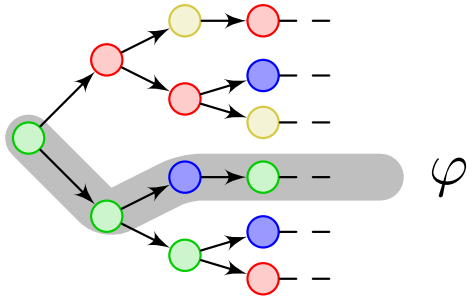
Aφ



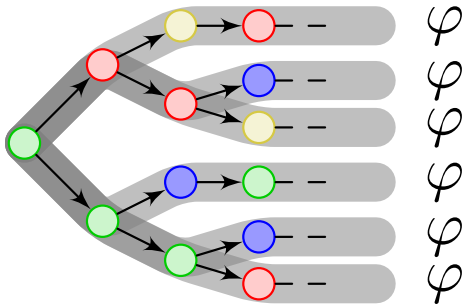
Computation-Tree Logic (CTL)

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

Eφ

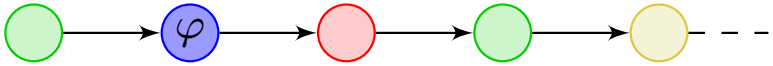


Aφ



- **temporal modalities:**

X φ



"next φ"

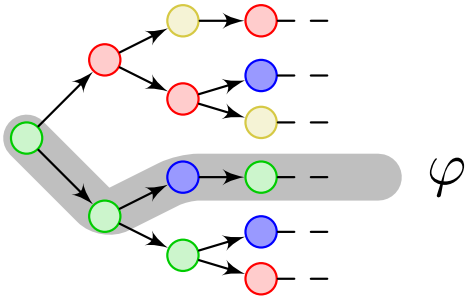
φ **U** ψ

"φ until ψ"

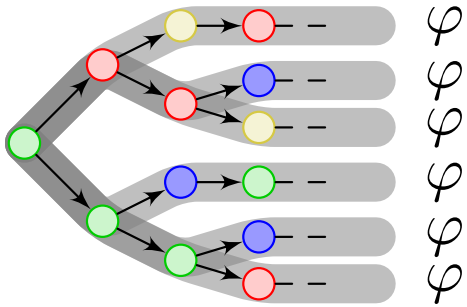
Computation-Tree Logic (CTL)

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

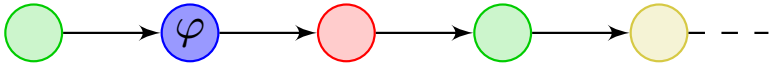
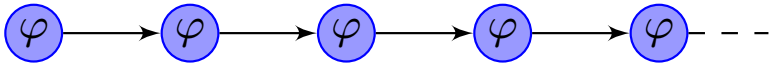
$\mathbf{E}\varphi$



$\mathbf{A}\varphi$



- **temporal modalities:**

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

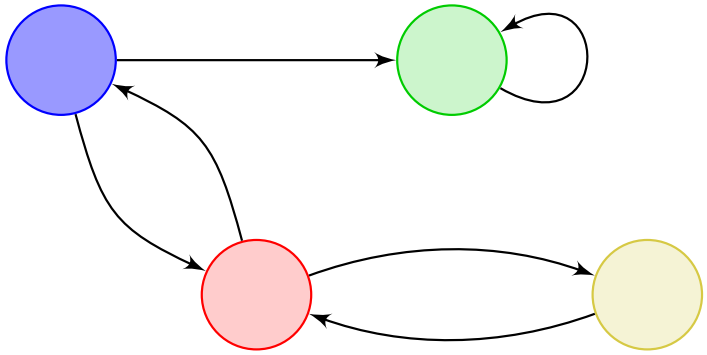
Examples of CTL formulas

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

Examples of CTL formulas

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

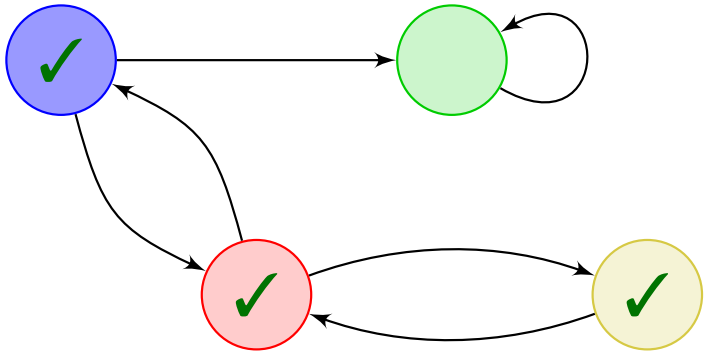
E F   is reachable



Examples of CTL formulas

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

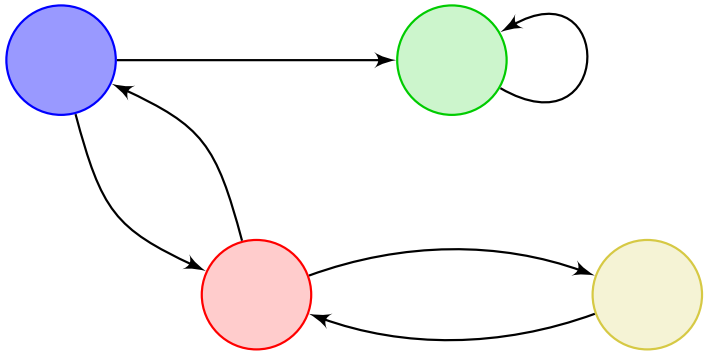
E F ● ● is reachable



Examples of CTL formulas

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

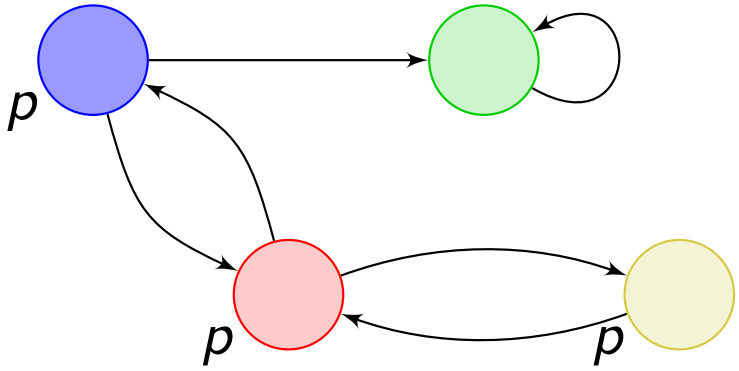
E G(**E F** ●) there is a path along which ● is always reachable



Examples of CTL formulas

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

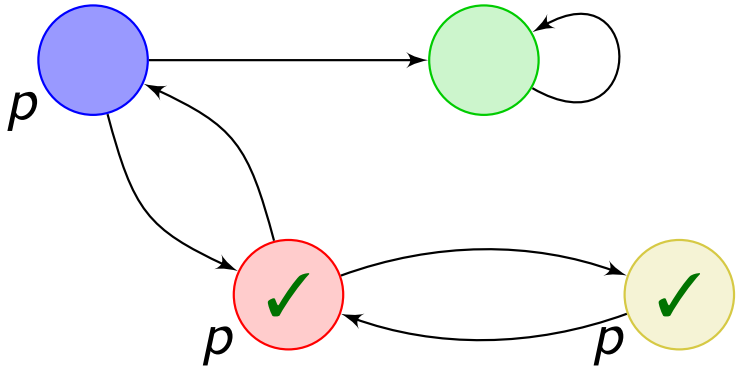
$\mathbf{E}\,\mathbf{G}(\underbrace{\mathbf{E}\,\mathbf{F}\,\bigcirc}_{p})$ there is a path along which $\bigcirc$ is always reachable



