MsATL: A Tool for SAT-Based ATL Satisfiability Checking

Demonstration

Artur Niewiadomski
Siedlce University, Faculty of Exact and Natural Sciences
artur.niewiadomski@uph.edu.pl

Magdalena Kacprzak
Bialystok University of Technology, Poland
m.kacprzak@pb.edu.pl

Damian Kurpiewski, Michał Knapik,
Wojciech Penczek
ICS PAS, Warsaw, Poland
{d.kurpiewski,m.knapik,penczek}@ipipan.waw.pl

Wojciech Jamroga
ICS PAS, Poland, and University of Luxembourg
wojciech.jamroga@uni.lu

ABSTRACT

We present MsATL: the first tool for deciding the satisfiability of Alternating-time Temporal Logic (ATL) with imperfect information. MsATL combines SAT Modulo Monotonic Theories solvers with existing ATL model checkers: MCMAS and STV. The tool can deal with various semantics of ATL, including perfect and imperfect information, and can handle additional practical requirements. MsATL can be applied for synthesis of games that conform to a given specification, with the synthesised game often being minimal.

ACM Reference Format:
Artur Niewiadomski, Magdalena Kacprzak, Damian Kurpiewski, Michał Knapik, Wojciech Penczek, and Wojciech Jamroga. 2020. MsATL: a Tool for SAT-Based ATL Satisfiability Checking. In *Proc. of the 19th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2020), Auckland, New Zealand, May 9–13, 2020,* IFAAMAS, 3 pages.

1 INTRODUCTION AND MOTIVATIONS

Multi-agent systems (MAS) are often viewed as a game between the human or artificial players. Building a formal specification of a designed system can provide various insights into the solved problem. A minimal model conforming to the specification is most valuable: we either obtain an implementable working example or a formally correct but non-acceptable design whose validation may be tractable. We are interested in synthesis of minimal game models that conform to a specification given in *Alternating-time temporal logic* (ATL) [1–3, 20]. Each constructive procedure for testing satisfiability is of high practical importance, as it can be used to synthesize models from specifications. It is employed by various branches of computer science, including Artificial Intelligence [25] and Applied Logic [9, 11], and Program Synthesis [23, 26]. Even if synthesis from scratch is not feasible, satisfiability-based approaches can be used in order to repair an "almost-correct" program [4, 15].

2 THEORETICAL BACKGROUND

Alternating-time temporal logic (ATL) [1–3] generalizes CTL [9] by replacing the path quantifiers E, A with *strategic modalities* $\langle\langle\Gamma\rangle\rangle$. Formally, the language of ATL is defined by the following grammar: $\varphi ::= \mathrm{p} \mid \neg\varphi \mid \varphi \wedge \varphi \mid \langle\langle\Gamma\rangle\rangle X\,\varphi \mid \langle\langle\Gamma\rangle\rangle\varphi U\varphi \mid \langle\langle\Gamma\rangle\rangle G\varphi$, for $\mathrm{p} \in PV$ (a set of proposition variables). Intuitively, $\langle\langle\Gamma\rangle\rangle\gamma$ expresses that the group of agents Γ has a collective strategy to enforce γ. "X" stands for "next," "G" for "always from now on," and U for "strong until." F ("sometimes in the future") is defined as $F\varphi \equiv (true)U\varphi$.

We interpret ATL over formal models of MAS. We assume that MAS consists of n agents, each assigned a set of *local states*, an *initial local state*, a set of *local actions*, a *local protocol* that assigns a non-empty set of available actions to each state, and a *local transition function* defining possible changes of local states. The global transition function is the composition of partial transition functions of all the agents. To describe the interaction between agents, we have chosen Moore synchronous models [3]. Moreover, for each global state, a set of propositions true in this state is defined.

A *strategy* of agent i is a conditional plan that specifies what i is going to do in any situation. To ensure decidability of ATL model checking [13, 18, 33], the main technique employed by MsATL, we focus on memoryless perfect and imperfect information strategies. Intuitively, a memoryless imperfect information strategy for i assigns a local action to each of its local states while a perfect information strategy for i assigns a local action to each global state. Thus, perfect information strategies give agent i full insight into other players' local states. For more details see [3].

