Variations on the semantics of Strategy Logic

Nicolas Markey

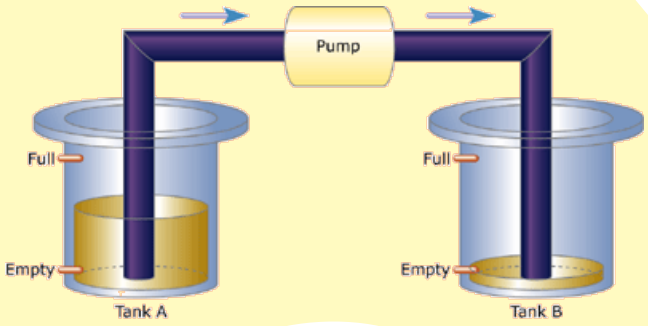
CNRS – IRISA (Univ. Rennes, France)

(based on joint works with Patricia Bouyer, Patrick Gardy, and François Laroussinie)

68nqrt – February 1, 2018

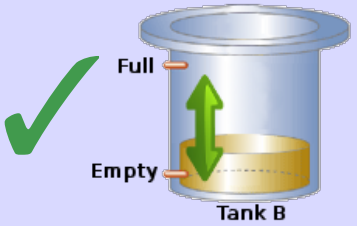
Model checking and synthesis

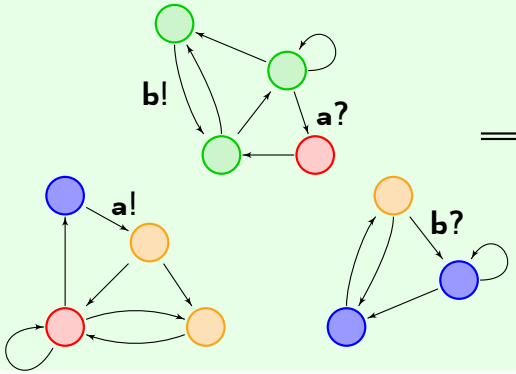
system:



[http://www.embedded.com]

property:



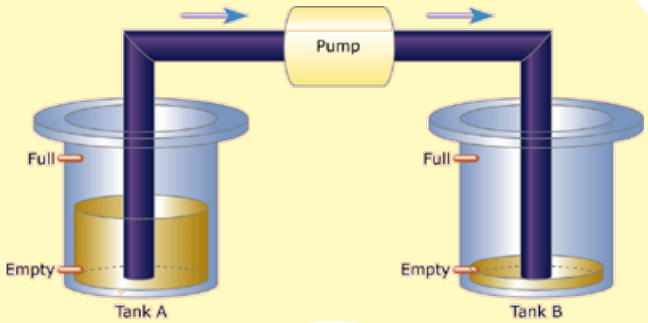


$\Longrightarrow$ model-checking algorithm $\Longleftarrow$ $\mathbf{A}\,\mathbf{G}(\neg\,\texttt{B.overfull} \wedge \neg\,\texttt{B.dried_up})$

yes/no

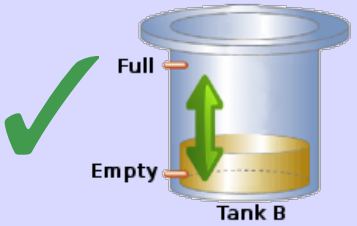
Model checking and synthesis

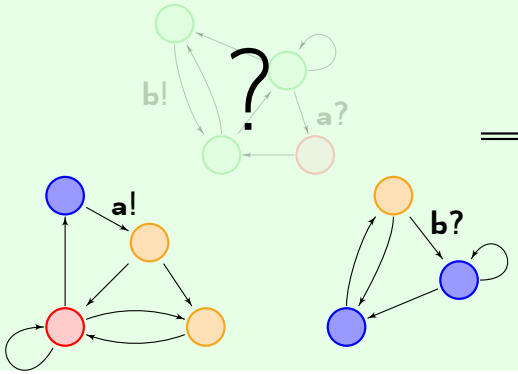
system:



[http://www.embedded.com]

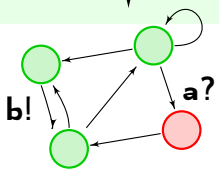
property:





synthesis algorithm

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



Outline of the presentation

1. Introduction
   - Computation-Tree Logic (CTL) and Quantified CTL
   - Alternating-time Temporal Logic (ATL)
2. Strategy Logic
   - Model checking Strategy Logic (SL)
   - Semantics of SL
     - When does history start?
     - What do strategies know about each other?
3. Conclusions and future works

Outline of the presentation

1. Introduction
   - Computation-Tree Logic (CTL) and Quantified CTL
   - Alternating-time Temporal Logic (ATL)
2. Strategy Logic
   - Model checking Strategy Logic (SL)
   - Semantics of SL
     - When does history start?
     - What do strategies know about each other?
3. Conclusions and future works

Computation-Tree Logic (CTL) [CE81]

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

[CE81] Clarke, Emerson. Design and Synthesis of Synchronization Skeletons... LOP, 1981.

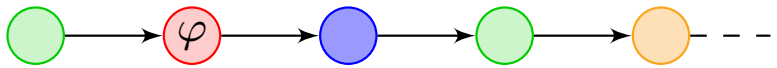
Computation-Tree Logic (CTL) [CE81]

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

[CE81] Clarke, Emerson. Design and Synthesis of Synchronization Skeletons... LOP, 1981.

Computation-Tree Logic (CTL) [CE81]

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

$\mathbf{X}\,\varphi$ 

"next φ"

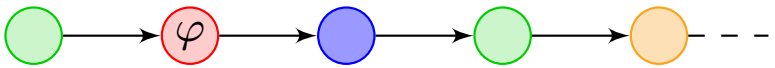
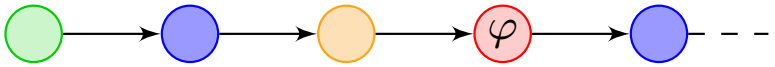
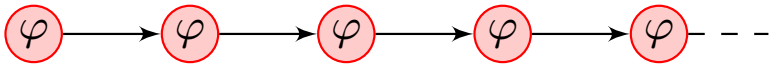
$\varphi\ \mathbf{U}\ \psi$

"φ until ψ"

[CE81] Clarke, Emerson. Design and Synthesis of Synchronization Skeletons... LOP, 1981.

Computation-Tree Logic (CTL) [CE81]

- **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"
$\texttt{true}\ \mathbf{U}\ \varphi \equiv \mathbf{F}\,\varphi$		"eventually φ"
$\neg\ \mathbf{F}\ \neg\varphi \equiv \mathbf{G}\,\varphi$		"always φ"

[CE81] Clarke, Emerson. Design and Synthesis of Synchronization Skeletons... LOP, 1981.

Computation-Tree Logic (CTL) [CE81]

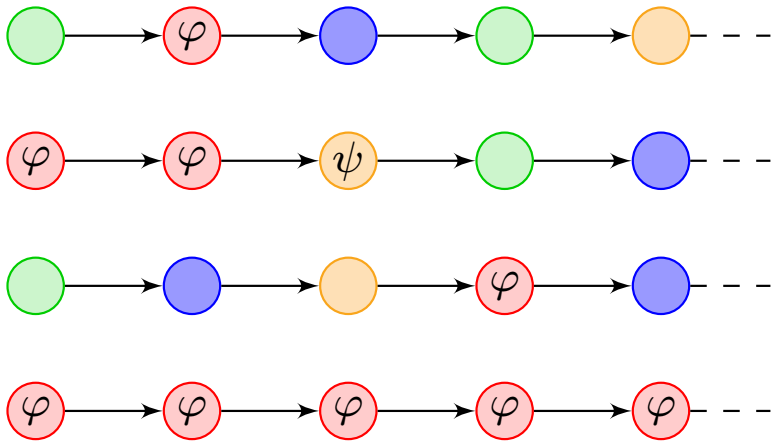
- **atomic propositions:** ◯, ◯, ...
- **boolean combinators:** $\neg\varphi$, $\varphi \vee \psi$, $\varphi \wedge \psi$, ...
- **temporal modalities:**

$\mathbf{X}\,\varphi$ — "next φ"

$\varphi\ \mathbf{U}\ \psi$ — "φ until ψ"

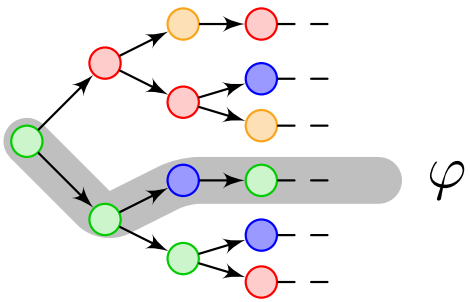
$\texttt{true}\ \mathbf{U}\ \varphi \equiv \mathbf{F}\,\varphi$ — "eventually φ"

$\neg\,\mathbf{F}\,\neg\varphi \equiv \mathbf{G}\,\varphi$ — "always φ"

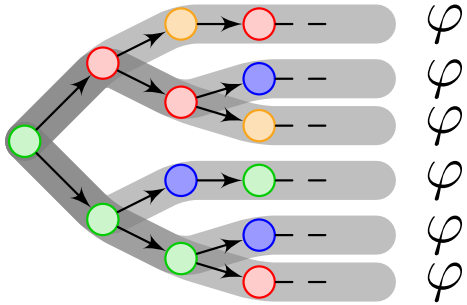


- **path quantifiers:**

$\mathbf{E}\varphi$



$\mathbf{A}\varphi$



[CE81] Clarke, Emerson. Design and Synthesis of Synchronization Skeletons... LOP, 1981.

Computation-Tree Logic (CTL) [CE81]

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

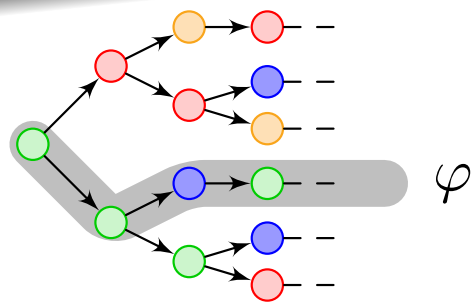
Examples

- **safety**: **AG** `safe`
  only `safe` states will be reached
- **fairness**: **AG AF** `fair`
  `fair` states will be visited infinitely many times
- **request-response**: **AG**(`request` $\Rightarrow$ **EF** `grant`)
  any `request` can eventually be granted

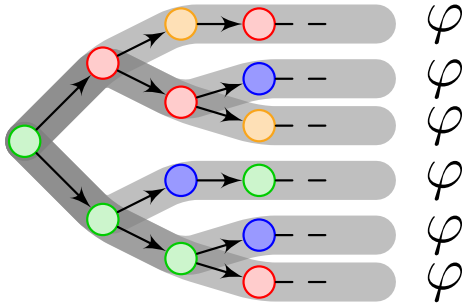
`true`

$\neg$ **F**

Eφ



Aφ



[CE81] Clarke, Emerson. Design and Synthesis of Synchronization Skeletons... LOP, 1981.

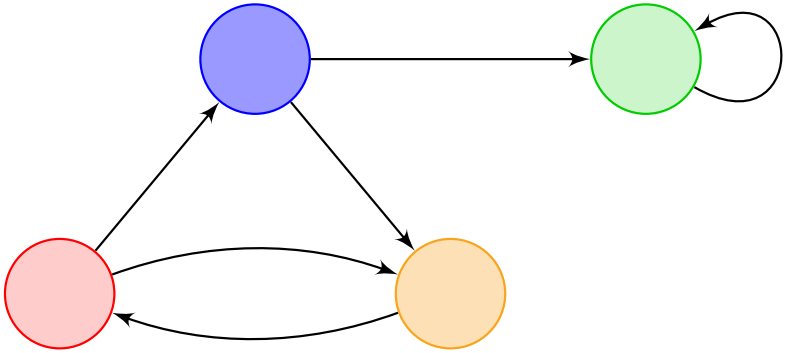
CTL model checking

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

CTL model checking

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

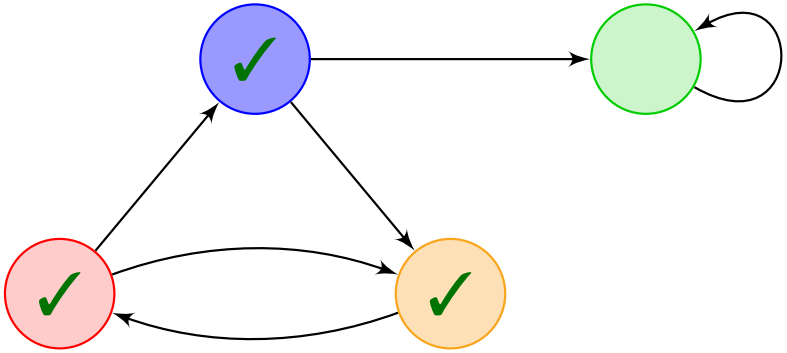
E F ● $\equiv$ some ● state is reachable



CTL model checking

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

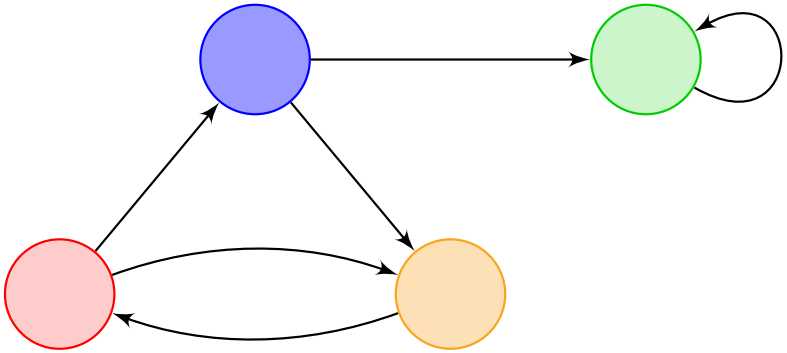
E F ● $\equiv$ some ● state is reachable



CTL model checking

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

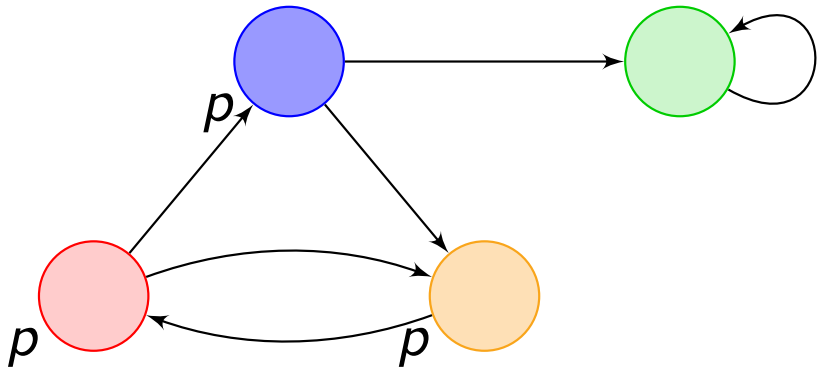
$\mathbf{E}\,\mathbf{G}(\neg$ ● $\wedge\ \mathbf{E}\,\mathbf{F}$ ●$)$ $\equiv$ there is a path along which ● is always reachable, but never reached



CTL model checking

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

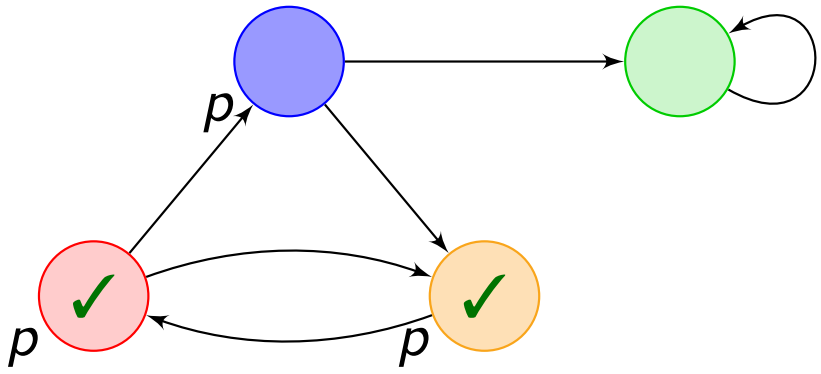
$\mathbf{E}\,\mathbf{G}(\neg$ ● $\wedge \underbrace{\mathbf{E}\,\mathbf{F}\,●}_{p})$ $\equiv$ there is a path along which ● is always reachable, but never reached



CTL model checking

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

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



CTL model checking

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... LOP, 1981.
[QS82] Queille, Sifakis. Specification and verification of concurrent systems in CESAR. SOP, 1982.

CTL model checking

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

Theorem ([CE81,QS82])

CTL model checking is PTIME*-complete.*

Remark ([KVW94])

CTL model checking on product structures is PSPACE-complete.

[CE81] Clarke, Emerson. Design and Synthesis of Synchronization Skeletons... LOP, 1981.
[QS82] Queille, Sifakis. Specification and verification of concurrent systems in CESAR. SOP, 1982.
[KVW94] Kupferman, Vardi, Wolper. An automata-theoretic approach to branching-time... CAV, 1994.

Quantified CTL [ES84,Kup95,Fre01,DLM12]

QCTL extends CTL with propositional quantifiers

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

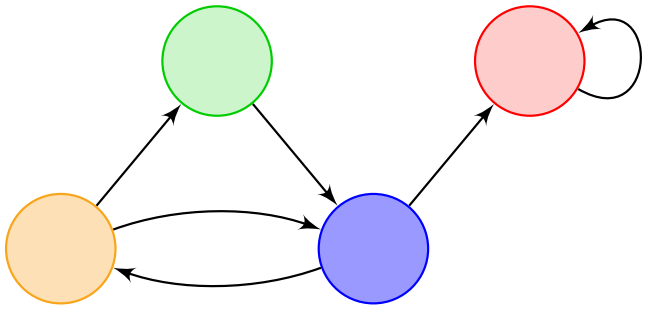
[ES84] Emerson and Sistla. Deciding Full Branching Time Logic. Information & Control, 1984.
[Kup95] Kupferman. Augmenting Branching Temporal Logics with Existential... CAV, 1995.
[Fre01] French. Decidability of Quantifed Propositional Branching Time Logics. AJCAI, 2001.
[DLM12] Da Costa, Laroussinie, Markey. Quantified CTL: ... CONCUR, 2012.

Quantified CTL [ES84,Kup95,Fre01,DLM12]

QCTL extends CTL with propositional quantifiers

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

$$\mathbf{E}\,\mathbf{F}\,\bigcirc \;\wedge\; \neg\Big(\exists p.\ \big[\mathbf{E}\,\mathbf{F}(\bigcirc \wedge p) \wedge \mathbf{E}\,\mathbf{F}(\bigcirc \wedge \neg p)\big]\Big)$$



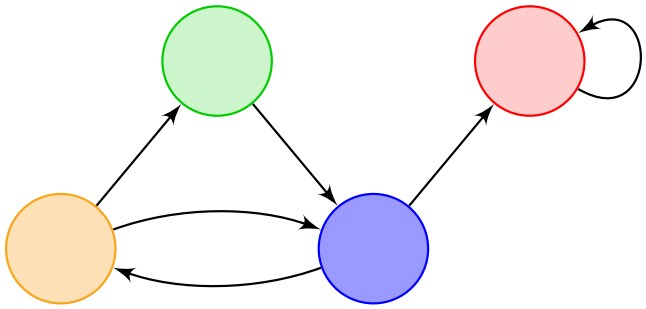
[ES84] Emerson and Sistla. Deciding Full Branching Time Logic. Information & Control, 1984.
[Kup95] Kupferman. Augmenting Branching Temporal Logics with Existential... CAV, 1995.
[Fre01] French. Decidability of Quantifed Propositional Branching Time Logics. AJCAI, 2001.
[DLM12] Da Costa, Laroussinie, Markey. Quantified CTL: ... CONCUR, 2012.