Examples of CTL formulas

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

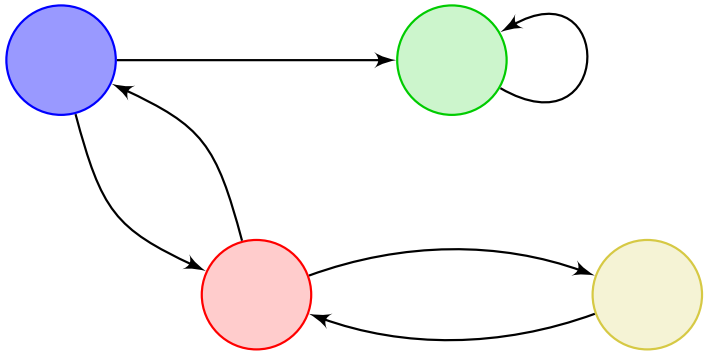
$\mathbf{E}\,\mathbf{G}(\underbrace{\mathbf{E}\,\mathbf{F}\,\bigcirc}_{p})$ there is a path along which $\bigcirc$ is always reachable



Examples of CTL formulas

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

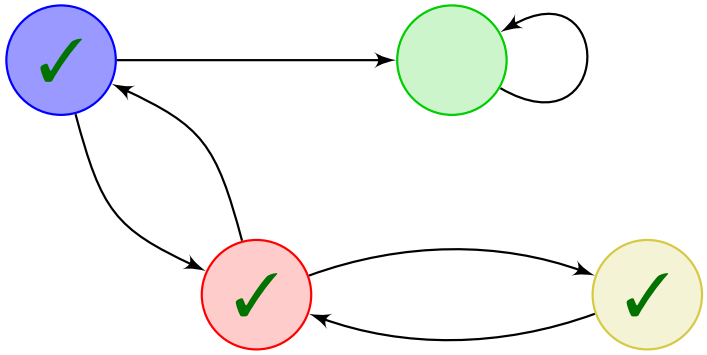
$\neg\, \mathbf{E}(\neg$ ● $)\ \mathbf{U}$ ● in order to reach ●, we have to visit ●



Examples of CTL formulas

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

$\neg\, \mathbf{E}(\neg$ ● $)\ \mathbf{U}$ ● in order to reach ●, we have to visit ●



Examples of 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.

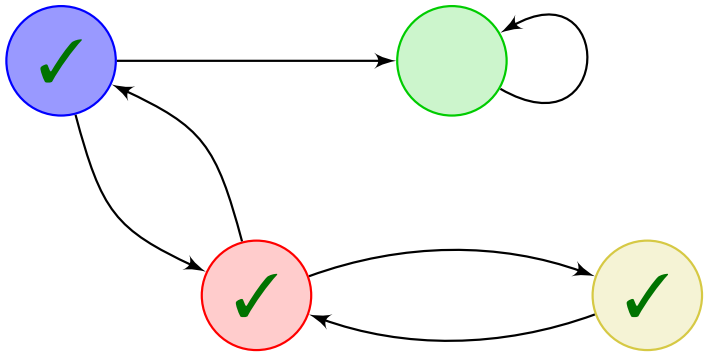
Examples of CTL formulas

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

Examples of CTL formulas

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

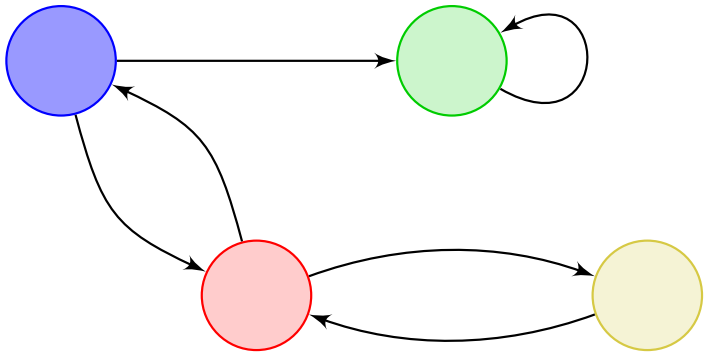
E G F ● there is a path visiting ● infinitely many times



Examples of CTL formulas

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

$\mathbf{A}(\mathbf{G}\,\mathbf{F}\,\bigcirc \Rightarrow \mathbf{G}\,\mathbf{F}\,\bigcirc)$ any path that visits $\bigcirc$ infinitely many times, also visits $\bigcirc$ infinitely many times



Examples of CTL formulas

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

Theorem ([EH86])

CTL* model checking is PSPACE*-complete.*

[EH86] Emerson, Halpern. "Sometimes" and "Not Never" Revisited: On Branching versus Linear Time Temporal Logic. J.ACM, 1986.

Outline of the presentation

1. Basics about CTL
   - expressing properties of reactive systems
   - efficient verification algorithms
2. Quantified CTL
   - CTL with quantification over atomic propositions
   - model checking and satisfiability are mostly decidable
3. Temporal logics for games: ATL and extensions
   - expressing properties of complex interacting systems
   - QCTL-based decision procedures for ATL_{sc}

Quantified CTL

[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.

[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.

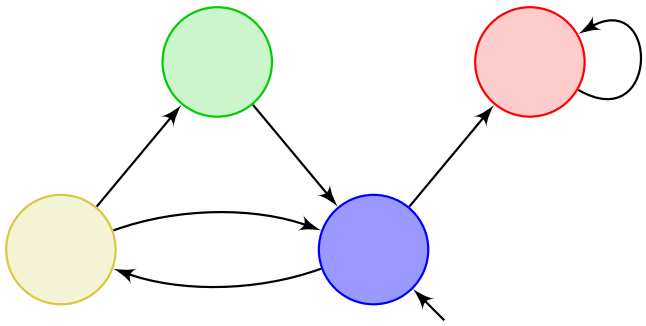
Quantified CTL

[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.

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



[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.

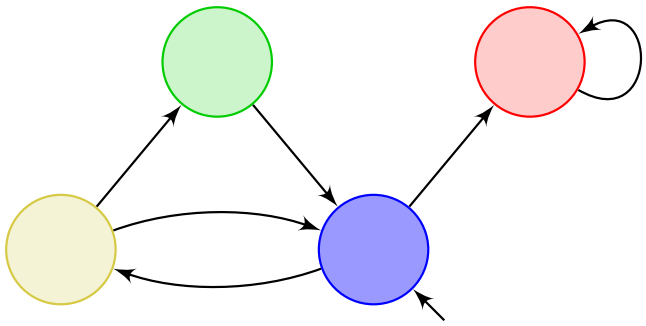
Quantified CTL

[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.

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



[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.

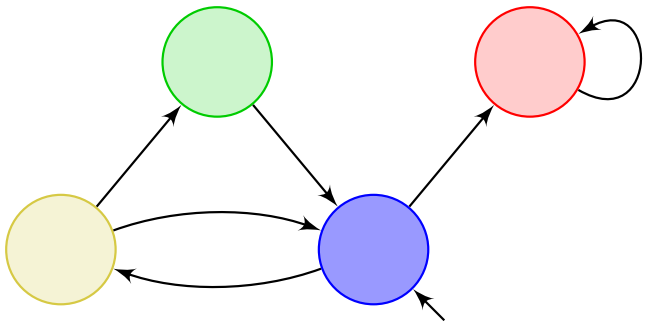
Quantified CTL

[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.

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



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

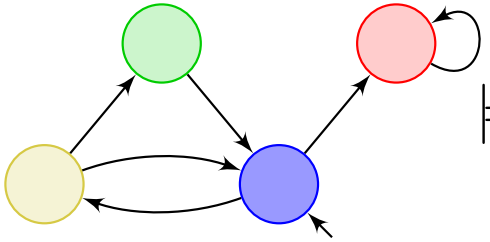
$\rightsquigarrow$ false if we label the computation tree;

[Kup95] Kupferman. Augmenting Branching Temporal Logics with Existential Quantification over Atomic Propositions. CAV, 1995.

