Proceedings of the 18th International Conference on Principles of Knowledge Representation and Reasoning
Main Track

Strategic Reasoning in Automated Mechanism Design

Bastien Maubert[1] , **Munyque Mittelmann**[2] , **Aniello Murano**[1] , **Laurent Perrussel**[2]
[1]Università degli Studi di Napoli "Federico II", Italy
[2]IRIT, Université de Toulouse 1 Capitole, France
bastien.maubert@gmail.com, munyque.mittelmann@irit.fr,
nello.murano@gmail.com, laurent.perrussel@irit.fr

Abstract

Mechanism Design aims at defining mechanisms that satisfy a predefined set of properties, and Auction Mechanisms are of foremost importance. Core properties of mechanisms, such as strategy-proofness or budget-balance, involve: (i) complex strategic concepts such as Nash equilibria, (ii) quantitative aspects such as utilities, and often (iii) imperfect information, with agents' private valuations. We demonstrate that Strategy Logic provides a formal framework fit to model mechanisms, express such properties, and verify them. To do so, we consider a quantitative and epistemic variant of Strategy Logic. We first show how to express the implementation of social choice functions. Second, we show how fundamental mechanism properties can be expressed as logical formulas, and thus evaluated by model checking. Finally, we prove that model checking for this particular variant of Strategy Logic can be done in polynomial space.

1 Introduction

Mechanism Design (MD) aims at defining game-like systems whose equilibria satisfy some desired properties, usually expressed in terms of incentive, utility or social welfare (Sandholm 2003). Some important types of mechanisms studied are voting systems and auctions. Verifying that a mechanism satisfies desired properties is usually done by hand, which can be a very difficult and time-consuming task for complex mechanisms and properties. For this reason, in the recent years some works started to investigate application of formal methods to the semi-automatic or automatic verification of some forms of mechanism design.

In the context of fully-automatic verification, "Logic for Mechanism Design - A Manifesto" (Pauly and Wooldridge 2003) is of particular interest. In this work the authors argue that strategic logics developed for the formal verification of multi-agent systems could be good candidates as formal frameworks to reason about mechanisms. They consider Alternating-time Temporal Logic (ATL) (Alur, Henzinger, and Kupferman 2002) and show with two case-studies based on voting systems that some relevant properties for the verification of such systems can be expressed in this logic. They conclude with a research agenda in which they detail features that are missing in ATL to make it really fit for mechanism design verification:

"*We need to incorporate more game-theoretic notions in the logics we use. While the logics discussed are capable of capturing some game theoretic notions, they are still too close to their computer science origins. For example, players' preferences, strategies, equilibrium notions, are all notions which so far are inadequately represented both in the underlying semantic models and in the logical languages used. It is also still an open question whether we will eventually end up with one general-purpose logic which functions as a standard, much the way first-order logic or modal logic do in computer science.*"

In the recent years much progress has been made in the field of logics for strategic reasoning. Strategy Logic (SL) (Chatterjee, Henzinger, and Piterman 2010; Mogavero et al. 2014) was introduced. Unlike ATL, which it subsumes, it allows for explicit manipulation of strategies, and it can express important concepts in non-zero-sum games, such as Nash equilibria. Its recent quantitative extension SL[$\mathcal{F}$] (Bouyer et al. 2019a) introduces values in models and functions in the language, enabling the reasoning about key game-theoretic concepts such as utilities and preferences. Several works have also considered extensions of SL with imperfect information, which is also an important feature in many systems of interest in mechanism design such as voting or auction systems.

In this work we argue that Strategy Logic is a good candidate for a general-purpose logic for mechanism design. To do so we focus on auction systems which, as argued in (Pauly and Wooldridge 2003), "*provide a good example of mechanisms which are (a) sufficiently complex to demonstrate the usefulness of formal verification, and (b) not too complex to make formal verification infeasible*".

