Synthesis of Mechanisms with Strategy Logic (Short Paper)

Munyque Mittelmann, Bastien Maubert, Aniello Murano, Laurent Perrussel

Synthesis of Mechanisms with Strategy Logic (Short Paper)

Munyque Mittelmann[1,2,*], Bastien Maubert[2], Aniello Murano[2] and Laurent Perrussel[1]

[1]IRIT - Université Toulouse 1 Capitole, France
[2]Università degli Studi di Napoli "Federico II", Italy

Abstract

Mechanism Design aims to design a game so that a desirable outcome is reached regardless of agents' self-interests. In this paper, we show how this problem can be rephrased as a synthesis problem, where mechanisms are automatically synthesized from a partial or complete specification in a high-level logical language. We show that Quantitative Strategy Logic is a perfect candidate for specifying mechanisms as it can express complex strategic and quantitative properties.

Keywords

Mechanism Design, Logics for Multi-Agent Systems, Strategic Reasoning.

1. Introduction

Mechanism Design is focused on the designing of games that aggregate agents' preferences towards a single joint decision. Such games should ensure a preferable behavior of (rational) players as well as desirable features of the decision [1]. Conitzer and Sandholm [2] introduced Automated Mechanism Design (AMD), whose goal is to automatically create mechanisms for solving a specific preference aggregation problem. AMD is usually tackled from an optimization and/or data-driven point of view (for instance, see [3, 4, 5]).

In this paper we argue that Strategy Logic [6, 7] is a good candidate for a general-purpose logic for mechanism design. More precisely, we present a recently proposed approach to AMD, which consists in automating the process of verifying and designing new mechanisms using formal methods and strategic reasoning. This approach was inspired by Wooldridge *et al.* (2007) [8], who advocated the use of Alternating-time Temporal Logic (ATL) [9] to reason about mechanisms. ATL lacks the ability to reason about quantitative aspects such as preferences, as well as game-theoretic concepts such as equilibria, which are key features for modeling and evaluating mechanisms, more precisely the ones with monetary transfers. For addressing these aspects, we consider Quantitative Strategy Logic (SL[$\mathcal{F}$]) [10]. SL[$\mathcal{F}$] is expressive enough to express complex solution concepts such as Nash equilibrium and properties about quantities.

Proceedings of the 23rd Italian Conference on Theoretical Computer Science, Rome, Italy, September 7-9, 2022
*Corresponding author.
munyque.mittelmann@ut-capitole.fr (M. Mittelmann); bastien.maubert@gmail.com (B. Maubert); nello.murano@gmail.com (A. Murano); laurent.perrussel@irit.fr (L. Perrussel)
https://sites.google.com/view/mittelmann (M. Mittelmann); http://www.bastien-maubert.fr/ (B. Maubert); http://people.na.infn.it/~murano/ (A. Murano); https://www.irit.fr/~Laurent.Perrussel/ (L. Perrussel)

First, we demonstrated how to represent and verify knowledge-based benchmarks and properties (such as efficiency and strategyproofness) in the newly proposed Epistemic SL[$\mathcal{F}$] (SLK[$\mathcal{F}$]) [11]. Previous extensions for imperfect information [12, 13, 14] focused on the qualitative versions of SL, and SLK[$\mathcal{F}$] is the first logic for strategic reasoning that combines quantitative aspects, imperfect information, and the ability to express complex concepts from game theory.

In a second stage, we considered SL[$\mathcal{F}$] with Natural Strategies [15] for reasoning with bounded recall [16]. This work offers a new perspective for reasoning about mechanisms based on the complexity of agents' strategies, which we illustrated by modeling the repeated keyword auction.

Finally, we reduced the design of deterministic mechanisms to SL[$\mathcal{F}$]-synthesis [17]. In this work, mechanisms are synthesized from a partial or complete specification expressed in a high-level logical language. The quantitative semantics of SL[$\mathcal{F}$] allows us to investigate the constructions of mechanisms that approximate such properties, which is not possible with standard Strategy Logic (SL) [6, 7]. Our approach enables generating optimal mechanisms from a SL[$\mathcal{F}$] specification, which may include requirements over the strategic behaviour of participants and quality of the outcome. In this communication paper, we focus on the results obtained in relation to synthesis of action-bounded mechanisms [17].

