Strategy RV: A Tool to Approximate ATL Model Checking under Imperfect Information and Perfect Recall

Angelo Ferrando\textsuperscript{1} and Vadim Malvone\textsuperscript{2}

1 University of Genova, Italy
2 Télécom Paris, France

System Correctness

- A very important problem in critical systems:
  - Safety: error costs lives (e.g. Therac-25).
  - Mission: error costs in terms of objectives (e.g. Ariane 5).
  - Business: failure cost in loss of money (e.g. Denver airport).
- In such systems failure is not an option.

The Model Checking Approach to System Verification

- Model system $S$ as a transition system $M_S$
- Specify property $P$ as formula $\varphi_P$
- Check that $M_S \models \varphi_P$

Problem: for Multi-agent Systems the model checking problem is undecidable in many cases of interest.

Multi-Agent Systems (MAS): Key aspects

- There are many agents (players) interacting among them.
- Each agent has a set of strategies.
- A strategy is a conditional plan that at each step of the game prescribes an action.
- The composition of strategies, one for each player, induces an unique computation.

The Role Played by Memory and Information

Depending on the memory, we distinguish between:
- imperfect recall strategies ($\text{IR}$) $\implies \sigma : St \rightarrow Act$;
- perfect recall strategies ($\text{PR}$) $\implies \sigma : St^+ \rightarrow Act$. 

Depending on the players’ information, we distinguish between:
- perfect information games ($\text{PI}$);
- imperfect information games ($\text{II}$).

Specification: Alternating-time temporal logic

State ($\varphi$) and path ($\psi$) formulas in ATL* are:

\[
\varphi ::= q \mid \neg \varphi \mid \varphi \land \varphi \mid < \Gamma > \psi
\]

\[
\psi ::= \neg \psi \mid \psi \land \psi \mid X \psi \mid (\psi | \psi)
\]

The strategy operator $< \Gamma > \psi$ is read as “the agents in coalition $\Gamma$ have a strategy to achieve $\ldots$”

Problem: Undecidability Imperfect information

<table>
<thead>
<tr>
<th></th>
<th>perfect information</th>
<th>imperfect information</th>
</tr>
</thead>
<tbody>
<tr>
<td>imperfect recall</td>
<td>PSPACE-complete</td>
<td>undecidable</td>
</tr>
<tr>
<td>perfect recall</td>
<td>2EXPSPACE-complete</td>
<td>PSPACE-complete</td>
</tr>
</tbody>
</table>

Our contribution

A tool to approximate the verification of Alternating-time Temporal Logic (ATL) under imperfect information and perfect recall, which is known to be undecidable, by using Runtime Verification.

High level idea

Given a model $M$ and a formula $\varphi$ in ATL*, we need:
1. to find the sub-models of $M$ in which there is perfect information (resp., imperfect recall strategies) and a sub-formula $\varphi'$ of $\varphi$ is satisfied;
2. to use monitors to check whether the temporal remaining part $\psi$ of $\varphi$ can be satisfied and a sub-model $M'$ identified by (1) can be reached.

Our verification procedure

\[\begin{align*}
\text{Data:} & \quad \text{a model } M, \text{ a property } \varphi, \text{ and a variable } c \\
\text{Result:} & \quad \text{the verification result }
\end{align*}\]

\[\begin{align*}
f_\varphi & = \{\} \\
\text{if} & \quad \text{choice } = 1 \quad \text{then} \\
|c_\varphi| & = \text{FindSubModelsWithPerfectInfo}(M, \varphi) \\
\text{else} & \quad \text{if} \quad \text{choice } = 1 \quad \text{then} \\
|c_\varphi| & = \text{FindSubModelsWithImperfectRecall}(M, \varphi) \\
\text{end} \\
\text{else} & \quad \text{if} \quad \text{choice } = 1 \quad \text{then} \\
|c_\varphi| & = \text{FindSubModelsWithPerfectInfo}(M, \varphi) \\
\text{end} \\
\text{return} & \quad \text{GenerateAndRunMonitors}(M, \varphi, c_\varphi \cup c_\varphi).
\end{align*}\]

Parsing of the input model

- The user inputs the Json model $M$ and formula $\varphi$ (left).
- The tool shows the graphical representation of $M$ (right).

Extraction, visualisation, and RV of sub-models

- Each sub-model is translated into its equivalent ISPL (Interpreted Systems Programming Language) program, and verified by MCMAS;
- The list of sub-models $(M_1, M_2, \ldots)$ satisfying a sub-formula $\varphi'$ of $\varphi$ is shown to the user (top left);
- By clicking a sub-model, its visualisation, along with the verified sub-formula $\varphi'$, are displayed (right);
- Finally, an execution trace can be reported by the user, and checked by a monitor on $M$ using the selected sub-model and the remaining part $\psi$ of $\varphi$ (bottom left).

Conclusions and future works

- We presented Strategy RV, a tool that, first extracts sub-models with perfect information and/or imperfect recall that satisfy a strategic objective; and then, it uses runtime verification to check the remaining temporal objectives and to reach one of the sub-models so generated.
- In future work we intend to improve the sub-models extraction and monitors generation.
- We plan to extend the approximation and monitoring techniques to more expressive languages for strategic reasoning.

contacts: angelo.ferrando@unige.it, vadim.malvone@telecom-paris.fr