Temporal logics for multi-agent systems

Nicolas Markey
LSV, CNRS & ENS Cachan, France

Journées Nationales
Lyon, 21-22 January 2013

Verification of computerised systems

- Computers are everywhere

Verification of computerised systems

- Computers are everywhere

- Bugs are everywhere...

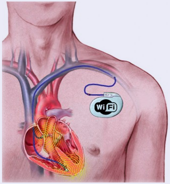

News

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

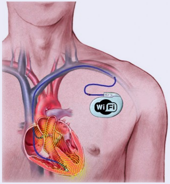
Toyota to recall Prius hybrids over ABS software

See video, below

By Martyn Williams

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

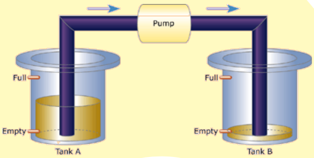
IDG News Service - Toyota plans to recall around 400,000 of its Prius hybrid cars to replace software that controls the antilock braking system (ABS), the auto maker said Tuesday.



- Verification should be everywhere!

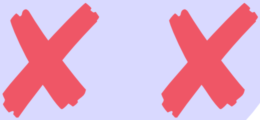
Model checking and synthesis

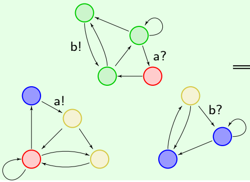
system:



[http://www.embedded.com]

property:



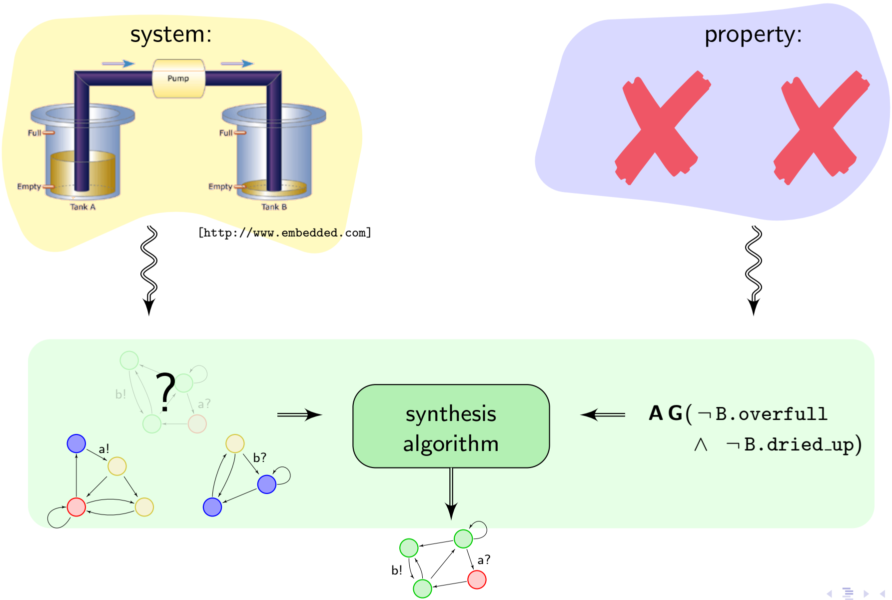


model-checking algorithm

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

yes/no

Model checking and synthesis



Outline of the presentation

1. Introduction
   $\leadsto$ formal verification model checking and synthesis

2. Classical temporal logics: CTL and LTL
   $\leadsto$ expressing properties of “closed” systems

3. Temporal logics for games: ATL
   $\leadsto$ expressing properties of interacting systems
   extensions to non-zero-sum games

Outline of the presentation

1. Introduction
   $\rightsquigarrow$ formal verification model checking and synthesis

2. Classical temporal logics: CTL and LTL
   $\rightsquigarrow$ expressing properties of “closed” systems

3. Temporal logics for games: ATL
   $\rightsquigarrow$ expressing properties of interacting systems
   extensions to non-zero-sum games

Computation-Tree Logic (CTL*)

- **atomic propositions:** , , …

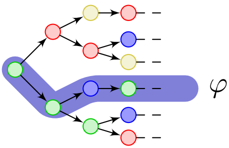
Computation-Tree Logic (CTL*)

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

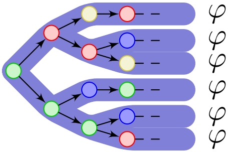
Computation-Tree Logic (CTL*)