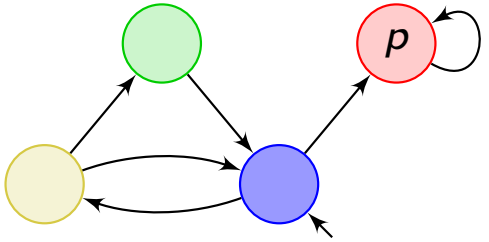
[Fre01] French. Decidability of Quantifed Propositional Branching Time Logics. AJCAI, 2001.

Semantics of QCTL

- structure semantics:



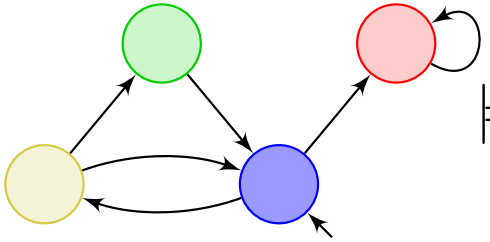
$\models_s \exists p.\varphi \quad \Leftrightarrow$



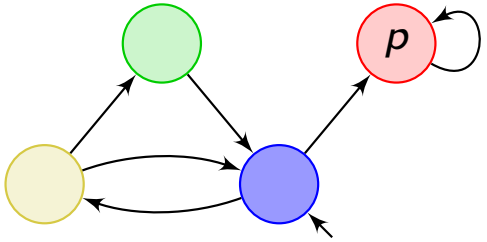
$\models \varphi$

Semantics of QCTL

- structure semantics:

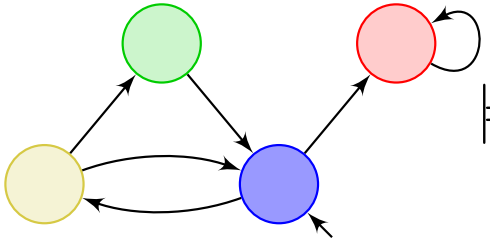


$\models_s \exists p.\varphi \quad \Leftrightarrow$

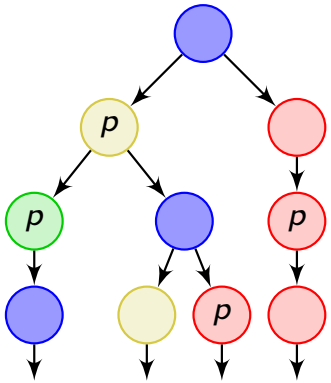


$\models \varphi$

- 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":

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

- QCTL can express (least or greatest) fixpoints:

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

Theorem

For both semantics, QCTL, QCTL* and MSO are equally expressive.

[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
- guess 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.

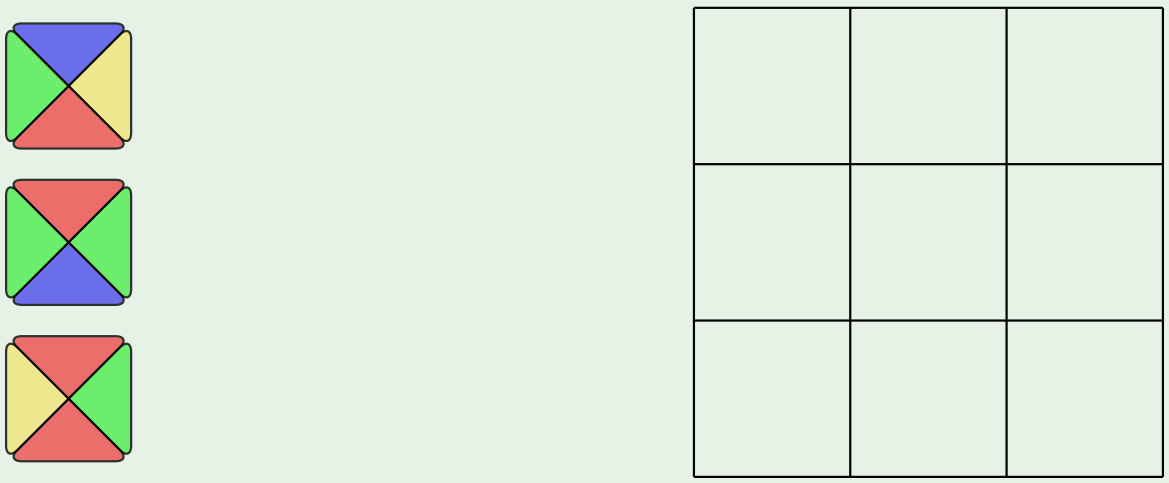
QCTL with structure semantics

Theorem

QCTL satisfiability for the structure semantics is undecidable.

Proof

Encode the problem of tiling finite square grids.



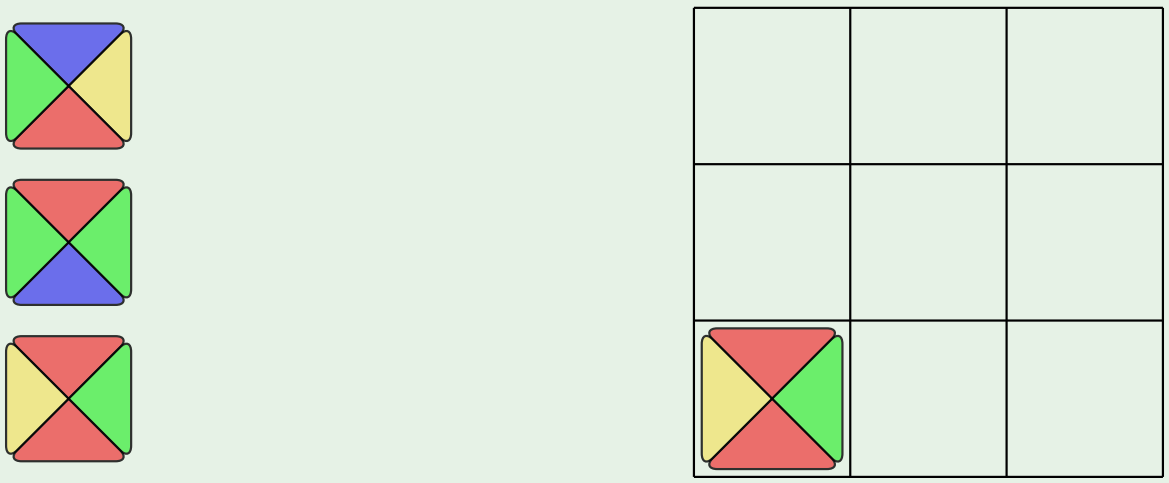
QCTL with structure semantics

Theorem

QCTL satisfiability for the structure semantics is undecidable.

Proof

Encode the problem of tiling finite square grids.



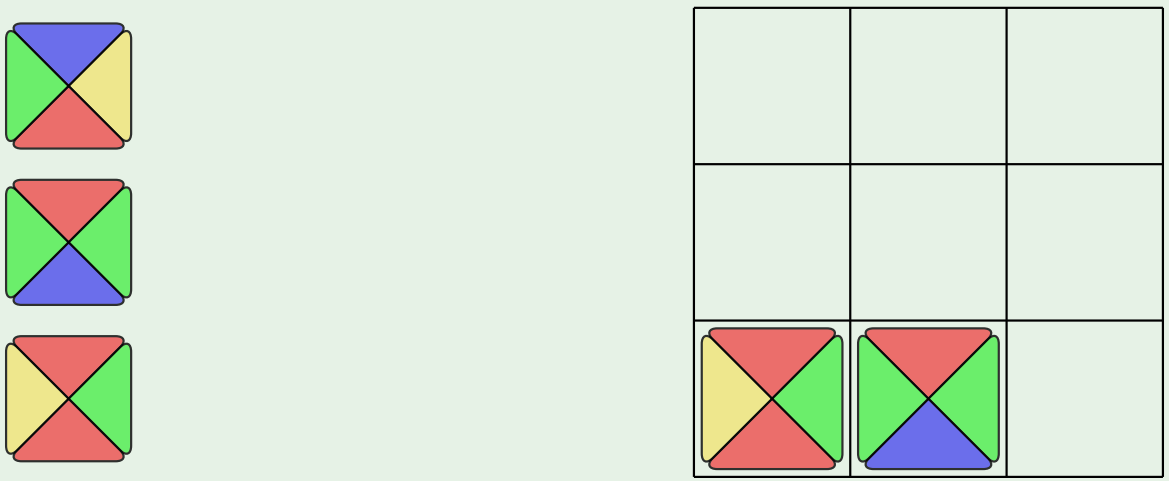
QCTL with structure semantics

Theorem

QCTL satisfiability for the structure semantics is undecidable.

