SMT4SMTL: a Tool for SMT-Based Satisfiability Checking of SMTL

Artur Niewiadomski  
University of Siedlce  
Siedlce, Poland  
artur.niewiadomski@uws.edu.pl

Maciej Nazarczuk  
University of Siedlce  
Siedlce, Poland  
maciej.nazarczuk@uws.edu.pl

Mateusz Przychodzki  
University of Siedlce  
Siedlce, Poland  
mateusz.przychodzki@uws.edu.pl

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

Wojciech Penczek  
Institute of Computer Science, PAS  
Warsaw, Poland  
w.penczek@ipipan.waw.pl

Andrzej Zbrzezny  
Jan Długosz University  
in Częstochowa, Poland  
a.zbrzezny@ujd.edu.pl

ABSTRACT

We present SMT4SMTL - the first tool for deciding the bounded satisfiability problem of Strategic Metric Temporal Logic (SMTL) and the existential fragment of Strategic Metric Temporal Logic (SMTL), interpreted over timed multi-agent systems represented by networks of timed automata. The tool combines Satisfiability Modulo Theories (SMT) techniques and Parametric Bounded Model Checking algorithms.

KEYWORDS

Strategic MTL; Bounded Satisfiability; SMT

1 INTRODUCTION AND CHALLENGES

The paper presents the tool SMT4SMTL for solving the bounded satisfiability problem for the existential fragment of Strategic Metric Temporal Logic (SMTL) [21] and MTL [6, 24]. This is a problem to decide whether an SMTL formula is satisfiable on a timed multi-agent system under some initial restrictions. SMT4SMTL implements a new method of SAT checking [21], which combines Satisfiability Modulo Theories (SMT) techniques and Parametric Bounded Model Checking algorithms. The bounded approach can be used only for the existential fragment of SMTL. Our method consists in synthesising a model as the product of agents represented by a Parametric Network of Timed Automata (PNTA), for an SMTL formula expressing the property. Parameters are used in guards, invariants, and to specify transitions and actions. Given an additional knowledge about the system’s structure, some parameters can be replaced with fixed values. We encode the SMTL formula and the runs of PNTA unfolded to a depth k, as an SMT problem instance, which is then checked for satisfiability by an SMT-solver. If the answer is SAT, all parameter values from a model are returned by the SMT-solver. Otherwise, the unfolding depth is increased.

The main challenge in solving the SAT problem of MTL and SMTL is its high complexity. The SAT problem for MTL, so for SMTL, is undecidable. In order to regain decidability of MTL, certain semantic and syntactic restrictions are introduced [36]. Semantic restrictions include adopting an integer-time model [6, 14, 16] or a bounded-variation dense-time model [42] for which SAT is decidable. Syntactic restrictions concern: punctuality or non-singularity [5], boundedness and safety [9, 34, 35]. Then, the SAT problem becomes decidable and is EXPSPACE-complete. To the best of our knowledge, SMT4SMTL is the only tool solving the SAT problem for SMTL. However, there are tools for solving the SAT problem for untimed strategic temporal logics [19, 20, 33].

2 APPLICATION DOMAIN

In recent years, significant efforts have been dedicated to verifying strategic properties in Multi-Agent Systems (MAS) [2, 17, 18, 22, 23, 37, 41] expressed using formalisms such as Alternating-time Temporal Logic ATL and ATL* [7], as well as its generalization Strategy Logic (SL) [31], also with several restrictions [1, 8, 28–30]. This research is supported by the tools like MOCHA [3], MCMAS [27], STV [25, 26], or MCMAS-SLK [10–12]. The primary emphasis lies on strategy synthesis, closely intertwined with model checking [11, 12]. Model synthesis, in turn, involves the automatic construction of a model for a given formula, thereby verifying the existence of such a model. SMTL finds application in specifying strategy-oriented behavior and troubleshooting in real-time systems, which are often complex and challenging to design, implement, and test, requiring specialized skills and expertise. These systems are engineered for real-time operation, demanding guaranteed responses within specified periods or meeting particular deadlines [38, 39]. For instance, controllers in airplanes, cars, or industrial plants are expected to complete tasks within reliable time constraints [43]. Additional potential applications encompass real-time communication and real-time strategy games.

3 THEORETICAL BACKGROUND

Strategic Metric Temporal Logic (SMTL) [21] extends Metric Temporal Logic (MTL) by strategy operators $\langle A \rangle \exists$ and $\langle A \rangle \forall$ (preceeding MTL formulae) for specifying existential and universal (resp.) strategic properties. Let $p \in AP$ be an atomic proposition, $Id$ be...
a set agents, and $\mathcal{H}$ the set of all the intervals in $\mathbb{R}_+$ of the form $[a, b], (a, b], [a, b], (a, b), [a, b], (a, \infty)$, or $[a, \infty)$, where $a, b \in \mathbb{N}$ and $a < b$, and let $I \subseteq \mathcal{H}$. The syntax of SMTL (MTL) is defined by the formulae $\phi$ (resp.) as follows:

$$\phi := p \mid \neg \phi \mid \phi \land \phi \mid (\langle A \rangle \psi) \mid (\langle \mathcal{A} \rangle \psi),$$

$$\psi := p \mid \neg p \mid \psi \land \psi \mid \psi \lor \psi \mid \psi \mathcal{U} \psi \mid G \psi,$$

where $A \subseteq Id$.