Quantified CTL [ES84,Kup95,Fre01,DLM12]

QCTL extends CTL with propositional quantifiers

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

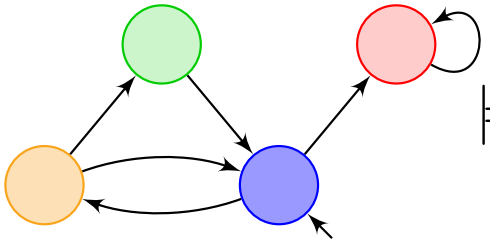
$$\mathbf{E}\,\mathbf{F}\,\bigcirc \;\wedge\; \neg\Big(\exists p.\ \big[\mathbf{E}\,\mathbf{F}(\bigcirc \wedge p) \wedge \mathbf{E}\,\mathbf{F}(\bigcirc \wedge \neg p)\big]\Big) \equiv \mathsf{uniq}(\bigcirc)$$



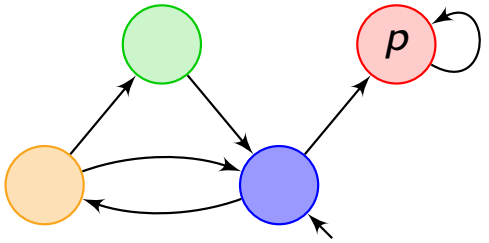
[ES84] Emerson and Sistla. Deciding Full Branching Time Logic. Information & Control, 1984.
[Kup95] Kupferman. Augmenting Branching Temporal Logics with Existential... CAV, 1995.
[Fre01] French. Decidability of Quantifed Propositional Branching Time Logics. AJCAI, 2001.
[DLM12] Da Costa, Laroussinie, Markey. Quantified CTL: ... CONCUR, 2012.

Semantics of QCTL

- structure semantics:



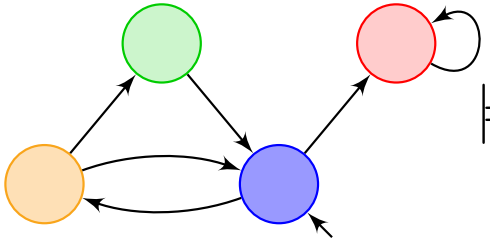
$\models_s \exists p.\varphi \quad \Leftrightarrow$



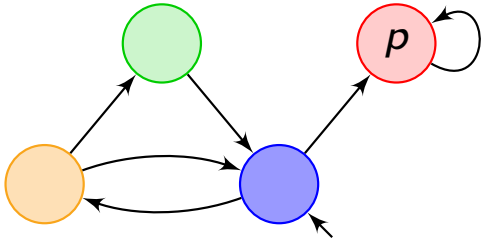
$\models \varphi$

Semantics of QCTL

- structure semantics:

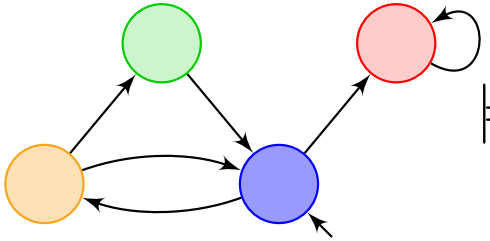


$\models_s \exists p.\varphi \quad \Leftrightarrow$

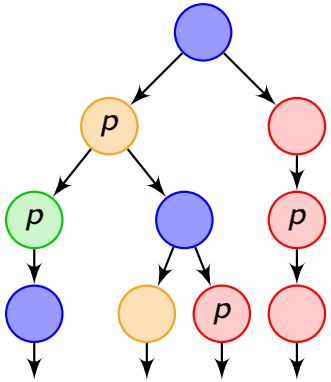


$\models \varphi$

- tree semantics:



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



$\models \varphi$

Expressiveness of QCTL

- QCTL can "count":

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

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:

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

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

QCTL and MSO are equally expressive.

Decision problems for QCTL [LM14]

Theorem

Under the structure semantics:

- *model checking is* PSPACE*-complete;*
- *satisfiability is undecidable.*

[LM14] Laroussinie, Markey. Quantified CTL: expressiveness and complexity. LMCS, 2014.

Decision problems for QCTL [LM14]

Theorem

Under the structure semantics:

- *model checking is* PSPACE*-complete;*
- *satisfiability is undecidable.*

Under the tree semantics: for formulas with k nested quantifiers,

- *model checking is* k-EXPTIME*-complete;*
- *satisfiability is* $(k+1)$-EXPTIME*-complete.*

[LM14] Laroussinie, Markey. Quantified CTL: expressiveness and complexity. LMCS, 2014.

Outline of the presentation

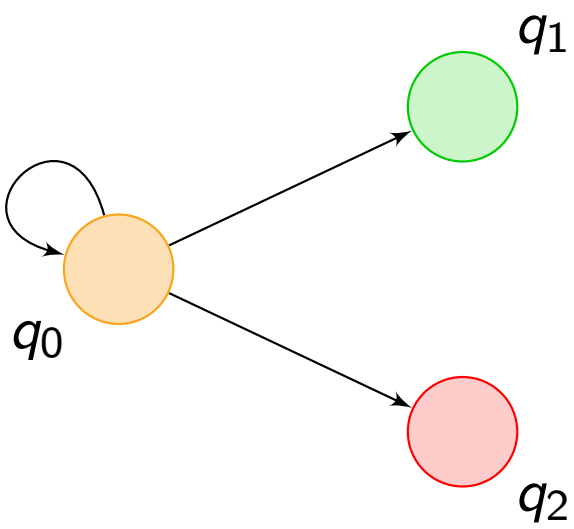
1. Introduction
   - Computation-Tree Logic (CTL) and Quantified CTL
   - Alternating-time Temporal Logic (ATL)
2. Strategy Logic
   - Model checking Strategy Logic (SL)
   - Semantics of SL
     - When does history start?
     - What do strategies know about each other?
3. Conclusions and future works

Games on graphs

Concurrent games

A concurrent game is made of

- a transition system;

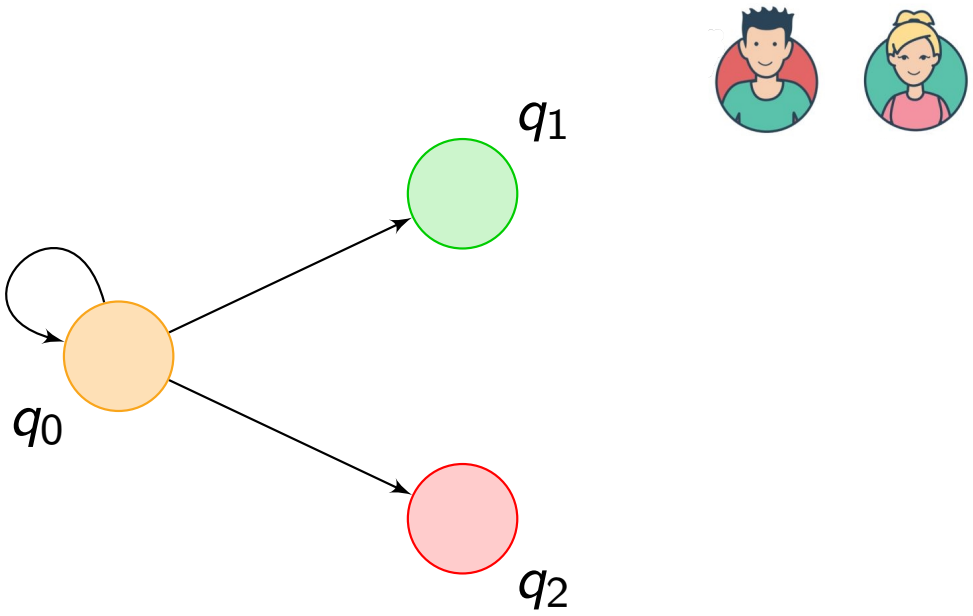


Games on graphs

Concurrent games

A concurrent game is made of

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

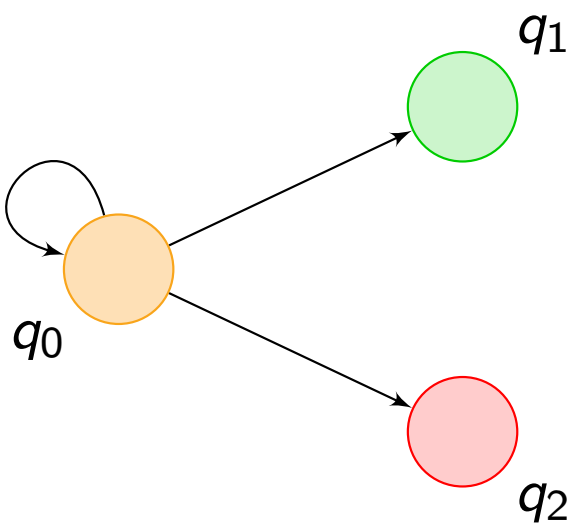


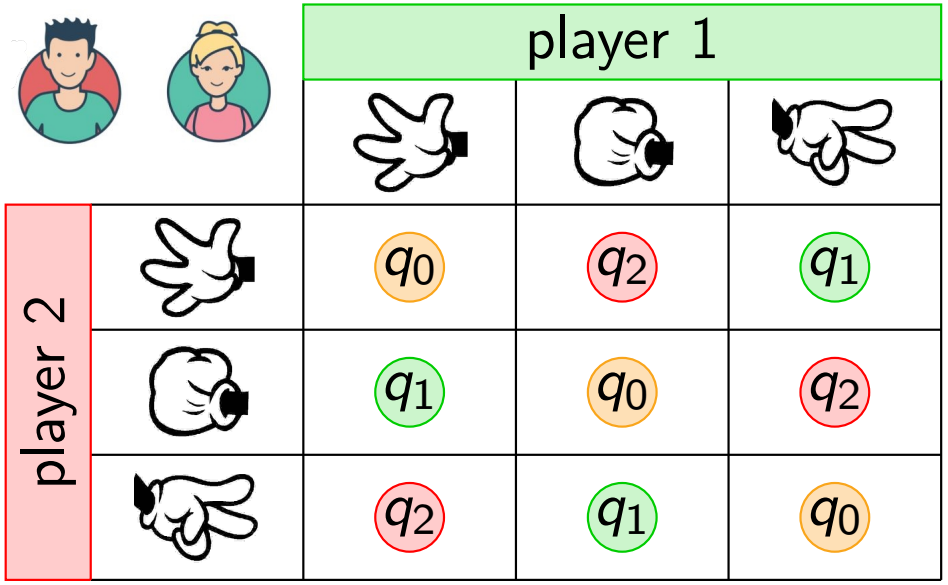
Games on graphs

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.





Games on graphs

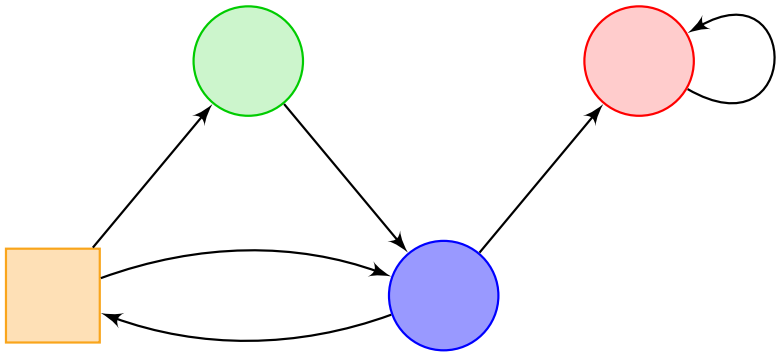
Concurrent games

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

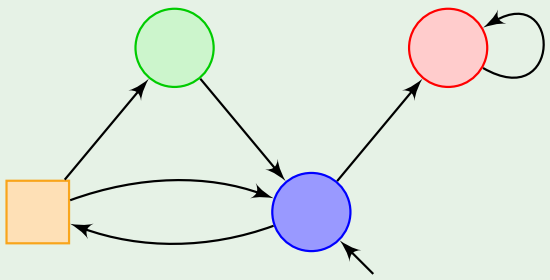
A (pure) strategy for a given player is a function telling which action to play depending on what has happened previously.

Reasoning about open systems

Strategies

A (pure) strategy for a given player is a function telling which action to play depending on what has happened previously.

Example



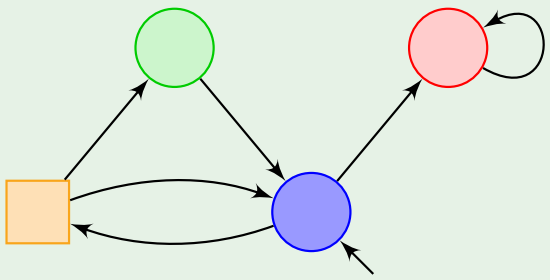
Reasoning about open systems

Strategies

A (pure) strategy for a given player is a function telling which action to play depending on what has happened previously.

Example

Strategy for player



Reasoning about open systems

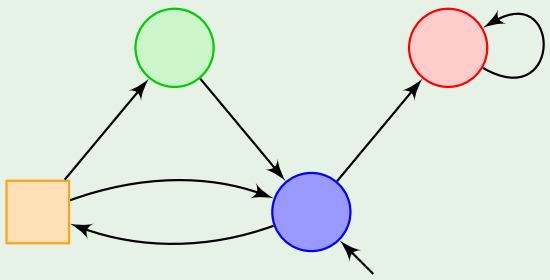
Strategies

A (pure) strategy for a given player is a function telling which action to play depending on what has happened previously.

Example

Strategy for player

alternately go to and (starting with).



Reasoning about open systems

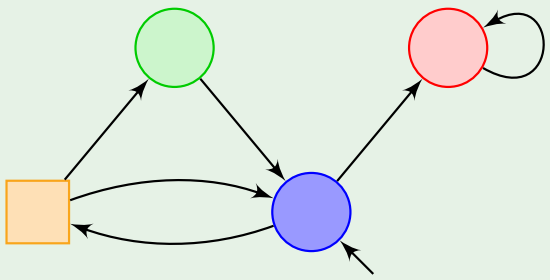
Strategies

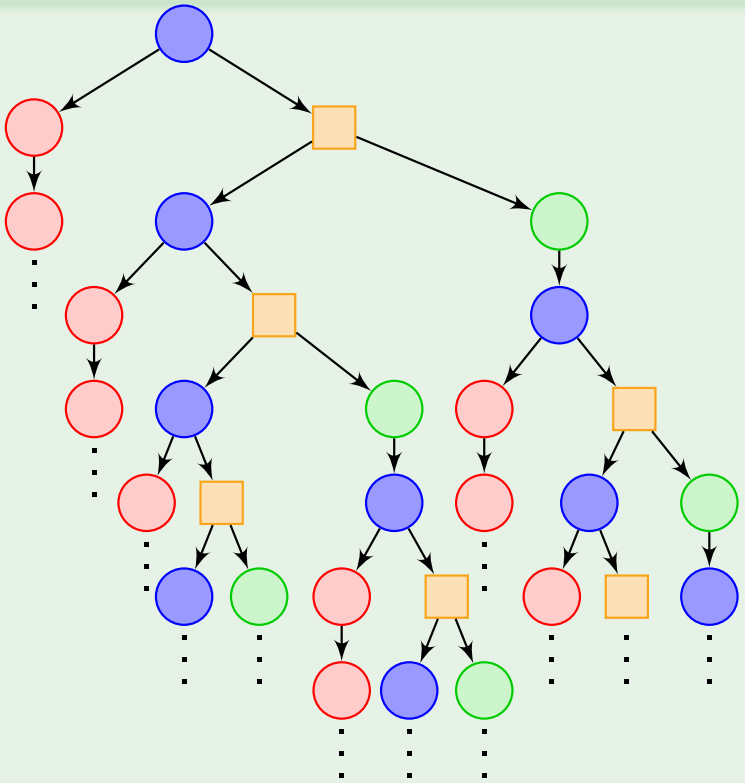
A (pure) strategy for a given player is a function telling which action to play depending on what has happened previously.

Example

Strategy for player

alternately go to and (starting with).





Reasoning about open systems

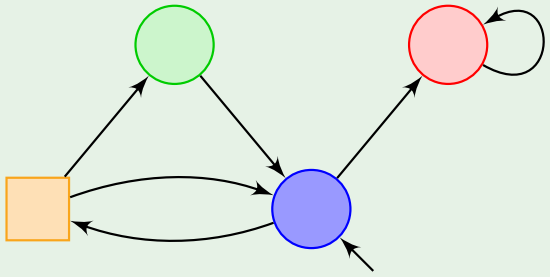
Strategies

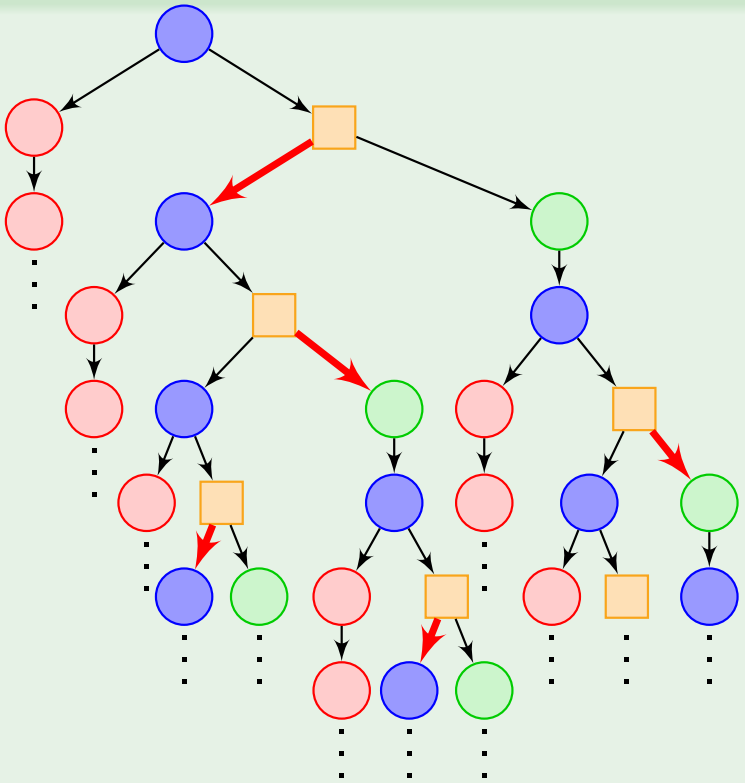
A (pure) strategy for a given player is a function telling which action to play depending on what has happened previously.

Example

Strategy for player

alternately go to and (starting with).