The problem we are solving is to decide (in a possibly most efficient way) whether an ATL formula is satisfiable. This means, given an ATL formula ϕ, we check if there exists a model M with an initial state ι in which the formula holds, i.e., $M, \iota \models \phi$. In what follows we call this decision problem ATL$_i$SAT (resp. ATL$_I$SAT) for imperfect (resp. perfect) information semantics of ATL. For more details about the theory behind MsATL see [21].

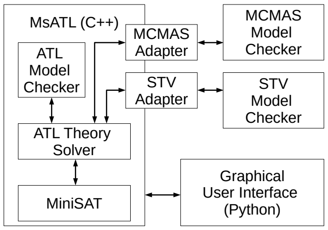
3 CHALLENGES

The main problem we are facing is a very high complexity of ATL$_I$SAT and unknown complexity of ATL$_i$SAT, which makes non-symbolic approaches, in principle, inefficient. The complexity of ATL$_I$SAT was first proved to be EXPTIME-complete [16, 34] for a fixed number of agents and later extended to the general case in [35]. The satisfiability of perfect information ATL*, a generalisation of perfect information ATL, is 2EXPTIME-complete [32]. The results

---

Proc. of the 19th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2020), B. An, N. Yorke-Smith, A. El Fallah Seghrouchni, G. Sukthankar (eds.), May 9–13, 2020, Auckland, New Zealand. © 2020 International Foundation for Autonomous Agents and Multiagent Systems (www.ifaamas.org). All rights reserved.

The work of M. Kacprzak was supported by the Bialystok University of Technology, Poland, as part of the research grant WZ/WI/1/2019 of the Faculty of Computer Science and funded by the resources for research by Ministry of Science and Higher Education, Poland. M. Knapik and W. Penczek acknowledge support from Luxembourg/Polish FNR/NCBiR project STV and CNRS/PAS project PARTIES.



Id	K	C	MsATL [sec.]	TATL [sec.]
1	9	13	0.22	0.58
2	13	19	0.23	6.2
3	17	25	0.24	29.7
4	20	31	0.31	74.6
5	23	35	0.32	229.1
6	26	41	0.34	551.9
7	30	49	0.38	1381.5
8	33	55	0.43	3947.8

Figure 1: The toolset architecture and experimental results

employ techniques based on alternating tree automata. A practically implementable tableau-based constructive decision method for ATL_ISAT was described in [17]. Subsequently, the tableau-based method was extended for checking ATL* [10] and ATEL [6], an epistemic extension of ATL [22].

Thus, there are two known methods for deciding ATL_ISAT, either by using alternating tree automata or tableau. The first one is of rather theoretical importance. The tableau-based procedure has been implemented [10], but it runs in 2EXPTIME and does not guarantee finding minimal models. For ATL_iSAT, it is not even known whether the problem is decidable. A hint of its difficulty is given in [18, 33] where model checking of ATL with imperfect information is shown to be Δ_2^P-complete. No less importantly, the logic has no standard fixed-point characterisation [7, 12]. Note that the previous solutions are applicable only to perfect information ATL, while the research of imperfect information ATL is growing rapidly. Our tool can deal with both variants of ATL with perfect- and imperfect information.

4 ARCHITECTURE AND TECHNOLOGY

MsATL employs a SAT solver and ATL model checkers to check $\mathrm{ATL}_{i/I}$SAT, and follows the concept of SAT Modulo Monotonic Theories solvers [5]. While some parts of MsATL's architecture (Fig. 1) are inspired by an earlier design for CTL [24], the tackled problem is more complex, as outlined in Sec. 3. The core components of our system are: the SAT-solver - modified MiniSAT [14], the ATL theory solver - a module interacting with the SAT-solver, and an ATL model checker (embedded or external). The SAT-solver is liable for manipulating variables representing agents' local transitions and the valuations of propositions over global states. The main task of the ATL theory solver is to check if the current partial valuation maintained by the SAT-solver represents a class of models that possibly contains a model satisfying the formula. We use external model checkers depending on the used semantics. For memoryless perfect information strategies we use our verifier and MCMAS model-checker [30, 31]. For memoryless imperfect information strategies, we use STV - the most recent tool for verification of strategic abilities under imperfect information [19, 27, 28].