The operators $\mathcal{U}$ and $G$ are read as “until in the interval $\Gamma$” and “always in the interval $\Gamma$”, respectively. The operator $\mathcal{U}$ is defined in the standard way: $F \psi = true \mathcal{U} \psi$, where $true = v \lor \neg v$, for some $p \in \mathcal{AP}$.

Intuitively, $\langle A \rangle \psi$ means that the agents of $A$ have a collective strategy s.t. it is possible to ensure $\psi$, while $\langle \mathcal{A} \rangle \psi$ means that the agents of $A$ have a strategy to inevitably ensure $\psi$. The fragment of SMTL is called existential if it does not contain the subformulas $\langle \mathcal{A} \rangle \psi$ and the negation is applied to the propositions only.

SMTL is interpreted over concrete models of Continuous MAS (CMAS), where each agent is represented by a timed automaton [4] with asynchronous, strongly monotonic semantics and continuous time. Thus, we assume that CMAS consists of $n$ agents, each assigned a set of local actions, a set of local states, an initial local state, a set of local clocks, a local transition relation defining possible changes of local states (clocks can be reset), a local protocol that assigns a non-empty set of available actions to each state, and a state invariant function that assigns clock constraints to the local states. The global states are tuples of the local states, and the global transition relation is defined by the asynchronous composition of the local transition relations of all agents. A strategy of agent $i$ is a conditional plan that specifies what $i$ is going to do in any situation. For more details see [21].

The problem we are addressing is the determination of the satisfiability of an existential SMTL formula $\phi$, i.e., SMT4SMTL checks whether there is a model $M$ for $\phi$. This is achieved by defining a PNTA with meta-parameters specifying the number of timed automata, as well as the number of their local states and transitions. This network and $\phi$ are encoded in SMT using Boolean, Integer, and Real variables. Finding a model involves determining values for these variables. The algorithm terminates when either a model satisfying $\phi$ is discovered or when the memory/time limit is reached.

4 ARCHITECTURE AND TECHNOLOGY

There are three main modules of SMT4SMTL: GUI, BMC [21], and Z3 SMT-solver[32]. GUI is a user friendly, interactive web client implemented in TypeScript on the top of SvelteKit [15] and Cytoscape [13] libraries. It allows to edit graphically a formula as well as PNTA and start the computations on the server side. The BMC module, implemented in C++, encodes the problem using smtlibv2 standard, which is then checked for satisfiability by the Z3 SMT-solver. When the computations are complete, the GUI visualizes the results (see Fig. 1). For more details the reader is referred to the tool website: https://smtl.ii.uw.edu.pl/.

5 EXPERIMENTAL EVALUATION

Tab. 1 displays an evaluation of SMT4SMTL performance using the timed version of Dining Philosophers problem of [21] and the formula $\alpha = \langle (LcK) \rangle \exists (F_{[1,E_1]} (\land j \in odd_p Eating j)) \land (\land j \in odd_p (F_{[0,\infty]}$)

<table>
<thead>
<tr>
<th>Var</th>
<th>$p$</th>
<th>$n$</th>
<th>$k$</th>
<th>Time</th>
<th>Mem</th>
</tr>
</thead>
<tbody>
<tr>
<td>bSAT</td>
<td>2</td>
<td>5</td>
<td>8</td>
<td>0.6 / 212</td>
<td>0.2</td>
</tr>
<tr>
<td>3</td>
<td>7</td>
<td>8</td>
<td>2.8 / 58.5</td>
<td>0.3</td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>9</td>
<td>8</td>
<td>8.6 / 1416</td>
<td>1.6</td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>11</td>
<td>8</td>
<td>23 / 3420</td>
<td>3.4</td>
<td></td>
</tr>
<tr>
<td>Synth</td>
<td>2</td>
<td>5</td>
<td>12</td>
<td>1.3 / 38</td>
<td>0.1</td>
</tr>
<tr>
<td>3</td>
<td>7</td>
<td>12</td>
<td>6.0 / 126</td>
<td>0.4</td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>9</td>
<td>24</td>
<td>78 / 298</td>
<td>0.5</td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>11</td>
<td>24</td>
<td>202 / 491</td>
<td>0.9</td>
<td></td>
</tr>
</tbody>
</table>

6 CONCLUSIONS

Our tool implements a novel technique for bounded satisfiability checking of a fragment of SMTL. This marks a breakthrough in the field, as SMT4SMTL stands out as the first tool capable of synthesizing systems specified within a fragment of SMTL. The method can also be applied to partially specified systems. The experiments conducted demonstrate a high potential for this approach.

ACKNOWLEDGMENTS

M. Kacprzak was supported by the Bialystok University of Technology within the grant WZ/WI-IIT/2/2022. W. Penczek was supported by NCBR Poland under the PoLLux/FNR-CORE project SpaceVote (POLUX-XI/14/SpaceVote/2023) and PHC Polonium project MoCCa.