2. Quantitative Strategy Logic

Let us first present SL[$\mathcal{F}$] [10] syntax and semantics.

Definition 1. The syntax of SL[$\mathcal{F}$] is defined by the grammar

$$\varphi ::= p \mid \exists s.\, \varphi \mid (\mathsf{i}, s)\varphi \mid f(\varphi, ..., \varphi) \mid \mathbf{X}\varphi \mid \varphi \mathbf{U} \varphi$$

where $p \in \mathrm{AP}$ is an atomic proposition, $s \in \mathrm{Var}$ is a strategy variable, $\mathsf{i} \in \mathrm{N}$ is an agent, and $f \in \mathcal{F}$ is a function over $[-1, 1]$.

The intuitive reading of the operators is as follows: $\exists s.\, \varphi$ means that there exists a strategy such that φ holds; $(\mathsf{i}, s)\varphi$ means that when strategy s is assigned to agent i, φ holds; $\mathbf{X}$ and $\mathbf{U}$ are the usual temporal operators "next" and "until". The meaning of $f(\varphi_1, ..., \varphi_n)$ depends on the function f. We use $\top$, $\vee$, and $\neg$ to denote, respectively, function 1, function $x, y \mapsto \max(x, y)$ and function $x \mapsto -x$.

Definition 2. A *weighted concurrent game structure* (wCGS) is a tuple $\mathcal{G} = (\mathcal{B}, V, v_\iota, \delta, \ell)$ where (i) $\mathcal{B}$ is a finite set of *actions*; (ii) V is a finite set of *positions*; (iii) $v_\iota \subseteq V$ is an *initial position*; (iv) $\delta : V \times \mathcal{B}^{\mathrm{N}} \to V$ is a *transition function*; (v) $\ell : V \times \mathrm{AP} \to [-1, 1]$ is a *weight function*.

In a position $v \in V$, each player i chooses an action $a_\mathsf{i} \in \mathcal{B}$, and the game proceeds to position $\delta(v, \boldsymbol{a})$ where $\boldsymbol{a}$ is the *action profile* $(a_\mathsf{i})_{\mathsf{i} \in \mathrm{N}}$. We write $\boldsymbol{o}$ for a tuple of objects $(o_\mathsf{i})_{\mathsf{i} \in \mathrm{N}}$, one for each agent, and such tuples are called *profiles*. Given a profile $\boldsymbol{o}$ and $\mathsf{i} \in \mathrm{N}$, we let o_i be agent i's component, and $\boldsymbol{o}_{-\mathsf{i}}$ is $(o_\mathsf{r})_{\mathsf{r} \neq \mathsf{i}}$. Similarly, we let $\mathrm{N}_{-\mathsf{i}} = \mathrm{N} \setminus \{\mathsf{i}\}$.

A *play* $\pi = v_1 v_2 ...$ is an infinite sequence of positions such that for every $i \geq 1$ there exists an action profile $\boldsymbol{a}$ such that $\delta(v_i, \boldsymbol{a}) = v_{i+1}$. We write $\pi_i = v_i$ for the position at index i in

play π. A *history* h is a finite prefix of a play. A *strategy* is a function $\sigma : \text{Hist} \to \mathcal{B}$ that maps each history to an action. We let *Str* be the set of strategies. An *assignment* $\mathcal{A} : \mathrm{N} \cup \mathrm{Var} \to \mathit{Str}$ is a function from players and variables to strategies. For an assignment $\mathcal{A}$, an agent i and a strategy σ for i, $\mathcal{A}[a \mapsto \sigma]$ is the assignment that maps a to σ and is otherwise equal to $\mathcal{A}$, and $\mathcal{A}[s \mapsto \sigma]$ is defined similarly, where s is a variable. For an assignment $\mathcal{A}$ and a history h, we let $\text{Out}(\mathcal{A}, h)$ be the unique play that continues h following the strategies assigned by $\mathcal{A}$.

