STV+AGR: Towards Verification of Strategic Ability Using Assume-Guarantee Reasoning

Damian Kurpiewski[1,2], Łukasz Mikulski[2,1], and Wojciech Jamroga[1,3]

[1] Institute of Computer Science, Polish Academy of Sciences, Warsaw, Poland
[2] Faculty of Mathematics and Computer Science, Nicolaus Copernicus University, Toruń, Poland
[3] Interdisciplinary Centre for Security, Reliability and Trust, SnT, University of Luxembourg, Luxembourg

Abstract. We present a substantially expanded version of the open source tool **STV** for strategy synthesis and verification of strategic abilities. The new version provides a web interface and support for assume-guarantee verification of multi-agent systems.

Keywords: model checking, assume-guarantee reasoning, strategic ability

1 Introduction

Model checking of multi-agent systems (MAS) allows for formal (and, ideally, automated) verification of their relevant properties. Algorithms and tools for model checking of *strategic abilities* [1,19,17] have been in development for over 20 years [2,7,4,13,3,9,12]. Unfortunately, the problem is hard, especially in the realistic case of agents with imperfect information [19,6].

In this paper, we propose a new extension of our open source experimental tool **STV** [12] that facilitates compositional model checking of strategic properties in asynchronous MAS through assume-guarantee reasoning (AGR) [18,5]. The extension is based on the results in [16], itself an adaptation of the AGR framework for liveness specifications from [14,15].

Many important properties of MAS refer to *strategic abilities* of agents and teams. For example, the **ATL*** formula $\langle\langle taxi, passg\rangle\rangle$F $\mathsf{destination}$ expresses that the cab and the passenger have a joint strategy to arrive at the destination, no matter what the other agents do, while $\langle\langle taxi\rangle\rangle$G$\neg\mathsf{fatality}$ says that the autonomous cab can drive in such a way that no one gets ever killed. Another intuitive set of strategic requirements is provided by properties of secure voting systems [20]. As shown by case studies [8,10] practical verification of such properties is still infeasible due to state-space and strategy-space explosion. **STV+AGR** addresses the specification and verification of such properties, as well as a user-friendly creation of models to be verified.

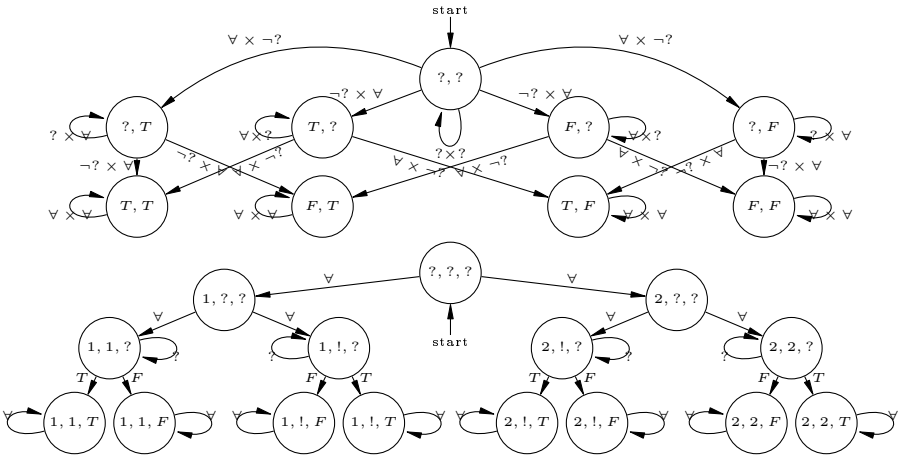

Fig. 1. Two modules: a coercer[2] (up) and a voter (down)

2 Simple Voting Scenario

To present the capabilities of **STV+AGR**, we designed an asynchronous version of the Simple Voting scenario [9]. The model consists of two types of agents, presented in Figure 1, and described below.

Voter. Every voter agent has three local variables: $vote_i$, $reported_i$ and $pstatus_i$, where ? means no decision while ! means that the voter decided not to share her vote with the coercer. Each voter i can also see the value of the pun_i variable of the coercer. The voter first casts her vote, then decides whether to share its value with the coercer. Finally, she waits for the coercer's decision on punishment.

Coercer. The coercer[k] has one local variable for each of k voters: pun_i. Moreover, he can observe the value of $reported_i$ for each voter and has two available actions per voter: to punish the voter or to refrain from punishment.

3 Formal Background

Modules. The main part of the input is given by a set of asynchronous modules inspired by [15], where local states are labelled with valuations of state variables. The transitions are valuations of input variables controlled by the other modules. The multi-agent system is defined by a composition of its modules.

Strategies. A strategy is a conditional plan that specifies what the agent(s) are going to do in every possible situation. Here, we consider the case of *imperfect information memoryless strategies*, represented by functions from the agent's local states (or, equivalently, its epistemic indistinguishability classes) to its available actions. The *outcome* of a strategy from state q consists of all the infinite paths starting from q and consistent with the strategy.