MsATL is modular: we can freely attach other model checkers to expand its capabilities. It also easily outperforms the only other tool [10] over ATL_ISAT (see Fig. 1 (right) and Sec. 5). MsATL can be used standalone or via GUI. The MsATL input requires at least: the number of (1) agents, (2) local states for each agent, (3) proposition variables, (4) an ATL formula to be checked for satisfiability, (5) the model checking engine. In the case of imperfect information, a list of observable propositions for each agent is also needed. For more details please refer to http://monosatatl.epizy.com/ and video demonstration of MsATL at https://youtu.be/HSW-i80VEHs.

Table 1: Satisfiability for imperfect information - the results

Id	G	K	C	L=2	L=3	L=4	L=5
1	1	2	4	12.1	37.2	88.8	226
2	2	3	9	16.4	52.7	167	542
3	3	3	6	15.8	56.6	163	559
4	3	4	6	22.9	68.1	194	746
5	4	7	6	35.8	124	285	795
6	5	13	13	70.9	265	647	2480
7	5	17	15	88.2	314	744	2365
8	5	21	18	106	383	1110	3470

5 EXPERIMENTAL EVALUATION

Fig. 1 (right) presents an evaluation of MsATL performance on randomly generated[2] ATL_ISAT instances. MsATL's performance is compared to the only other available tool TATL [10]. The meaning of the table columns, from left to right, is as follows. The first three contain formulas' ids; the number of nested strategy operators; and the total number of Boolean connectives. Next, we present computation times of both tools, in seconds.

Table 1 presents experimental results for randomly generated formulae of ATL_i with MsATL calling STV for the model checking subtask. The column 'G' is for the number of distinct groups of agents, and the columns marked 'L' contain computation times (sec.) for different numbers of local states per agent. While not comprehensive, the results show the potential of our method, especially for some classes of ATL formulae. The experiments have been performed on Intel i5-7200U CPU/16GB Linux machine.

Satisfiability in perfect information models implies satisfiability for imperfect information, but not vice versa [8]. To test MsATL on a (non-randomly generated) case that requires imperfect information, we used formula $\neg\phi$, where $\phi \equiv \big(\neg \mathsf{next} \wedge \langle\!\langle a \rangle\!\rangle F\, \mathsf{next} \wedge \langle\!\langle \emptyset \rangle\!\rangle G(\mathsf{next} \rightarrow \langle\!\langle 1 \rangle\!\rangle F\, \mathsf{win})\big) \rightarrow \langle\!\langle 1 \rangle\!\rangle F\, \mathsf{win}$. Intuitively, ϕ expresses that, if agent a can get to a "next" state, and whenever in "next" it has a follow-up strategy to win, then a must also have a single strategy to win.[3] Formulae like ϕ are known to be valid for ATL_I but not for ATL_i [8]. MsATL determined $\neg\phi$ to be satisfiable for ATL_i (in about 80 sec.) and unsatisfiable for ATL_I (in about 11 sec.), which demonstrates that both functionalities of MsATL are important.

6 CONCLUSIONS

The problem of deciding the ATL satisfiability is computationally hard and the existing techniques are still not satisfactory for practical solutions. MsATL implements a novel technique, applying symbolic methods and SAT Modulo Monotonic Theories solvers for checking the ATL satisfiability. The method is universal as it can be applied to different classes of multi-agent systems [29], also with additional restrictions, and ATL under various semantics. This is the first tool to synthesise systems under imperfect information of ATL. The experiments show a high potential for this approach.

---

[2] Due to the lack of standard benchmarks for testing the satisfiability of ATL, we have implemented an ATL formula generator.

[3] We could not use a more straightforward formalization, since MsATL calls STV for model checking, and STV does not admit the "nexttime" operator X.

REFERENCES

[1] R. Alur, T. A. Henzinger, and O. Kupferman. 1997. Alternating-Time Temporal Logic. In *Proc. of the 38th IEEE Symp. on Foundations of Computer Science (FOCS'97)*. IEEE Computer Society, 100–109.

[2] R. Alur, T. A. Henzinger, and O. Kupferman. 1998. Alternating-Time Temporal Logic. *LNCS* 1536 (1998), 23–60.