Definition 3. (Partial, see complete definition in [10]) Let $\mathcal{G} = (\mathcal{B}, V, \delta, \ell, V_\iota)$ be a wCGS, and $\mathcal{A}$ an assignment. The satisfaction value $[\![\varphi]\!]^{\mathcal{G}}_{\mathcal{A}}(h) \in [-1, 1]$ of an $\mathsf{SL}[\mathcal{F}]$ formula φ in a history h is defined as follows, where π denotes $\text{Out}(\mathcal{A}, h)$:

$$
\begin{aligned}
[\![p]\!]^{\mathcal{G}}_{\mathcal{A}}(h) &= \ell(\text{last}(h), p) \\
[\![\exists s.\, \varphi]\!]^{\mathcal{G}}_{\mathcal{A}}(h) &= \max_{\sigma \in \mathit{Str}} [\![\varphi]\!]^{\mathcal{G}}_{\mathcal{A}[s \mapsto \sigma]}(h) \\
[\![(\mathsf{i}, s)\varphi]\!]^{\mathcal{G}}_{\mathcal{A}}(h) &= [\![\varphi]\!]^{\mathcal{G}}_{\mathcal{A}[\mathsf{i} \mapsto \mathcal{A}(s)]}(h)
\end{aligned}
$$

We write $[\![\varphi]\!]^{\mathcal{G}}(h)$ when the satisfaction value of φ does not depend on the assignment. We also let $[\![\varphi]\!]^{\mathcal{G}} = [\![\varphi]\!]^{\mathcal{G}}(v_\iota)$. We can define the classic abbreviations: $\bot =_{\text{def}} \neg\top$, $\varphi \wedge \varphi' =_{\text{def}} \neg(\neg\varphi \vee \neg\varphi')$, $\varphi \to \varphi' =_{\text{def}} \neg\varphi \vee \varphi'$, $\mathbf{F}\psi =_{\text{def}} \top\mathbf{U}\psi$, $\mathbf{G}\psi =_{\text{def}} \neg\mathbf{F}\neg\psi$ and $\forall s.\, \varphi =_{\text{def}} \neg\exists s.\, \neg\varphi$. We also use $\mathbf{A}\varphi$ as a shorthand for a universal quantification on strategies and bindings for all agents.

3. Satisfiability and Synthesis of $\mathsf{SL}[\mathcal{F}]$

The satisfiability of SL is undecidable in general [18], but it is decidable when restricted to systems with a bounded number of actions [19], which we show to be also the case for $\mathsf{SL}[\mathcal{F}]$. We restrict our attention to models in which atomic propositions take values in a given finite set of possible values. Given a finite set $\mathcal{V} \subset [-1, 1]$ s.t. $\{-1, 1\} \subseteq \mathcal{V}$, the *$\mathcal{V}$-satisfiability problem* for $\mathsf{SL}[\mathcal{F}]$ is the restriction of the satisfiability problem to $\mathcal{V}$-weighted wCGS. We have that:

Theorem 1. *Let $\mathcal{V}$ be a finite set of values and $\mathcal{B}$ a finite set of actions. Then $\mathcal{V}$-satisfiability of $\mathsf{SL}[\mathcal{F}]$ over the wCGS $\mathcal{G} = (\mathcal{B}, V, v_\iota, \delta, \ell)$ is decidable.*

The algorithm for $\mathsf{SL}[\mathcal{F}]$ satisfiability to synthesize mechanisms that optimally satisfy the specification, in the sense that they achieve the best possible satisfaction value for the specification. First, we note that the algorithm for the satisfiability problem of $\mathsf{SL}[\mathcal{F}]$ can actually return a satisfying wCGS when one exists. Second, it is proved in [10] that given a finite set $\mathcal{V}$ of possible values for atomic propositions and a formula $\varphi \in \mathsf{SL}[\mathcal{F}]$ there is only a finite number of possible satisfaction values φ can take in any wCGS, and we can compute an over-approximation $\widetilde{\text{Val}}_{\varphi,\mathcal{V}}$ of this set.

Algorithm 1 synthesizes a wCGS that maximizes the satisfaction value of the given $\mathsf{SL}[\mathcal{F}]$ specification, in all cases where the satisfiability problem for $\mathsf{SL}[\mathcal{F}]$ can be solved and a witness produced. We now show how this can be used to solve automated mechanism design.