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

$\mathbf{E}\varphi$

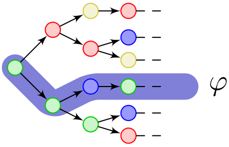


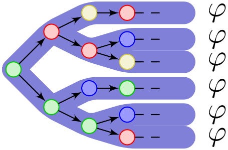
$\mathbf{A}\varphi$



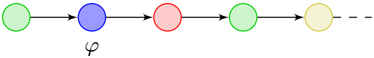
Computation-Tree Logic (CTL*)

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

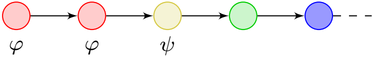
$\mathbf{E}\varphi$ 

$\mathbf{A}\varphi$ 

- **temporal modalities:**

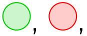
$\mathbf{X}\,\varphi$ 

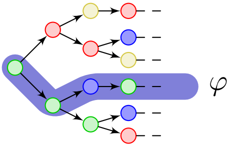
"next φ"

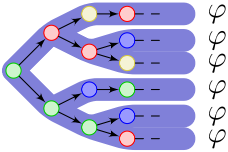
$\varphi\ \mathbf{U}\ \psi$ 

"φ until ψ"

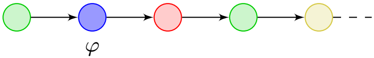
Computation-Tree Logic (CTL*)

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

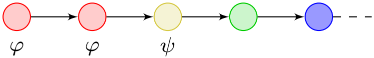
Eφ 

Aφ 

- **temporal modalities:**

X φ 

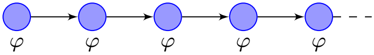
"next φ"

φ **U** ψ 

"φ until ψ"

`true` **U** $\varphi \equiv$ **F** φ 

"eventually φ"

$\neg$ **F** $\neg\varphi \equiv$ **G** φ 

"always φ"

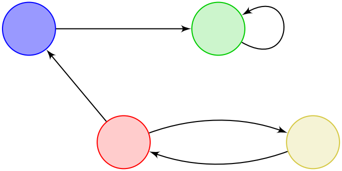
Fragments of CTL*

- CTL: each temporal modality is in the immediate scope of a path quantifier.

Fragments of CTL*

- CTL: each temporal modality is in the immediate scope of a path quantifier.

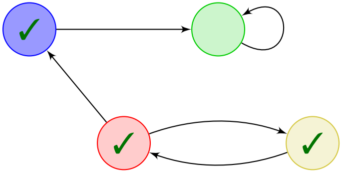
$\mathbf{E\,F}$ is reachable



Fragments of CTL*

- CTL: each temporal modality is in the immediate scope of a path quantifier.

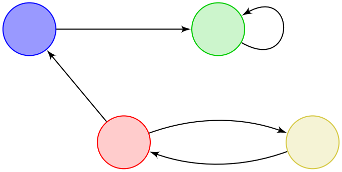
E F is reachable



Fragments of CTL*

- CTL: each temporal modality is in the immediate scope of a path quantifier.

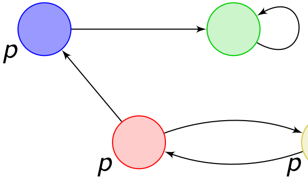
$\mathbf{E}\,\mathbf{G}(\mathbf{E}\,\mathbf{F}\,\bigcirc)$ there is a path along which $\bigcirc$ is always reachable



Fragments of CTL*

- CTL: each temporal modality is in the immediate scope of a path quantifier.

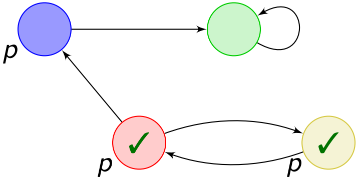
$\mathbf{E\,G}(\underbrace{\mathbf{E\,F}\,\bigcirc}_{p})$ there is a path along which $\bigcirc$ is always reachable



Fragments of CTL*

- CTL: each temporal modality is in the immediate scope of a path quantifier.