Proof

Encode the problem of tiling finite square grids.



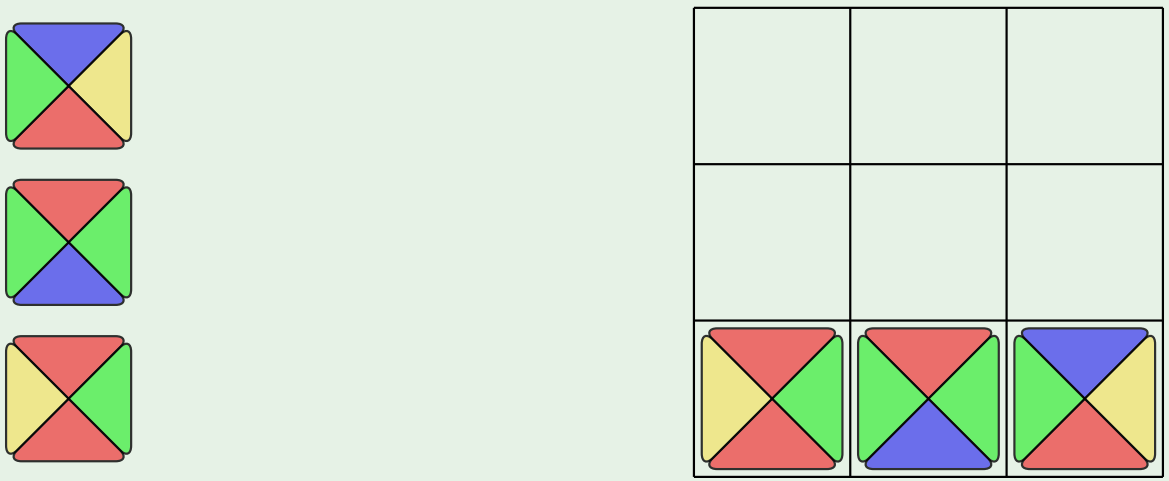
QCTL with structure semantics

Theorem

QCTL satisfiability for the structure semantics is undecidable.

Proof

Encode the problem of tiling finite square grids.



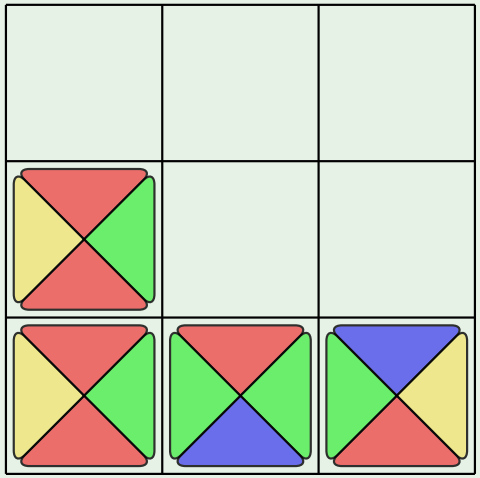
QCTL with structure semantics

Theorem

QCTL satisfiability for the structure semantics is undecidable.

Proof

Encode the problem of tiling finite square grids.



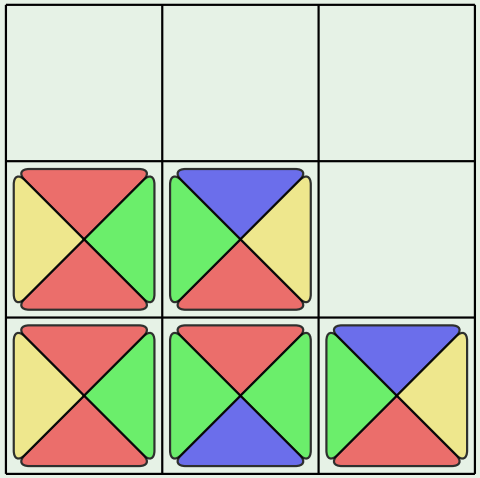
QCTL with structure semantics

Theorem

QCTL satisfiability for the structure semantics is undecidable.

Proof

Encode the problem of tiling finite square grids.



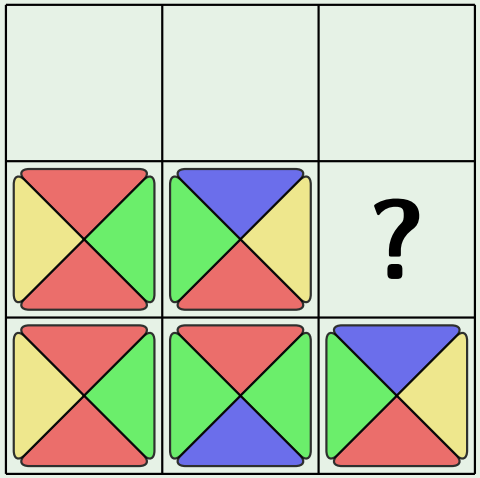
QCTL with structure semantics

Theorem

QCTL satisfiability for the structure semantics is undecidable.

Proof

Encode the problem of tiling finite square grids.



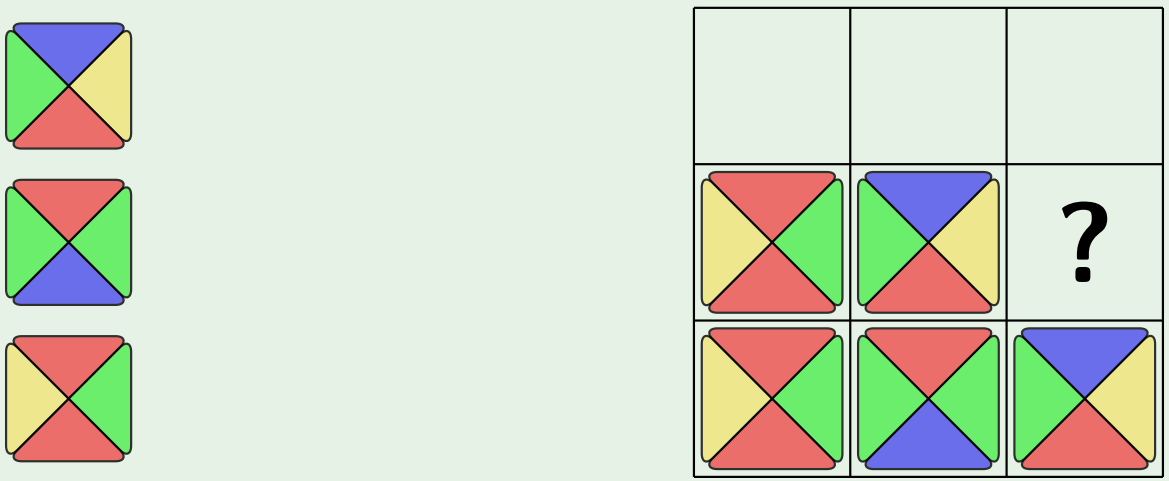
QCTL with structure semantics

Theorem

QCTL satisfiability for the structure semantics is undecidable.

Proof

Encode the problem of tiling finite square grids.



Given a set of tiles, whether any square grid can be tiled is undecidable.

QCTL with structure semantics

Theorem

QCTL satisfiability for the structure semantics is undecidable.