Reasoning about open systems

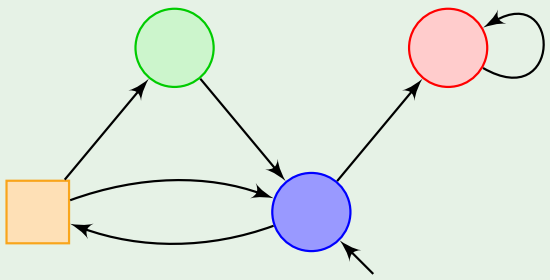
Strategies

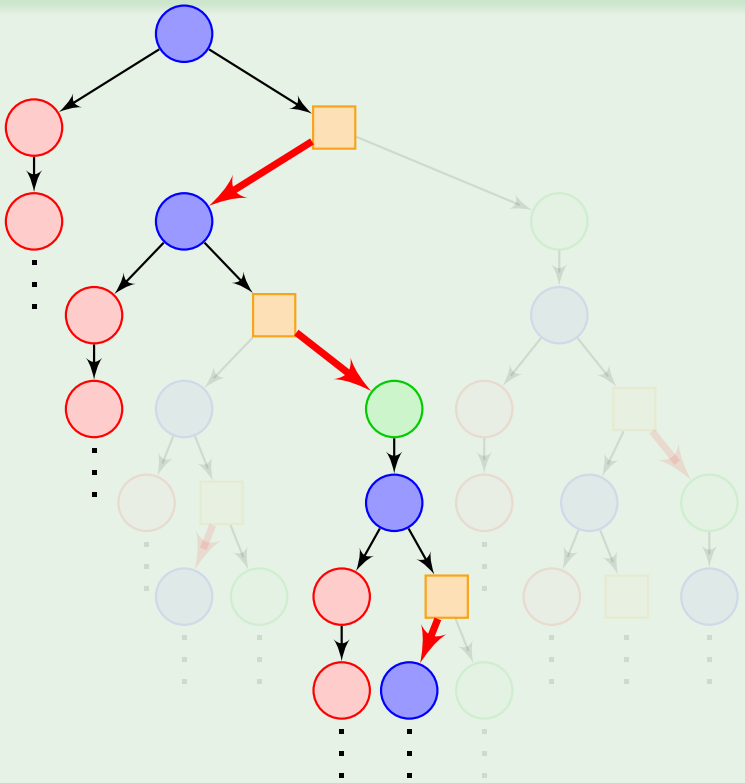
A (pure) strategy for a given player is a function telling which action to play depending on what has happened previously.

Example

Strategy for player

alternately go to and (starting with).





Temporal logics for games: ATL [AHK02]

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 [AHK02]

ATL extends CTL with strategy quantifiers

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

Semantics of $\langle\langle A \rangle\rangle\, \varphi$

Existential quantification (over strategies) implicitly includes a universal quantification (over outcomes):

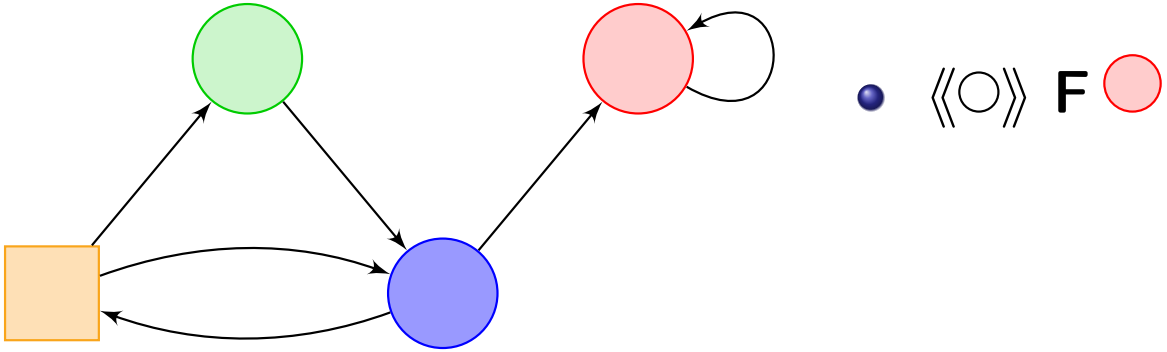
$$\mathcal{G}, \bigcirc \models \langle\langle A \rangle\rangle\, \varphi \quad \iff \quad \exists \sigma_A.\ \forall \pi \in \mathsf{Out}(\bigcirc, \sigma_A).\ \pi \models \varphi.$$

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

Temporal logics for games: ATL [AHK02]

ATL extends CTL with strategy quantifiers

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

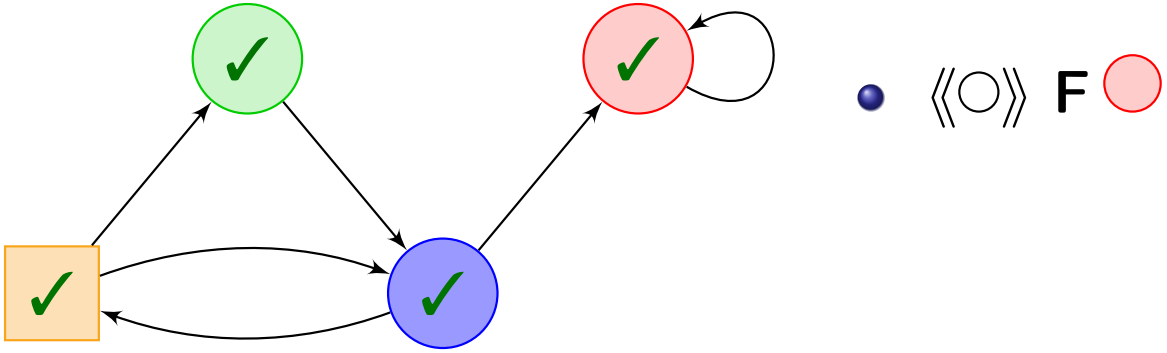


[AHK02] Alur, Henzinger, Kupferman. Alternating-time Temporal Logic. J. ACM, 2002.

Temporal logics for games: ATL [AHK02]

ATL extends CTL with strategy quantifiers

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

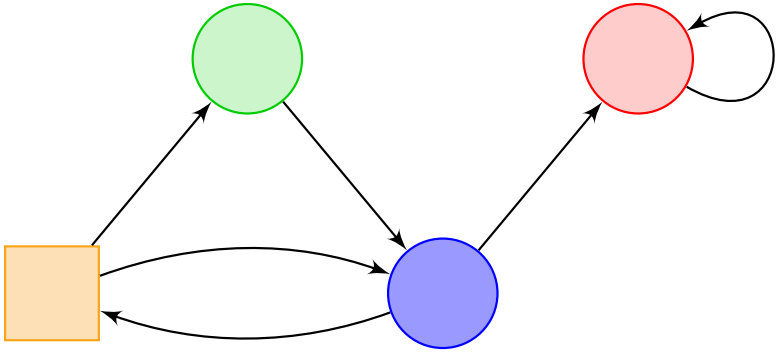


[AHK02] Alur, Henzinger, Kupferman. Alternating-time Temporal Logic. J. ACM, 2002.

Temporal logics for games: ATL [AHK02]

ATL extends CTL with strategy quantifiers

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



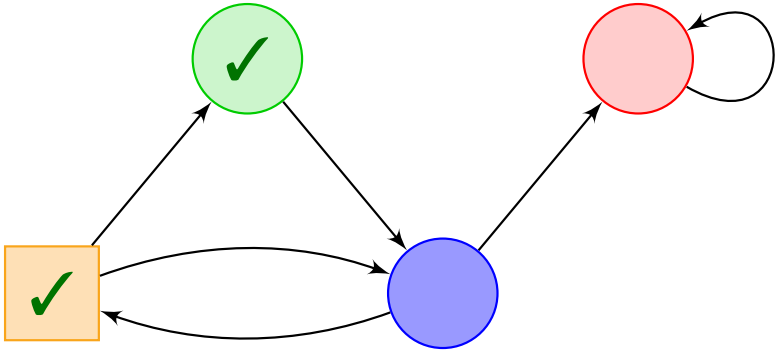
- $\langle\langle \bigcirc \rangle\rangle$ **F**
- $\langle\langle \Box \rangle\rangle$ **F**

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

Temporal logics for games: ATL [AHK02]

ATL extends CTL with strategy quantifiers

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



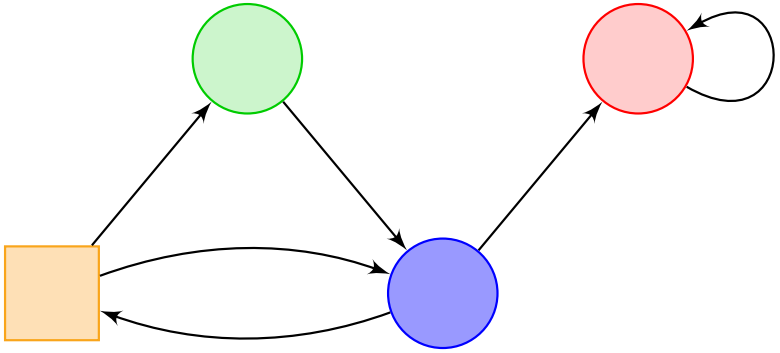
- $\langle\langle \bigcirc \rangle\rangle$ **F**
- $\langle\langle \Box \rangle\rangle$ **F**

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

Temporal logics for games: ATL [AHK02]

ATL extends CTL with strategy quantifiers

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



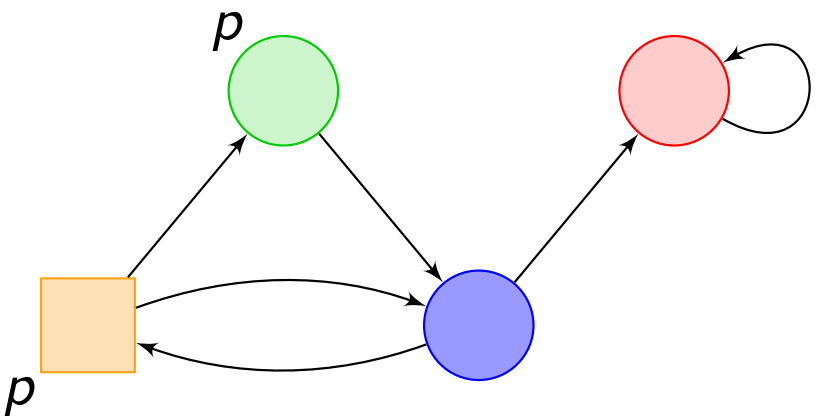
- $\langle\langle \bigcirc \rangle\rangle\ \mathbf{F}$
- $\langle\langle \Box \rangle\rangle\ \mathbf{F}$
- $\langle\langle \bigcirc \rangle\rangle\ \mathbf{G}(\ \langle\langle \Box \rangle\rangle\ \mathbf{F}\ )$

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

Temporal logics for games: ATL [AHK02]

ATL extends CTL with strategy quantifiers

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



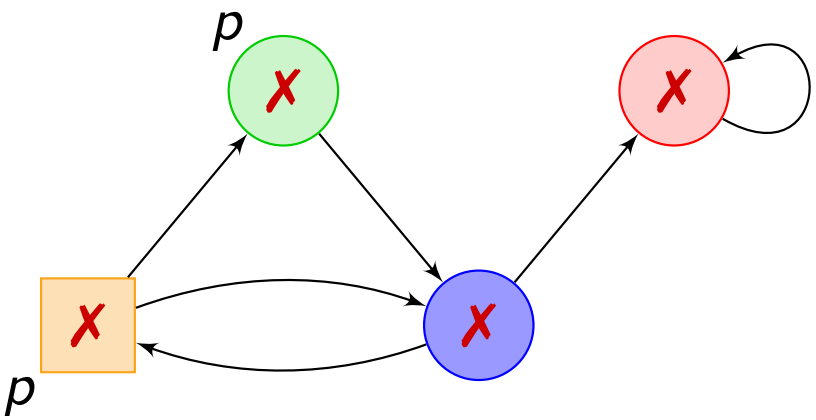
- $\langle\langle \bigcirc \rangle\rangle\, \mathbf{F}$
- $\langle\langle \Box \rangle\rangle\, \mathbf{F}$
- $\langle\langle \bigcirc \rangle\rangle\, \mathbf{G}(\underbrace{\langle\langle \Box \rangle\rangle\, \mathbf{F}}_{p}) \equiv \langle\langle \bigcirc \rangle\rangle\, \mathbf{G}\, p$

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

Temporal logics for games: ATL [AHK02]

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 [AHK02]

ATL extends CTL with strategy quantifiers

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

Theorem ([AHK02])

Model checking ATL is PTIME*-complete.*

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

Temporal logics for games: ATL [AHK02]

ATL extends CTL with strategy quantifiers

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

Theorem ([AHK02])

Model checking ATL is PTIME*-complete.*

Remark ([LMO08])

In PTIME only if the transition table is given explicitly (size $|\mathsf{Moves}|^{|\mathsf{Agt}|}$)

[AHK02] Alur, Henzinger, Kupferman. Alternating-time Temporal Logic. J. ACM, 2002.
[LMO08] Laroussinie, Markey, Oreiby. On the Expressiveness and Complexity of ATL. LMCS, 2008

Outline of the presentation

1. Introduction
   - Computation-Tree Logic (CTL) and Quantified CTL
   - Alternating-time Temporal Logic (ATL)
2. Strategy Logic
   - Model checking Strategy Logic (SL)
   - Semantics of SL
     - When does history start?
     - What do strategies know about each other?
3. Conclusions and future works

Outline of the presentation

1. Introduction
   - Computation-Tree Logic (CTL) and Quantified CTL
   - Alternating-time Temporal Logic (ATL)
2. Strategy Logic
   - Model checking Strategy Logic (SL)
   - Semantics of SL
     - When does history start?
     - What do strategies know about each other?
3. Conclusions and future works

Strategy Logic [CHP07,MMV10]

Strategy logic

Explicit quantification over strategies + strategy assignement:

$$\exists \sigma_A.\ \forall \sigma_B.\ \mathsf{assign}(A \to \sigma_A, B \to \sigma_B).\ \varphi$$

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

Strategy Logic [CHP07,MMV10]

Strategy logic

Explicit quantification over strategies + strategy assignement:

$$\underline{\exists \sigma_A.\ \forall \sigma_B}.\ \underline{\mathsf{assign}(A \to \sigma_A, B \to \sigma_B)}.\ \underline{\varphi}$$

Quantification over strategies

Assignment of strategies to players

Path property

[CHP07] Chatterjee, Henzinger, Piterman. Strategy Logic. CONCUR, 2007.

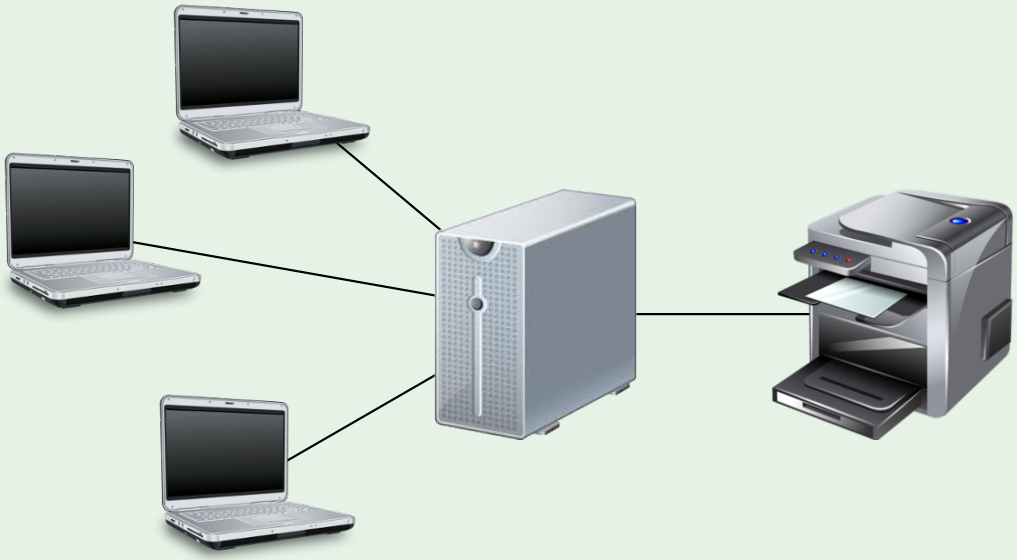
[MMV10] Mogavero, Murano, Vardi. Reasoning about strategies. FSTTCS, 2010.

Strategy Logic

Example

- client-server interactions

$$\exists \sigma_S.\ \exists \sigma_c.\ \text{assign}(S \to \sigma_S).\ (\mathbf{A}\,\mathbf{G}\ \text{mutual exclusion} \wedge \mathbf{A}\,\mathbf{G} \bigwedge_C \text{assign}(C \to \sigma_c).\ \mathbf{A}\,\mathbf{F}\ \text{access}_C)$$



Strategy Logic

Example

- client-server interactions
- existence of a pure boolean Nash equilibrium

$$\exists \sigma_1 \ldots \exists \sigma_n.\ \mathsf{assign}(A_1 \to \sigma_1, \ldots A_n \to \sigma_n).$$
$$\Big[\bigwedge_i ((\exists \sigma'_i.\mathsf{assign}(A_i \to \sigma'_i).\ \varphi_i) \Rightarrow \varphi_i.)\Big]$$

Strategy Logic

Example

- client-server interactions
- existence of a pure boolean Nash equilibrium
- existence of a dominant strategy for Player 1

$$\exists \sigma_1.\ \forall \sigma_2.\ \mathsf{assign}(A_2 \to \sigma_2).$$
$$\left[(\exists \sigma_1'.\ \mathsf{assign}(A_1 \to \sigma_1').\ \varphi_1) \Rightarrow (\mathsf{assign}(A_1 \to \sigma_1').\ \varphi_1)\right]$$

Strategy Logic

Example

- client-server interactions
- existence of a pure boolean Nash equilibrium
- existence of a dominant strategy for Player 1
- existence of an admissible strategy for Player 1

$$\exists \sigma_1.\ \neg\Big(\exists \sigma_1'.\ \forall \sigma_2.\ \mathsf{assign}(A_2 \to \sigma_2).$$
$$\big[\mathsf{assign}(A_1 \to \sigma_1).\ \varphi_1 \ \Rightarrow\ \mathsf{assign}(A_1 \to \sigma_1').\ \varphi_1\big] \wedge$$
$$\exists \sigma_2.\ \mathsf{assign}(A_2 \to \sigma_2).$$
$$\big[\mathsf{assign}(A_1 \to \sigma_1).\ \neg\varphi_1 \wedge \mathsf{assign}(A_1 \to \sigma_1').\ \varphi_1\big]\Big)$$

Model checking Strategy Logic

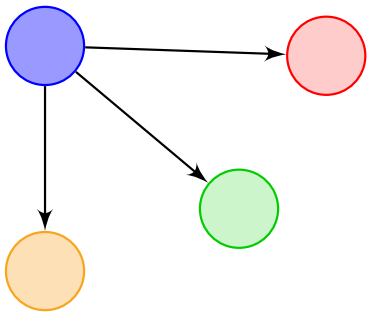
Theorem

Strategy logic can be translated into QCTL.

Model checking Strategy Logic

Theorem

Strategy logic can be translated into QCTL.

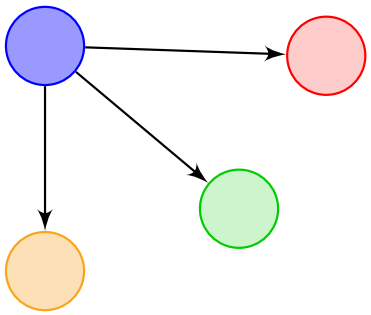


- players has moves m_1, ..., m_n;
- from the transition table, we can compute the set Next(, A, m_i) of states that can be reached from when player A plays m_i.

Model checking Strategy Logic

Theorem

Strategy logic can be translated into QCTL.