[3] R. Alur, T. A. Henzinger, and O. Kupferman. 2002. Alternating-Time Temporal Logic. *J. ACM* 49(5) (2002), 672–713.

[4] P. C. Attie, A. Cherri, K. Dak-Al-Bab, M. Sakr, and J. Saklawi. 2015. Model and program repair via SAT solving. In *13. ACM/IEEE International Conference on Formal Methods and Models for Codesign, MEMOCODE 2015, Austin, TX, USA, September 21-23, 2015*. IEEE, 148–157.

[5] S. Bayless, N. Bayless, H.H. Hoos, and A.J. Hu. 2015. SAT Modulo Monotonic Theories. In *Proceedings of the Twenty-Ninth AAAI Conference on Artificial Intelligence (AAAI'15)*. AAAI Press, 3702–3709.

[6] F. Belardinelli. 2014. Reasoning about Knowledge and Strategies: Epistemic Strategy Logic. In *Proceedings 2nd International Workshop on Strategic Reasoning, SR 2014, Grenoble, France, April 5-6, 2014*. 27–33.

[7] N. Bulling and W. Jamroga. 2011. Alternating Epistemic Mu-Calculus. In *Proceedings of IJCAI-11*. 109–114.

[8] N. Bulling and W. Jamroga. 2014. Comparing Variants of Strategic Ability: How Uncertainty and Memory Influence General Properties of Games. *Journal of Autonomous Agents and Multi-Agent Systems* 28, 3 (2014), 474–518.

[9] E.M. Clarke and E.A. Emerson. 1981. Design and Synthesis of Synchronization Skeletons Using Branching Time Temporal Logic. In *Proceedings of Logics of Programs Workshop (Lecture Notes in Computer Science)*, Vol. 131. 52–71.

[10] A. David. 2015. Deciding ATL* Satisfiability by Tableaux. In *International Conference on Automated Deduction*. Springer, 214–228.

[11] E. De Angelis, A. Pettorossi, and M. Proietti. 2012. Synthesizing Concurrent Programs Using Answer Set Programming. *Fundam. Inform.* 120, 3-4 (2012), 205–229.

[12] C. Dima, B. Maubert, and S. Pinchinat. 2015. Relating Paths in Transition Systems: The Fall of the Modal Mu-Calculus. In *Proceedings of MFCS (Lecture Notes in Computer Science)*, Vol. 9234. Springer, 179–191. https://doi.org/10.1007/978-3-662-48057-1_14

[13] C. Dima and F.L. Tiplea. 2011. Model-checking ATL under Imperfect Information and Perfect Recall Semantics is Undecidable. *CoRR* abs/1102.4225 (2011).

[14] N. Eén and N. Sörensson. 2003. An Extensible SAT-solver. In *Theory and Applications of Satisfiability Testing, 6th International Conference, SAT 2003. Santa Margherita Ligure, Italy, May 5-8, 2003 Selected Revised Papers (Lecture Notes in Computer Science)*, Vol. 2919. Springer, 502–518.

[15] D. Gopinath, M.Z. Malik, and S. Khurshid. 2011. Specification-Based Program Repair Using SAT. In *Tools and Algorithms for the Construction and Analysis of Systems - 17th International Conference, TACAS 2011, Held as Part of the Joint European Conferences on Theory and Practice of Software, ETAPS 2011, Saarbrücken, Germany, March 26-April 3, 2011. Proceedings (Lecture Notes in Computer Science)*, Vol. 6605. Springer, 173–188.

[16] V. Goranko and G. Van Drimmelen. 2006. Complete axiomatization and decidability of alternating-time temporal logic. *Theoretical Computer Science* 353, 1-3 (2006), 93–117.

[17] V. Goranko and D. Shkatov. 2009. Tableau-based decision procedures for logics of strategic ability in multiagent systems. *ACM Trans. Comput. Log.* 11, 1 (2009), 3:1–3:51.

[18] W. Jamroga and J. Dix. 2006. Model Checking ATL$_{ir}$ is Indeed Δ_2^P-complete. In *Proceedings of EUMAS'06 (CEUR Workshop Proceedings)*, Vol. 223. CEUR-WS.org.