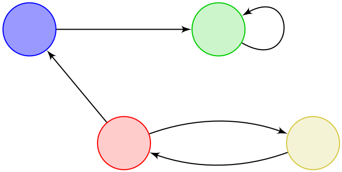
$\mathbf{E}\,\mathbf{G}(\underbrace{\mathbf{E}\,\mathbf{F}\,\bigcirc}_{p})$ there is a path along which $\bigcirc$ is always reachable



Fragments of CTL*

- CTL: each temporal modality is in the immediate scope of a path quantifier.

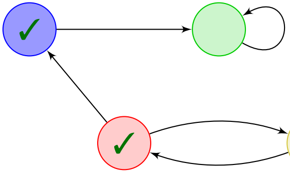
$\neg\,\mathbf{E}(\neg \bigcirc)\ \mathbf{U}\ \bigcirc$ in order to reach $\bigcirc$, we have to visit $\bigcirc$



Fragments of CTL*

- CTL: each temporal modality is in the immediate scope of a path quantifier.

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



Fragments of CTL*

- CTL: each temporal modality is in the immediate scope of a path quantifier.

Theorem

CTL model checking is PTIME*-complete.*
CTL symbolic model checking is PSPACE*-complete.*

Fragments of CTL*

- CTL: each temporal modality is in the immediate scope of a path quantifier.

Theorem

CTL model checking is PTIME*-complete.*
CTL symbolic model checking is PSPACE*-complete.*

- LTL: $\mathbf{E}\varphi$ or $\mathbf{A}\varphi$, where φ has no path quantifier.

Fragments of CTL*

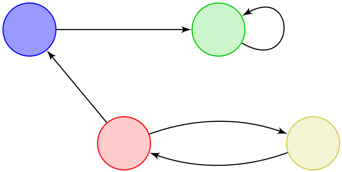
- CTL: each temporal modality is in the immediate scope of a path quantifier.

Theorem

CTL model checking is PTIME*-complete.*
CTL symbolic model checking is PSPACE*-complete.*

- LTL: $\mathbf{E}\varphi$ or $\mathbf{A}\varphi$, where φ has no path quantifier.

$\mathbf{E}(\mathbf{G}\,\mathbf{F}\ )$ there is a path visiting infinitely many times



Fragments of CTL*

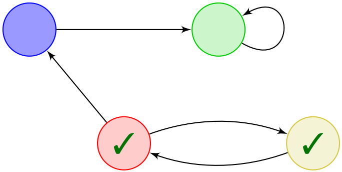
- CTL: each temporal modality is in the immediate scope of a path quantifier.

Theorem

CTL model checking is PTIME*-complete.*
CTL symbolic model checking is PSPACE*-complete.*

- LTL: $\mathbf{E}\varphi$ or $\mathbf{A}\varphi$, where φ has no path quantifier.

$\mathbf{E}(\mathbf{G}\,\mathbf{F}\ \bigcirc)$ there is a path visiting $\bigcirc$ infinitely many times



Fragments of CTL*

- CTL: each temporal modality is in the immediate scope of a path quantifier.

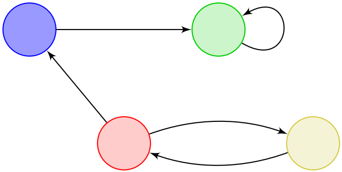
Theorem

CTL model checking is PTIME*-complete.*
CTL symbolic model checking is PSPACE*-complete.*

- LTL: $\mathbf{E}\varphi$ or $\mathbf{A}\varphi$, where φ has no path quantifier.

$$\mathbf{A}[(\mathbf{F}\ \bigcirc) \Rightarrow (\mathbf{F}\ \mathbf{G}\ \neg \bigcirc)]$$

any path that visits $\bigcirc$ visits $\bigcirc$ finitely many times



Fragments of CTL*

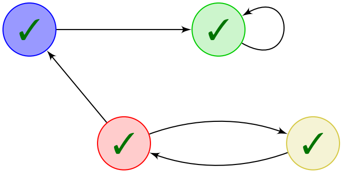
- CTL: each temporal modality is in the immediate scope of a path quantifier.

Theorem

CTL model checking is PTIME*-complete.*
CTL symbolic model checking is PSPACE*-complete.*

- LTL: $\mathbf{E}\varphi$ or $\mathbf{A}\varphi$, where φ has no path quantifier.

