Model Checking Strategic Ability

Why, What, and Especially: How?

Wojtek Jamroga
University of Luxembourg & Polish Academy of Sciences

Politechnika Warszawska, 13/03/2023

Outline

Outline

Specification and Verification of Strategic Ability

- Many important properties are based on **strategic ability**
- Functionality ≈ ability of authorized users to complete tasks
- Security ≈ inability of unauthorized users to complete tasks
- Relevant for causality, responsibility, blameworthiness, etc.

Specification and Verification of Strategic Ability

- Many important properties are based on **strategic ability**
- Functionality $\approx$ ability of authorized users to complete tasks
- Security $\approx$ inability of unauthorized users to complete tasks
- Relevant for causality, responsibility, blameworthiness, etc.

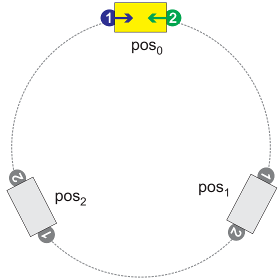
- One can try to formalize such properties in modal logics of strategic ability, such as ATL or Strategy Logic
- ...and verify them by model checking

Outline

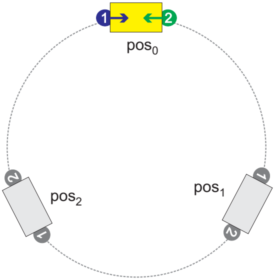
Models of Multi-Agent Systems

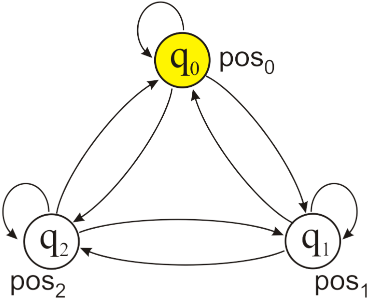
- How to model a distributed system? $\leadsto$ **transition graph**
- Nodes represent states of the system (or situations)
- Arrows correspond to changes of state