More precisely, we consider a new variant of Strategy Logic with quantitative features, imperfect information and epistemic operators, that we call SLK[$\mathcal{F}$]. Because it is enough for many auction scenarios, we focus on memoryless strategies, that do not depend on the past but only on the current state. We first show how mechanisms can be cast as concurrent-game structures (Definition 5). We then show how SLK[$\mathcal{F}$] can express that a mechanism implements a social choice function (Theorem 1), a fundamental concept in mechanism design. This then allows us to express in SLK[$\mathcal{F}$] whether a mechanism satisfies desired properties. We illustrate this with a number of important properties often required in auctions, or more generally in mechanism design, that characterize the desired behaviour of the par-

ticipants or the chosen output of the mechanism: strategy proofness (SP), individual rationality (IR), efficiency (EF), budget-balance (BB) and Pareto optimality (PO) (Nisan et al. 2007). SP expresses that each participant's dominant strategy is to truthfully report her preferences. IR means that they have an incentive to participate in the mechanism, EF says that the chosen outcome gives a high social welfare, BB means that the mechanism satisfies monetary balance, and PO means no agent can strictly improve her utility without making at least one other agent decrease her utility. We then consider epistemic aspects and show how, thanks to the epistemic operators in $\mathsf{SLK}[\mathcal{F}]$, we can express properties relating agents' revenues with their beliefs about other agents' preferences. Verifying that a mechanism satisfies a property then consists in model checking an $\mathsf{SLK}[\mathcal{F}]$ formula, which we show can be done in PSPACE for memoryless strategies (and is thus a PSPACE-complete problem).

Organization We discuss relevant related work in Section 2. In Section 3 we describe the Epistemic Quantitative Strategy Logic $\mathsf{SLK}[\mathcal{F}]$. Next, in Section 4, we show how an auction can be represented as a weighted concurrent game structure and how classical concepts from MD are expressed as $\mathsf{SLK}[\mathcal{F}]$-formulas. Section 5 establishes the complexity of model checking, and Section 6 concludes and discusses issues for future work.

2 Related Work

Different approaches to automating verification of auctions are found in the literature. Some works from computer-aided verification (Caminati et al. 2015; Barthe et al. 2016; Kerber, Lange, and Rowat 2016) express mechanisms in high-level specification languages, which can express rich features including probabilistic aspects. The drawback of this high expressivity is that, in contrast with model checking, verification is then not fully automatic, but only assisted by a reasoner such as Isabelle or Coq. Troquard *et al*. (2011) show how to reason about voting rules properties such as strategy-proofness in a formalism that allows fully automatic verification. However, the logic they use can only model one-shot mechanisms and thus does not capture multi-stage auctions such as Dutch Auctions. The works closest to ours are (Pauly and Wooldridge 2003; Wooldridge et al. 2007) which, as we already discussed, advocate the use of strategic logics to reason about (possibly multi-stage) mechanisms by considering Alternating-time Temporal Logic (ATL) (Alur, Henzinger, and Kupferman 2002). They discuss that ATL lacks the ability to reason about quantitative aspects such as preferences, and game-theoretic concepts such as equilibria. The first order extension of ATL (Belardinelli and Lomuscio 2016) allows one to capture some quantitative aspects, and the authors demonstrate how an English auction may be represented, and strategic properties such as manipulation and collusion verified. However, key strategic concepts such as dominance can not be expressed in the logic.

Our logic $\mathsf{SLK}[\mathcal{F}]$ is rooted in a rich line of work on logics for strategic reasoning, starting with ATL (Alur, Henzinger, and Kupferman 2002), the foundational language for strategic reasoning in multi-agent systems. ATL has been extended in various directions, considering for instance strategy contexts (Laroussinie and Markey 2015) or adding imperfect information and epistemic operators (Jamroga and Bulling 2011). Strategy Logic (SL) (Chatterjee, Henzinger, and Piterman 2010; Mogavero et al. 2014) was then proposed which, by treating strategies as first-order variables, can express complex game-theoretic concepts. SL has been extended to handle imperfect information and knowledge operators (Berthon et al. 2021; Belardinelli et al. 2020; Maubert and Murano 2018), but none of these logics can account for quantitative aspects. Recently, $\mathsf{SL}[\mathcal{F}]$ (Bouyer et al. 2019a) was introduced as a quantitative extension of SL. By introducing quantitative values in the models and functions in the language, it enables the reasoning about all key concepts involved in auctions: utilities, payments, goods and quantities. In this work we merge both lines by combining quantitative aspects and imperfect information in $\mathsf{SLK}[\mathcal{F}]$.