Proof

Reduction: is there a finite Kripke structure such that

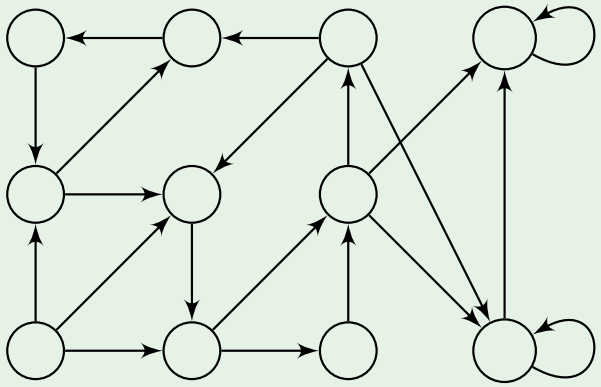
QCTL with structure semantics

Theorem

QCTL satisfiability for the structure semantics is undecidable.

Proof

Reduction: is there a finite Kripke structure such that



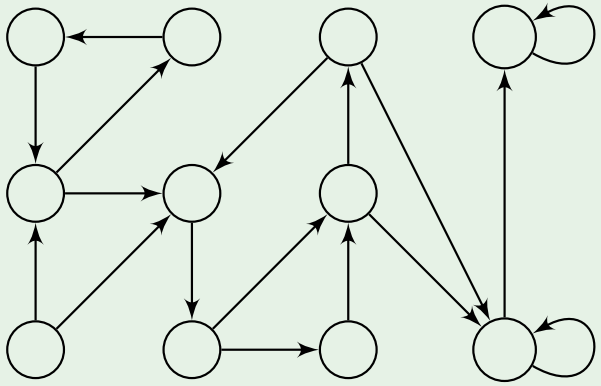
QCTL with structure semantics

Theorem

QCTL satisfiability for the structure semantics is undecidable.

Proof

Reduction: is there a finite Kripke structure such that



- each state has one or two successors

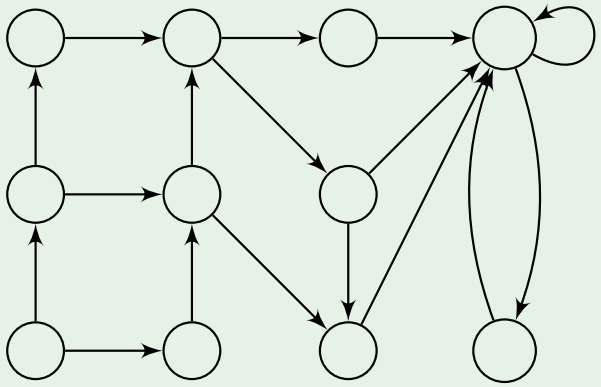
QCTL with structure semantics

Theorem

QCTL satisfiability for the structure semantics is undecidable.

Proof

Reduction: is there a finite Kripke structure such that



- two successors of the same state have a common successor

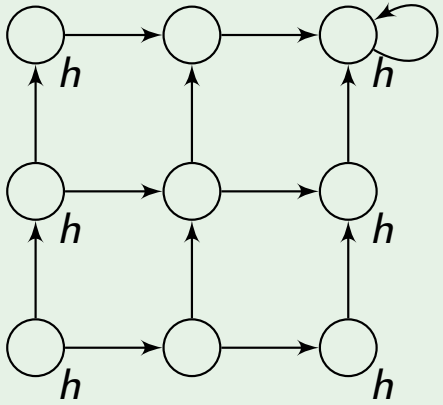
QCTL with structure semantics

Theorem

QCTL satisfiability for the structure semantics is undecidable.

Proof

Reduction: is there a finite Kripke structure such that



- [... many more conditions ...]

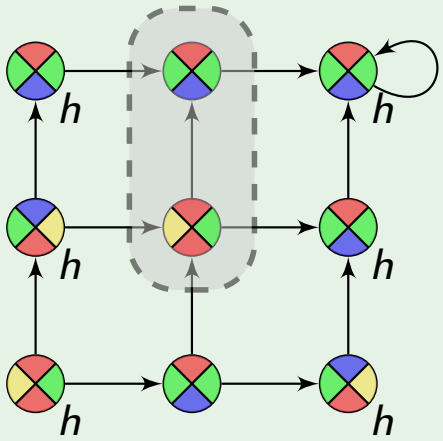
QCTL with structure semantics

Theorem

QCTL satisfiability for the structure semantics is undecidable.

Proof

Reduction: is there a finite Kripke structure such that



- for any tiling, there is a position where the neighbouring tiles do not match

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 tree automata:

- polynomial-size automata for CTL;
- boolean combinators can be handled easily;
- quantification is handled by projection, which requires alternation removal (exponential blowup).

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

Outline of the presentation

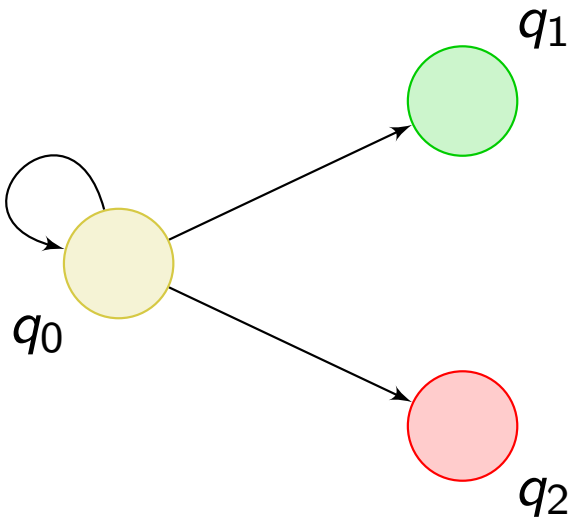
1. Basics about CTL
   - expressing properties of reactive systems
   - efficient verification algorithms
2. Quantified CTL
   - CTL with quantification over atomic propositions
   - model checking and satisfiability are mostly decidable
3. Temporal logics for games: ATL and extensions
   - expressing properties of complex interacting systems
   - QCTL-based decision procedures for ATL_{sc}

Reasoning about multi-agent systems

Concurrent games

A concurrent game is made of

- a transition system;