Example: Robots and Carriage

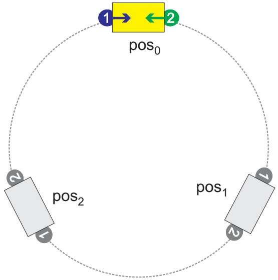


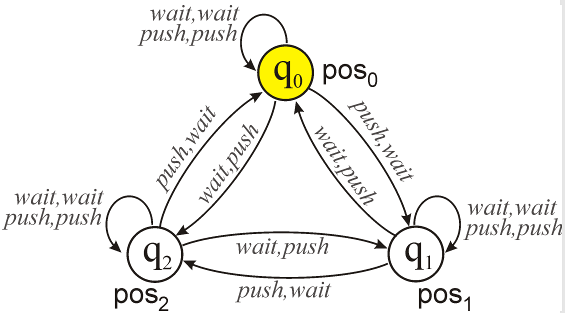
Example: Robots and Carriage



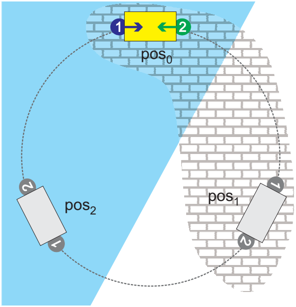


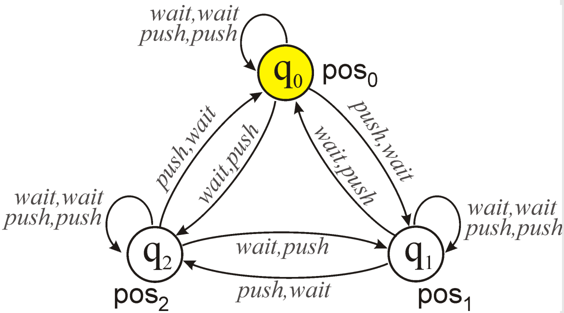
Example: Robots and Carriage



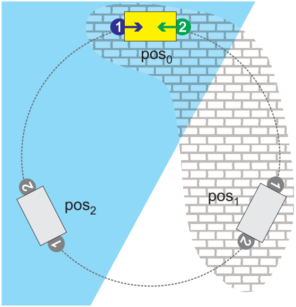


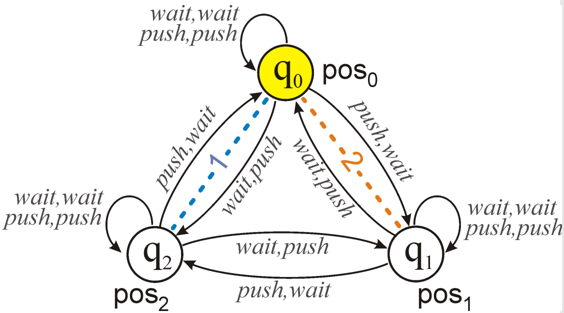
Example: Robots and Carriage





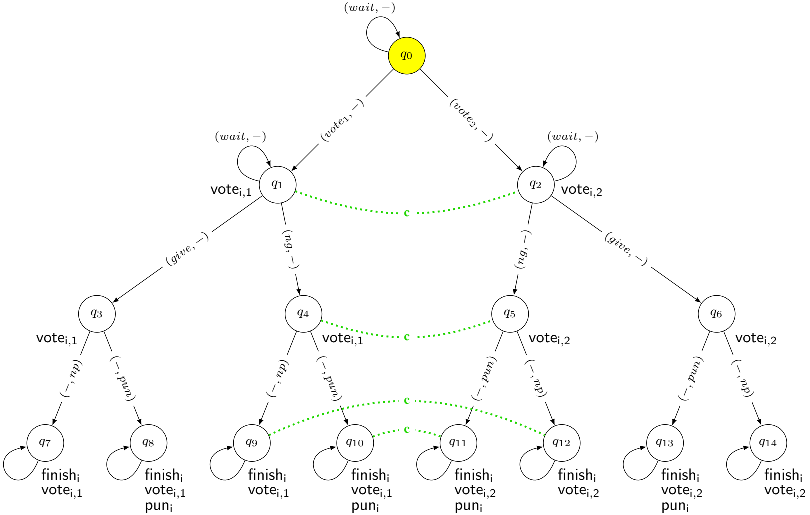
Example: Robots and Carriage



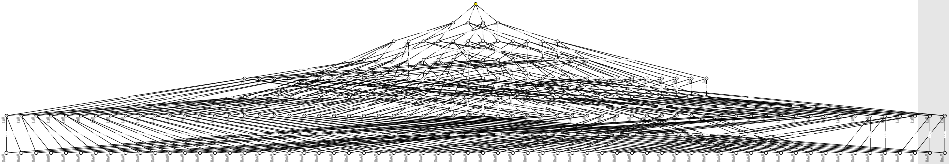


Example: Voting and Coercion

Example: Voting and Coercion



Example: Voting and Coercion



Outline

Logical Specification of Strategic Abilities

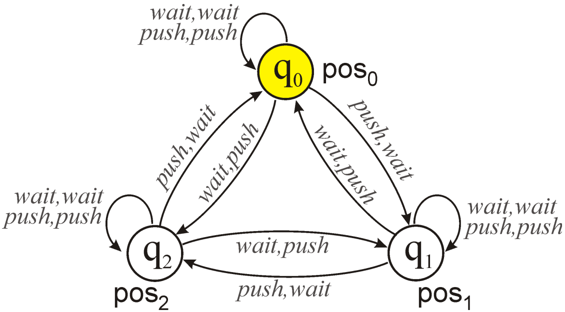
ATL: What Agents Can Achieve

- ATL: Alternating-time Temporal Logic [Alur et al. 1997-2002]
- Temporal logic meets game theory
- Main idea: cooperation modalities

$\langle\langle A \rangle\rangle \Phi$: coalition A has a collective strategy to enforce Φ

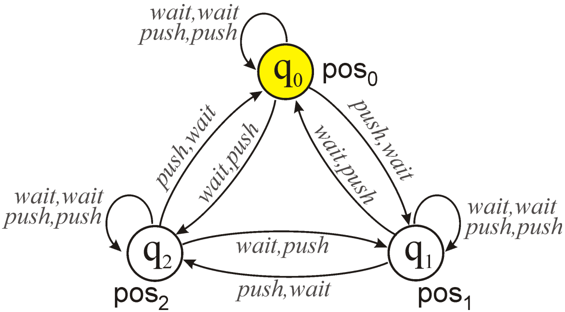
$\leadsto$ Φ can include temporal operators: $\bigcirc$ (next), $\Diamond$ (sometime in the future), $\Box$ (always in the future), U (strong until)

Example: Robots and Carriage



$\langle\langle 1 \rangle\rangle \Diamond \mathsf{pos}_1$

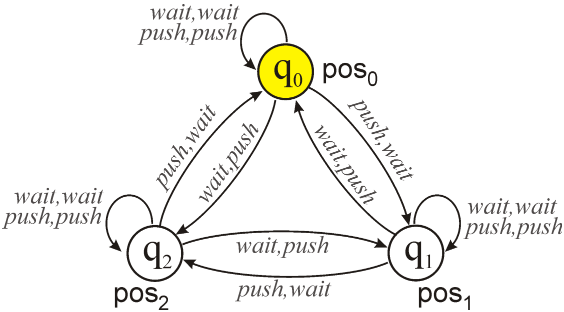
Example: Robots and Carriage



$\langle\langle 1 \rangle\rangle \Diamond \mathsf{pos}_1$

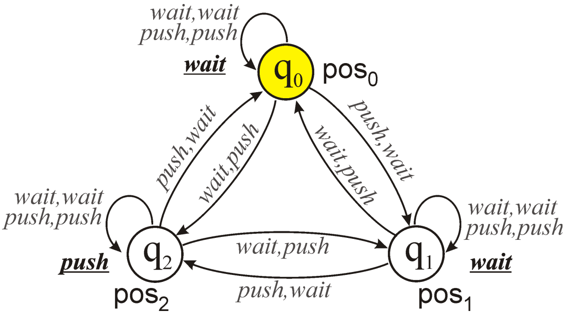
No!

Example: Robots and Carriage



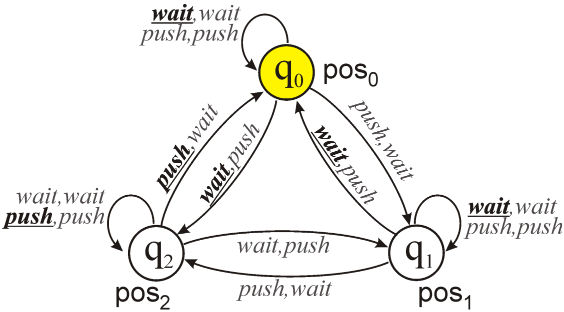
$\langle\langle 1 \rangle\rangle \Box \neg \mathsf{pos}_1$

Example: Robots and Carriage



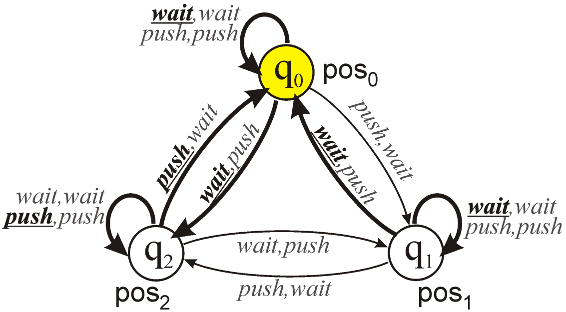
$\langle\langle 1 \rangle\rangle \Box \neg \mathsf{pos}_1$

