Model Checking Strategic Ability

Why, What, and Especially: How?

Wojtek Jamroga, Polish Academy of Sciences

LIPN @ Univeristy Paris 13, Villetaneuse, 15/06/2018
Outline

1. Introduction
2. Modeling Multi-Agent Systems
3. Logical Specification of Strategic Abilities
4. Model Checking
5. Approximate Model Checking
6. Model Reductions
7. Conclusions
Outline

1. Introduction
2. Modeling Multi-Agent Systems
3. Logical Specification of Strategic Abilities
4. Model Checking
5. Approximate Model Checking
6. Model Reductions
7. Conclusions
Specification and Verification of Strategic Ability

- Many important properties are based on **strategic ability**
- **Functionality** ≈ ability of authorized users to complete some tasks
- **Security** ≈ inability of unauthorized users to complete certain tasks
Many important properties are based on strategic ability

- **Functionality** \(\approx\) ability of authorized users to complete some tasks
- **Security** \(\approx\) inability of unauthorized users to complete certain tasks

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

1. Introduction
2. Modeling Multi-Agent Systems
3. Logical Specification of Strategic Abilities
4. Model Checking
5. Approximate Model Checking
6. Model Reductions
7. Conclusions
Models of Multi-Agent Systems

- How to model a distributed system? \( \sim \) 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

1. Introduction
2. Modeling Multi-Agent Systems
3. Logical Specification of Strategic Abilities
4. Model Checking
5. Approximate Model Checking
6. Model Reductions
7. Conclusions
Logical Specification of Strategic Abilities
ATL: What Agents Can Achieve

- Temporal logic meets game theory
- Main idea: cooperation modalities

\[ \langle A \rangle \Phi: \text{coalition } A \text{ has a collective strategy to enforce } \Phi \]

\[ \Rightarrow \Phi \text{ can include temporal operators: } X \text{ (next), } F \text{ (sometime in the future), } G \text{ (always in the future), } U \text{ (strong until)} \]
Example: Robots and Carriage

\[ (1) \square \text{pos}_1 \]
Example: Robots and Carriage

\[ \langle 1 \rangle F \text{pos}_1 \]

No!
Example: Robots and Carriage

\[ \langle 1 \rangle \text{G} \neg \text{pos}_1 \]
Example: Robots and Carriage

\[ G \neg \text{pos}_1 \]
Example: Robots and Carriage

\[ \langle 1 \rangle \mathcal{G} \neg \text{pos}_1 \]
Example: Robots and Carriage

$\langle 1 \rangle G \neg pos_1$
Example: Robots and Carriage

\[ \langle 1 \rangle G \neg \text{pos}_1 \]
Example: Robots and Carriage

\[ \langle 1 \rangle \mathbf{G} \neg \mathbf{pos}_1 \]

Yes!
Example: Voting and Coercion

$\langle c \rangle F (\text{pun} \leftrightarrow \neg \text{vote}_{i,1})$
Outline

1 Introduction
2 Modeling Multi-Agent Systems
3 Logical Specification of Strategic Abilities
4 Model Checking
5 Approximate Model Checking
6 Model Reductions
7 Conclusions
Model Checking ATL
We want to implement function $mcheck(M, \varphi)$ such that:

$$mcheck(M, \varphi) = \begin{cases} \top & \text{if } M \models \varphi \\ \bot & \text{else} \end{cases}$$
We want to implement function $mcheck(M, \varphi)$ such that:

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

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:

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

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
Simple Rocket Domain: Verification of $\langle\langle 1, 2 \rangle\rangle F \text{ caP}$
Simple Rocket Domain: Verification of $\langle\langle 1, 2\rangle\rangle F \text{caP}$
Simple Rocket Domain: Verification of $\langle\langle 1, 2\rangle\rangle F \text{ caP}$
Simple Rocket Domain: Verification of $\langle 1, 2 \rangle F caP$
Simple Rocket Domain: Verification of $\langle\langle 1, 2 \rangle\rangle F \text{ caP}$
Simple Rocket Domain: Verification of $\langle 1, 2 \rangle F \operatorname{caP}$
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: approximate verification
Not That Easy...

Note: the main source of complexity is the size of the model!
Not That Easy...

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

Possible way out: use smaller models \( \rightsquigarrow \text{model reductions} \)
Outline

1. Introduction
2. Modeling Multi-Agent Systems
3. Logical Specification of Strategic Abilities
4. Model Checking
5. Approximate Model Checking
6. Model Reductions
7. Conclusions
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 upper bound = lower bound, we get the exact answer!
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 upper bound = lower bound, we get the exact answer!