Related to auction representation, let us stress (Yadav and Thangarajah 2016) combining the Prometheus tool (Padgham, Thangarajah, and Winikoff 2008) and specification in ATL of an auction-based multi-agent system. An interesting lesson concerns scalability which is shown as "reasonable" via experimental results. Based on the Game Description Language (Genesereth and Thielscher 2014), (Mittelmann and Perrussel 2020) proposes an Auction Description Language, which is general enough for representing numerous kinds of auctions. However, there is no strategic or epistemic dimension in the language as it focuses on auctions rules and not on participant's behavior.

3 Epistemic $\mathsf{SL}[\mathcal{F}]$

$\mathsf{SL}[\mathcal{F}]$ (Bouyer et al. 2019a) introduces quantitative aspects in SL, but it lacks the ability to handle imperfect information inherent to the auction scenarios that we aim at modeling, where agents may ignore other agents' preferences for instance. We thus introduce $\mathsf{SLK}[\mathcal{F}]$, which extends $\mathsf{SL}[\mathcal{F}]$ with imperfect information and knowledge operators. A notable difference is that while $\mathsf{SL}[\mathcal{F}]$ considers all values to be in [0,1], we slightly generalize the setting to allow for negative values in [-1,0] as well. This allows us to naturally capture, for instance, double-sided auctions, where sellers are agents with negative types, allocations and payments, while positive value are used for buyers.

For the remainder of the paper, we fix a set of atomic propositions AP, a set of agents Ag and a set of strategy variables Var, except when stated otherwise. We let n be the number of agents in Ag.

Definition 1. Let $\mathcal{F} \subseteq \{f\colon [-1,1]^m \to [-1,1] \mid m \in \mathbb{N}\}$ be a set of functions over $[-1,1]$ of possibly different arities. The syntax of $\mathsf{SLK}[\mathcal{F}]$ is defined by the following grammar:

$$\varphi ::= p \mid \exists s_a.\, \varphi \mid (a, s_a)\varphi \mid K_a\varphi \mid f(\varphi, ..., \varphi) \mid \mathbf{X}\varphi \mid \varphi \mathbf{U} \varphi$$

where $p \in$ AP, $s_a \in$ Var, $a \in$ Ag, and $f \in \mathcal{F}$.

The intuitive reading of the operators is as follows: $\exists s_a.\, \varphi$ means that there exists a strategy for agent a such that φ holds; $(a, s_a)\varphi$ means that when strategy s_a is assigned to

agent a, φ holds; $K_a\varphi$ means that agent a knows that 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 $x \mapsto -x$.

Remark 1. In (Bouyer et al. 2019a), values are meant to represent degrees of truth value in $[0, 1]$ where 0 corresponds to "false" and 1 corresponds to "true", as in Fuzzy Logics. In this setting, negation $\neg$ is the function $x \mapsto 1 - x$. Here we consider instead values in $[-1, 1]$. This does not affect the semantics of the logic, nor the model-checking problem, and it allows us to consider negative quantities. For instance, a positive value may denote that an agent is receiving something, while a negative value represents that she is giving something. The main difference is that "false" now corresponds to -1, and negation is function $x \mapsto -x$.

A variable is *free* in formula φ if it is bound to an agent without being quantified upon, and an agent a is free in φ if φ contains a temporal operator ($\mathbf{X}$ or $\mathbf{U}$) that is not in the scope of any binding for a. The set of free variables and agents in φ is written free(φ), and a formula φ is a *sentence* if free(φ) $= \emptyset$.

The strategy quantifier $\exists s_a.\, \varphi$ quantifies on strategies *for agent* a. Except in its original formulation (Chatterjee, Henzinger, and Piterman 2010), variants of SL do not specify for which agent a strategy is at the level of strategy quantification, and this allows assigning the same strategy to different agents. However in the imperfect-information setting, we need to know with respect to which observation relation a strategy should be uniform. In (Berthon et al. 2021) this is done by parameterizing strategy quantifiers with observation relations. Here we adopt a slightly less general but more intuitive notation, by parameterizing directly with the agent who will use the strategy. This is enough for our purposes, because we will not need to share a same strategy between different agents, and we consider that the observation relation for each agent is fixed, as reflected by the following definition.