Example: Robots and Carriage



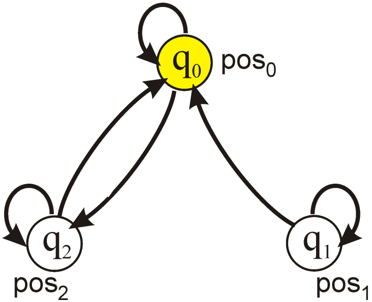
$\langle\langle 1 \rangle\rangle \Box \neg \mathsf{pos}_1$

Example: Robots and Carriage



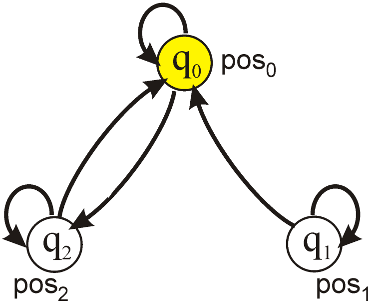
$\langle\langle 1 \rangle\rangle \Box \neg \mathsf{pos}_1$

Example: Robots and Carriage



$\langle\langle 1 \rangle\rangle \Box \neg \mathsf{pos}_1$

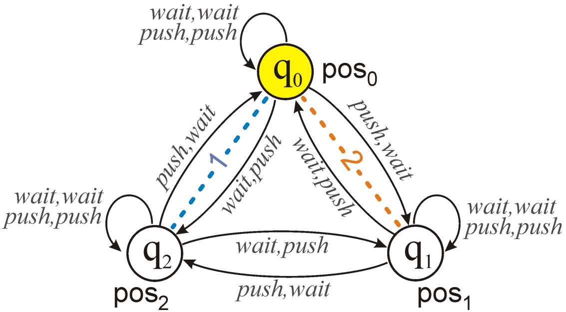
Example: Robots and Carriage



$\langle\langle 1 \rangle\rangle \Box \neg \mathsf{pos}_1$

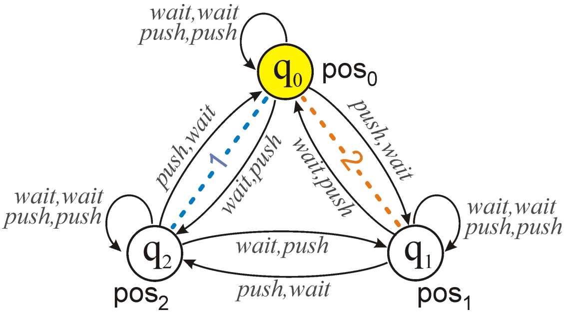
Yes!

Example: Robots and Carriage



$\langle\langle 1 \rangle\rangle \Box \neg \mathsf{pos}_1$

Example: Robots and Carriage



$\langle\langle 1 \rangle\rangle \Box \neg \mathsf{pos}_1$

No!

Outline

Model Checking ATL

Model Checking ATL

We want to implement function $mcheck(M, \varphi)$ such that:

$$mcheck(M, \varphi) = \begin{cases} \top & \text{if} \quad M \models \varphi \\ \bot & \text{else} \end{cases}$$

Model Checking ATL

We want to implement function $mcheck(M, \varphi)$ such that:

$$mcheck(M, \varphi) = \begin{cases} \top & \text{if} \quad M \models \varphi \\ \bot & \text{else} \end{cases}$$

Algorithms and even tools exist
So, let's specify and model-check!

Model Checking ATL

We want to implement function $mcheck(M, \varphi)$ such that:

$$mcheck(M, \varphi) = \begin{cases} \top & \text{if} \quad M \models \varphi \\ \bot & \text{else} \end{cases}$$

Algorithms and even tools exist
So, let's specify and model-check!

Not that easy...

Not That Easy...

Caveat: there are serious complexity obstacles:

- State-space explosion
- Transition-space explosion
- Invalidity of fixpoint equivalences for imperfect information

Not That Easy...

Caveat: there are serious complexity obstacles:

- State-space explosion
- Transition-space explosion
- Invalidity of fixpoint equivalences for imperfect information

Model checking strategic ability for agents with imperfect information ranges from NP-complete to undecidable, depending on the exact syntax, semantics, and representation of models.

Not That Easy...

Caveat: there are serious complexity obstacles:

- State-space explosion
- Transition-space explosion
- Invalidity of fixpoint equivalences for imperfect information

Model checking strategic ability for agents with imperfect information ranges from NP-complete to undecidable, depending on the exact syntax, semantics, and representation of models.

Possible way out: **incomplete verification**

Incomplete Model Checking

Two ideas:

- Approximate model checking
- Brute force search with local optimization

Incomplete Model Checking

Two ideas:

- Approximate model checking
- Brute force search with local optimization

Note: the main source of complexity is the size of the model!

Incomplete Model Checking

Two ideas:

- Approximate model checking
- Brute force search with local optimization

Note: the main source of complexity is the size of the model!
Possible way out: use smaller models $\rightsquigarrow$ **model reductions**

Outline

Approximate Verification of Strategic Ability

- Exact verification of strategic abilities is hard
- Idea: try to find formulae that **approximate the truth value of the given specification** (i.e., upper bound and lower bound)

Approximate Verification of Strategic Ability

- Exact verification of strategic abilities is hard
- Idea: try to find formulae that **approximate the truth value of the given specification** (i.e., upper bound and lower bound)
- ...and which are easier to compute 🙂

Approximate Verification of Strategic Ability

- Exact verification of strategic abilities is hard
- Idea: try to find formulae that **approximate the truth value of the given specification** (i.e., upper bound and lower bound)
- ...and which are easier to compute
- If lower bound = upper bound, we get the exact answer!