Approximate Verification of Strategic Ability

Approximation Semantics

\[
tr(p) = p,
tr(\neg \phi) = \neg TR(\phi),
tr(\phi \land \psi) = tr(\phi) \land tr(\psi),
tr(\langle A \rangle \phi) = \langle A \rangle tr(\phi),
tr(\langle \langle A \rangle \rangle G \phi) = \nu Z. (C_A tr(\phi) \land \langle A \rangle \bullet Z),
tr(\langle \langle A \rangle \rangle \psi U \phi) = \mu Z. (E_A tr(\phi) \lor (C_A tr(\psi) \land \langle A \rangle \bullet Z)).
\]

\[
TR(p) = p,
TR(\neg \phi) = \neg tr(\phi),
TR(\phi \land \psi) = TR(\phi) \land TR(\psi),
TR(\langle A \rangle \phi) = E_A \langle \langle A \rangle \rangle万亿 X TR(\phi),
TR(\langle \langle A \rangle \rangle G \phi) = E_A \langle \langle A \rangle \rangle万亿 G TR(\phi),
TR(\langle \langle A \rangle \rangle \psi U \phi) = E_A \langle \langle A \rangle \rangle万亿 TR(\psi) U TR(\phi).
\]
Approximate Verification of Strategic Ability

Theorem (Jamroga et al. 2017)

For every pointed model $M$ and ATL formula $\varphi$:

$$M \models tr(\varphi) \implies M \models \varphi \implies M \models TR(\varphi).$$
Approximate Verification of Strategic Ability

Theorem (Jamroga et al. 2017)

For every pointed model $M$ and ATL formula $\phi$:

$$M \models tr(\phi) \implies M \models \phi \implies M \models TR(\phi).$$

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

Approximate Verification of Strategic Ability
Approximate Verification of Strategic Ability
## Experimental Results

<table>
<thead>
<tr>
<th>No. of cards</th>
<th>#states</th>
<th>Approximate verification</th>
<th>Exact verification</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>lower</td>
<td>upper</td>
</tr>
<tr>
<td>4</td>
<td>11</td>
<td>0.0001</td>
<td>7e-05</td>
</tr>
<tr>
<td>8</td>
<td>310</td>
<td>0.002</td>
<td>0.001</td>
</tr>
<tr>
<td>12</td>
<td>12626</td>
<td>0.16</td>
<td>0.05</td>
</tr>
<tr>
<td>16</td>
<td>534722</td>
<td>172.07</td>
<td>2.61</td>
</tr>
<tr>
<td>20</td>
<td>2443467</td>
<td>76 h</td>
<td>1929</td>
</tr>
</tbody>
</table>

Formula: $\langle S \rangle F \text{ win}$

Time in seconds, unless explicitly indicated

timeout $\approx 45h$
### Experimental Results for Absent-Minded Declarer

<table>
<thead>
<tr>
<th>No. of cards</th>
<th>#states</th>
<th>Approximate verification</th>
<th>Exact verification</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>lower</td>
<td>upper</td>
</tr>
<tr>
<td>4</td>
<td>19</td>
<td>0.0003</td>
<td>0.0003</td>
</tr>
<tr>
<td>8</td>
<td>774</td>
<td>0.01</td>
<td>0.02</td>
</tr>
<tr>
<td>12</td>
<td>51865</td>
<td>29.31</td>
<td>2.45</td>
</tr>
</tbody>
</table>

Formula: $\langle S \rangle F \text{ win}$

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

1. Introduction
2. Modeling Multi-Agent Systems
3. Logical Specification of Strategic Abilities
4. Model Checking
5. Approximate Model Checking
6. Model Reductions
7. Conclusions
Bisimulation-Based Model Reduction

Given are:

- \( G, G' \): two iCGS's, sharing the set of agents \( \mathcal{A} \) and the set of atoms \( \mathcal{A}P \)
- coalition \( A \subseteq \mathcal{A} \)
- relation \( \Rightarrow_A \subseteq S \times S' \) between the states of \( G \) and \( G' \).
Bisimulation-Based Model Reduction

Given are:
- \( G, G' \): two iCGS’s, sharing the set of agents \( A \) and the set of atoms \( AP \)
- coalition \( A \subseteq A \)
- relation \( \Rightarrow_A \subseteq S \times S' \) between the states of \( G \) and \( G' \).

Strategy simulator

A simulator of partial strategies for coalition \( A \) with respect to \( \Rightarrow_A \) is any family of functions

\[
ST = \{ ST_{C_A(q), C_A(q')} : PStr_A(C_A(q)) \rightarrow PStr_A(C_A(q')) \mid q \Rightarrow_A q' \}.
\]

The idea is that \( ST_{C_A(q), C_A(q')} \) “transforms” each partial strategy \( \sigma_A \) that works on the neighborhood of \( q \) in model \( G \) into a corresponding strategy \( \sigma'_A \) that works on the neighborhood of \( q' \) in model \( G' \).
**Bisimulation-Based Model Reduction**

**Simulation for ATL$_{ir}$**

$\Rightarrow_A \subseteq S \times S'$ is a simulation for $A$ iff there exists a simulator of partial strategies $ST$ such that $q \Rightarrow_A q'$ 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 $r \Rightarrow_A r'$.
3. For any states $r \in C_A(q)$ and $r' \in C'_A(q')$ such that $r \Rightarrow_A r'$, every partial strategy $\sigma_A \in PStr_A(C_A(q))$, and every state $s' \in \text{succ}(r', ST(\sigma_A))$, there exists a state $s \in \text{succ}(r, \sigma_A)$ such that $s \Rightarrow_A s'$.