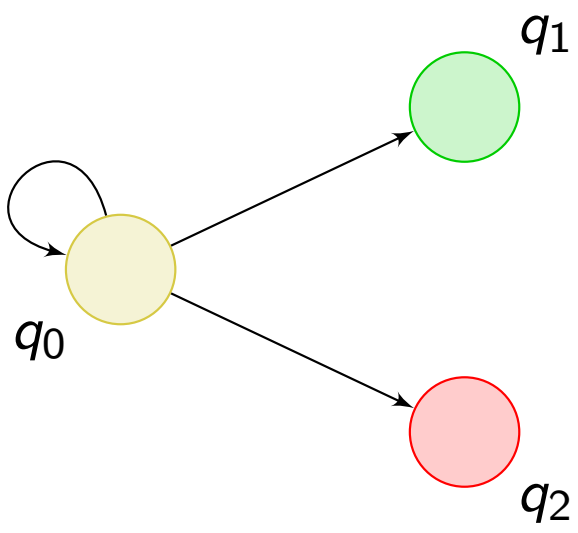


Reasoning about multi-agent systems

Concurrent games

A concurrent game is made of

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

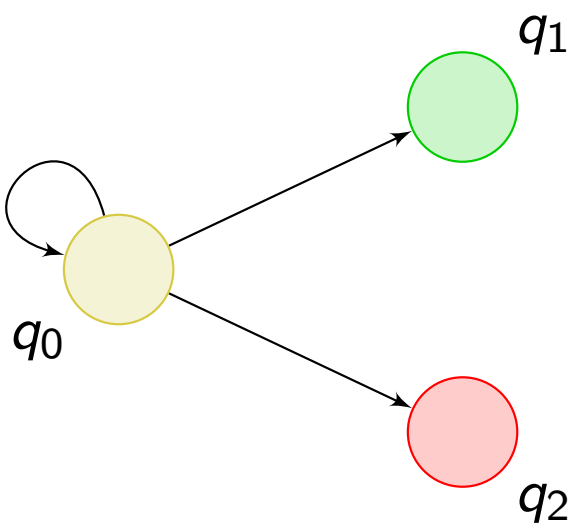


Reasoning about multi-agent systems

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 multi-agent systems

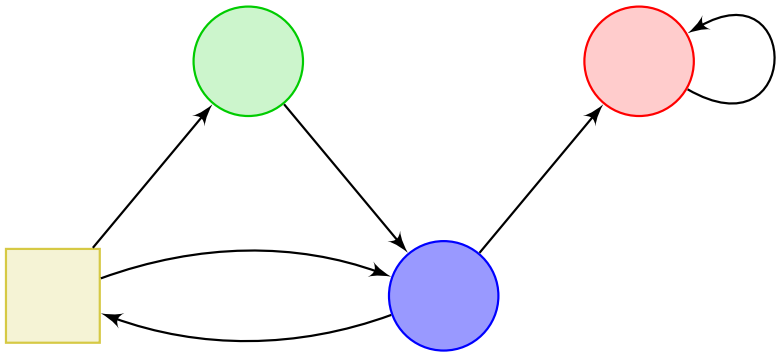
Concurrent games

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

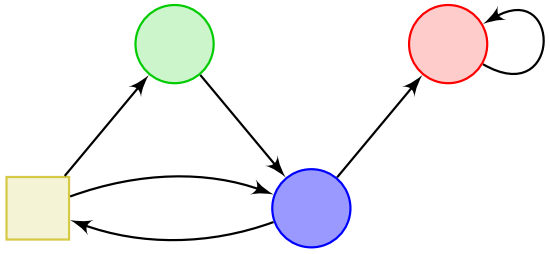
A strategy for a given player is a function telling what to play depending on what has happened previously.

Reasoning about open systems

Strategies

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

Strategy for player □:
alternately go to ● and ●.

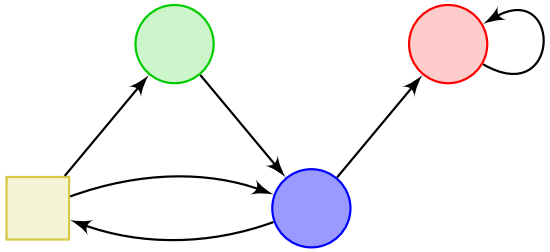


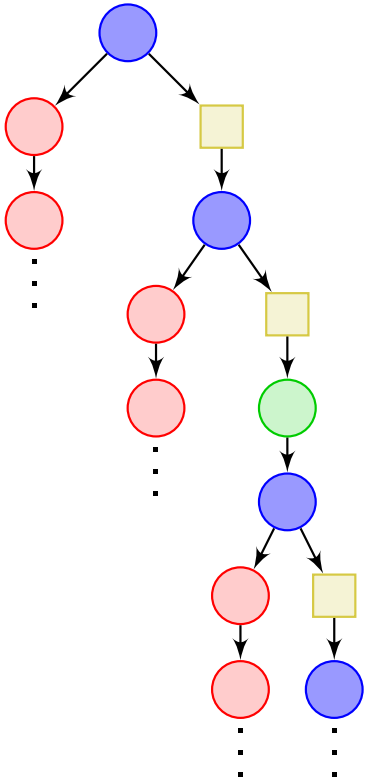
Reasoning about open systems

Strategies

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

Strategy for player □:
alternately go to ● and ●.





Temporal logics for games: ATL

ATL extends CTL with strategy quantifiers

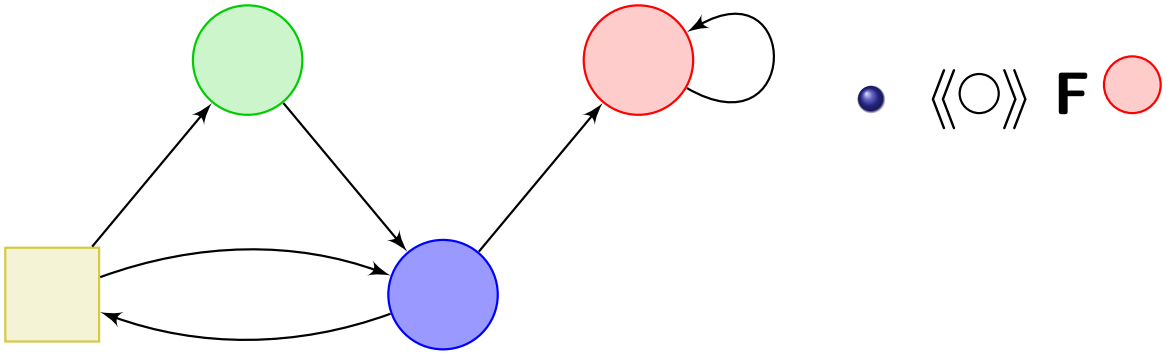
$\langle\langle A \rangle\rangle\, \varphi$ expresses that A has a strategy to enforce φ.

[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.

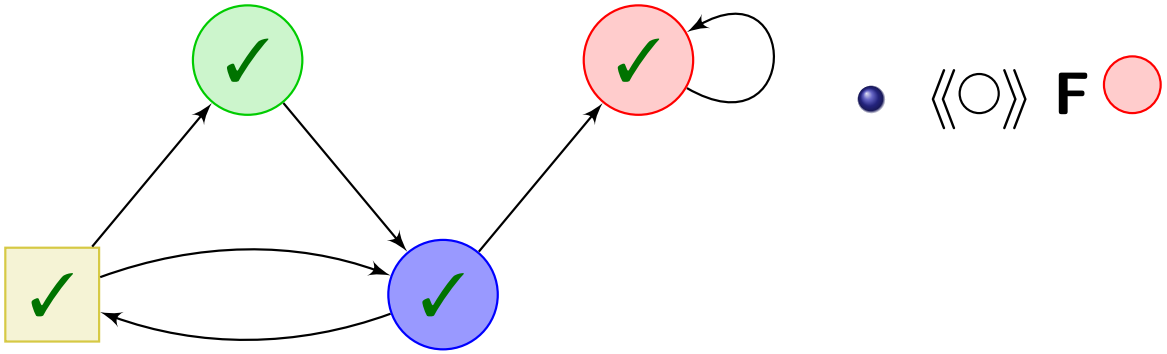


[AHK02] Alur, Henzinger, Kupferman. Alternating-time Temporal Logic. J. ACM, 2002.

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.

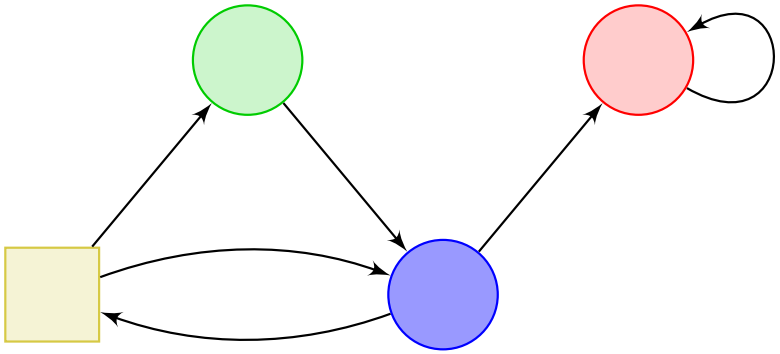


[AHK02] Alur, Henzinger, Kupferman. Alternating-time Temporal Logic. J. ACM, 2002.

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.