Definition 2. A *weighted concurrent game structure with imperfect information* (wCGS) is a tuple $\mathcal{G} = (\{\mathrm{Ac}_a\}_{a\in\mathrm{Ag}}, V, \delta, \ell, V_\iota, \{\sim_a\}_{a\in\mathrm{Ag}})$ where

- Ac_a is a finite set of *actions* for agent a;
- V is a finite set of *positions*;
- $\delta : V \times (\prod_{a\in\mathrm{Ag}} \mathrm{Ac}_a) \to V$ is a *transition function*;
- $\ell : V \times \mathrm{AP} \to [-1, 1]$ is a *weight function*;
- $V_\iota \subseteq V$ is a set of *initial positions*;
- $\sim_a \subseteq V \times V$ is an equivalence relation called the *observation relation* of agent a.

For a collection of objects indexed by agents in Ag, we may omit the index set and write, e.g., $\{\sim_a\}$ for $\{\sim_a\}_{a\in\mathrm{Ag}}$. We will also often write $\boldsymbol{o}$ for a tuple of objects $(o_a)_{a\in\mathrm{Ag}}$, one for each agent, and such tuples are called *profiles*. Given a profile $\boldsymbol{o}$ and $a \in \mathrm{Ag}$, we let o_a be agent a's component, and $\boldsymbol{o}_{-a}$ is $(o_b)_{b\neq a}$. Similarly, we let $\mathrm{Ag}_{-a} = \mathrm{Ag} \setminus \{a\}$.

Action profiles In a position $v \in V$, each player a chooses an action $c_a \in \mathrm{Ac}_a$, and the game proceeds to position $\delta(v, \boldsymbol{c})$ where $\boldsymbol{c}$ is the *action profile* $(c_a)_{a\in\mathrm{Ag}}$.

Plays A *play* $\pi = v_0v_1v_2...$ is an infinite sequence of positions such that for every $i \geq 0$ there exists an action profile $\boldsymbol{c}$ such that $\delta(v_i, \boldsymbol{c}) = v_{i+1}$. We write $\pi_i = v_i$ for the position at index i in play π.

Strategies A (memoryless) *strategy* for agent a is a function $\sigma : V \to \mathrm{Ac}_a$ that maps each position to an action. A strategy σ for agent a is *uniform* if, for all positions v, v' such that $v \sim_a v'$, we have $\sigma(v) = \sigma(v')$. We let Str_a be the set of uniform strategies for agent a, and $\mathit{Str} = \cup_{a\in\mathrm{Ag}}\mathit{Str}_a$. Because there are finitely many positions, *Str* is finite.

Assignments An *assignment* $\mathcal{A} : \mathrm{Ag} \cup \mathrm{Var} \to \mathit{Str}$ is a function from players and variables to strategies. For an assignment $\mathcal{A}$, an agent a and a strategy σ for a, $\mathcal{A}[a \mapsto \sigma]$ 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.

Outcomes For an assignment $\mathcal{A}$ and a position v, we let $\mathrm{Out}(\mathcal{A}, v)$ be the unique play that starts in v and follows the strategies assigned by $\mathcal{A}$. Formally, $\mathrm{Out}(\mathcal{A}, v)$ is the play $v_0v_1...$ such that $v_0 = v$ and for all $i \geq 0$, $v_{i+1} = \delta(v_i, \boldsymbol{c})$ where for all $a \in \mathrm{Ag}$, $\boldsymbol{c}_a = \mathcal{A}(a)(v_0...v_i)$.

Definition 3. Let $\mathcal{G} = (\{\mathrm{Ac}_a\}_{a\in\mathrm{Ag}}, V, \delta, \ell, V_\iota, \{\sim_a\}_{a\in\mathrm{Ag}})$ be a wCGS, and $\mathcal{A}$ an assignment. The satisfaction value $[\![\varphi]\!]^{\mathcal{G}}_{\mathcal{A}}(v) \in [-1, 1]$ of an $\mathsf{SL}[\mathcal{F}]$ formula φ in a position v is defined as follows, where π denotes $\mathrm{Out}(v, \mathcal{A})$:

$$
\begin{aligned}
[\![p]\!]^{\mathcal{G}}_{\mathcal{A}}(v) &= \ell(v, p)\\
[\![\exists s_a.\, \varphi]\!]^{\mathcal{G}}_{\mathcal{A}}(v) &= \max_{\sigma\in\mathit{Str}_a} [\![\varphi]\!]^{\mathcal{G}}_{\mathcal{A}[s_a\mapsto\sigma]}(v)\\
[\![(a, s_a)\varphi]\!]^{\mathcal{G}}_{\mathcal{A}}(v) &= [\![\varphi]\!]^{\mathcal{G}}_{\mathcal{A}[a\mapsto\mathcal{A}(s_a)]}(v)\\
[\![K_a\varphi]\!]^{\mathcal{G}}_{\mathcal{A}}(v) &= \min_{v'\sim_a v} [\![\varphi]\!]^{\mathcal{G}}_{\mathcal{A}}(v')\\
[\![f(\varphi_1,..., \varphi_m)]\!]^{\mathcal{G}}_{\mathcal{A}}(v) &= f([\![\varphi_1]\!]^{\mathcal{G}}_{\mathcal{A}}(v), ..., [\![\varphi_m]\!]^{\mathcal{G}}_{\mathcal{A}}(v))\\
[\![\mathbf{X}\varphi]\!]^{\mathcal{G}}_{\mathcal{A}}(v) &= [\![\varphi]\!]^{\mathcal{G}}_{\mathcal{A}}(\pi_1)\\
[\![\varphi_1\mathbf{U}\varphi_2]\!]^{\mathcal{G}}_{\mathcal{A}}(v) &= \sup_{i\geq 0} \min\Big([\![\varphi_2]\!]^{\mathcal{G}}_{\mathcal{A}}(\pi_i), \min_{0\leq j<i} [\![\varphi_1]\!]^{\mathcal{G}}_{\mathcal{A}}(\pi_j)\Big)
\end{aligned}
$$

If φ is a sentence, its satisfaction value does not depend on the assignment, and we write $[\![\varphi]\!]^{\mathcal{G}}(v)$ for $[\![\varphi]\!]^{\mathcal{G}}_{\mathcal{A}}(v)$ where $\mathcal{A}$ is any assignment. We also let $[\![\varphi]\!]^{\mathcal{G}} = \min_{v\in V_\iota} [\![\varphi]\!]^{\mathcal{G}}(v)$.

We can define the following classic abbreviations: $\bot := \neg\top$, $\varphi \wedge \varphi' := \neg(\neg\varphi \vee \neg\varphi')$, $\varphi \to \varphi' := \neg\varphi \vee \varphi'$, $\mathbf{F}\psi := \top\mathbf{U}\psi$, $\mathbf{G}\psi := \neg\mathbf{F}\neg\psi$ and $\forall s.\, \varphi := \neg\exists s.\, \neg\varphi$, and check that they correspond to the intuition. For instance, $\wedge$ corresponds to $\min$, $\mathbf{F}\psi$ computes the maximum of the satisfaction value of ψ over all future points in time, $\mathbf{G}\psi$ computes the minimum of these values, and $\forall s.\, \varphi$ minimizes the value of φ over all possible strategies s.

Remark 2. In the particular case where atomic propositions only take values in $\{-1, 1\}$ and $\mathcal{F}$ consists of the functions $x \mapsto -x$ (negation) and $x, y \mapsto \max(x, y)$ (disjunction), $\mathsf{SLK}[\mathcal{F}]$ corresponds to usual Boolean-valued SLK with memoryless agents.

4 Reasoning about Auction Mechanisms

We now show how $\mathsf{SLK}[\mathcal{F}]$ can be used to express all important concepts and properties from mechanism design.

4.1 Social Choice Functions

We first recall social choice functions, used to formalize how to choose one outcome among several alternatives, based on individual preferences of the agents.

Let Alt be a finite set of *alternatives*. Since our focus is on characterizing mechanisms with monetary transfers, we assume that each alternative in Alt is of the form $\alpha = (x, \boldsymbol{p})$ where $x \in \mathcal{X}$ is a *choice* from a finite set of choices $\mathcal{X}$, and $p_a \in [-1, 1]$ is a payment for agent a.

For each agent $a \in \text{Ag}$, let also $\Theta_a \subset [-1, 1]$ be a finite set of possible *types* for a. We let $\boldsymbol{\Theta} = \prod_{a \in \text{Ag}} \Theta_a$, and we note $\boldsymbol{\theta} = (\theta_a)_{a \in \text{Ag}} \in \boldsymbol{\Theta}$ for a type profile, which assigns a type θ_a to each agent a. The type θ_a of an agent a determines how she values each choice $x \in \mathcal{X}$; this is represented by a *valuation function* $\mathsf{v}_a : \mathcal{X} \times \Theta_a \to [-1, 1]$.