Approximate Verification of Strategic Ability

Approximation Semantics

$$LB(p) = p,$$
$$LB(\neg\phi) = \neg UB(\phi),$$
$$LB(\phi \wedge \psi) = LB(\phi) \wedge LB(\psi),$$
$$LB(\langle A\rangle\phi) = \langle A\rangle LB(\phi),$$
$$LB(\langle\!\langle A\rangle\!\rangle\Box\phi) = \nu Z.(C_A LB(\phi) \wedge \langle A\rangle^{\bullet} Z),$$
$$LB(\langle\!\langle A\rangle\!\rangle\psi \,\mathrm{U}\, \phi) = \mu Z.\big(E_A LB(\phi) \vee (C_A LB(\psi) \wedge \langle A\rangle^{\bullet} Z)\big).$$

$$UB(p) = p,$$
$$UB(\neg\phi) = \neg LB(\phi),$$
$$UB(\phi \wedge \psi) = UB(\phi) \wedge UB(\psi),$$
$$UB(\langle A\rangle\phi) = E_A \langle\!\langle A\rangle\!\rangle_{\mathrm{Ir}} \bigcirc UB(\phi),$$
$$UB(\langle\!\langle A\rangle\!\rangle\Box\phi) = E_A \langle\!\langle A\rangle\!\rangle_{\mathrm{Ir}} \Box\, UB(\phi),$$
$$UB(\langle\!\langle A\rangle\!\rangle\psi \,\mathrm{U}\, \phi) = E_A \langle\!\langle A\rangle\!\rangle_{\mathrm{Ir}}\, UB(\psi) \,\mathrm{U}\, UB(\phi).$$

Approximate Verification of Strategic Ability

Theorem (Jamroga, Knapik, Kurpiewski, & Mikulski 2019)

For every pointed model M and ATL formula φ:

$$M \models LB(\varphi) \implies M \models \varphi \implies M \models UB(\varphi).$$

Approximate Verification of Strategic Ability

Theorem (Jamroga, Knapik, Kurpiewski, & Mikulski 2019)

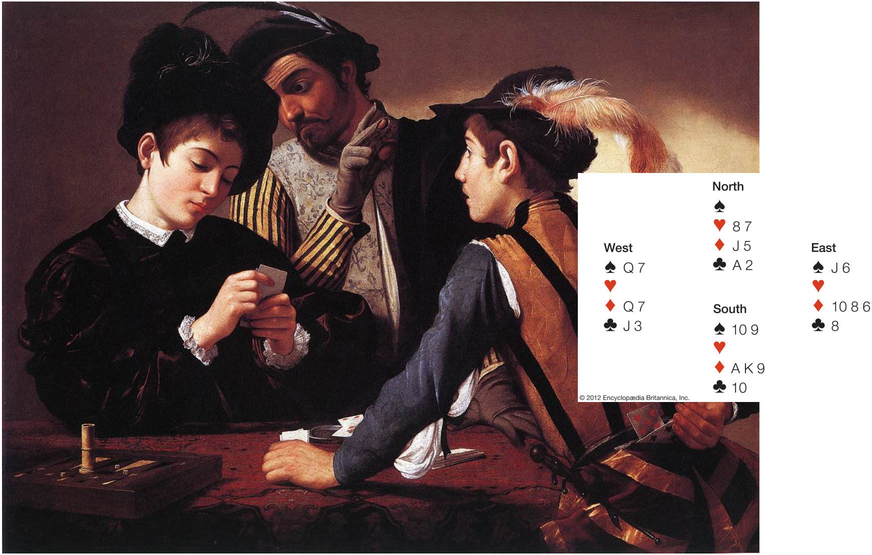
For every pointed model M and ATL formula φ:

$$M \models LB(\varphi) \implies M \models \varphi \implies M \models UB(\varphi).$$

- Benchmark: card play (similar mathematical structure to coercion in a voting protocol!)

Approximate Verification of Strategic Ability

Approximate Verification of Strategic Ability



Experimental Results

#cards	#states	Approximate verification				Exact (MCMAS)
		tgen	lower	upper	match	
4	11	<0.01	<0.01	<0.01	100%	0.12
8	346	0.01	<0.01	<0.01	100%	2.42 h*
12	12953	0.7	0.07	0.01	100%	timeout
16	617897	35.2	348.4	0.7	100%	timeout
20*	2443467	132.0	8815.7	4.2	100%	timeout

Formula: $\langle\langle \mathbf{S} \rangle\rangle \Diamond \mathsf{win}$

Time in seconds, unless explicitly indicated
timeout $\approx$ 45h

Experimental Results with Optimized Data Structures

#cards	#states	Approximate verification				Exact (MCMAS)
		tgen	lower	upper	match	
4	11	<0.01	<0.01	<0.01	100%	0.12
8	346	<0.01	<0.01	<0.01	100%	2.42 h*
12	12953	0.06	<0.01	<0.01	100%	timeout
16	617897	4.6	0.6	0.3	100%	timeout
$20^{\star}$	2443467	34.0	3.0	2.0	100%	timeout
20	1.5 e7	124.0	8.5	6.0	100%	timeout
$24^{\star}$	7 e7	3779.0	667.0	78.0	100%	timeout