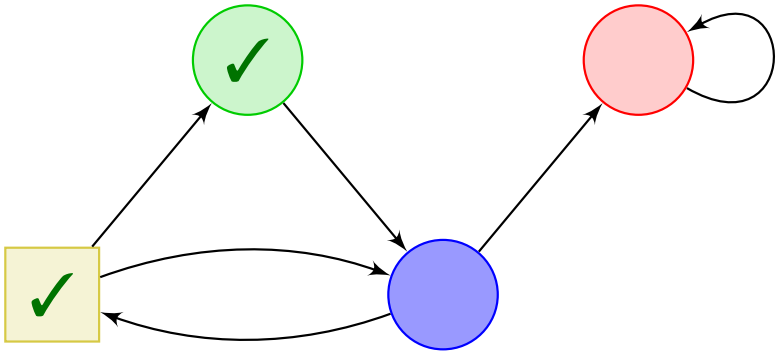
- $\langle\langle \bigcirc \rangle\rangle\, \mathbf{F}$
- $\langle\langle \Box \rangle\rangle\, \mathbf{F}$

[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.



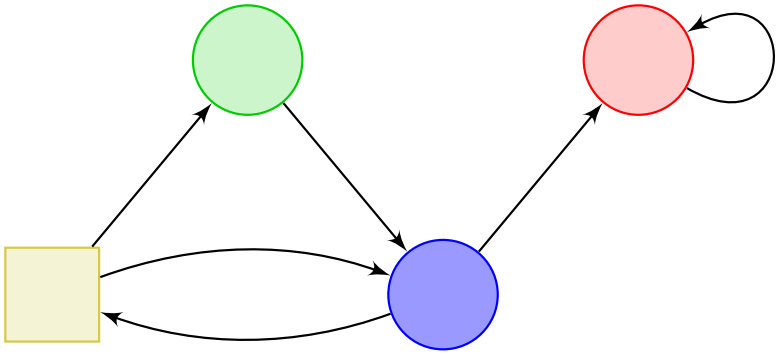
- $\langle\langle \bigcirc \rangle\rangle\, \mathbf{F}$
- $\langle\langle \Box \rangle\rangle\, \mathbf{F}$

[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.



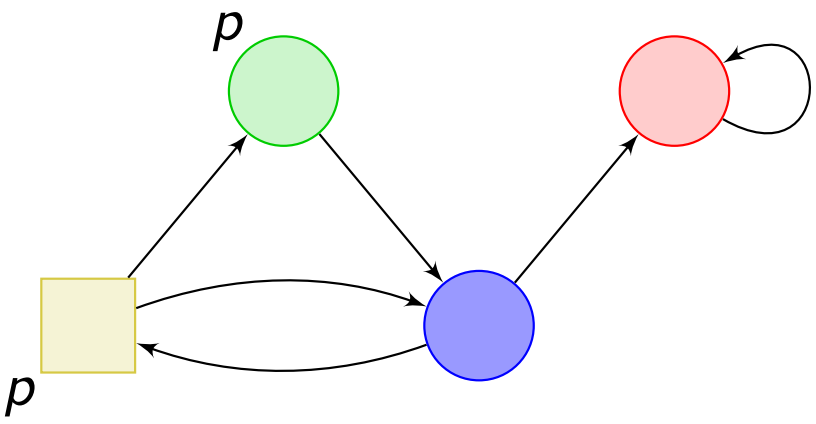
- $\langle\langle \bigcirc \rangle\rangle\ \mathbf{F}$
- $\langle\langle \Box \rangle\rangle\ \mathbf{F}$
- $\langle\langle \bigcirc \rangle\rangle\ \mathbf{G}(\ \langle\langle \Box \rangle\rangle\ \mathbf{F}\ )$

[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.



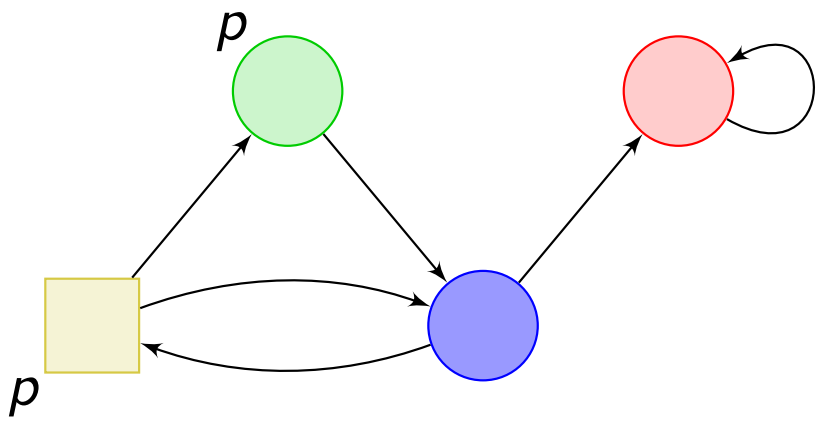
- $\langle\langle \bigcirc \rangle\rangle\, \mathbf{F}$
- $\langle\langle \Box \rangle\rangle\, \mathbf{F}$
- $\langle\langle \bigcirc \rangle\rangle\, \mathbf{G}(\underbrace{\langle\langle \Box \rangle\rangle\, \mathbf{F}}_{p}) \equiv \langle\langle \bigcirc \rangle\rangle\, \mathbf{G}\, p$

[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.



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

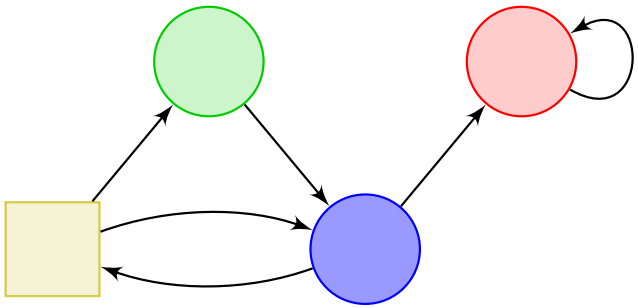
Theorem

Model checking ATL is PTIME-complete.

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

ATL with strategy contexts

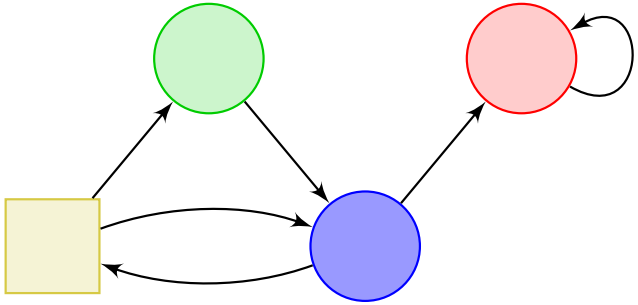
[BDLM09]



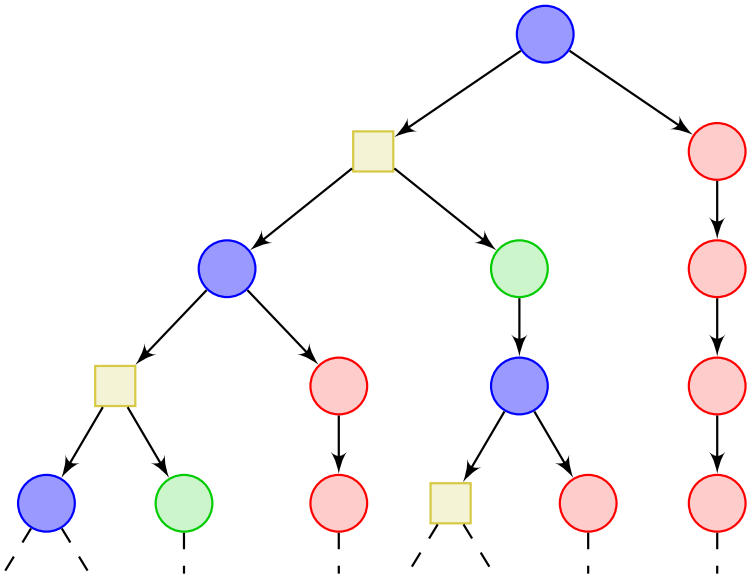
$\langle\langle \bigcirc \rangle\rangle\ \mathbf{G}(\ \langle\langle \Box \rangle\rangle\ \mathbf{F}$ ●$)$

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

ATL with strategy contexts [BDLM09]