Example 1. For instance, in a one-sided auction with one good, a choice describes who wins the good. This can be modelled by letting $\mathcal{X} = \{-1, 1\}^{\text{Ag}}$, and considering that choice $(x_a)_{a \in \text{Ag}}$ represents that agent a wins the good if $x_a = 1$. In a two-sided auction (i.e., with buyers and sellers) with multiple copies of a good, a choice describes how many items each agent sells or buys. We then let $\mathcal{X} = [-1, 1]^{\text{Ag}}$, and a choice $(x_a)_{a \in \text{Ag}}$ means that agent a buys x_a items if $x_a \geq 0$, and sells $-x_a$ items if $x_a \leq 0$ (values are normalized in [-1,1]).

The type θ_a of agent a reflects how much the agent desires the good. In one-sided auctions with choices defined as $\mathcal{X} = \{-1, 1\}^{\text{Ag}}$ as in Example 1, one could define the valuation of agent a as $\mathsf{v}_a(x, \theta_a) = x_a \cdot \theta_a$. With this definition, a type θ_a close to 1 models an agent very interested in the good, who will have a high valuation if she receives it, the good, and a low one if she does not. A type close to -1 represents an agent who strongly does not want of this good, who has high valuation if she does not receive the good, and low valuation if she does. Finally, type 0 represents an indifferent agent, who receives valuation 0 in all possible choices.

In double-sided auctions with multiple goods (of a same type), with choices modeled as $\mathcal{X} = [-1, 1]^{\text{Ag}}$, a positive type indicates the quantity an agent is willing to buy, while a negative type denotes a selling quantity. Defining the valuation of agent a as $\mathsf{v}_a(x, \theta_a) = x_a \cdot \theta_a$, the sign of this value indicates if the outcome corresponds to what the agent intended to do (selling or buying), and its norm indicates the extent of her satisfaction or dissatisfaction.

The (quasi-linear) *utility* of agent a with type θ_a for an alternative $\alpha = (x, \boldsymbol{p})$ is defined as

$$u_a(\alpha, \theta_a) = \mathsf{v}_a(x, \theta_a) - p_a$$

That is, the utility for agent a is the difference between how much she valuates the choice x and her payment p_a.

Definition 4. A *social choice function* (SCF) $\mathsf{f} : \boldsymbol{\Theta} \to \text{Alt}$ is a function that, given a type profile $\boldsymbol{\theta}$, chooses an alternative $\mathsf{f}(\boldsymbol{\theta}) \in \text{Alt}$. We can split a social choice function as follows: $\mathsf{f} = (\mathsf{x}, \{\mathsf{p}_a\})$, where $\mathsf{x} : \boldsymbol{\Theta} \to \mathcal{X}$ is a *choice function* and for each a, $\mathsf{p}_a : \boldsymbol{\Theta} \to [-1, 1]$ is a *payment function* for agent a.

Example 2. The first-price social choice function $\mathsf{f}_{\mathsf{fp}} = (\mathsf{x}, \{\mathsf{p}_a\})$ is defined as follows. The allocation choice is defined as $\mathsf{x}(\boldsymbol{\theta}) = (x_1, ..., x_{\mathsf{n}})$, where $x_a = 1$ if θ_a is the highest type in $\boldsymbol{\theta}$ and $x_a = 0$ otherwise. In case two agents $a \neq b$ have the highest type, $x_a = 1$ iff $a \prec b$. The payment function for agent a is defined as $\mathsf{p}_a(\boldsymbol{\theta}) = x_a \cdot \theta_a$.

In the next sections, we describe how to represent mechanisms as wCGS and how to determinate whether a wCGS implements a SCF.

4.2 Mechanisms as wCGS

While social choice functions describe what is the desired outcome given agents' preferences (types), a mechanism describes agents' actions and their outcome. A mechanism consists of a description of the agents' possible actions, and a description of the alternatives that result from them. Some mechanisms are "one-shot", meaning that the final alternative is reached after each agent has chosen one action, while others may contain multiple stages. Also they may involve agents holding some private information. Weighted concurrent game structures can very naturally model complex one-shot or multi-stage mechanisms with imperfect information, and we provide a general definition of mechanisms as a class of concurrent game structures with special atomic propositions to represent types, allocations, payments etc.