Formula: $\langle\langle \mathbf{S} \rangle\rangle \Diamond \mathsf{win}$

Time in seconds, unless explicitly indicated
timeout $\approx$ 45h

Experimental Results for Absent-Minded Declarer

#cards	#states	Approximate verification				Exact (MCMAS)
		tgen	lower	upper	match	
4	19	$<$0.01	$<$0.01	$<$0.01	100%	9.68 h*
8	713	0.04	0.01	$<$0.01	100%	timeout
12	52843	5.2	18.6	0.6	80%	timeout
16	memout					timeout

Formula: $\langle\langle \mathbf{S} \rangle\rangle \Diamond \mathsf{win}$

Time in seconds, unless explicitly indicated
timeout $\approx$ 45h

DominoDFS

- When no better idea, try brute-force search for a winning strategy
- Give luck a chance $\rightsquigarrow$ Depth-First Search (DFS)

DominoDFS

- When no better idea, try brute-force search for a winning strategy
- Give luck a chance $\leadsto$ Depth-First Search (DFS)
- Idea: optimize by discarding dominated strategies

DominoDFS

- When no better idea, try brute-force search for a winning strategy
- Give luck a chance $\leadsto$ Depth-First Search (DFS)
- Idea: optimize by discarding dominated partial strategies

DominoDFS

- When no better idea, try brute-force search for a winning strategy
- Give luck a chance $\rightsquigarrow$ Depth-First Search (DFS)
- Idea: optimize by discarding dominated partial strategies
- Additional advantage: might work where fixpoint approximation is guaranteed to fail

Strategic Domination

- Consider a partial strategy σ_a defined in epistemic class $[q]_{\sim_a}$
- The context of σ_a is a partial (possibly nondeterministic) strategy σ_a^C defined everywhere outside $[q]_{\sim_a}$
- Then, (σ_a, σ_a^C) specify a full (possibly nondeterministic) strategy of agent a

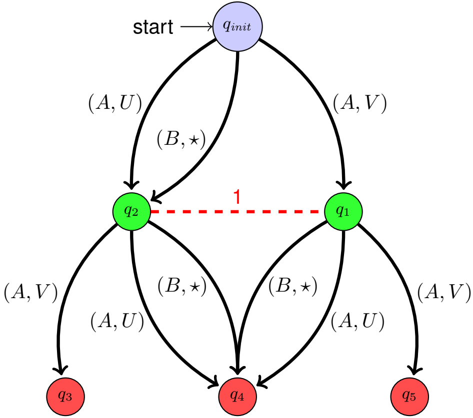
Strategic Domination

- Consider a partial strategy σ_a defined in epistemic class $[q]_{\sim_a}$
- The context of σ_a is a partial (possibly nondeterministic) strategy σ_a^C defined everywhere outside $[q]_{\sim_a}$
- Then, (σ_a, σ_a^C) specify a full (possibly nondeterministic) strategy of agent a

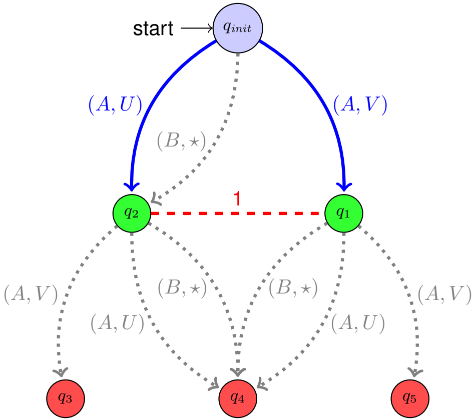
Strategic Domination

Partial strategy σ_a **dominates** σ_a' with respect to context σ_a^C iff the outcome paths of (σ_a', σ_a^C) strictly subsume those of (σ_a, σ_a^C).

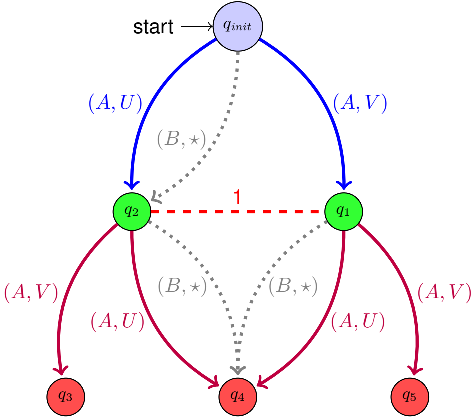
Domination-Based Depth-First Strategy Search



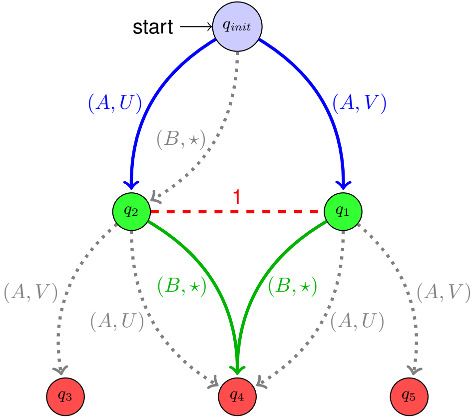
Domination-Based Depth-First Strategy Search



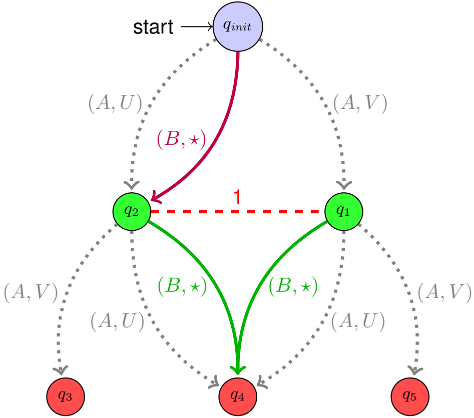
Domination-Based Depth-First Strategy Search