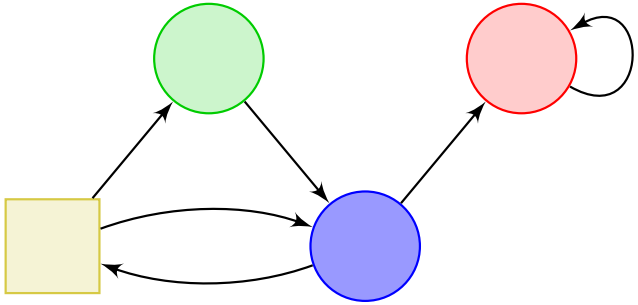
$\langle\langle\bigcirc\rangle\rangle\ \mathbf{G}(\ \langle\langle\Box\rangle\rangle\ \mathbf{F}\ \bigcirc)$



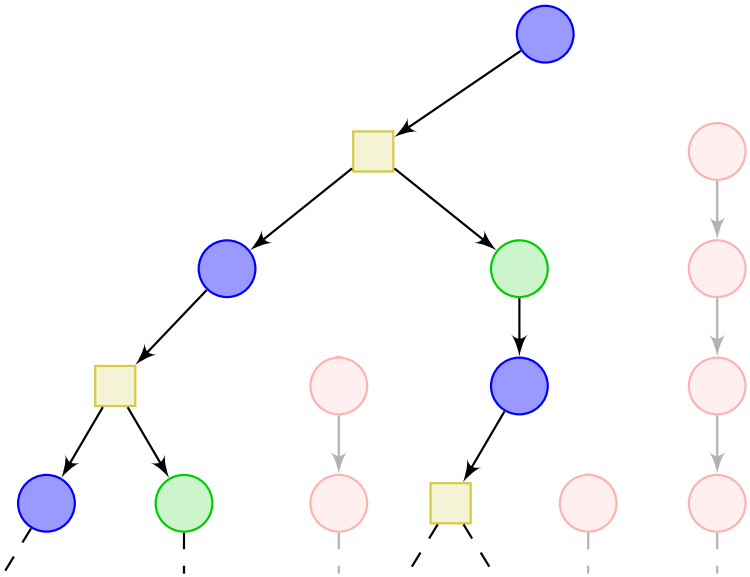
- consider the following strategy of Player ●: "always go to □";

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

ATL with strategy contexts [BDLM09]



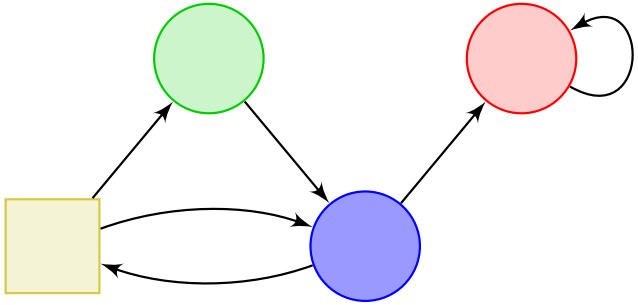
$\langle\langle\bigcirc\rangle\rangle\ \mathbf{G}(\ \langle\langle\Box\rangle\rangle\ \mathbf{F}\ \bigcirc)$



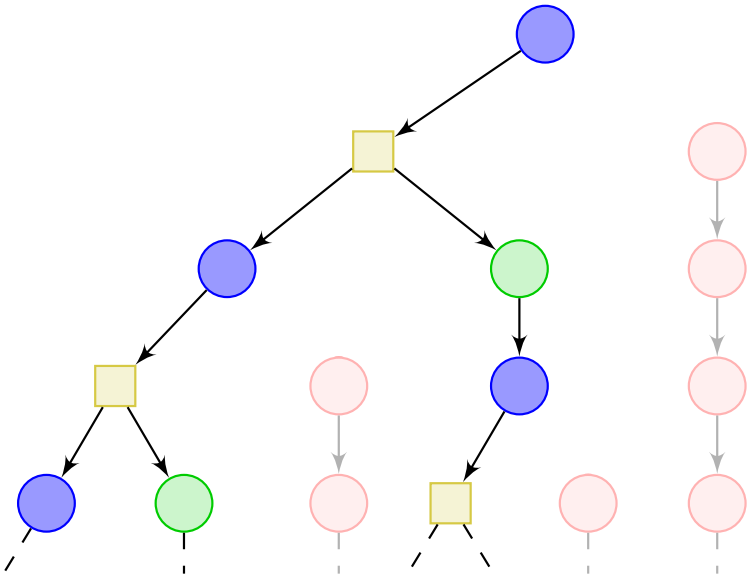
- consider the following strategy of Player ●: "always go to □";

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

ATL with strategy contexts [BDLM09]



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



- 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.

What ATL_{sc} can express

- Client-server interactions for accessing a shared resource:

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

What ATL_{sc} can express

- Client-server interactions for accessing a shared resource:

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

- Existence of Nash equilibria:

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

What ATL_{sc} can express

- Client-server interactions for accessing a shared resource:

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

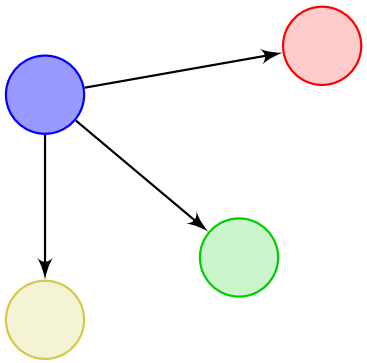
- Existence of Nash equilibria:

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

- Existence of dominating strategy:

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

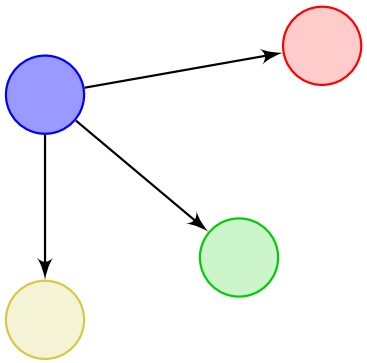
Translating ATL_{sc} into QCTL



- 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:

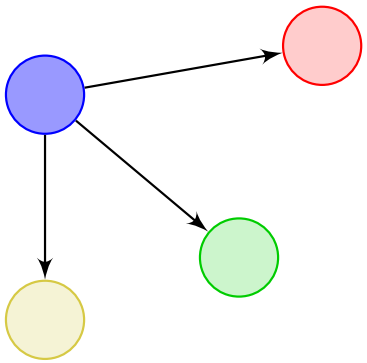
$\exists m_1^A.\ \exists m_2^A \ldots \exists m_n^A.$

- this corresponds to a strategy: $\mathbf{A}\,\mathbf{G}(m_i^A \Leftrightarrow \bigwedge \neg\, m_j^A)$;
- 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.

Corollary

ATL_{sc}^0 (memoryless quantification) model checking is decidable.

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

What about satisfiability?

Theorem

QCTL satisfiability is decidable.

What about satisfiability?

Theorem

QCTL satisfiability is decidable.

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.

But

Theorem (TW12)

ATL_{sc} satisfiability is undecidable.

Why?

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

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

Satisfiability for turn-based games

Theorem (LM13b)

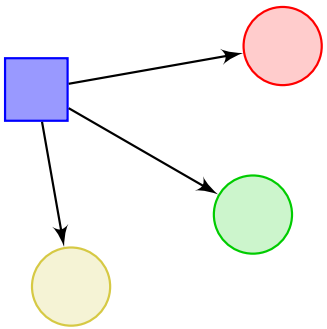
When restricted to turn-based games, ATL_{sc} satisfiability is decidable.

[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 □ has moves ●, ● and ●.
- 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:

$\exists \mathrm{mov}_A.$

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

[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.

Theorem

Model checking ATL_{sc} with only memoryless quantification is PSPACE-*complete.*

[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

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

Strategy logic can also be translated into QCTL.

Theorem

- *Strategy-logic satisfiability is decidable when restricted to turn-based games.*
- *Memoryless strategy-logic satisfiability is undecidable.*

[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;

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;

Future directions

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