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

SL can be translated as follows:

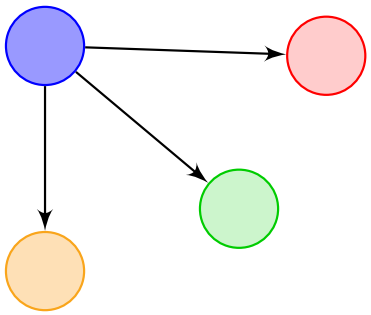
- encoding of $\exists\sigma.\ \psi$:

$$\exists m_1^\sigma\ \exists m_2^\sigma \ldots \exists m_k^\sigma.\ \mathbf{A}\,\mathbf{G}(m_i^\sigma \Leftrightarrow \bigwedge \neg\, m_j^\sigma) \wedge \hat{\psi}$$

Model checking Strategy Logic

Theorem

Strategy logic can be translated into QCTL.



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

SL can be translated as follows:

- encoding of $\exists\sigma.\ \psi$:

$$\exists m_1^\sigma\ \exists m_2^\sigma \ldots \exists m_k^\sigma.\ \mathbf{A}\,\mathbf{G}(m_i^\sigma \Leftrightarrow \bigwedge \neg\, m_j^\sigma) \wedge \hat{\psi}$$

- encoding of assign(α). φ (for full binding α: Agt $\rightarrow$ Strat):

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

Model checking Strategy Logic

Theorem

Strategy logic can be translated into QCTL.

Theorem ([CHP07,MMV10,DLM12,LM13])

- *Strategy-Logic model-checking is decidable.*
- *Strategy-Logic satisfiability is undecidable over concurrent games.*
- *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.
[DLM12] Da Costa, Laroussinie, Markey. Quantified CTL: ... CONCUR, 2012.
[LM13] Laroussinie, Markey. Satisfiability of ATL with strategy contexts. Gandalf, 2013.

Outline of the presentation

1. Introduction
   - Computation-Tree Logic (CTL) and Quantified CTL
   - Alternating-time Temporal Logic (ATL)
2. Strategy Logic
   - Model checking Strategy Logic (SL)
   - Semantics of SL
     - When does history start?
     - What do strategies know about each other?
3. Conclusions and future works

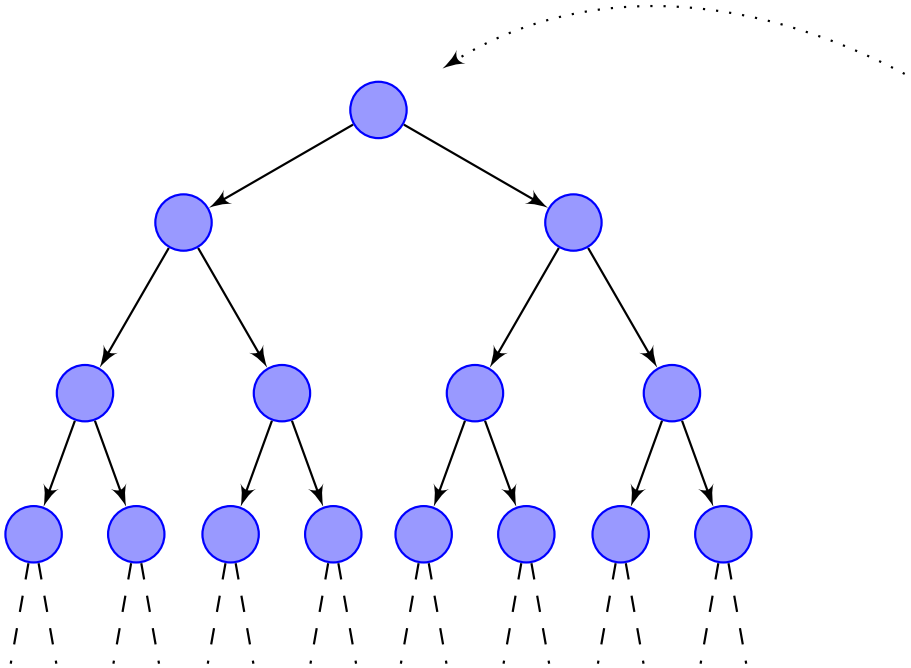
What is the exact semantics? [BGM16]

$$\exists \sigma.\ \mathbf{AG}(\mathsf{assign}(A \to \sigma).\ \varphi)$$

[BGM16] Bouyer, Gardy, Markey. On the semantics of SL. IPL, 2016.

What is the exact semantics? [BGM16]

$$\exists\sigma.\ \mathbf{AG}(\text{assign}(A \to \sigma).\ \varphi)$$

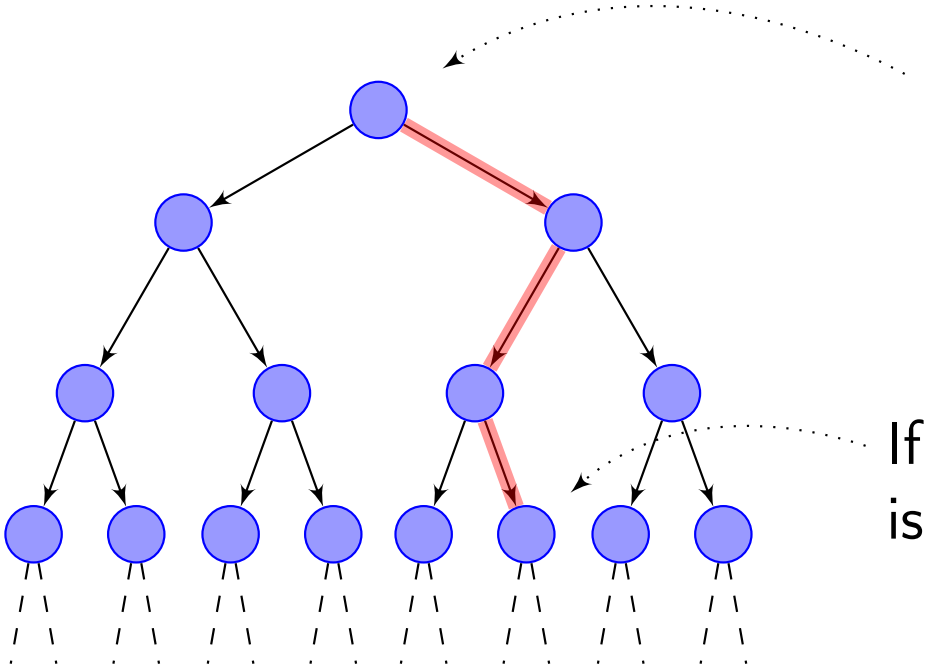


Strategy σ is selected here

[BGM16] Bouyer, Gardy, Markey. On the semantics of SL. IPL, 2016.

What is the exact semantics? [BGM16]

$$\exists\sigma.\ \mathbf{AG}(\text{assign}(A \to \sigma).\ \varphi)$$



Strategy σ is selected here

If strategy σ is assigned here,
is the prefix part of the history?

[BGM16] Bouyer, Gardy, Markey. On the semantics of SL. IPL, 2016.

What is the exact semantics? [BGM16]

$$\exists \sigma.\ \mathbf{AG}(\text{assign}(A \to \sigma).\ \varphi)$$

Example

If σ prescribes to alternate between m_0 and m_1, starting with m_0:

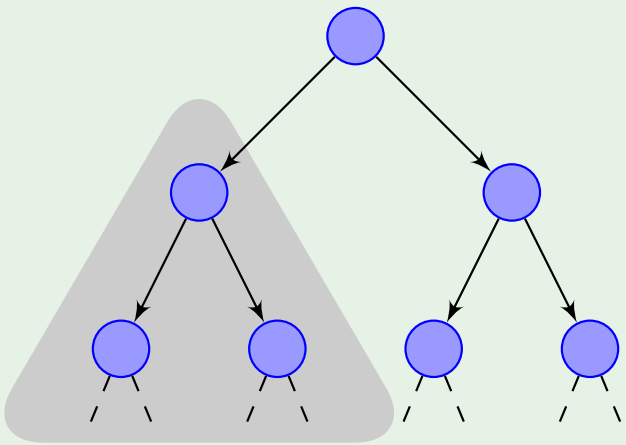
[BGM16] Bouyer, Gardy, Markey. On the semantics of SL. IPL, 2016.

What is the exact semantics? [BGM16]

$$\exists\sigma.\ \mathbf{AG}(\text{assign}(A \to \sigma).\ \varphi)$$

Example

If σ prescribes to alternate between m_0 and m_1, starting with m_0:



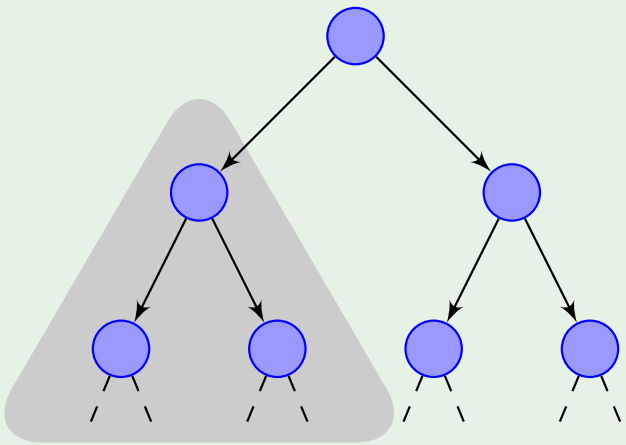
[BGM16] Bouyer, Gardy, Markey. On the semantics of SL. IPL, 2016.

What is the exact semantics? [BGM16]

$$\exists\sigma.\ \mathbf{AG}(\text{assign}(A \to \sigma).\ \varphi)$$

Example

If σ prescribes to alternate between m_0 and m_1, starting with m_0:



History starts when strategy is selected.

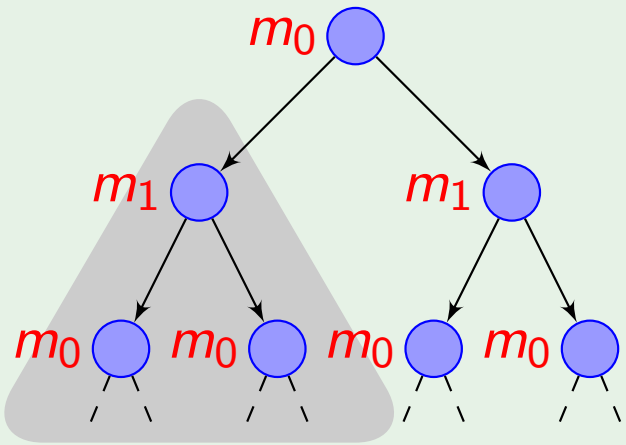
[BGM16] Bouyer, Gardy, Markey. On the semantics of SL. IPL, 2016.

What is the exact semantics? [BGM16]

$$\exists\sigma.\ \mathbf{AG}(\text{assign}(A \to \sigma).\ \varphi)$$

Example

If σ prescribes to alternate between m_0 and m_1, starting with m_0:



History starts when strategy is selected.

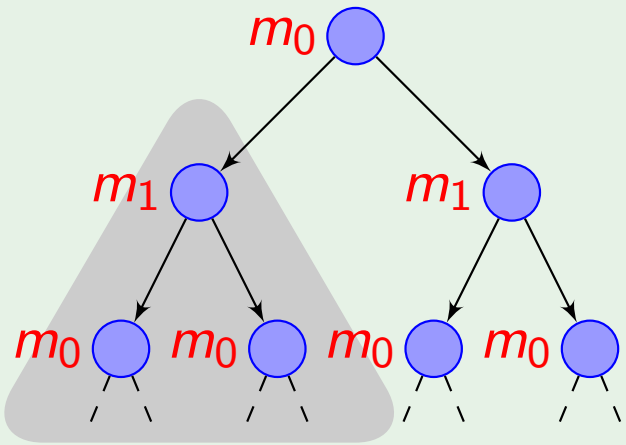
[BGM16] Bouyer, Gardy, Markey. On the semantics of SL. IPL, 2016.

What is the exact semantics? [BGM16]

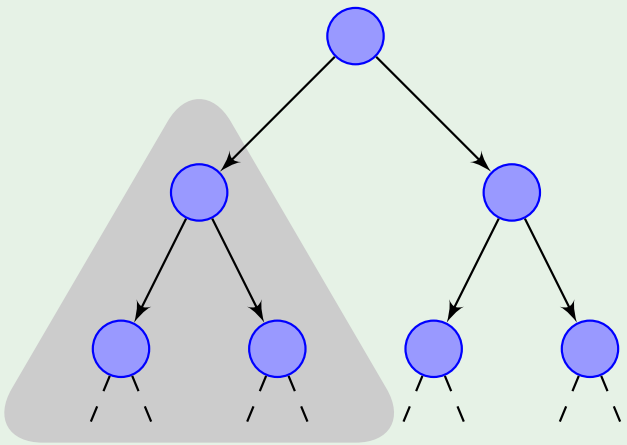
$$\exists\sigma.\ \mathbf{AG}(\text{assign}(A \to \sigma).\ \varphi)$$

Example

If σ prescribes to alternate between m_0 and m_1, starting with m_0:



History starts when strategy is selected.



History starts when strategy is assigned.

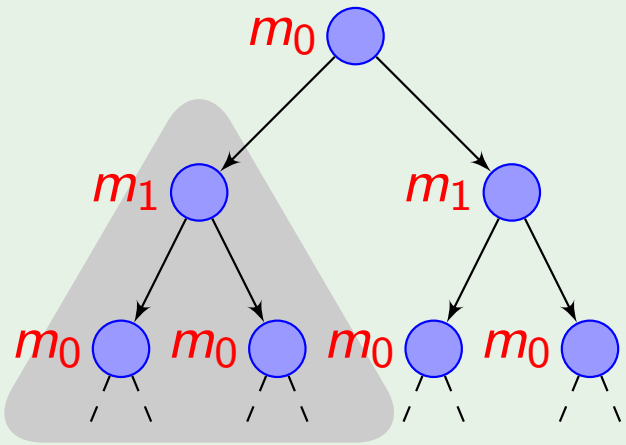
[BGM16] Bouyer, Gardy, Markey. On the semantics of SL. IPL, 2016.

What is the exact semantics? [BGM16]

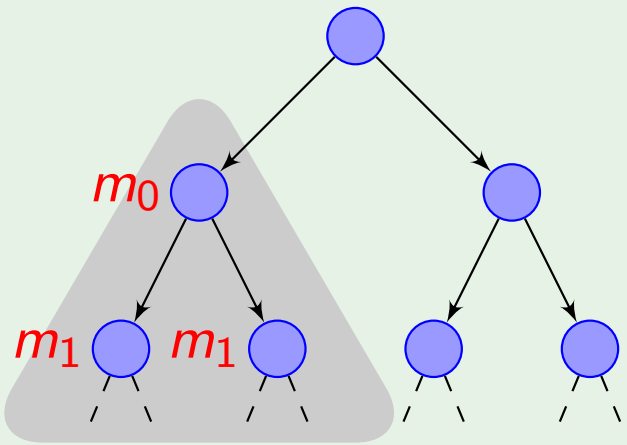
$$\exists\sigma.\ \mathbf{AG}(\text{assign}(A \to \sigma).\ \varphi)$$

Example

If σ prescribes to alternate between m_0 and m_1, starting with m_0:



History starts when strategy is selected.



History starts when strategy is assigned.

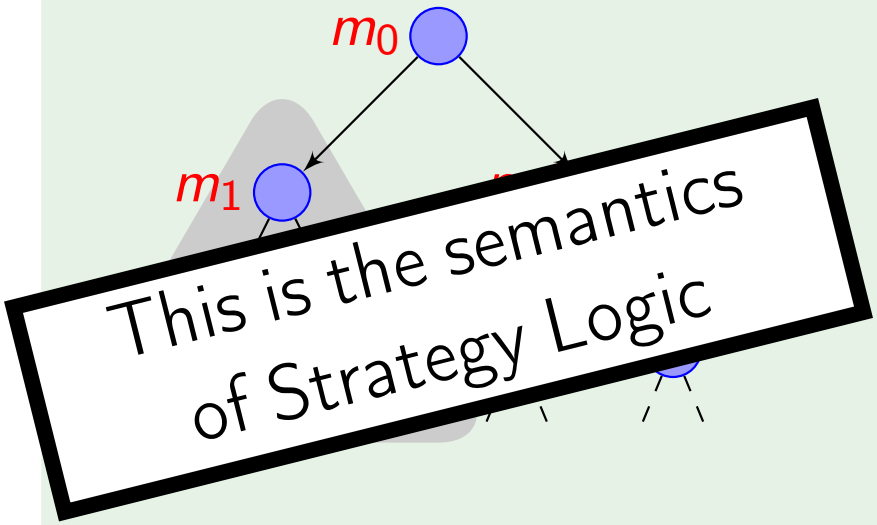
[BGM16] Bouyer, Gardy, Markey. On the semantics of SL. IPL, 2016.

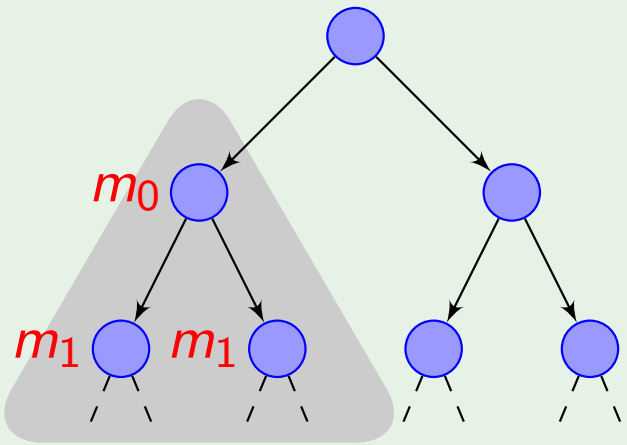
What is the exact semantics? [BGM16]

$$\exists\sigma.\ \mathbf{AG}(\text{assign}(A \to \sigma).\ \varphi)$$

Example

If σ prescribes to alternate between m_0 and m_1, starting with m_0:





History starts when strategy is selected.

History starts when strategy is assigned.

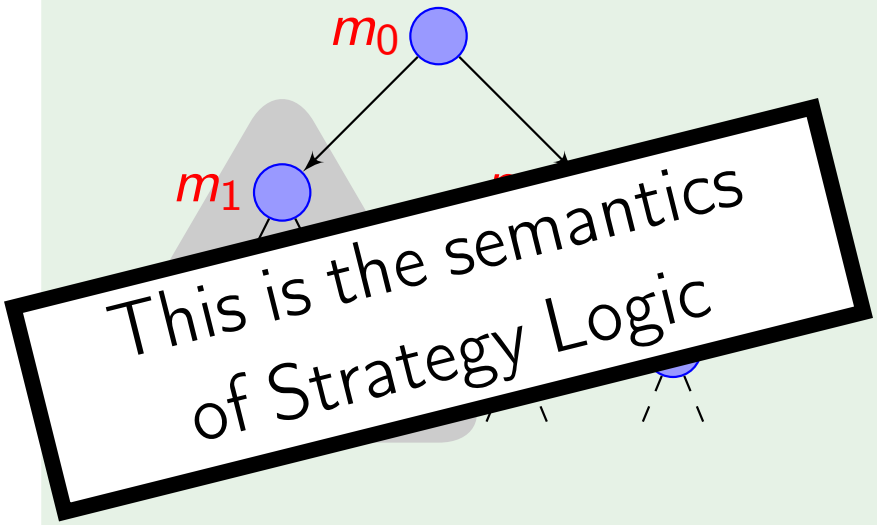
[BGM16] Bouyer, Gardy, Markey. On the semantics of SL. IPL, 2016.