Domination-Based Depth-First Strategy Search



Domination-Based Depth-First Strategy Search



Experimental Results: Bridge Endplay

#cards	DominoDFS	MCMAS	Approx.	Approx. opt.
4	0.0006	0.12	0.0008	< 0.0001
8	0.01	8712	0.01	< 0.0001
12	0.8	timeout	0.8	0.06
16	160	timeout	384	5.5
20	1373	timeout	8951	39
24	memout	timeout	memout	4524

Experimental Results: Castles

#agents	DominoDFS	MCMAS	brute force (SMC)	Approx.
$(1,1,1)$	0.3	65	63	—
$(2,1,1)$	1.5	12898	184	—
$(2,2,1)$	25	timeout	4923	—
$(2,2,2)$	160	timeout	timeout	—
$(3,2,2)$	2688	timeout	timeout	—
$(3,3,2)$	timeout	timeout	timeout	—

Fixpoint approximation bound to be inconclusive!

Outline

Model Reductions

Model Reductions

Bisimulation-Based Model Reduction

Given are:

- M, M': two iCGS's, sharing the set of agents $\mathbb{A}\mathrm{gt}$ and the set of atoms AP
- coalition $A \subseteq \mathbb{A}\mathrm{gt}$
- relation $\Rrightarrow_A \subseteq S \times S'$ between the states of M and M'.

Bisimulation-Based Model Reduction

Given are:

- M, M': two iCGS's, sharing the set of agents $\mathbb{A}\mathrm{gt}$ and the set of atoms AP
- coalition $A \subseteq \mathbb{A}\mathrm{gt}$
- relation $\Rrightarrow_A \subseteq S \times S'$ between the states of M and M'.

Strategy simulator

A simulator of partial strategies for coalition A with respect to $\Rrightarrow_A$ is any family of functions

$$ST = \{ST_{C_A(q),C_A(q')} : PStr_A(C_A(q)) \to PStr_A(C_A(q')) \quad | \quad q \Rrightarrow_A q'\}.$$

The idea is that $ST_{C_A(q),C_A(q')}$ "transforms" each partial strategy σ_A that works on the neighborhood of q in model M into a corresponding strategy σ'_A that works on the neighborhood of q' in model M'.

Bisimulation-Based Model Reduction

Simulation for ATL$_{ir}$

$\Rrightarrow_A \subseteq S \times S'$ is a **simulation for A** iff there exists a simulator of partial strategies ST such that $q \Rrightarrow_A q'$ implies the following:

1. $\pi(q) = \pi'(q')$;
2. For every $i \in A$ and $r' \in S'$, if $q' \sim'_i r'$ then for some $r \in S$ we have that $q \sim_i r$ and $r \Rrightarrow_A r'$.
3. For any states $r \in C_A(q)$ and $r' \in C'_A(q')$ such that $r \Rrightarrow_A r'$, every partial strategy $\sigma_A \in PStr_A(C_A(q))$, and every state $s' \in succ(r', ST(\sigma_A))$, there exists a state $s \in succ(r, \sigma_A)$ such that $s \Rrightarrow_A s'$.

Bisimulation for ATL$_{ir}$

A relation $\Leftrightarrow_A$ is a **bisimulation for A** iff both $\Rrightarrow_A$ and $\Rrightarrow_A^{-1} = \{(q', q) \mid q \Rrightarrow_A q'\}$ are simulations.

Bisimulation-Based Model Reduction

Preservation Theorem for ATL$_{ir}$ (Belardinelli, Condurache, Dima, Jamroga, & Knapik 2021)

If $\Leftrightarrow_A$ is a bisimulation for A and $q \Leftrightarrow_A q'$, then for every A-formula φ,

$$(M, q) \models \varphi \quad \text{if and only if} \quad (M', q') \models \varphi.$$

Bisimulation-Based Model Reduction

Preservation Theorem for ATL$_{ir}$ (Belardinelli, Condurache, Dima, Jamroga, & Knapik 2021)

If $\Leftrightarrow_A$ is a bisimulation for A and $q \Leftrightarrow_A q'$, then for every A-formula φ,

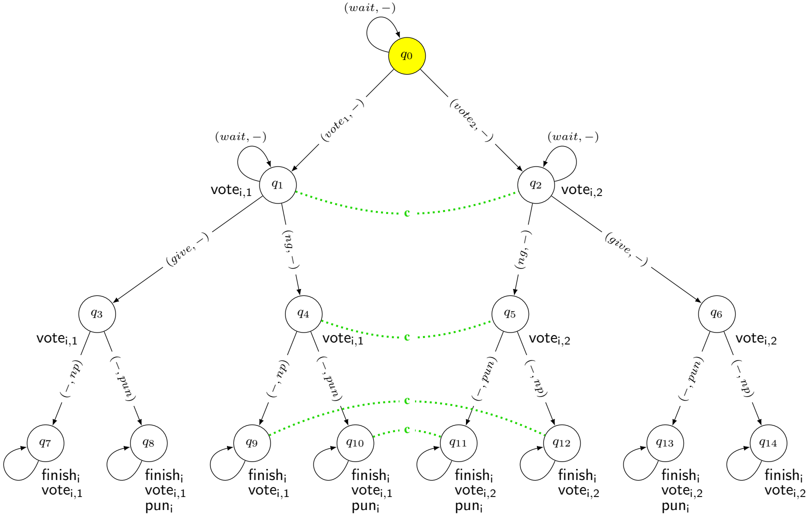
$$(M, q) \models \varphi \quad \text{if and only if} \quad (M', q') \models \varphi.$$