$\mathbf{A}[(\mathbf{F}\bigcirc) \Rightarrow (\mathbf{F}\,\mathbf{G}\,\neg\bigcirc)]$ any path that visits $\bigcirc$ visits $\bigcirc$ finitely many times



Fragments of CTL*

- CTL: each temporal modality is in the immediate scope of a path quantifier.

Theorem

CTL model checking is PTIME*-complete.*
CTL symbolic model checking is PSPACE*-complete.*

- LTL: $\mathbf{E}\varphi$ or $\mathbf{A}\varphi$, where φ has no path quantifier.

Theorem

LTL (symbolic) model checking is PSPACE*-complete.*

Fragments of CTL*

- CTL: each temporal modality is in the immediate scope of a path quantifier.

Theorem

CTL model checking is PTIME*-complete.*
CTL symbolic model checking is PSPACE*-complete.*

- LTL: $\mathbf{E}\varphi$ or $\mathbf{A}\varphi$, where φ has no path quantifier.

Theorem

LTL (symbolic) model checking is PSPACE*-complete.*

Theorem

CTL (symbolic) model checking is* PSPACE*-complete.*

Outline of the presentation

1. Introduction
   $\rightsquigarrow$ formal verification model checking and synthesis

2. Classical temporal logics: CTL and LTL
   $\rightsquigarrow$ expressing properties of "closed" systems

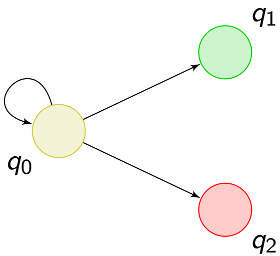
3. Temporal logics for games: ATL
   $\rightsquigarrow$ expressing properties of interacting systems
   extensions to non-zero-sum games

Reasoning about multi-agent systems

Concurrent games

A concurrent game is made of

- a transition system;

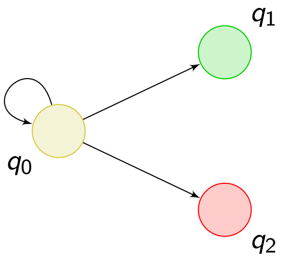


Reasoning about multi-agent systems

Concurrent games

A concurrent game is made of

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

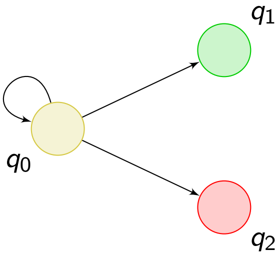


Reasoning about multi-agent systems

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

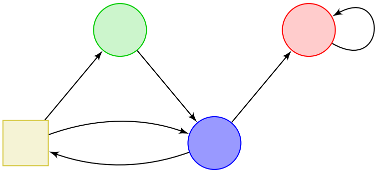
Concurrent games

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

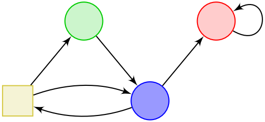
A strategy for a given player is a function telling what to play depending on what has happened previously.

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 .

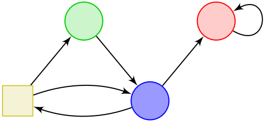


Reasoning about open systems

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 .

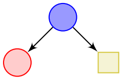


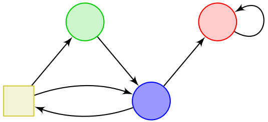
Reasoning about open systems

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 .



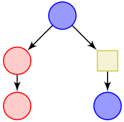


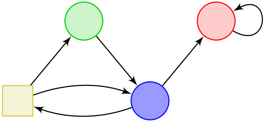
Reasoning about open systems

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 .



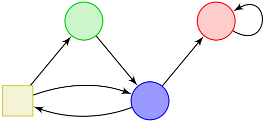


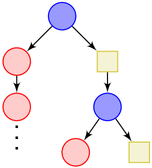
Reasoning about open systems

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 .



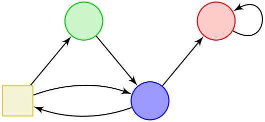


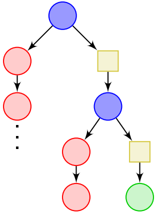
Reasoning about open systems

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 .



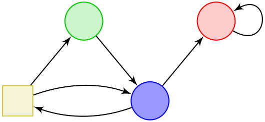


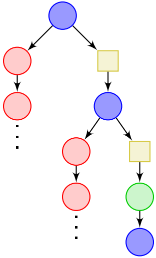
Reasoning about open systems

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 .