What is the exact semantics? [BGM16]

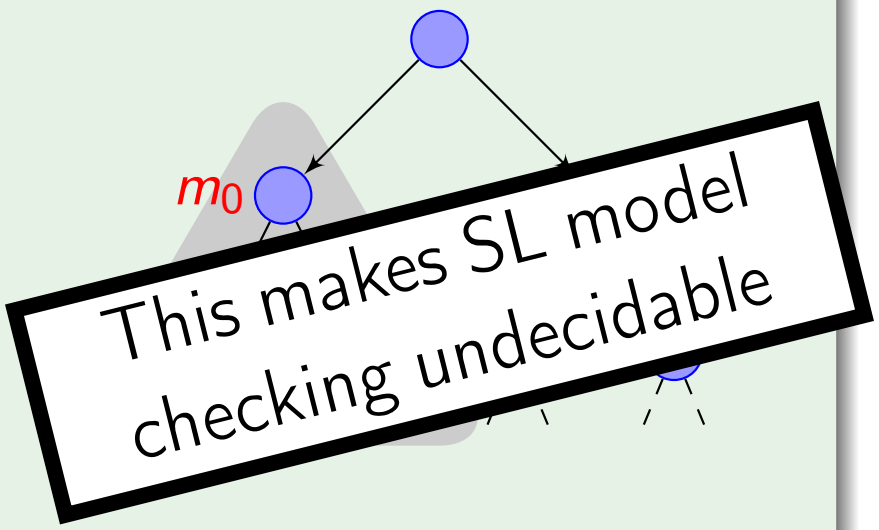
$$\exists\sigma.\ \mathbf{AG}(\text{assign}(A \to \sigma).\ \varphi)$$

Example

If σ prescribes to alternate between m_0 and m_1, starting with m_0:



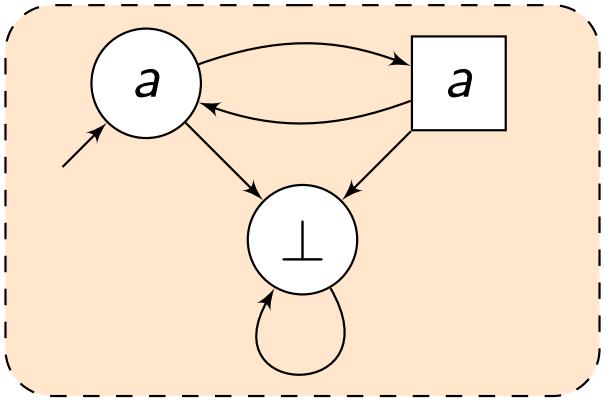
History starts when strategy is selected.



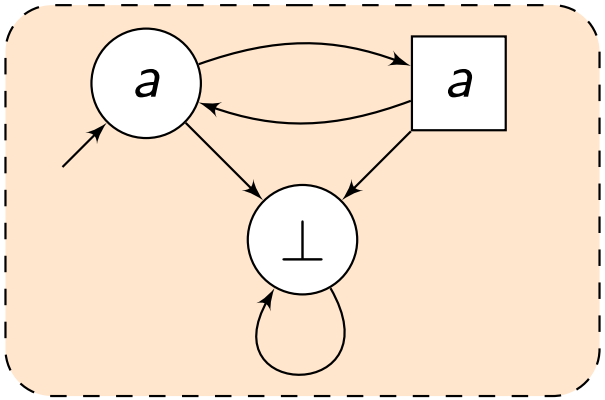
History starts when strategy is assigned.

[BGM16] Bouyer, Gardy, Markey. On the semantics of SL. IPL, 2016.

Encoding of a 2-counter machine

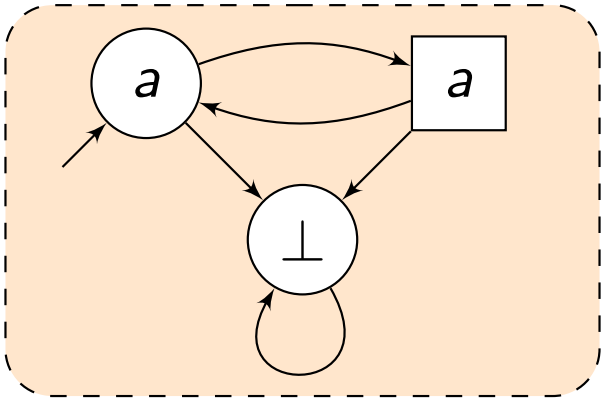


Encoding of a 2-counter machine



Strategy $\sigma_{\bigcirc}$ encodes counter value k if it plays k times to $\boxed{a}$ before playing to $\perp$ (similarly for $\sigma_{\square}$).

Encoding of a 2-counter machine



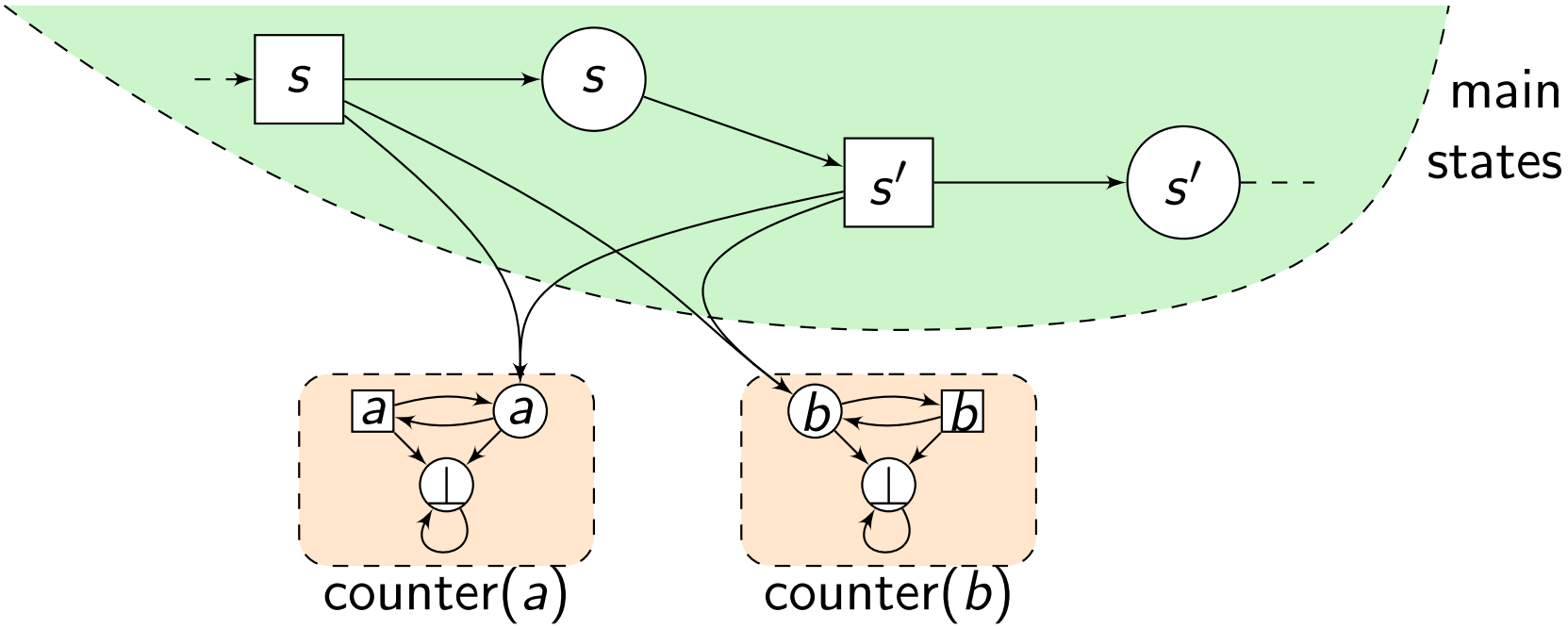
Strategy $\sigma_{\bigcirc}$ encodes counter value k if it plays k times to $\boxed{a}$ before playing to $\bot$ (similarly for $\sigma_{\square}$).

Lemma

Two strategies $\sigma_{\bigcirc}$ and $\sigma_{\square}$ encode the same counter value iff, when entering the game above and applying these strategies, it holds

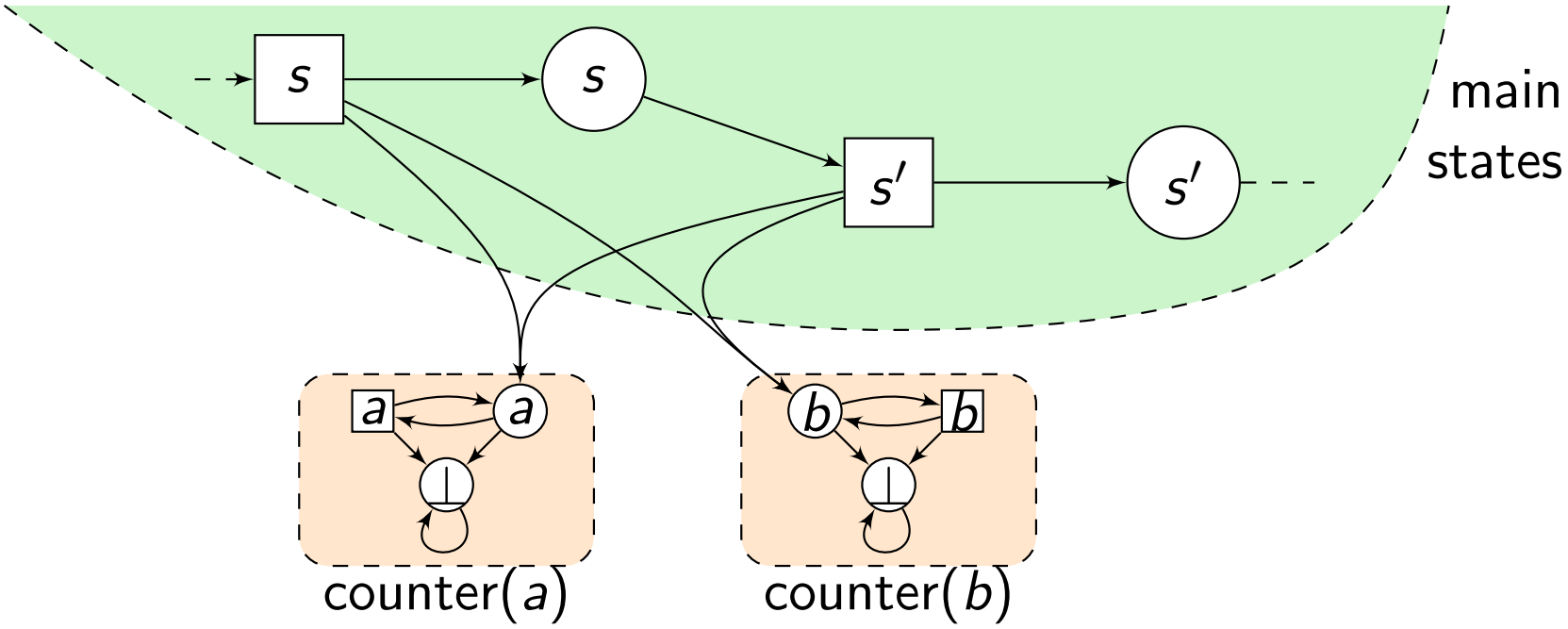
$$\mathbf{G}(\boxed{a} \Rightarrow \mathbf{X}\,a) \wedge \mathbf{F}\left[(a \wedge \mathbf{X}\,\bot) \wedge \neg\langle\cdot\bigcirc\cdot\rangle\,\neg\,\mathbf{X}\,\mathbf{X}\,\bot\right] \qquad (\varphi_{=})$$

Encoding of a 2-counter machine



- A strategy for ○ encodes a run of the 2-counter machine;

Encoding of a 2-counter machine



- A strategy for ○ encodes a run of the 2-counter machine;
- At each step, Player □ can check correct values of counters

$$\boxed{s} \Rightarrow \exists \sigma_{\square}.\ \mathbf{E}\,\mathbf{X}(a \wedge \text{assign}(\square \to \sigma_{\square}).\varphi_{=}) \wedge$$
$$\mathbf{E}\,\mathbf{X}\,\mathbf{E}\,\mathbf{X}\, s' \wedge \mathbf{E}\,\mathbf{X}(a \wedge \text{assign}(\square \to \sigma_{\square}).\varphi_{+1})$$

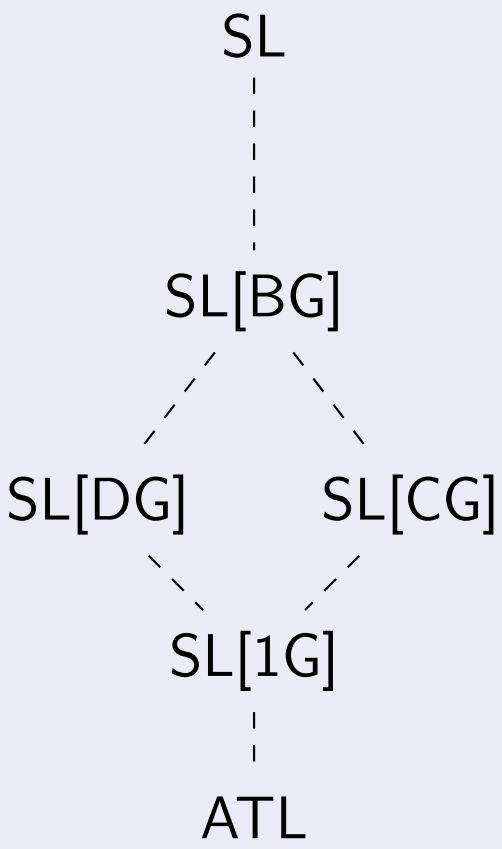
Outline of the presentation

1. Introduction
   - Computation-Tree Logic (CTL) and Quantified CTL
   - Alternating-time Temporal Logic (ATL)
2. Strategy Logic
   - Model checking Strategy Logic (SL)
   - Semantics of SL
     - When does history start?
     - What do strategies know about each other?
3. Conclusions and future works

Strategy dependences [MMPV11,GBM18]

Note

From this point on, we only consider formulas in SL[BG]:

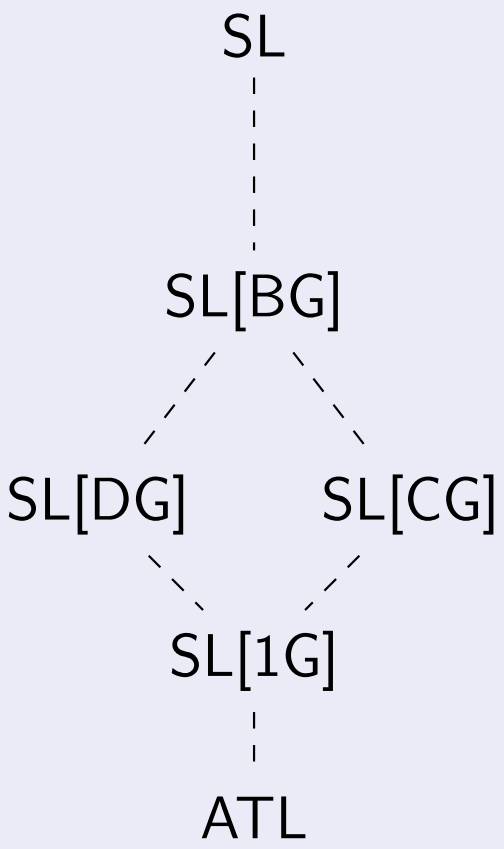


[MMPV11] Mogavero, Murano, Perelli, Vardi. Reasonning about strategies: ... J. ACM, 2011.
[GBM18] Gardy, Bouyer, Markey. Dependences in Strategy Logic. STACS, 2018.

Strategy dependences [MMPV11,GBM18]

Note

From this point on, we only consider formulas in SL[BG]:



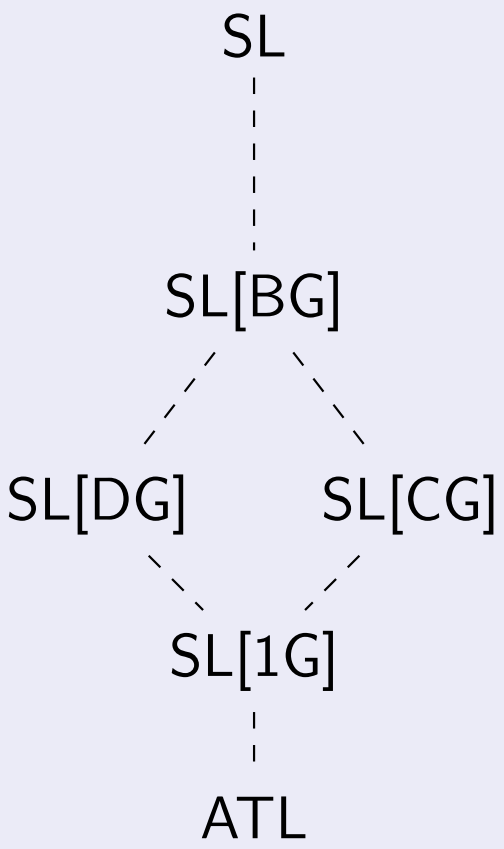
$$\underbrace{(Q_i\sigma_i).}_{\text{quantifier block}}\ \underbrace{\mathsf{assign}(A_j \mapsto \sigma_i).\ \varphi_{\mathsf{LTL}}}_{\text{goal}}$$

[MMPV11] Mogavero, Murano, Perelli, Vardi. Reasonning about strategies: ... J. ACM, 2011.
[GBM18] Gardy, Bouyer, Markey. Dependences in Strategy Logic. STACS, 2018.

Strategy dependences [MMPV11,GBM18]

Note

From this point on, we only consider formulas in SL[BG]:



disjunction/conjunction of goals

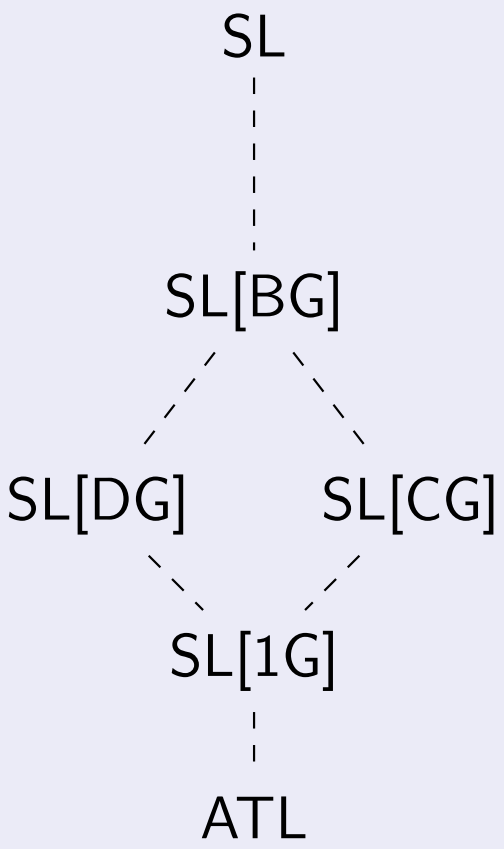
$$\underbrace{(Q_i\sigma_i).}_{\text{quantifier block}}\ \underbrace{\mathsf{assign}(A_j \mapsto \sigma_i).\ \varphi_{\mathsf{LTL}}}_{\text{goal}}$$

[MMPV11] Mogavero, Murano, Perelli, Vardi. Reasonning about strategies: ... J. ACM, 2011.
[GBM18] Gardy, Bouyer, Markey. Dependences in Strategy Logic. STACS, 2018.