Corollary

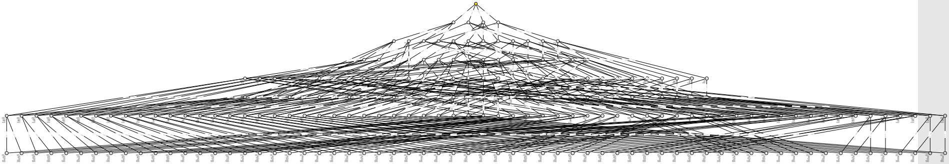
If $\Leftrightarrow$ is a bisimulation for every $A \subseteq \mathbb{A}\mathrm{gt}$, and $q \Leftrightarrow q'$, then for every formula φ,

$$(M, q) \models \varphi \quad \text{if and only if} \quad (M', q') \models \varphi.$$

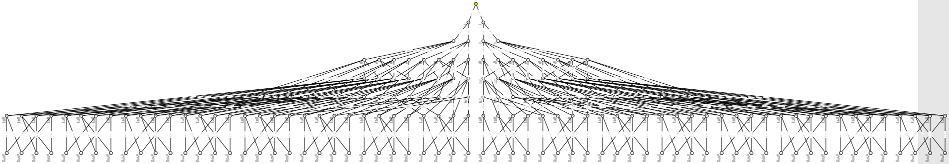
Reduction for Voting and Coercion



Reduction for Voting and Coercion



Reduction for Voting and Coercion



Bisimulation-Based Model Reduction: Summary

- Strong preservation result for the bisimulation (preserves the truth of all ATL_{ir} formulae)
- Can provide very significant reduction

Bisimulation-Based Model Reduction: Summary

- Strong preservation result for the bisimulation (preserves the truth of all ATL$_{ir}$ formulae)
- Can provide very significant reduction **when you know where to look**

Bisimulation-Based Model Reduction: Summary

- Strong preservation result for the bisimulation (preserves the truth of all ATL_{ir} formulae)
- Can provide very significant reduction **when you know where to look**
- ...But: the conditions are very strong $\leadsto$ limited applicability
- ...And reduced model + bisimulation must be crafted by hand

Bisimulation-Based Model Reduction: Summary

- Strong preservation result for the bisimulation (preserves the truth of all ATL_{ir} formulae)
- Can provide very significant reduction **when you know where to look**
- ...But: the conditions are very strong $\leadsto$ limited applicability
- ...And reduced model + bisimulation must be crafted by hand
- No methodology/algorithm for **automated reduction**

Partial Order Reduction

- **Partial order reduction (POR)**: a method of generating reduced models that preserve the formulae of logic $\mathcal{L}$
- For each infinite path, the reduced model contains at least one $\mathcal{L}$-equivalent path (but as few as possible!)

Partial Order Reduction

- **Partial order reduction (POR)**: a method of generating reduced models that preserve the formulae of logic $\mathcal{L}$
- For each infinite path, the reduced model contains at least one $\mathcal{L}$-equivalent path (but as few as possible!)
- Idea for $\text{LTL}_{-\bigcirc}$: take only one arbitrary interleaving of independent actions

Partial Order Reduction for LTL$_{-\bigcirc}$

Algorithm DFS-POR

A stack represents the path $\pi = g_0 a_0 g_1 a_1 \cdots g_n$ currently being visited.
For g_n, the following three operations are executed in a loop:

1. Compute the set $en(g_n) \subseteq Act$ of enabled actions.
2. Select (heuristically) a subset $E(g_n) \subseteq en(g_n)$ of necessary actions.
3. For any action $a \in E(g_n)$, compute the successor state g' of g_n such that $g_n \xrightarrow{a} g'$, and add g' to the stack.
   Recursively proceed to explore the submodel originating at g'.
4. Remove g_n from the stack.

Partial Order Reduction for LTL$_{-\bigcirc}$

Conditions for selection of $E(g)$

C1 No action $a \in Act \setminus E(g)$ that is dependent on an action in $E(g)$ can be executed before an action in $E(g)$ is executed.

C2 On every cycle in the constructed state graph there is at least one node g for which $E(g) = en(g)$.

C3 Each action in $E(g)$ is invisible, i.e., it does not change $V(g)$.

Partial Order Reduction for LTL$_{-\bigcirc}$

Theorem (Peled 1993)

For every formula φ of LTL$_{-\bigcirc}$:

$$M \models \varphi \quad \text{iff} \quad DFS(M) \models \varphi.$$

Partial Order Reduction for $\mathrm{LTL}_{-\bigcirc}$

Theorem (Peled 1993)

For every formula φ of $\mathrm{LTL}_{-\bigcirc}$:

$$M \models \varphi \quad \text{iff} \quad DFS(M) \models \varphi.$$

What about **ATL**?

It would seem that a much stronger (and hence less useful) reduction is needed, as ATL is much more expressive than LTL...

Surprise!

Partial Order Reduction for Strategic Abilities

Th. (Jamroga, Penczek, Sidoruk, Dembinski & Mazurkiewicz '20)

For every formula φ of **ATL**$_{-\bigcirc}$ without nested strategic operators, interpreted over imperfect information strategies:

$$M \models \varphi \quad \text{iff} \quad DFS(M) \models \varphi.$$

Partial Order Reduction for Strategic Abilities

Th. (Jamroga, Penczek, Sidoruk, Dembinski & Mazurkiewicz '20)

For every formula φ of **ATL**$_{-\bigcirc}$ without nested strategic operators, interpreted over imperfect information strategies:

$$M \models \varphi \qquad \text{iff} \qquad DFS(M) \models \varphi.$$

Also:

The same does **not** hold for ATL$_{-\bigcirc}$ interpreted over perfect information strategies.