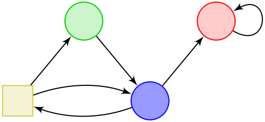


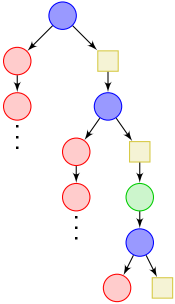
Reasoning about open systems

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 .



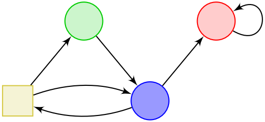


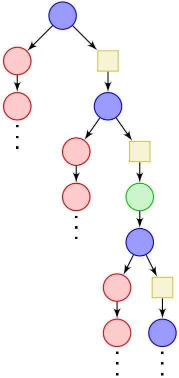
Reasoning about open systems

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 .





Alternating-time Temporal Logic

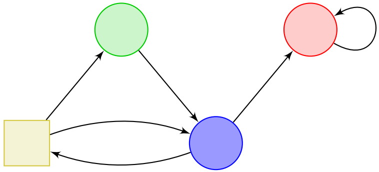
ATL formulas are built inductively using atomic propositions, Boolean combinations, temporal modalities **X** and **U**, and strategy quantifiers:

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

Alternating-time Temporal Logic

ATL formulas are built inductively using atomic propositions, Boolean combinations, temporal modalities **X** and **U**, and strategy quantifiers:

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

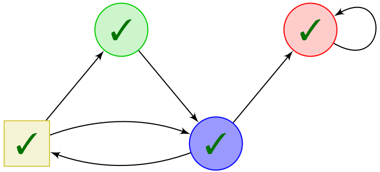


- $\langle\!\langle \bigcirc \rangle\!\rangle\ \mathbf{F}\ \bigcirc$

Alternating-time Temporal Logic

ATL formulas are built inductively using atomic propositions, Boolean combinations, temporal modalities **X** and **U**, and strategy quantifiers:

$$\langle\langle A \rangle\rangle\, \varphi \text{ expresses that } A \text{ has a strategy to enforce } \varphi.$$

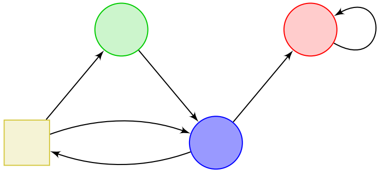


- $\langle\langle \bigcirc \rangle\rangle\ \mathbf{F}\ \bigcirc$

Alternating-time Temporal Logic

ATL formulas are built inductively using atomic propositions, Boolean combinations, temporal modalities **X** and **U**, and strategy quantifiers:

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

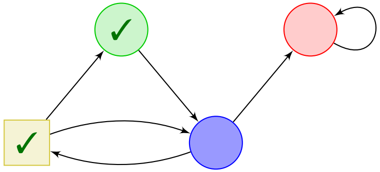


- $\langle\!\langle \bigcirc \rangle\!\rangle$ **F** $\bigcirc$
- $\langle\!\langle \square \rangle\!\rangle$ **F** $\bigcirc$

Alternating-time Temporal Logic

ATL formulas are built inductively using atomic propositions, Boolean combinations, temporal modalities **X** and **U**, and strategy quantifiers:

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

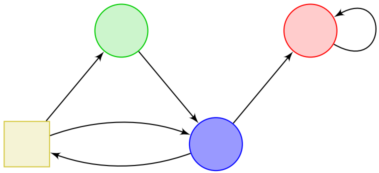


- $\langle\!\langle \bigcirc \rangle\!\rangle$ **F** 
- $\langle\!\langle \square \rangle\!\rangle$ **F** 

Alternating-time Temporal Logic

ATL formulas are built inductively using atomic propositions, Boolean combinations, temporal modalities **X** and **U**, and strategy quantifiers:

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

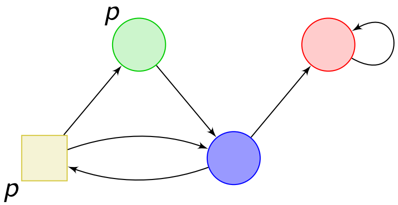


- $\langle\!\langle \bigcirc \rangle\!\rangle\ \mathbf{F}\ \bigcirc$
- $\langle\!\langle \square \rangle\!\rangle\ \mathbf{F}\ \bigcirc$
- $\langle\!\langle \bigcirc \rangle\!\rangle\ \mathbf{G} (\ \langle\!\langle \square \rangle\!\rangle\ \mathbf{F}\ \bigcirc)$