Strategy dependences [MMPV11,GBM18]

Note

From this point on, we only consider formulas in SL[BG]:



boolean combination of goals

disjunction/conjunction of goals

$\underline{(Q_i\sigma_i).}\ \underline{\mathsf{assign}(A_j \mapsto \sigma_i).\ \varphi_{\mathsf{LTL}}}$

quantifier block

goal

[MMPV11] Mogavero, Murano, Perelli, Vardi. Reasonning about strategies: ... J. ACM, 2011.

[GBM18] Gardy, Bouyer, Markey. Dependences in Strategy Logic. STACS, 2018.

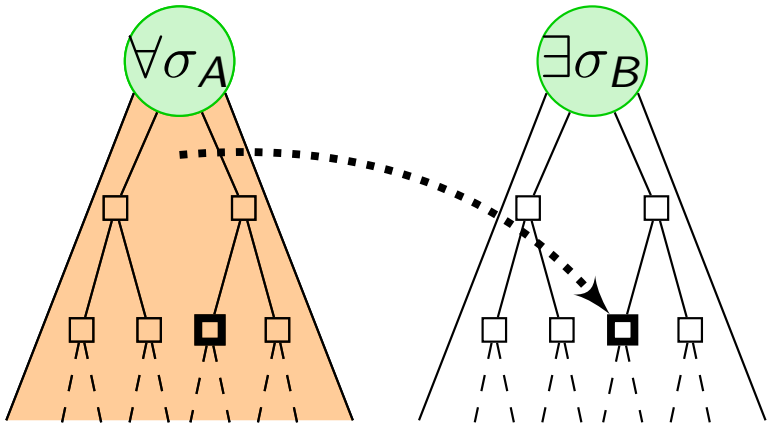
Strategy dependences [MMPV11,GBM18]

$$\forall \sigma_A \exists \sigma_B.\ \mathsf{assign}(A \mapsto \sigma_A; B \mapsto \sigma_B).\ \varphi$$

[MMPV11] Mogavero, Murano, Perelli, Vardi. Reasonning about strategies: ... J. ACM, 2011.
[GBM18] Gardy, Bouyer, Markey. Dependences in Strategy Logic. STACS, 2018.

Strategy dependences [MMPV11,GBM18]

$$\forall \sigma_A \exists \sigma_B.\ \mathrm{assign}(A \mapsto \sigma_A; B \mapsto \sigma_B).\ \varphi$$

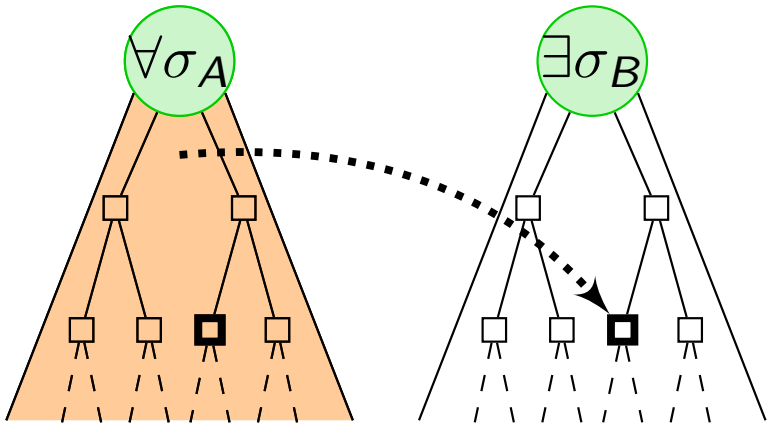


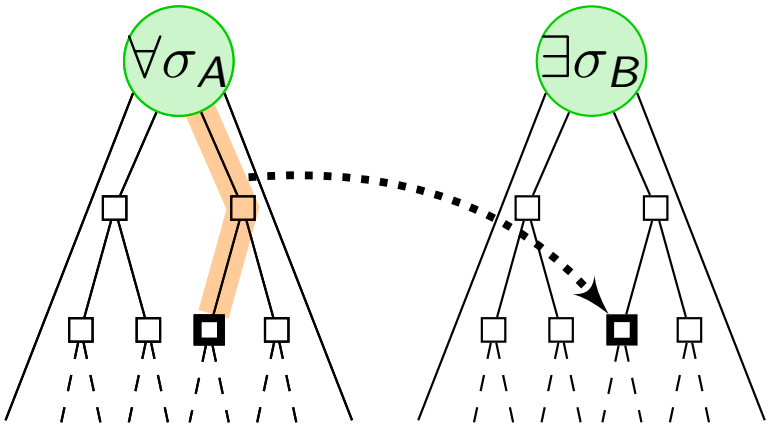
Strategy σ_B at node n may depend on the whole strategy σ_A.

[MMPV11] Mogavero, Murano, Perelli, Vardi. Reasonning about strategies: ... J. ACM, 2011.
[GBM18] Gardy, Bouyer, Markey. Dependences in Strategy Logic. STACS, 2018.

Strategy dependences [MMPV11,GBM18]

$$\forall \sigma_A \exists \sigma_B.\ \mathsf{assign}(A \mapsto \sigma_A; B \mapsto \sigma_B).\ \varphi$$





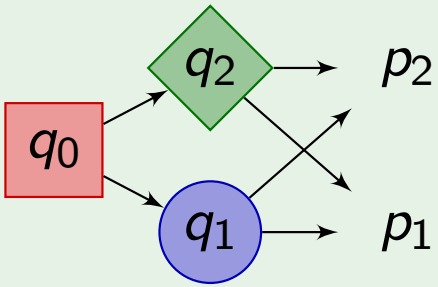
Strategy σ_B at node n may depend on the whole strategy σ_A.

Strategy σ_B at node n only depends on σ_A along the branch to n.

[MMPV11] Mogavero, Murano, Perelli, Vardi. Reasonning about strategies: ... J. ACM, 2011.
[GBM18] Gardy, Bouyer, Markey. Dependences in Strategy Logic. STACS, 2018.

Dependence maps (Skolemization)

Example

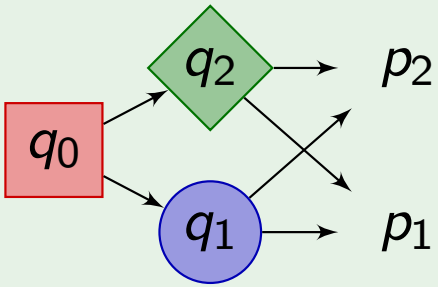


$$\varphi = \forall y.\exists z.\forall x_A.\forall x_B.$$

$$\bigvee \begin{cases} \text{assign}(\square \mapsto x_A; \bigcirc \mapsto y; \diamond \mapsto z).\ \mathbf{F}\ p_1 \\ \text{assign}(\square \mapsto x_B; \bigcirc \mapsto y; \diamond \mapsto z).\ \mathbf{F}\ p_2 \end{cases}$$

Dependence maps (Skolemization)

Example



$$\varphi = \forall y.\exists z.\forall x_A.\forall x_B.$$

$$\bigvee \begin{cases} \text{assign}(\square \mapsto x_A; \bigcirc \mapsto y; \diamond \mapsto z).\ \mathbf{F}\ p_1 \\ \text{assign}(\square \mapsto x_B; \bigcirc \mapsto y; \diamond \mapsto z).\ \mathbf{F}\ p_2 \end{cases}$$

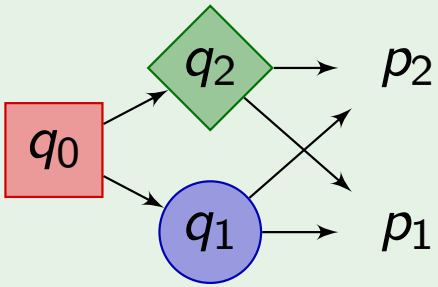
Dependence maps

Dependence maps select $\exists$-strategies from $\forall$-strategies:

$$\begin{array}{rcl} \theta\colon (\mathcal{V}^{\forall} \to \text{Strats}) & \to & (\mathcal{V}^{\exists} \to \text{Strats}) \\ w & \mapsto & \theta(w) \end{array}$$

Dependence maps (Skolemization)

Example



$$\varphi = \forall y.\exists z.\forall x_A.\forall x_B.$$

$$\bigvee \begin{cases} \text{assign}(\square \mapsto x_A; \bigcirc \mapsto y; \diamond \mapsto z).\ \mathbf{F}\ p_1 \\ \text{assign}(\square \mapsto x_B; \bigcirc \mapsto y; \diamond \mapsto z).\ \mathbf{F}\ p_2 \end{cases}$$

Dependence maps

Dependence maps select $\exists$-strategies from $\forall$-strategies:

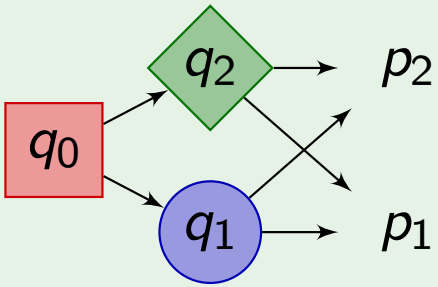
$$\begin{array}{rcl} \theta\colon (\mathcal{V}^{\forall} \to \text{Strats}) & \to & (\mathcal{V}^{\exists} \to \text{Strats}) \\ w & \mapsto & \theta(w) \end{array}$$

Then

$$q_0 \models (Q_i \sigma_i).\ \Phi \qquad \Leftrightarrow \qquad \exists \theta.\ \forall w.\ q_0 \models_{\theta(w) \cup w} \Phi.$$

Dependence maps (Skolemization)

Example



$$\varphi = \forall y.\exists z.\forall x_A.\forall x_B.$$

$$\bigvee \begin{cases} \text{assign}(\square \mapsto x_A; \bigcirc \mapsto y; \diamond \mapsto z).\ \mathbf{F}\ p_1 \\ \text{assign}(\square \mapsto x_B; \bigcirc \mapsto y; \diamond \mapsto z).\ \mathbf{F}\ p_2 \end{cases}$$

Dependence maps

Dependence maps select $\exists$-strategies from $\forall$-strategies:

$$\begin{array}{rcl} \theta\colon (\mathcal{V}^\forall \to \text{Strats}) & \to & (\mathcal{V}^\exists \to \text{Strats}) \\ w & \mapsto & \theta(w) \end{array}$$

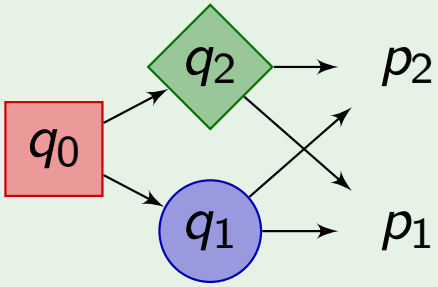
Then

$$q_0 \models (Q_i \sigma_i).\ \Phi \qquad \Leftrightarrow \qquad \exists \theta.\ \forall w.\ q_0 \models_{\theta(w) \cup w} \Phi.$$

$q_0 \not\models \varphi$ is (a priori) not equivalent to $q_0 \models \neg\varphi$.

Dependence maps (Skolemization)

Example

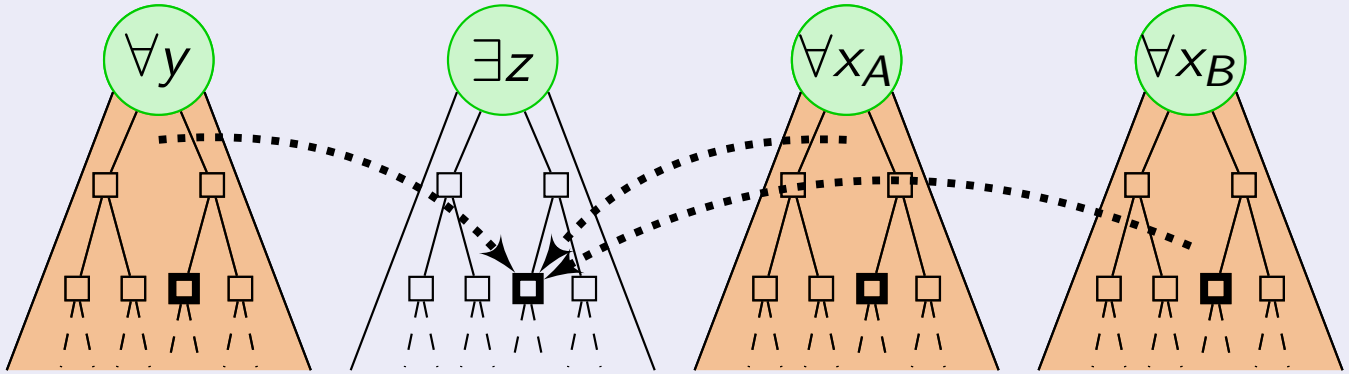


$$\varphi = \forall y.\exists z.\forall x_A.\forall x_B.$$

$$\bigvee \begin{cases} \text{assign}(\square \mapsto x_A; \bigcirc \mapsto y; \diamond \mapsto z).\ \mathbf{F}\ p_1 \\ \text{assign}(\square \mapsto x_B; \bigcirc \mapsto y; \diamond \mapsto z).\ \mathbf{F}\ p_2 \end{cases}$$

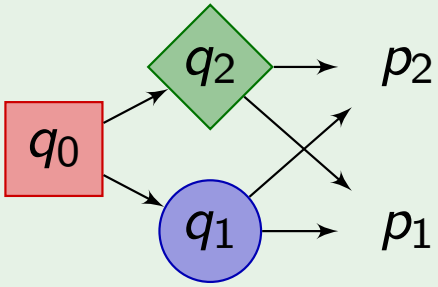
Dependence maps

- $\forall\exists$ SL semantics:
  - no further constraints: existentially-quantified variables may depend on all universally-quantified variables.



Dependence maps (Skolemization)

Example



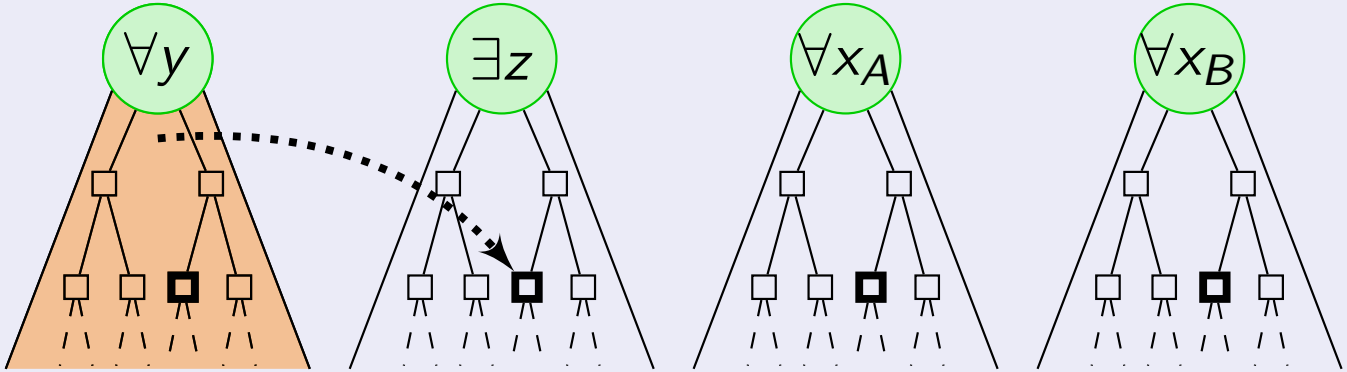
$$\varphi = \forall y.\exists z.\forall x_A.\forall x_B.$$

$$\bigvee \begin{cases} \text{assign}(\square \mapsto x_A; \bigcirc \mapsto y; \diamond \mapsto z).\ \mathbf{F}\ p_1 \\ \text{assign}(\square \mapsto x_B; \bigcirc \mapsto y; \diamond \mapsto z).\ \mathbf{F}\ p_2 \end{cases}$$

Dependence maps

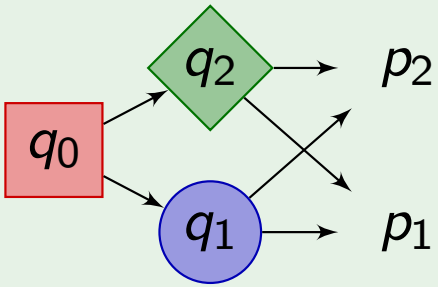
- classical SL semantics ($\models^C$):
  - depend on previously-quantified variables:

$$\text{if } w(y) = w'(y), \text{ then } \theta(w)(z) = \theta(w')(z)$$



Dependence maps (Skolemization)

Example



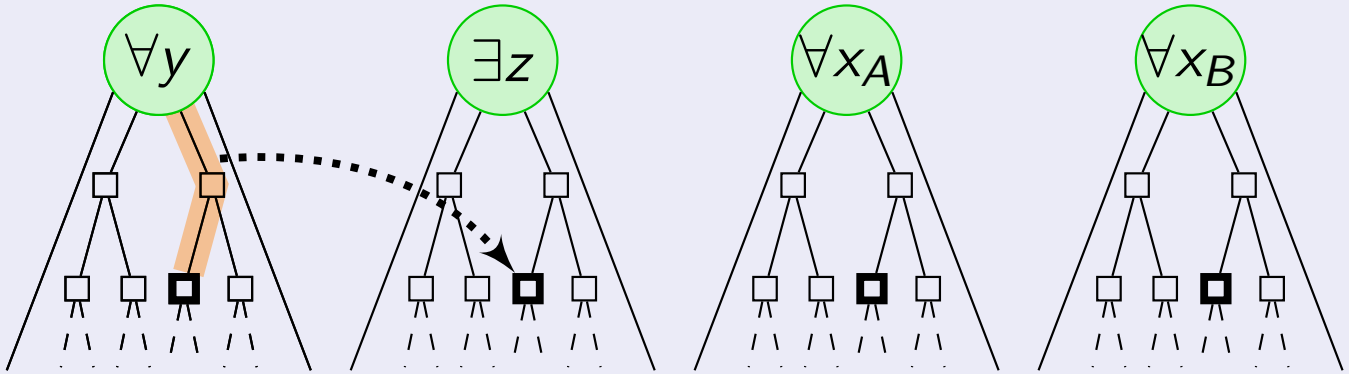
$$\varphi = \forall y.\exists z.\forall x_A.\forall x_B.$$

$$\bigvee \begin{cases} \text{assign}(\square \mapsto x_A; \bigcirc \mapsto y; \diamond \mapsto z).\ \mathbf{F}\, p_1 \\ \text{assign}(\square \mapsto x_B; \bigcirc \mapsto y; \diamond \mapsto z).\ \mathbf{F}\, p_2 \end{cases}$$

Dependence maps

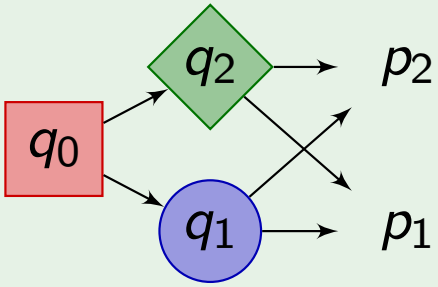
- elementary semantics ($\models^E$):
  - depend on previously-quantified variables along current prefix:

$$\text{for all } n, \text{ if } w(y)(m) = w'(y)(m) \text{ for all } m \preceq n, \text{ then } \theta(w)(z)(n) = \theta(w')(z)(n)$$



Dependence maps (Skolemization)

Example



$$\varphi = \forall y.\exists z.\forall x_A.\forall x_B.$$

$$\bigvee \begin{cases} \text{assign}(\square \mapsto x_A; \bigcirc \mapsto y; \diamond \mapsto z).\ \mathbf{F}\ p_1 \\ \text{assign}(\square \mapsto x_B; \bigcirc \mapsto y; \diamond \mapsto z).\ \mathbf{F}\ p_2 \end{cases}$$

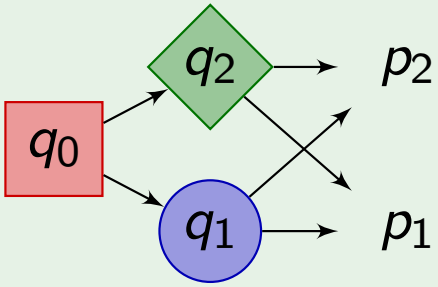
Theorem

For any $\varphi \in SL[BG]$:

$$q_0 \models^C \varphi \ \Leftrightarrow\ q_0 \models \varphi \qquad\qquad q_0 \models^E \varphi \ \Rightarrow\ q_0 \models^C \varphi$$

Dependence maps (Skolemization)

Example



$$\varphi = \forall y.\exists z.\forall x_A.\forall x_B.$$

$$\bigvee \begin{cases} \text{assign}(\square \mapsto x_A; \bigcirc \mapsto y; \diamond \mapsto z).\ \mathbf{F}\ p_1 \\ \text{assign}(\square \mapsto x_B; \bigcirc \mapsto y; \diamond \mapsto z).\ \mathbf{F}\ p_2 \end{cases}$$

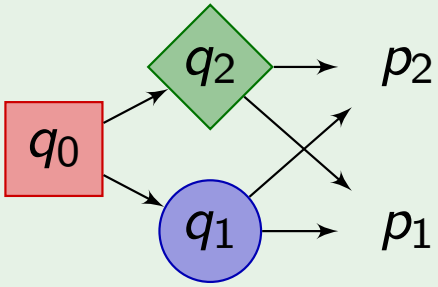
Theorem

For any $\varphi \in SL[BG]$:

$$q_0 \models^C \varphi \ \Leftrightarrow\ q_0 \models \varphi \qquad\qquad q_0 \models^E \varphi \ \Rightarrow\ q_0 \models^C \varphi$$

Dependence maps (Skolemization)

Example



$$\varphi = \forall y.\exists z.\forall x_A.\forall x_B.$$

$$\bigvee \begin{cases} \text{assign}(\square \mapsto x_A; \bigcirc \mapsto y; \diamond \mapsto z).\ \mathbf{F}\ p_1 \\ \text{assign}(\square \mapsto x_B; \bigcirc \mapsto y; \diamond \mapsto z).\ \mathbf{F}\ p_2 \end{cases}$$

- $q_0 \models^C \varphi$: z plays to p_1 iff y plays to p_1.