[19] W. Jamroga, M. Knapik, D. Kurpiewski, and Ł. Mikulski. 2019. Approximate verification of strategic abilities under imperfect information. *Artif. Intell.* 277 (2019).

[20] W. Jamroga, W. Penczek, P. Dembiński, and A. Mazurkiewicz. 2018. Towards Partial Order Reductions for Strategic Ability. In *Proceedings of the 17th International Conference on Autonomous Agents and MultiAgent Systems (AAMAS '18)*. 156–165.

[21] M. Kacprzak, A. Niewiadomski, and W. Penczek. 2020. SAT-Based ATL Satisfiability Checking. (2020). arXiv:cs.LO/2002.03117

[22] M. Kacprzak and W. Penczek. 2004. A Sat-Based Approach to Unbounded Model Checking for Alternating-Time Temporal Epistemic Logic. *Synthese* 142, 2 (2004), 203–227.

[23] G. Katz and D. Peled. 2017. Synthesizing, correcting and improving code, using model checking-based genetic programming. *STTT* 19, 4 (2017), 449–464.

[24] T. Klenze, S. Bayless, and A.J. Hu. 2016. Fast, Flexible, and Minimal CTL Synthesis via SMT. In *Computer Aided Verification*, S. Chaudhuri and A. Farzan (Eds.). Springer International Publishing, 136–156.

[25] J. R. Koza. 1993. *Genetic programming - on the programming of computers by means of natural selection*. MIT Press.

[26] K. Krawiec, I. Bladek, J. Swan, and J. H. Drake. 2018. Counterexample-Driven Genetic Programming: Stochastic Synthesis of Provably Correct Programs. In *Proceedings of the Twenty-Seventh International Joint Conference on Artificial Intelligence, IJCAI 2018, July 13-19, 2018, Stockholm, Sweden*. ijcai.org, 5304–5308.

[27] D. Kurpiewski, W. Jamroga, and M. Knapik. 2019. STV: Model Checking for Strategies under Imperfect Information. In *Proceedings of the 18th International Conference on Autonomous Agents and MultiAgent Systems, AAMAS '19, Montreal, QC, Canada, May 13-17, 2019*. 2372–2374.

[28] D. Kurpiewski, M. Knapik, and W. Jamroga. 2019. On Domination and Control in Strategic Ability. In *Proceedings of the 18th International Conference on Autonomous Agents and MultiAgent Systems, AAMAS '19, Montreal, QC, Canada, May 13-17, 2019*. 197–205.

[29] A. Lomuscio, W. Penczek, and H. Qu. 2010. Partial Order Reductions for Model Checking Temporal-epistemic Logics over Interleaved Multi-agent Systems. *Fundam. Inform.* 101, 1-2 (2010), 71–90.

[30] A. Lomuscio, H. Qu, and F. Raimondi. 2017. MCMAS: an open-source model checker for the verification of multi-agent systems. *International Journal on Software Tools for Technology Transfer* 19, 1 (2017), 9–30.

[31] A. Lomuscio and F. Raimondi. 2006. Model checking knowledge, strategies, and games in multi-agent systems. In *5th International Joint Conference on Autonomous Agents and Multiagent Systems (AAMAS 2006), Hakodate, Japan, May 8-12, 2006*. 161–168.

[32] S. Schewe. 2008. ATL* Satisfiability Is 2EXPTIME-Complete. In *Automata, Languages and Programming, 35th International Colloquium, ICALP 2008, Reykjavik, Iceland, July 7-11, 2008, Proceedings, Part II - Track B: Logic, Semantics, and Theory of Programming & Track C: Security and Cryptography Foundations*. 373–385.

[33] P. Y. Schobbens. 2004. Alternating-Time Logic with Imperfect Recall. *Electronic Notes in Theoretical Computer Science* 85, 2 (2004), 82–93.

[34] G. van Drimmelen. 2003. Satisfiability in alternating-time temporal logic. In *18th Annual IEEE Symposium of Logic in Computer Science, 2003. Proceedings*. IEEE, 208–217.

[35] D. Walther, C. Lutz, F. Wolter, and M. Wooldridge. 2006. ATL satisfiability is indeed EXPTIME-complete. *Journal of Logic and Computation* 16, 6 (2006), 765–787.