Alternating-time Temporal Logic

ATL formulas are built inductively using atomic propositions, Boolean combinations, temporal modalities **X** and **U**, and strategy quantifiers:

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

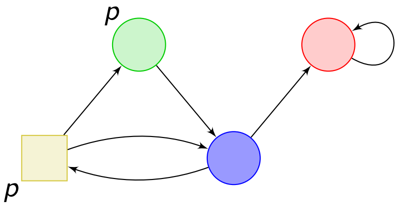


- $\langle\!\langle \bigcirc \rangle\!\rangle\ \mathbf{F}\ \bigcirc$
- $\langle\!\langle \square \rangle\!\rangle\ \mathbf{F}\ \bigcirc$
- $\langle\!\langle \bigcirc \rangle\!\rangle\ \mathbf{G}(\underbrace{\langle\!\langle \square \rangle\!\rangle\ \mathbf{F}\ \bigcirc}_{p}) \equiv \langle\!\langle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

Alternating-time Temporal Logic

ATL formulas are built inductively using atomic propositions, Boolean combinations, temporal modalities $\mathbf{X}$ and $\mathbf{U}$, and strategy quantifiers:

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



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

Alternating-time Temporal Logic

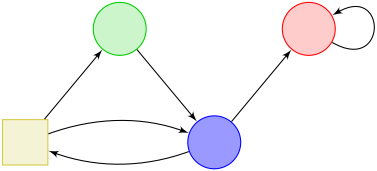
ATL formulas are built inductively using atomic propositions, Boolean combinations, temporal modalities **X** and **U**, and strategy quantifiers:

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

Theorem

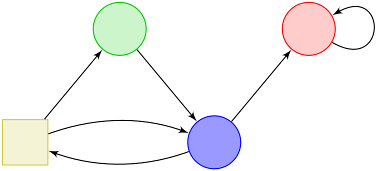
ATL model checking is PTIME*-complete.*
ATL symbolic model checking is EXPTIME*-complete.*

Another semantics: ATL with strategy contexts

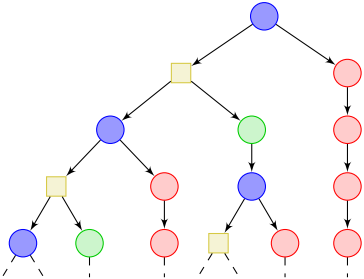


$$\langle\langle \bigcirc \rangle\rangle\ \mathbf{G}(\ \langle\langle \square \rangle\rangle\ \mathbf{F}\ \bigcirc)$$

Another semantics: ATL with strategy contexts

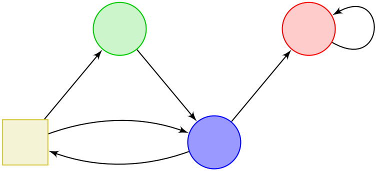


$$\langle\!\langle \bigcirc \rangle\!\rangle\ \mathbf{G}(\ \langle\!\langle \square \rangle\!\rangle\ \mathbf{F}\ \bigcirc)$$

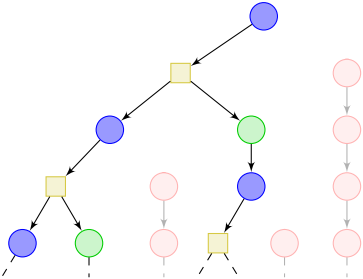


- consider the following strategy of Player $\bigcirc$: “always go to $\square$”;

Another semantics: ATL with strategy contexts

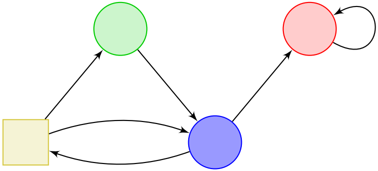


$\langle\!\langle \bigcirc \rangle\!\rangle\ \mathbf{G}(\ \langle\!\langle \Box \rangle\!\rangle\ \mathbf{F}\ \bigcirc)$

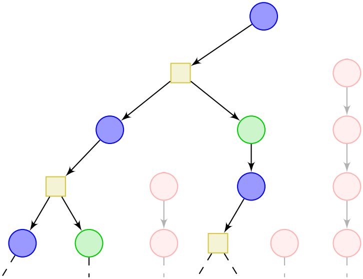


- consider the following strategy of Player $\bigcirc$: “always go to $\Box$”;

Another semantics: ATL with strategy contexts



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



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

ATL with strategy contexts

Definition

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

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

ATL with strategy contexts

Theorem

ATL_{sc} is strictly more expressive than ATL.