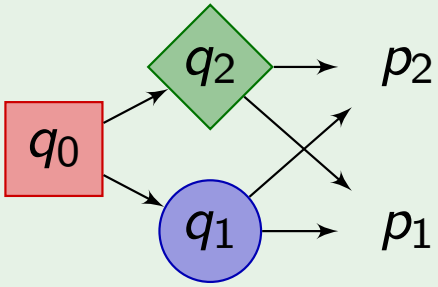
Theorem

For any $\varphi \in SL[BG]$:

$$q_0 \models^C \varphi \ \Leftrightarrow\ q_0 \models \varphi \qquad\qquad q_0 \models^E \varphi \ \Rightarrow\ q_0 \models^C \varphi$$

Dependence maps (Skolemization)

Example



$$\varphi = \forall y.\exists z.\forall x_A.\forall x_B.$$

$$\bigvee \begin{cases} \text{assign}(\square \mapsto x_A; \bigcirc \mapsto y; \diamond \mapsto z).\ \mathbf{F}\ p_1 \\ \text{assign}(\square \mapsto x_B; \bigcirc \mapsto y; \diamond \mapsto z).\ \mathbf{F}\ p_2 \end{cases}$$

- $q_0 \models^C \varphi$: z plays to p_1 iff y plays to p_1.
- $q_0 \not\models^E \varphi$: y only plays in q_1, so z will not depend on y.

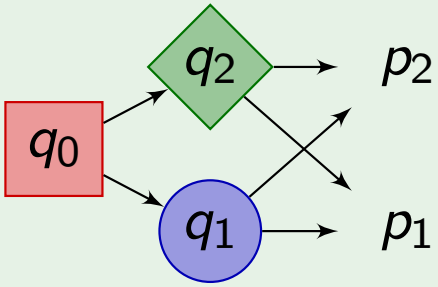
Theorem

For any $\varphi \in SL[BG]$:

$$q_0 \models^C \varphi \ \Leftrightarrow\ q_0 \models \varphi \qquad\qquad q_0 \models^E \varphi \ \Rightarrow\ q_0 \models^C \varphi$$

Dependence maps (Skolemization)

Example



$$\varphi = \forall y.\exists z.\forall x_A.\forall x_B.$$

$$\bigvee \begin{cases} \text{assign}(\square \mapsto x_A; \bigcirc \mapsto y; \diamond \mapsto z).\ \mathbf{F}\, p_1 \\ \text{assign}(\square \mapsto x_B; \bigcirc \mapsto y; \diamond \mapsto z).\ \mathbf{F}\, p_2 \end{cases}$$

- $q_0 \models^C \varphi$: z plays to p_1 iff y plays to p_1.
- $q_0 \not\models^E \varphi$: y only plays in q_1, so z will not depend on y.
- $q_0 \not\models^E \neg\varphi$: because otherwise $q_0 \models^C \neg\varphi$...

Theorem

For any $\varphi \in SL[BG]$:

$$q_0 \models^C \varphi \;\Leftrightarrow\; q_0 \models \varphi \qquad\qquad q_0 \models^E \varphi \;\Rightarrow\; q_0 \models^C \varphi$$

Elementary semantics

Theorem

For any $\varphi \in SL[1G]$, $\quad q_0 \models^E \varphi$ *iff* $q_0 \not\models^E \neg\varphi$.

Elementary semantics

Theorem

For any $\varphi \in SL[1G]$, $\quad q_0 \models^E \varphi$ *iff* $q_0 \not\models^E \neg\varphi$.

Proof

From previous theorem:

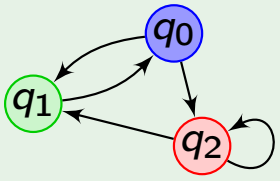
$$q_0 \models^E \varphi \;\Rightarrow\; q_0 \models^C \varphi \;\Rightarrow\; q_0 \not\models^C \neg\varphi \;\Rightarrow\; q_0 \not\models^E \neg\varphi.$$

Elementary semantics

Theorem

For any $\varphi \in SL[1G]$, $\quad q_0 \models^E \varphi$ *iff* $q_0 \not\models^E \neg\varphi$.

Proof



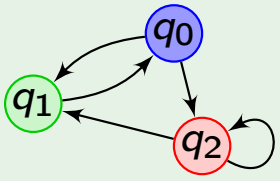
$$\models^E \quad \forall x.\ \exists y.\ \mathsf{assign}(A \mapsto x, B \mapsto y).\ \ \psi$$

Elementary semantics

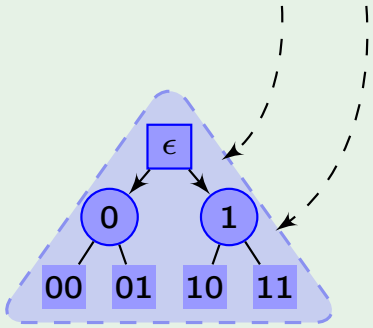
Theorem

For any $\varphi \in SL[1G]$, $\quad q_0 \models^E \varphi$ *iff* $q_0 \not\models^E \neg\varphi$.

Proof



$$\models^E \quad \forall x.\ \exists y.\ \text{assign}(A \mapsto x, B \mapsto y).\ \ \psi$$

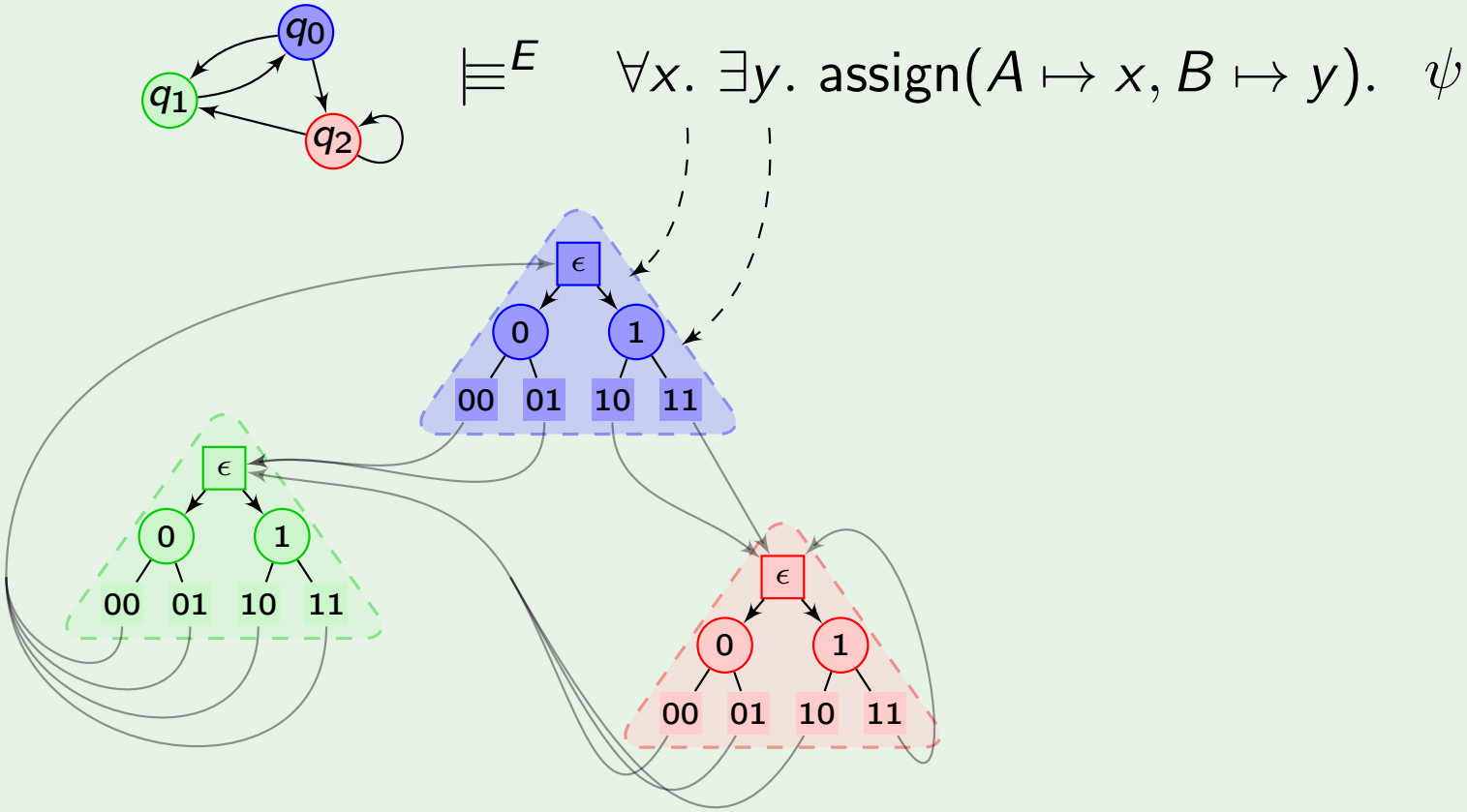


Elementary semantics

Theorem

For any $\varphi \in SL[1G]$, $\quad q_0 \models^E \varphi$ *iff* $q_0 \not\models^E \neg\varphi$.

Proof

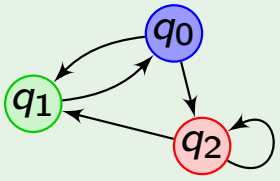


Elementary semantics

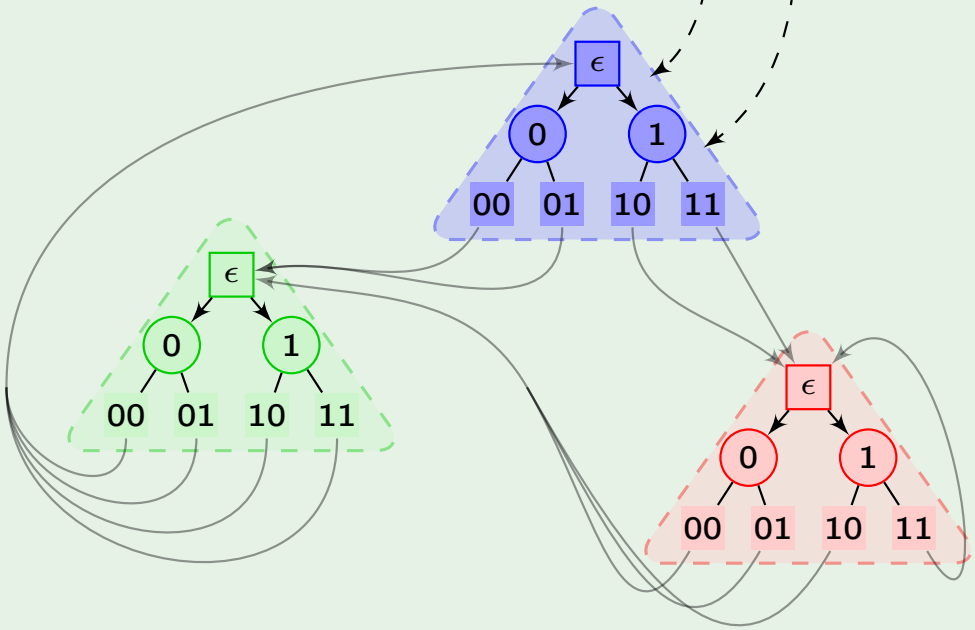
Theorem

For any $\varphi \in SL[1G]$, $\quad q_0 \models^E \varphi$ *iff* $q_0 \not\models^E \neg\varphi$.

Proof



$$\models^E \quad \forall x.\ \exists y.\ \text{assign}(A \mapsto x, B \mapsto y).\ \ \psi$$



$\times \qquad \mathcal{A}_\psi$

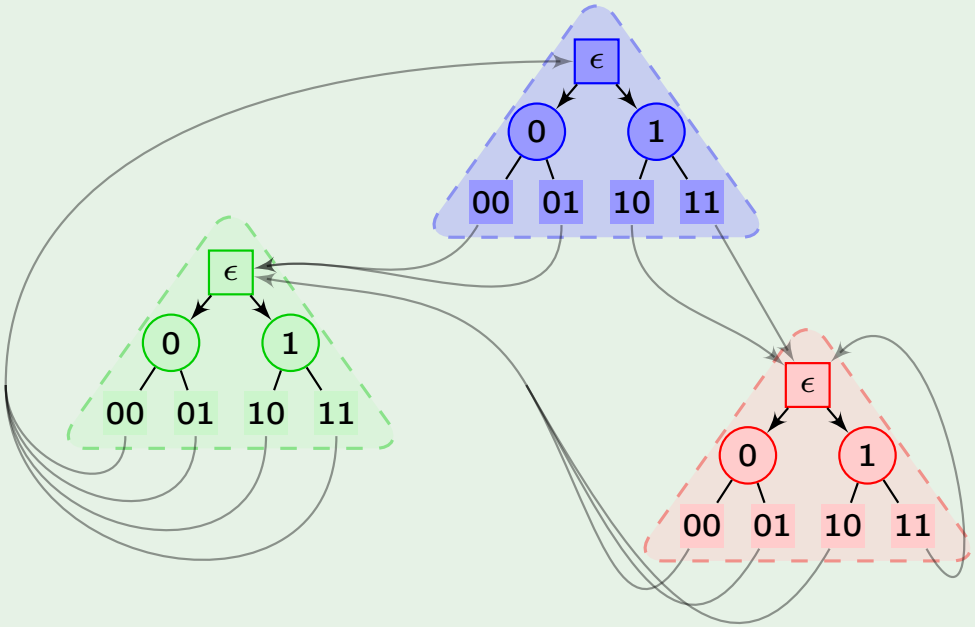
deterministic parity automaton

Elementary semantics

Theorem

For any $\varphi \in SL[1G]$, $\quad q_0 \models^E \varphi$ *iff* $q_0 \not\models^E \neg\varphi$.

Proof



$\times \qquad \mathcal{A}_\psi$

deterministic parity automaton

Player $P_\exists$ wins $\Rightarrow$ $q_0 \models^E \varphi$; $\qquad$ Player $P_\forall$ wins $\Rightarrow$ $q_0 \models^E \neg\varphi$

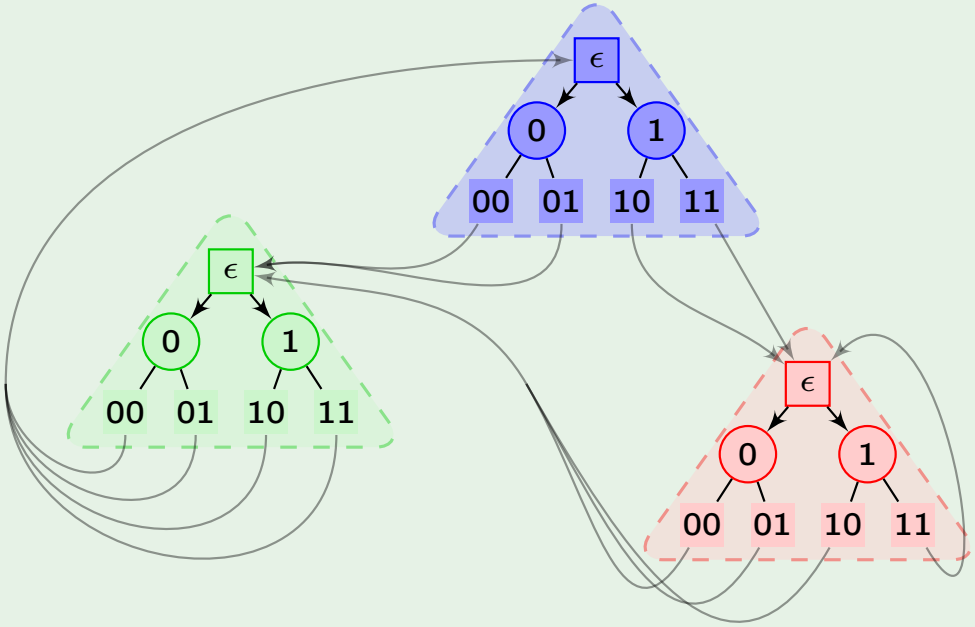
Result follows by determinacy of turn-based parity games.

Elementary semantics

Theorem

For any $\varphi \in SL[1G]$, $\quad q_0 \models^E \varphi$ *iff* $q_0 \not\models^E \neg\varphi$.