**Bisimulation for ATL$_{ir}$**

A relation $\Leftrightarrow_A$ is a bisimulation for $A$ iff both $\Rightarrow_A$ and $\Rightarrow_A^{-1} = \{(q', q) \mid q \Rightarrow_A q'\}$ are simulations.
Bisimulation-Based Model Reduction

Preservation Theorem for $\text{ATL}_{ir}$ (I)

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

$$(G, q) \models \varphi \text{ if and only if } (G', q') \models \varphi.$$
Bisimulation-Based Model Reduction

**Preservation Theorem for ATL_{ir} (I)**

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

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

**Preservation Theorem for ATL_{ir} (II)**

If $\equiv$ is a bisimulation for every $A \subseteq A$, and $q \equiv_A q'$, then for every formula $\varphi$,

$$(G, q) \models \varphi \quad \text{if and only if} \quad (G', q') \models \varphi.$$
Bisimulation-Based Model Reduction

Preservation Theorem for ATL_{ir} (I)
If $\equiv_A$ is a bisimulation for $A$ and $q \equiv_A q'$, then for every $A$-formula $\varphi$,

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

Preservation Theorem for ATL_{ir} (II)
If $\equiv$ is a bisimulation for every $A \subseteq \mathcal{A}$, and $q \equiv_A q'$, then for every formula $\varphi$,

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

Reduction for Voting and Coercion

\[
\begin{align*}
q_0 & \\
q_1 & (\text{vote}_i, -) \\
q_2 & (\text{vote}_i, -) \\
q_3 & (\text{vote}_i, 1) \\
q_4 & (\text{vote}_i, 1) \\
q_5 & (\text{vote}_i, 2) \\
q_6 & (\text{vote}_i, 2) \\
q_7 & (\text{finish}_i, \text{vote}_i, 1) \\
q_8 & (\text{finish}_i, \text{vote}_i, 1) \\
q_9 & (\text{finish}_i, \text{vote}_i, 1) \\
q_{10} & (\text{finish}_i, \text{vote}_i, 1) \\
q_{11} & (\text{finish}_i, \text{vote}_i, 2) \\
q_{12} & (\text{finish}_i, \text{vote}_i, 2) \\
q_{13} & (\text{finish}_i, \text{vote}_i, 2) \\
q_{14} & (\text{finish}_i, \text{vote}_i, 2)
\end{align*}
\]
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 \( \text{all } \text{ATL}_{ir} \text{ formulae} \))
- 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 $\text{ATL}_{ir}$ formulae)
- Can provide very significant reduction when you know where to look
- ...But: the conditions are also very strong $\leadsto$ limited applicability
- ...And the reduced model + bisimulation must be crafted by hand
Bisimulation-Based Model Reduction: Summary

- Strong preservation result for the bisimulation (preserves the truth of all $\text{ATL}_{ir}$ formulae)
- Can provide very significant reduction when you know where to look
- ...But: the conditions are also very strong $\leadsto$ limited applicability
- ...And the 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 $\text{LTL}_{\neg X}$: take only one arbitrary interleaving of independent actions
Partial Order Reduction for LTL−X

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:
Partial Order Reduction for LTL$_X$

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.
Partial Order Reduction for LTL$_{-X}$

**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.
Partial Order Reduction for LTL$_{-X}$

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 $g'$. 
Partial Order Reduction for LTL_{\neg X}

Algorithm DFS-POR

A stack represents the path $\pi = g_0a_0g_1a_1\cdots g_n$ 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 $\text{LTL}_{\neg x}$

**Conditions for selection of $E(g)$**

**C1** No action $a \in \text{Act} \setminus E(g)$ that is dependent on an action in $E(g)$ can be executed before an action in $E(g)$ is executed.
Partial Order Reduction for LTL\_x

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 $E(g) = en(g)$. 
Partial Order Reduction for LTL$_x$

**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\(_{-X}\)

**Theorem (Peled 1993)**

For every formula \(\varphi\) of LTL\(_{-X}\):

\[ M \models \varphi \iff DFS(M) \models \varphi. \]

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...
Partial Order Reduction for LTL\_x

Theorem (Peled 1993)

For every formula $\varphi$ of LTL\_x:

$$ M \models \varphi \iff DFS(M) \models \varphi. $$

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

Theorem (Jamroga et al. 2017)

For every formula $\varphi$ of $\text{ATL}_{\neg X}$ without nested strategic operators, interpreted over imperfect information strategies:

$$ M \models \varphi \iff \text{DFS}(M) \models \varphi. $$
Partial Order Reduction for Strategic Abilities

Note also:

**Theorem 1 (Jamroga et al. 2017)**

The same does not hold for ATL$_{X}$ interpreted over perfect information strategies.

Outline

1. Introduction
2. Modeling Multi-Agent Systems
3. Logical Specification of Strategic Abilities
4. Model Checking
5. Approximate Model Checking
6. Model Reductions
7. Conclusions
Conclusions

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