ATL with strategy contexts

Theorem

ATL_{sc} is strictly more expressive than ATL.

Proof

$$\langle\!\langle A \rangle\!\rangle\, \varphi \equiv \;\wr\mathrm{Agt}\wr\; \langle\cdot A \cdot\rangle\, \hat{\varphi}$$

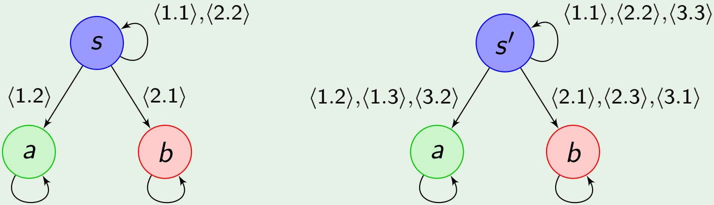
ATL with strategy contexts

Theorem

ATL_{sc} is strictly more expressive than ATL.

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.



What ATL_{sc} can express

- All ATL^* properties;

What ATL_{sc} can express

- All ATL* properties;
- Client-server interactions for accessing a shared resource:

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

What ATL_{sc} can express

- All ATL* properties;
- 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:

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

What ATL_{sc} can express

- All ATL* properties;
- Client-server interactions for accessing a shared resource:

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

- 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:

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

Model checking ATL_{sc}

Theorem

Given a CGS $\mathcal{C}$, a state ℓ_0 and an ATL_{sc} formula φ, we can build an alternating parity tree automaton $\mathcal{A}$ s.t.

$$\mathcal{L}(\mathcal{A}) \neq \varnothing \quad \Leftrightarrow \quad \mathcal{C}, \ell_0 \models_{\varnothing} \varphi.$$

$\mathcal{A}$ has size d-exponential, where d is the maximal number of nested quantifiers.

Model checking ATL_{sc}

Theorem

Given a CGS $\mathcal{C}$, a state ℓ_0 and an ATL_{sc} formula φ, we can build an alternating parity tree automaton $\mathcal{A}$ s.t.

$$\mathcal{L}(\mathcal{A}) \neq \varnothing \quad \Leftrightarrow \quad \mathcal{C}, \ell_0 \models_{\varnothing} \varphi.$$

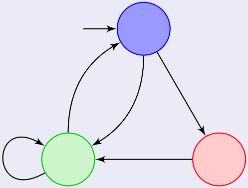
$\mathcal{A}$ has size d-exponential, where d is the maximal number of nested quantifiers.

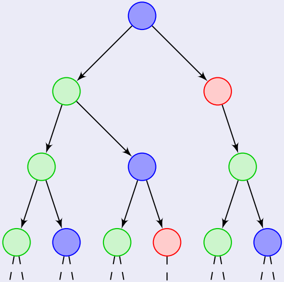
Theorem

Model checking ATL_{sc} is d-EXPTIME*-complete.*

Model checking ATL$_{sc}$

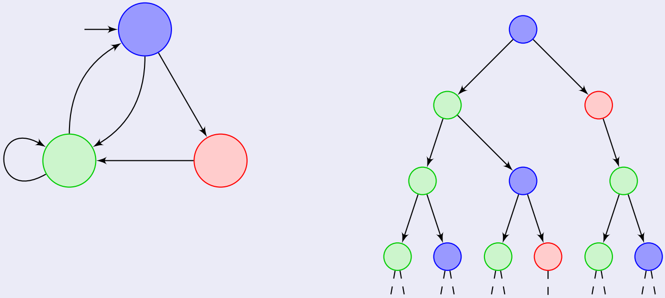
Tree-automata approach





Model checking ATL$_{sc}$

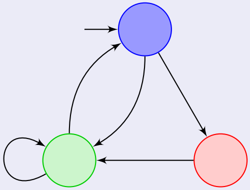
Tree-automata approach

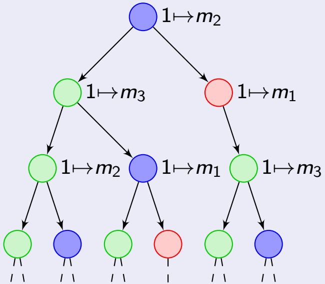


- The unwinding tree is accepted by a deterministic tree automaton;