Partial Order Reduction for Strategic Abilities

Th. (Jamroga, Penczek, Sidoruk, Dembinski & Mazurkiewicz '20)

For every formula φ of **ATL**$_{-\bigcirc}$ without nested strategic operators, interpreted over imperfect information strategies:

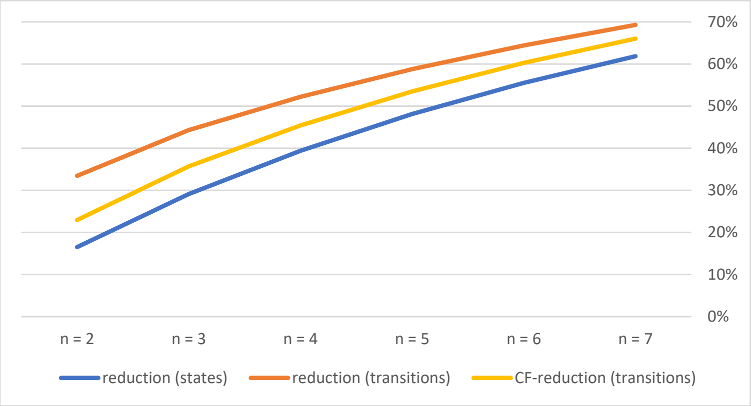
$$M \models \varphi \quad \text{iff} \quad DFS(M) \models \varphi.$$

Also:

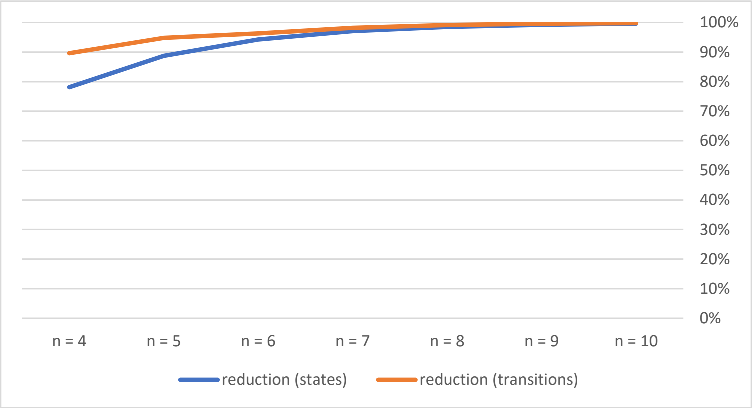
The same does **not** hold for ATL$_{-\bigcirc}$ interpreted over perfect information strategies.

How good are the reductions in practice?

Experimental Results: Asynchronous Simple Voting



Experimental Results: Trains, Gate, and Controller



Partial Order Reduction: Summary

- For some strategic abilities, we get an effective automated model reduction off the shelf for free
- There is **free lunch** out there! 

Partial Order Reduction: Summary

- For some strategic abilities, we get an effective automated model reduction off the shelf for free
- There is **free lunch** out there! 

- The trick is... someone must have already paid for the lunch
- What remained was to **prove that we are eligible** to get it (nontrivial!)

Outline

Conclusions

- Model checking is one of success stories in computer science and AI
- Still, verification of realistic systems faces a complexity barrier
- Way out: incomplete model checking and model reductions

Conclusions

- Model checking is one of success stories in computer science and AI
- Still, verification of realistic systems faces a complexity barrier
- Way out: incomplete model checking and model reductions
- For some strategic abilities, we get an automated model reduction for free

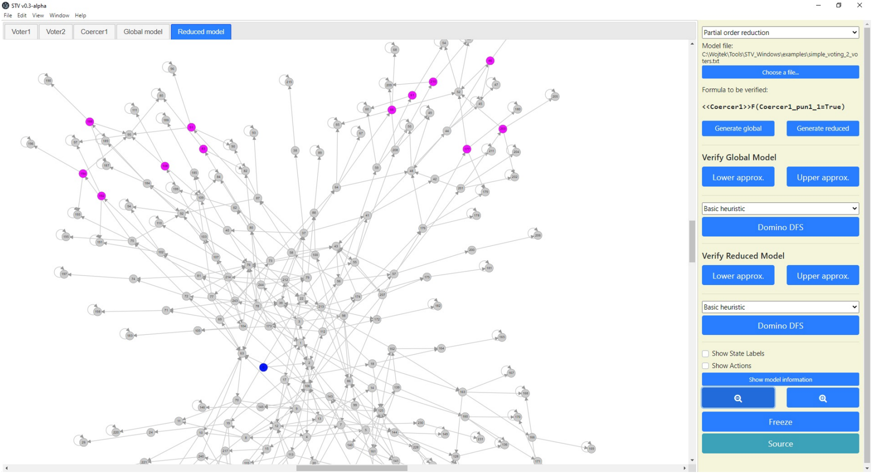
Conclusions

- Model checking is one of success stories in computer science and AI
- Still, verification of realistic systems faces a complexity barrier
- Way out: incomplete model checking and model reductions
- For some strategic abilities, we get an **effective** automated model reduction for free

STrategic Verifier (STV)

- Experimental model checker developed at ICS PAS
- Implemented techniques:
  1. standard fixpoint algorithm for perfect info games
  2. brute force DFS
  3. domination-based strategy search
  4. fixpoint approximation
  5. Under development: assume-guarantee reasoning, parallelized DFS, on the fly model generation, and state abstraction
  6. bisimulation checking for bisimulation-based reduction
  7. partial-order reduction
  8. benchmarks and examples
- Includes lightweight GUI and web-based interface

STrategic Verifier (STV)



STrategic Verifier (STV)

Current build: https://github.com/blackbat13/stv

Desktop version for Windows:
https://github.com/blackbat13/stv/releases/download/v0.3.1-alpha/stv-v0.3-alpha-win32-x64.zip

Web version: http://stv.cs-htiew.com/


Thank you