Algorithm 1 $synthesis(\Phi, \mathcal{V})$

Input: a SL$[\mathcal{F}]$-formula Φ and a set of possible values for atomic propositions $\mathcal{V}$.
Output: a wCGS $\mathcal{G}$ such that $[\![\Phi]\!]^{\mathcal{G}}$ is maximal

1: Compute $\widetilde{\text{Val}}_{\Phi,\mathcal{V}}$
2: Let $\nu_1, ..., \nu_n$ be a decreasing enumeration of $\widetilde{\text{Val}}_{\Phi,\mathcal{V}}$
3: **for** $i \leftarrow 1$ to n **do**
4: Solve $\mathcal{V}$-satisfiability for Φ and $\varepsilon = \nu_i$
5: **if** there exists $\mathcal{G}$ such that $[\![\Phi]\!]^{\mathcal{G}} \geq \nu_i$ **then**
return $\mathcal{G}$

4. Synthesis for Mechanism Design

We first recall basic concepts used to formalize mechanisms, which determine how to choose one option among several alternatives, based on agents' strategies. We assume that each alternative is a tuple (x, p) where $x \in \mathcal{X}$ is a *choice* from a finite set of choices $\mathcal{X} \subset [-1, 1]$, $\mathrm{p} = (\mathrm{p_i})_{\mathrm{i} \in \mathrm{N}}$, and $\mathrm{p_i} \in [-1, 1]$ is the payment for agent i. For each agent $\mathrm{i} \in \mathrm{N}$, let also $\Theta_\mathrm{i} \subset [-1, 1]$ be a finite set of possible *types* for i. We let $\boldsymbol{\Theta} = \prod_{\mathrm{i} \in \mathrm{N}} \Theta_\mathrm{i}$, and we note $\boldsymbol{\theta} = (\theta_\mathrm{i})_{\mathrm{i} \in \mathrm{N}} \in \boldsymbol{\Theta}$ for a type profile, which assigns a type θ_i to each agent i. The type θ_i of an agent i determines how she values each choice $x \in \mathcal{X}$; this is represented by a *valuation function* $\mathsf{v_i} : \mathcal{X} \times \Theta_\mathrm{i} \to [-1, 1]$. A mechanism consists of a description of the agents' possible strategies, and a description of the alternatives that result from them. As shown in [11], we can represent mechanisms as wCGS and verify their equilibrium outcome.

SL$[\mathcal{F}]$ can express a variety of important notions in mechanism design, such as strategyproofness, individual rationality, and efficiency [11]. We recall the formulas for some of these notions. Let $\boldsymbol{\theta} = (\theta_\mathrm{i})_{\mathrm{i} \in \mathrm{N}}$ be a type profile in $\boldsymbol{\Theta}$.

First the agent's utility is denoted by the SL$[\mathcal{F}]$ formula $\mathrm{util_i}(\theta_\mathrm{i}) =_{\text{def}} \mathsf{v_i}(\mathrm{choice}, \theta_\mathrm{i}) - \mathrm{pay_i}$.

Efficiency can be expressed as follows: $\mathrm{EF}(\boldsymbol{\theta}) =_{\text{def}} \sum_{\mathrm{i} \in \mathrm{N}} \mathsf{v_i}(\mathrm{choice}, \theta_\mathrm{i}) = \mathrm{maxv}_{\boldsymbol{\theta}}$, where $\mathrm{maxv}_{\boldsymbol{\theta}} = \max_{x \in \mathcal{X}} \sum_{\mathrm{i} \in \mathrm{N}} \mathsf{v_i}(x, \theta_\mathrm{i})$ is a constant in $\mathcal{F}$.

We also recall the SL$[\mathcal{F}]$-formula that characterizes Nash equilibria:

$$\mathrm{NE}(\boldsymbol{s}, \boldsymbol{\theta}) =_{\text{def}} \bigwedge_{\mathrm{i} \in \mathrm{N}} \forall t. \left[(\mathrm{N}_{-\mathrm{i}}, \boldsymbol{s}_{-\mathrm{i}})(\mathrm{i}, t)\mathbf{F}(\mathrm{term} \wedge \mathrm{util_i}(\theta_\mathrm{i})) \leq (\mathrm{N}, \boldsymbol{s})\mathbf{F}(\mathrm{term} \wedge \mathrm{util_i}(\theta_\mathrm{i}))\right]$$

where $\boldsymbol{s} = (s_\mathrm{i})_{\mathrm{i} \in \mathrm{N}}$ is a profile of strategy variables.