Logic. Given a model M and a state q in the model, the $\mathbf{1ATL^*_{-X}}$ [11] formula $\langle\langle a \rangle\rangle \varphi$ holds in the pointed model (M, q) iff there exists a strategy for agent a that makes φ true on all the outcome paths starting from any state indistinguishable from q. The semantics of coalitional abilities is analogous, for joint strategies of coalitions. Following this concept, the formula $\langle\langle C \rangle\rangle \varphi$ holds in (M, q) iff there exists a joint strategy (a set of strategies for every agent) of coalition C that makes φ true on all the outcome paths starting from any state indistinguishable from q.

Assume-guarantee reasoning. The theory behind the present extension of the **STV** tool is discussed in the companion paper [16]. The main idea is to cope with the state-space explosion by decomposing the goal φ of coalition C into local goals $\varphi_i, i \in C$, and verify them one by one against abstractions of each agent's environment. An abstraction for i is obtained by defining a single module, called the *assumption*, which *guarantees* that all the paths present in the original system have their counterparts in the composition of module i and its associated assumption. Those counterparts may reduce finite fragments of considered paths if nothing noticeable to the agents takes place. Moreover, we use a distance between modules, based on shared synchronization actions, so that only "close" agents are taken into account when preparing the assumption for i. This way, one can deduce the existence of a joint strategy to obtain φ from the existence of individual strategies that achieve local goals φ_i.

Automated generation of assumptions. The main difficulty in using assume-guarantee reasoning is how to define the right assumptions for the relevant modules. To this end, we propose an automated procedure that generates the assumptions, based on the subset of modules that are "close" to the given module M_i. The abstraction is obtained by composing all the "close" modules, abstracting away their state labels and variables except for the inputs of M_i, as well as removing all their input variables which are not state variables of M_i.

4 Technology

STV+AGR does *explicit-state model checking*. That is, the global states and transitions of the model are represented explicitly in the memory of the verification process. The tool includes the following new functionalities.

User-defined input. The user can load and parse the input specification from a text file that defines the groups of modules. The modules are local automata representing the agents. The groups define the partition for the assume-guarantee verification. Each group that describes the part of the coalition must also define the formula to be verified.

Web-based graphical interface. The generated models and the verification results are visualized in the intuitive web-based graphical interface. The GUI is implemented in Typescript and uses the Angular framework.

Evaluation. The assumption-guarantee scheme has been evaluated on the asynchronous variant of Simple Voting, using the formula $\varphi \equiv \langle\langle \mathit{Voter}_1 \rangle\rangle \mathrm{G}(\neg \mathsf{pstatus}_1 \vee$

V	Monolithic model checking				Assume-guarantee verification			
	#st	#tr	DFS	Apprx	#st	#tr	DFS	Apprx
2	529	2216	<0.1	<0.1/Yes	161	528	<0.1	<0.1/Yes
3	12167	127558	<0.1	0.8/Yes	1127	7830	<0.1	<0.1/Yes
4	2.79e5	6.73e6	memout		7889	1.08e5	<0.1	0.8/Yes
5	memout				5.52e4	1.45e6	<0.1	11/Yes

Table 1. Results of assume-guarantee verification (times given in seconds)

$\mathsf{voted}_1 = 1$). Note that the coalition consisted of only one agent, which made the decomposition of the formula trivial. The results are presented in Table 1. The first column describes the configuration of the benchmark, i.e., the number of voters. Then, we report the performance of model checking algorithms that operate on the explicit model of the whole system vs. assume-guarantee verification. *DFS* is a straightforward implementation of depth-first strategy synthesis. *Apprx* refers to the method of fixpoint-approximation [9]; besides the time, we also report if the approximation was conclusive.

Usage. The tool is available at stv.cs-htiew.com. The video demonstration of the tool is available at youtu.be/1DrmSRK1fBA. Example specifications can be found at stv-docs.cs-htiew.com. The presented tool **STV+AGR** allows to: Generate and display the composition of a set of modules into the model of a multi-agent system; Generate and display the automatic assumption, given a module and a distance bound; Provide local specifications for modules, and compute the global specification as their conjunction; Verify a $\mathbf{1ATL^*_{-X}}$ formula for a given system (using the verification methods available in the **STV** package); Verify a $\mathbf{1ATL^*_{-X}}$ formula for a composition of a module and its automatic assumption (using the methods in **STV**); Verify a $\mathbf{1ATL^*_{-X}}$ formula for a composition of a module and a user-defined assumption (using the methods in **STV**); Display the verification result, including the relevant truth values and the winning strategy.