Model checking ATL$_{sc}$

Tree-automata approach

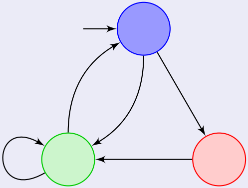


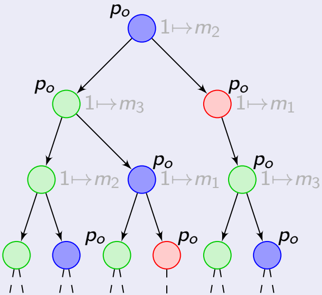


- A strategy is encoded as a labelling of the unwinding tree;

Model checking ATL$_{sc}$

Tree-automata approach

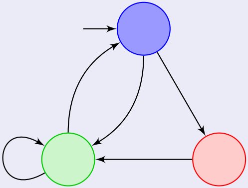


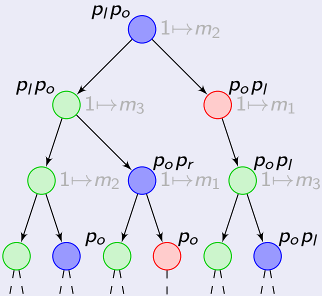


- We can mark outcomes corresponding to selected strategies;

Model checking ATL$_{sc}$

Tree-automata approach

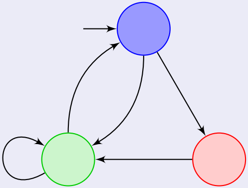


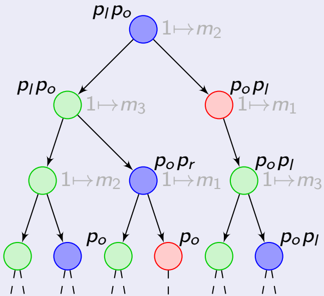


- We mark the tree with extra propositions p_l and p_r, and require that it satisfies $\mathbf{A}(\mathbf{G}\, p_o \Rightarrow p_l \;\mathbf{U}\; p_r)$;

Model checking ATL$_{sc}$

Tree-automata approach

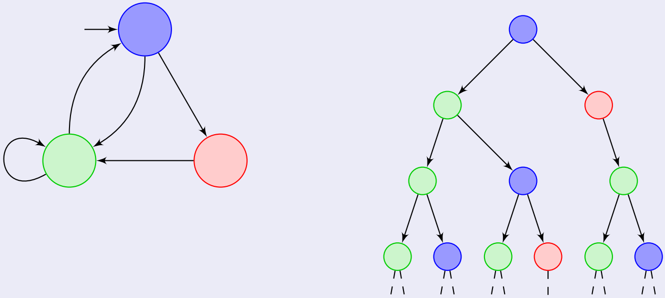




- We require that subtrees rooted at a p_l or p_r node is accepted by the automaton for φ or φ', respectively;

Model checking ATL$_{sc}$

Tree-automata approach



- We can build a tree automaton accepting all trees that *can be labelled* with correct strategies. This requires turning the alternating tree automaton into a non-deterministic one, which yields an exponential-size automaton.

Conclusions

- Our results on ATL_{sc}:
  - ATL_{sc} is a natural semantic extension of the popular ATL;
  - ATL_{sc} is much more expressive: equilibria, client-server interactions... Well-suited for non-zero-sum objectives;
  - There is a price for this expressiveness: high complexity of the model-checking algorithm.

Conclusions

- Our results on ATL_{sc}:
  - ATL_{sc} is a natural semantic extension of the popular ATL;
  - ATL_{sc} is much more expressive: equilibria, client-server interactions... Well-suited for non-zero-sum objectives;
  - There is a price for this expressiveness: high complexity of the model-checking algorithm.

- Future works:
  - study satisfiability of ATL_{sc};
  - behavioural equivalence for ATL_{sc}.
  - handle stochastic strategies, partial observation, ...