We now illustrate the mechanism synthesis problem by considering rules based on the Japanese auction. We let $\mathrm{wins_i} \in (-1, 1]$ be a constant value denoting the choice in which the agent i is the winner, with $\mathrm{wins_i} \neq \mathrm{wins_r}$ for any $\mathrm{r} \neq \mathrm{i}$. We consider the choice set $\mathcal{X} = \{\mathrm{wins_i} : \mathrm{i} \in \mathrm{N}\} \cup \{-1\}$, where -1 specifies the case where there is no winner at the end of the game.

Example 1. In the Japanese auction, the price is repeatedly raised by the auctioneer until only one bidder remains. The remaining bidder wins the item at the final price [20]. Let us fix a price increment $\mathsf{inc} > 0$. There are only two possible actions, accept (acc) or decline (dec), so that the set $\mathcal{B} = \{\mathrm{acc}, \mathrm{dec}\}$ is indeed bounded. Furthermore, we let

$\Phi = \{\text{price}, \text{sold}, \text{initial}, \text{choice}, \text{bid}_i, \text{pay}_i, \text{term} : i \in N\}$, where price denotes the current price, initial denotes whether the position is the initial one, sold specifies whether the item was sold, bid_i specifies whether i is an active bidder, choice and pay_i denote respec. the choice elected by the mechanism, and the payment of agent i. The proposition term specifies whether a position is terminal. The following $\mathsf{SL}[\mathcal{F}]$-formulae are a partial description of a mechanism, inspired by the Japanese auction. The meaning of Rules J1-J8 is intuitive. Rule J9 specifies that for all type profiles there should exist a NE whose outcome is IR and EF.

J1. $\mathbf{AG}((\text{initial} \rightarrow \text{price} = 0 \wedge \neg\text{sold} \wedge \neg\text{term}) \wedge (\mathbf{XG}\neg\text{initial} \wedge \mathbf{F}\ \text{term}))$

J2. $\mathbf{AG}(\text{sold} \leftrightarrow \text{choice} \neq -1)$

J3. $\mathbf{AG}((\neg\text{sold} \wedge \text{price} + \text{inc} \leq 1) \rightarrow (\text{price} + \text{inc} = \mathbf{X}\text{price} \wedge \neg\mathbf{X}\text{term}))$

J4. $\mathbf{AG}((\text{sold} \vee \text{price} + \text{inc} > 1) \rightarrow (\text{price} = \mathbf{X}\text{price} \wedge \mathbf{X}\text{term}))$

J5. $\mathbf{AG}(\text{choice} = \text{wins}_i \leftrightarrow \text{bid}_i \wedge \bigwedge_{r \neq i} \neg\text{bid}_i)$

J6. $\mathbf{AG}(\text{choice} = -1 \leftrightarrow \neg(\bigvee_{i \in N}(\text{bid}_i \wedge \bigwedge_{r \neq i} \neg\text{bid}_i)))$

J7. $\mathbf{AG}\big(\bigwedge_{i \in N}(\text{choice} = \text{wins}_i \rightarrow \text{pay}_i = \text{price})\big)$

J8. $\mathbf{AG}\big(\bigwedge_{i \in N}(\text{choice} \neq \text{wins}_i \rightarrow \text{pay}_i = 0)\big)$

J9. $\bigwedge_{\boldsymbol{\theta} \in \boldsymbol{\Theta}}(\exists \boldsymbol{s}.\, \text{NE}(\boldsymbol{s}, \boldsymbol{\theta}) \wedge \mathbf{F}(\text{term} \wedge \text{IR}(\boldsymbol{\theta}) \wedge \text{EF}(\boldsymbol{\theta})))$

We denote by Σ_{jpn} the conjunction of Rules J1-J9. Algorithm 1 constructs a wCGS that maximizes the satisfaction value of Σ_{jpn}. We show that this value is 1, meaning that there exists a mechanism that is individually rational and efficient for some Nash equilibrium, for all type profiles [17].

5. Conclusion

We present a novel approach for AMD based on Strategy Logic and formal methods, which builds an important bridge between logics for strategic reasoning in MAS and economic theory (in particular, computational social choice and mechanisms design). In the results presented here, in which mechanisms can be automatically generated from partial or complete specifications in a rich logical language. The great expressiveness of the specification language $\mathsf{SL}[\mathcal{F}]$ makes our approach of automated synthesis very general, unlike previous proposals. Another advantage is the use of formal methods, which are developed to guarantee their correctness by construction. While mechanism synthesis from $\mathsf{SL}[\mathcal{F}]$ specifications is undecidable, we solve it when the number of actions is bounded.