5 Conclusions

Much complexity of model checking for strategic abilities is due to the size of the model of the system. **STV+AGR** addresses the challenge by implementing a compositional model checking scheme, called assume-guarantee verification. No less importantly, our tool supports user-friendly modelling of MAS, and automated generation of abstractions that are used as assumptions in the scheme.

Acknowledgement The authors thank Witold Pazderski and Yan Kim for assistance with the web interface. The work was supported by NCBR Poland and FNR Luxembourg under the PolLux/FNR-CORE project STV (POLLUX-VII/1/2019), as well as the CHIST-ERA grant CHIST-ERA-19-XAI-010 by NCN Poland (2020/02/Y/ST6/00064). The work of Damian Kurpiewski was also supported by the CNRS IEA project MoSART.

References

1. Alur, R., Henzinger, T., Kupferman, O.: Alternating-time Temporal Logic. J. of the ACM **49**, 672–713 (2002)
2. Alur, R., Henzinger, T., Mang, F., Qadeer, S., Rajamani, S., Tasiran, S.: MOCHA: Modularity in model checking. In: Proc. of CAV'98. LNCS, vol. 1427, pp. 521–525. Springer (1998)
3. Belardinelli, F., Lomuscio, A., Murano, A., Rubin, S.: Verification of multi-agent systems with imperfect information and public actions. In: Proc. of AAMAS'17. pp. 1268–1276 (2017)
4. Cermák, P., Lomuscio, A., Mogavero, F., Murano, A.: MCMAS-SLK: A model checker for the verification of strategy logic specifications. In: Proc. of CAV'14. LNCS, vol. 8559, pp. 525–532. Springer (2014)
5. Clarke, E., Long, D., McMillan, K.: Compositional model checking. In: Proc. of LICS'89. pp. 353–362. IEEE Comput. Soc. Press (1989)
6. Dima, C., Tiplea, F.: Model-checking ATL under imperfect information and perfect recall semantics is undecidable. CoRR **abs/1102.4225** (2011)
7. Huang, X., van der Meyden, R.: Symbolic model checking epistemic strategy logic. In: Proc. of AAAI'14. pp. 1426–1432 (2014)
8. Jamroga, W., Kim, Y., Kurpiewski, D., Ryan, P.: Towards model checking of voting protocols in uppaal. In: Proc. of E-Vote-ID'20. LNCS, vol. 12455, pp. 129–146. Springer (2020)
9. Jamroga, W., Knapik, ., Kurpiewski, D., Mikulski, Ł.: Approximate verification of strategic abilities under imperfect information. Artif. Int. **277** (2019)
10. Jamroga, W., Kurpiewski, D., Malvone, V.: Natural strategic abilities in voting protocols. In: Proc. of STAST'20. LNCS, vol. 12455, pp. 129–146. Springer (2021)
11. Jamroga, W., Penczek, W., Sidoruk, T., Dembinski, P., Mazurkiewicz, A.: Towards partial order reductions for strategic ability. J. Artif. Intell. Res. **68**, 817–850 (2020)
12. Kurpiewski, D., Pazderski, W., Jamroga, W., Kim, Y.: STV+Reductions: Towards practical verification of strategic ability using model reductions. In: Proc. of AAMAS'21. pp. 1770–1772. ACM (2021)
13. Lomuscio, A., Qu, H., Raimondi, F.: MCMAS: An open-source model checker for the verification of multi-agent systems. Int. J. Soft. Tools Tech. Trans. **19**(1), 9–30 (2017)
14. Lomuscio, A., Strulo, B., Walker, N., Wu, P.: Assume-guarantee reasoning with local specifications. In: Proc. of ICFEM'10. LNCS, vol. 6447, pp. 204–219. Springer (2010)
15. Lomuscio, A., Strulo, B., Walker, N., Wu, P.: Assume-guarantee reasoning with local specifications. Int. J. Found. Comput. Sci. **24**(4), 419–444 (2013)
16. Mikulski, Ł., Jamroga, W., Kurpiewski, D.: Assume-guarantee verification of strategic ability. In: Proceedings of PRIMA (2022), to appear
17. Mogavero, F., Murano, A., Perelli, G., Vardi, M.: Reasoning about strategies: On the model-checking problem. ACM Trans. Comp. Log. **15**(4), 1–42 (2014)
18. Pnueli, A.: In transition from global to modular temporal reasoning about programs. In: Logics and Models of Concurrent Systems. NATO ASI Series, vol. 13, pp. 123–144. Springer (1984)
19. Schobbens, P.: Alternating-time logic with imperfect recall. Electr. Not. Theor. Comput. Sci. **85**(2), 82–93 (2004)
20. Tabatabaei, M., Jamroga, W., Ryan, P.Y.A.: Expressing receipt-freeness and coercion-resistance in logics of strategic ability: Preliminary attempt. In: Proc. of PrAISe@ECAI'16. pp. 1:1–1:8. ACM (2016)