Proof



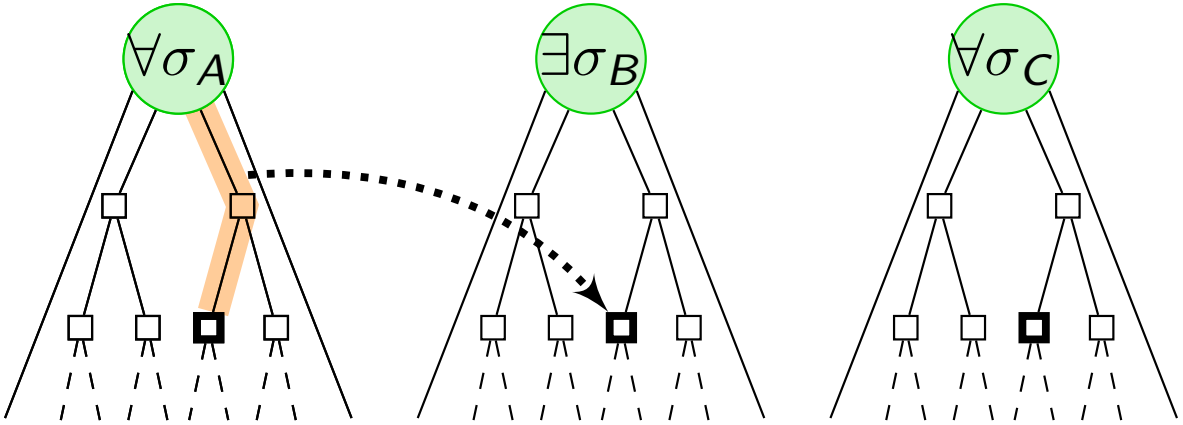
$\times \qquad \mathcal{A}_\psi$

deterministic parity automaton

Theorem

Model checking SL[1G] with $\models^E$ *is 2*EXPTIME*-complete.*

Beyond elementary dependences [GBM18]

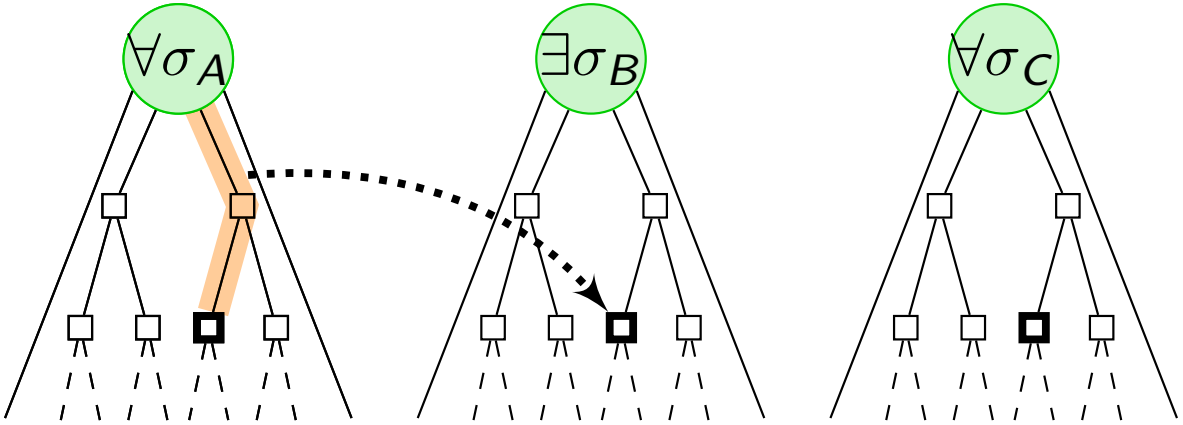


Elementary semantics

Strategy σ_B at node n only depends on σ_A along the branch to n.

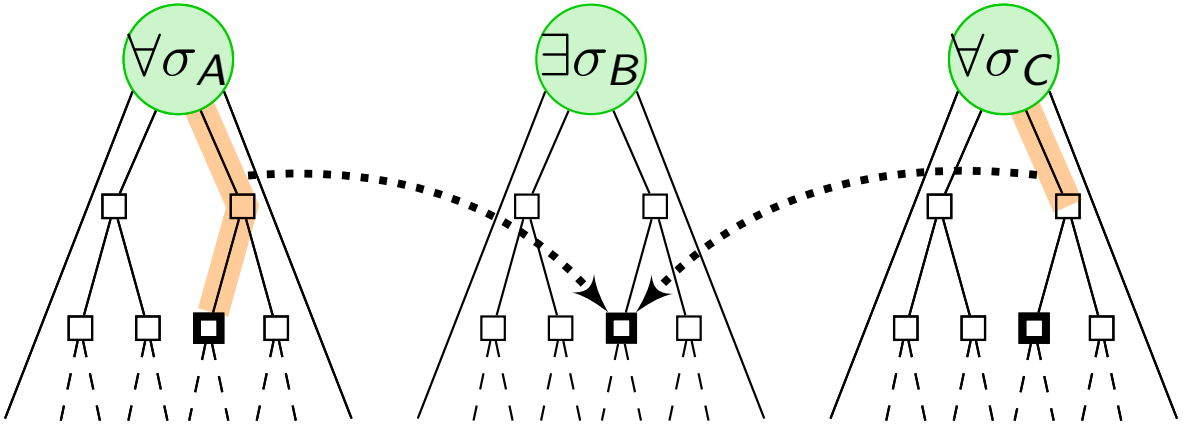
[GBM18] Gardy, Bouyer, Markey. Dependences in Strategy Logic. STACS, 2018.

Beyond elementary dependences [GBM18]



Elementary semantics

Strategy σ_B at node n only depends on σ_A along the branch to n.

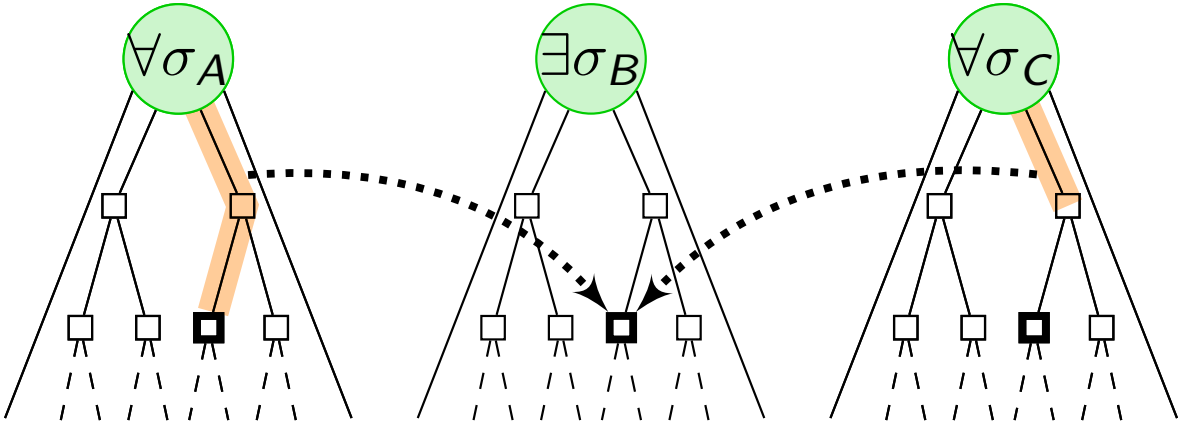


Timeline semantics

Strategy σ_B also depends on σ_C along strict prefixes of n.

[GBM18] Gardy, Bouyer, Markey. Dependences in Strategy Logic. STACS, 2018.

Beyond elementary dependences [GBM18]



Timeline semantics

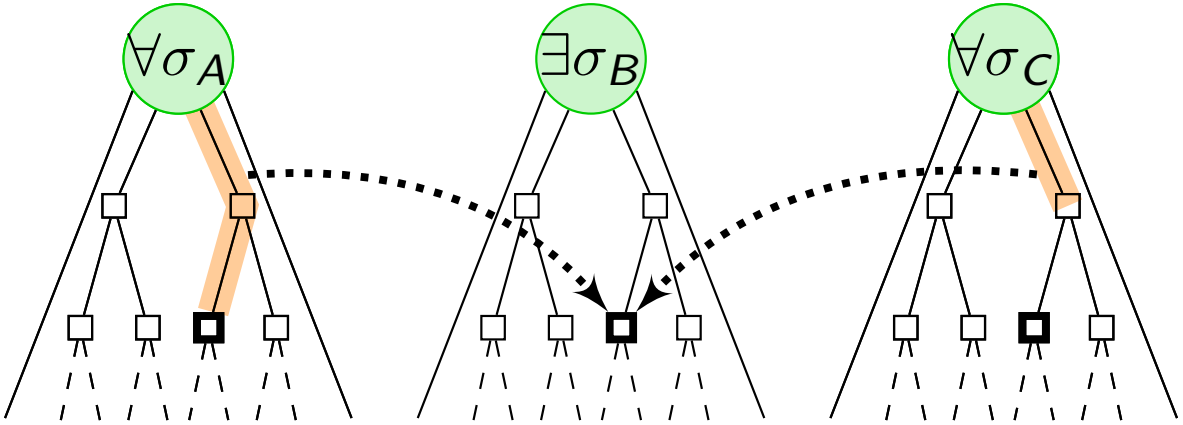
Strategy σ_B also depends on σ_C along strict prefixes of n.

Proposition

For any $\varphi \in SL[BG]$, $\quad q_0 \models^E \varphi \;\Rightarrow\; q_0 \models^T \varphi$

[GBM18] Gardy, Bouyer, Markey. Dependences in Strategy Logic. STACS, 2018.

Beyond elementary dependences [GBM18]



Timeline semantics

Strategy σ_B also depends on σ_C along strict prefixes of n.

Proposition

For any $\varphi \in SL[BG]$, $\quad q_0 \models^E \varphi \;\Rightarrow\; q_0 \models^T \varphi$

Proposition

For any $\varphi \in SL[1G]$,

$$q_0 \models^E \varphi \iff q_0 \models^T \varphi$$
$$q_0 \models^E \neg\varphi \iff q_0 \not\models^T \varphi$$

[GBM18] Gardy, Bouyer, Markey. Dependences in Strategy Logic. STACS, 2018.

Beyond elementary dependences

Theorem

For a large fragment of SL[BG] (containing SL[CG] and SL[DG]),

$$q_0 \models^T \varphi \;\Leftrightarrow\; q_0 \not\models^T \neg\varphi.$$

Beyond elementary dependences

Theorem

For a large fragment of SL[BG] (containing SL[CG] and SL[DG]),

$$q_0 \models^T \varphi \;\Leftrightarrow\; q_0 \not\models^T \neg\varphi.$$

Semi-stable sets

For $s, f, g \in \{0,1\}^n$, we let

$$\begin{aligned} \bar{s} &\colon i \mapsto 1 - s(i) \\ f \curlyvee g &\colon \max\{f(i), g(i)\} \\ f \curlywedge g &\colon \min\{f(i), g(i)\} \end{aligned}$$

Beyond elementary dependences

Theorem

For a large fragment of SL[BG] (containing SL[CG] and SL[DG]),

$$q_0 \models^T \varphi \;\Leftrightarrow\; q_0 \not\models^T \neg\varphi.$$

Semi-stable sets

$F \subseteq \{0,1\}^n$ is semi-stable if

$$\forall f, g \in F.\ \forall s \in \{0,1\}^n.\ (f \curlywedge s) \curlyvee (g \curlywedge \bar{s}) \in F \quad \text{or} \quad (f \curlywedge \bar{s}) \curlyvee (g \curlywedge s) \in F$$

Beyond elementary dependences

Theorem

For a large fragment of SL[BG] (containing SL[CG] and SL[DG]),

$$q_0 \models^T \varphi \;\Leftrightarrow\; q_0 \not\models^T \neg\varphi.$$

Semi-stable sets

- any singleton is semi-stable;
- $\{(1,1),(0,0)\}$ is not semi-stable in $\{0,1\}^2$.
- The complement of a semi-stable set is semi-stable.

Beyond elementary dependences

Theorem

For a large fragment of SL[BG] (containing SL[CG] and SL[DG]),

$$q_0 \models^T \varphi \;\Leftrightarrow\; q_0 \not\models^T \neg\varphi.$$

Proof (sketch of proof)

- SL[EG]: fragment of SL[BG]:

$$(Q_i\sigma_i)_i.\ \text{Boolean-Combination}(G_1, \ldots, G_n)$$

where Boolean Combination defines a semi-stable set.

Beyond elementary dependences

Theorem

For a large fragment of SL[BG] (containing SL[CG] and SL[DG]),

$$q_0 \models^T \varphi \;\Leftrightarrow\; q_0 \not\models^T \neg\varphi.$$

Proof (sketch of proof)

- SL[EG]: fragment of SL[BG]:

$$(Q_i\sigma_i)_i.\ \text{Boolean-Combination}(G_1, \ldots, G_n)$$

where Boolean Combination defines a semi-stable set.

Proposition

SL[EG] contains SL[CG] and SL[DG], and is closed under negation.

Beyond elementary dependences

Theorem

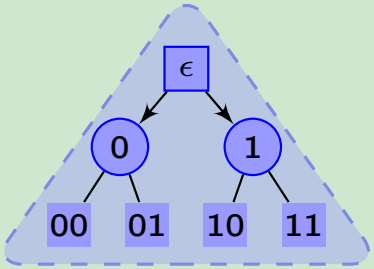
For a large fragment of SL[BG] (containing SL[CG] and SL[DG]),

$$q_0 \models^T \varphi \;\Leftrightarrow\; q_0 \not\models^T \neg\varphi.$$

Proof (sketch of proof)

- extending the construction for SL[1G] with several goals:

several goals $\Rightarrow$ possibly several outcomes



$$\forall x.\ \exists y.\ \left[\begin{array}{c} \mathrm{assign}(A \mapsto x; B \mapsto y).\ \varphi_1 \\ \Longleftrightarrow \\ \mathrm{assign}(A \mapsto y; B \mapsto x).\ \varphi_2 \end{array}\right]$$

Beyond elementary dependences

Theorem

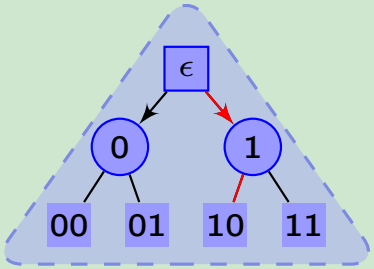
For a large fragment of SL[BG] (containing SL[CG] and SL[DG]),

$$q_0 \models^T \varphi \;\Leftrightarrow\; q_0 \not\models^T \neg\varphi.$$

Proof (sketch of proof)

- extending the construction for SL[1G] with several goals:

several goals $\Rightarrow$ possibly several outcomes



$$\forall x.\ \exists y.\ \left[\begin{array}{c} \mathrm{assign}(A \mapsto x; B \mapsto y).\ \varphi_1 \\ \Longleftrightarrow \\ \mathrm{assign}(A \mapsto y; B \mapsto x).\ \varphi_2 \end{array}\right]$$

Beyond elementary dependences

Theorem

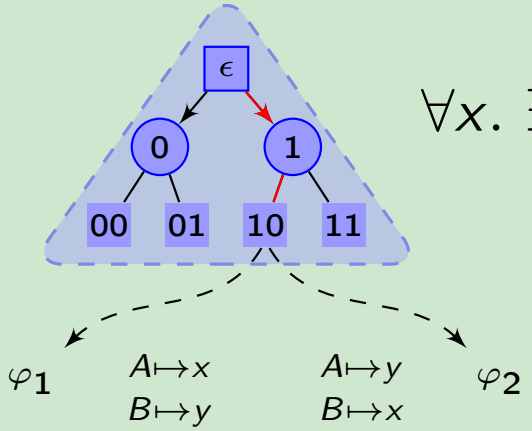
For a large fragment of SL[BG] (containing SL[CG] and SL[DG]),

$$q_0 \models^T \varphi \;\Leftrightarrow\; q_0 \not\models^T \neg\varphi.$$

Proof (sketch of proof)

- extending the construction for SL[1G] with several goals:

several goals $\Rightarrow$ possibly several outcomes



$$\forall x.\ \exists y.\ \left[\begin{array}{c} \text{assign}(A \mapsto x; B \mapsto y).\ \varphi_1 \\ \Longleftrightarrow \\ \text{assign}(A \mapsto y; B \mapsto x).\ \varphi_2 \end{array}\right]$$

Beyond elementary dependences

Theorem

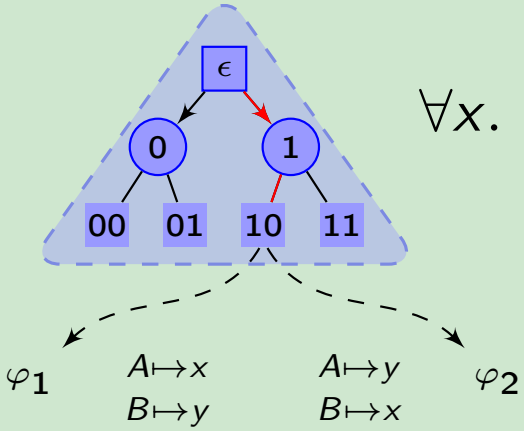
For a large fragment of SL[BG] (containing SL[CG] and SL[DG]),

$$q_0 \models^T \varphi \;\Leftrightarrow\; q_0 \not\models^T \neg\varphi.$$

Proof (sketch of proof)

- extending the construction for SL[1G] with several goals:

several goals $\Rightarrow$ possibly several outcomes



$$\forall x.\ \exists y.\ \left[\begin{array}{c} \text{assign}(A \mapsto x; B \mapsto y).\ \varphi_1 \\ \Longleftrightarrow \\ \text{assign}(A \mapsto y; B \mapsto x).\ \varphi_2 \end{array}\right]$$

... but this formula is not in SL[EG].

Beyond elementary dependences

Theorem

For a large fragment of SL[BG] (containing SL[CG] and SL[DG]),

$$q_0 \models^T \varphi \;\Leftrightarrow\; q_0 \not\models^T \neg\varphi.$$

Proof (sketch of proof)

- solution for SL[EG]:
  - optimal behaviour along one outcome is independent of what happens along other outcomes;
  - inductively compute optimal sets of goals to be fulfilled depending on the situation.

Beyond elementary dependences

Theorem

For a large fragment of SL[BG] (containing SL[CG] and SL[DG]),

$$q_0 \models^T \varphi \;\Leftrightarrow\; q_0 \not\models^T \neg\varphi.$$

Proof (sketch of proof)

- solution for SL[EG]:
  - optimal behaviour along one outcome is independent of what happens along other outcomes;
  - inductively compute optimal sets of goals to be fulfilled depending on the situation.

Theorem

*Model checking SL[EG] is 2*EXPTIME*-complete (for $\models^T$).*

Outline of the presentation

1. Introduction
   - Computation-Tree Logic (CTL) and Quantified CTL
   - Alternating-time Temporal Logic (ATL)
2. Strategy Logic
   - Model checking Strategy Logic (SL)
   - Semantics of SL
     - When does history start?
     - What do strategies know about each other?
3. Conclusions and future works

Conclusions and future works

Conclusions

- Strategy Logic is a very powerful logic for specifying properties of games on graphs;
- Understanding its exact semantics is difficult:
  $\rightsquigarrow$ does it correspond to what users need/expect?

Conclusions and future works

Conclusions

- Strategy Logic is a very powerful logic for specifying properties of games on graphs;
- Understanding its exact semantics is difficult:
  $\rightsquigarrow$ does it correspond to what users need/expect?

Future directions

- direct reduction to turn-based parity games for SL[EG];
- tree-automata approach for timeline semantics;
- model checking SL[BG] for $\models^T$; and beyond SL[BG]?

Conclusions and future works

Conclusions

- Strategy Logic is a very powerful logic for specifying properties of games on graphs;
- Understanding its exact semantics is difficult:
  $\rightsquigarrow$ does it correspond to what users need/expect?

Future directions

- direct reduction to turn-based parity games for SL[EG];
- tree-automata approach for timeline semantics;
- model checking SL[BG] for $\models^T$; and beyond SL[BG]?
- better understanding of semantics of SL;
- randomized strategies;
- extensions with quantities (e.g. time, energy, ...).