Acknowledgments

This work is part of a paper accepted at IJCAI-ECAI 22. This research is supported by the ANR project AGAPE ANR-18-CE23-0013, the PRIN project RIPER (No. 20203FFYLK), and the EU ICT-48 2020 project TAILOR (No. 952215).

References

[1] N. Nisan, T. Roughgarden, É. Tardos, V. Vazirani, Algorithmic Game Theory, Cambridge University Press, 2007.

[2] T. Sandholm, Automated mechanism design: A new application area for search algorithms, in: Principles and Practice of Constraint Programming – CP 2003, 2003.

[3] W. Shen, P. Tang, S. Zuo, Automated mechanism design via neural networks, in: Proc. of the Int. Conf. on Autonomous Agents and Multi-Agent Systems (AAMAS 2019), 2019.

[4] P. Dütting, Z. Feng, H. Narasimhan, D. Parkes, S. S. Ravindranath, Optimal auctions through deep learning, in: Proc. of the Int. Conf. on Machine Learning (ICML 2019), 2019.

[5] H. Narasimhan, S. B. Agarwal, D. C. Parkes, Automated mechanism design without money via machine learning, in: Proc. of IJCAI-2016, 2016.

[6] K. Chatterjee, T. A. Henzinger, N. Piterman, Strategy logic, Information and Computation 208 (2010) 677–693.

[7] F. Mogavero, A. Murano, G. Perelli, M. Y. Vardi, Reasoning about strategies: On the model-checking problem, ACM Trans. on Computational Logic (TOCL) 15 (2014) 1–47.

[8] M. Wooldridge, T. Ågotnes, P. Dunne, W. Van der Hoek, Logic for automated mechanism design-a progress report, in: Proc. of AAAI Conference on Artificial Intelligence (AAAI 2007), 2007.

[9] R. Alur, T. A. Henzinger, O. Kupferman, Alternating-time temporal logic, Journal of the ACM 49 (2002) 672–713.

[10] P. Bouyer, O. Kupferman, N. Markey, B. Maubert, A. Murano, G. Perelli, Reasoning about quality and fuzziness of strategic behaviours, in: Proc. of the Int. Joint Conf. on AI (IJCAI 2019), 2019.

[11] B. Maubert, M. Mittelmann, A. Murano, L. Perrussel, Strategic reasoning in automated mechanism design, in: Proc. of the Int. Conference on Principles of Knowledge Representation and Reasoning (KR 2021), 2021, pp. 487–496.

[12] F. Belardinelli, A. Lomuscio, A. Murano, S. Rubin, Verification of multi-agent systems with public actions against strategy logic, AI 285 (2020).

[13] R. Berthon, B. Maubert, A. Murano, S. Rubin, M. Vardi, Strategy logic with imperfect information, ACM Trans. on Computational Logic 22 (2021).

[14] P. Cermák, A. Lomuscio, F. Mogavero, A. Murano, Practical verification of multi-agent systems against SLK specifications, Information and Computation 261 (2018) 588–614.

[15] W. Jamroga, V. Malvone, A. Murano, Natural strategic ability, AI 277 (2019) 103170.

[16] F. Belardinelli, W. Jamroga, V. Malvone, M. Mittelmann, A. Murano, L. Perrussel, Reasoning about human-friendly strategies in repeated keyword auctions, in: AAMAS-22, 2022.

[17] M. Mittelmann, B. Maubert, A. Murano, L. Perrussel, Automated synthesis of mechanisms, in: Proc. of the Int. Joint Conf. on AI (IJCAI 2022), 2022.

[18] F. Mogavero, A. Murano, G. Perelli, M. Y. Vardi, Reasoning about strategies: on the satisfiability problem, Logical Methods in Computer Science 13 (2017).

[19] F. Laroussinie, N. Markey, Augmenting ATL with strategy contexts, Information and Computation 245 (2015) 98–123. doi:10.1016/j.ic.2014.12.020.

[20] P. Klemperer, Auction theory: A guide to the literature, Journal of Economic Surveys 13 (1999) 227–286.