Since we focus on allocation problems (in particular, auctions), choices $\mathcal{X}$ represent allocations of goods of different types from the set $Gt = \{1, ..., \mathsf{m}\}$. An *allocative choice* (Parkes and Ungar 2001) is a tuple $(x_a)_{a \in \text{Ag}} \in \mathcal{X}$ where $x_a = (x_{a,1}, ..., x_{a,\mathsf{m}})$ denotes the allocation for agent a, and $x_{a,g} \in [-1, 1]$ is the amount of goods of type g allocated to agent a (normalized in $[-1, 1]$). Note that, in the case of one type of goods ($\mathsf{m} = 1$), we obtain $\mathcal{X} = [-1, 1]^{\text{Ag}}$ as in Example 1.

Definition 5. Let $\text{AP} \supseteq \{\text{all}_{a,g}, \text{pay}_a, \text{ter}, \text{type}_a : a \in \text{Ag}, g \in Gt\}$ be a set of atomic propositions, where $\text{all}_{a,g}$, type_a, pay_a denote, respectively, how many units of the good g are allocated to agent a, the type of a, and her payment. The proposition ter specifies whether a state is terminal. A *mechanism* is a wCGS over the atomic propositions AP that satisfies the following:

(i) there is one initial position $v_\iota^{\boldsymbol{\theta}}$ for each possible type profile $\boldsymbol{\theta} \in \boldsymbol{\Theta}$;

(ii) types remain unchanged through transitions, i.e. if $\delta(v, \boldsymbol{c}) = v'$ then $\ell(v, \text{type}_a) = \ell(v', \text{type}_a)$ for each a;

(iii) each agent knows her own type: if $v \sim_a v'$, then $\ell(v, \text{type}_a) = \ell(v', \text{type}_a)$;

(iv) every play eventually reaches a *terminal position*, i.e., a sink[1] where proposition ter has value 1;

(v) in all non-terminal positions, ter has value -1.

A type profile $\boldsymbol{\theta}$ together with a strategy profile $\boldsymbol{\sigma}$ determines a unique terminal position $v(\boldsymbol{\theta}, \boldsymbol{\sigma})$, which is the terminal position reached from $v_\iota^{\boldsymbol{\theta}}$ via $\boldsymbol{\sigma}$. The values of propositions $\text{all}_{a,g}$ and pay_a in terminal position $v(\boldsymbol{\theta}, \boldsymbol{\sigma})$ encode an alternative that we write $\mathcal{G}[\boldsymbol{\theta}, \boldsymbol{\sigma}]$.

[1] A sink is a position that loops for all action profiles.

We now illustrate with an example how this formal definition of mechanisms captures complex iterative mechanisms with quantitative aspects and imperfect information.

Example 3 (Dutch auction). A Dutch auction is an iterative protocol with decreasing price; hereafter we assume the single good and single unit case. Initially, the auctioneer proposes a high asking price. This price is gradually lowered until some bidder accepts to purchase the good. The auction then ends and the object is sold to this bidder at the given price (Krishna 2009). In case of draw, the winner is determined with respect to an arbitrary order $\prec$ among the agents.

Let us fix a price decrement $\mathsf{dec} \in (0, 1]$ and, for each agent $a \in \mathrm{Ag}$, (i) a finite set of possible types $\Theta_a \subset [0, 1]$, and (ii) her real type $\theta_a \in \Theta_a$. Agent a's valuation is $\mathsf{v}_a(x) = x_a \cdot \theta_a$.

Define the mechanism $\mathcal{G}_{\mathsf{dut}} = (\{\mathrm{Ac}_a\}, V, \delta, \ell, V_\iota, \{\sim_a\})$ over $\mathrm{AP} = \{\mathsf{price}, \mathsf{all}_a, \mathsf{pay}_a, \mathsf{ter}, \mathsf{type}_a : a \in \mathrm{Ag}\}$, where:

- $\mathrm{Ac}_a = \{\mathsf{bid}, \mathsf{wait}\}$ for each $a \in \mathrm{Ag}$,
- V consists of positions of the form $\langle \mathsf{p}, \{x_a\}, \mathsf{t}, \{\theta_a\}\rangle$ with $\mathsf{p} \in \{1 - x \cdot \mathsf{dec} : 0 \leq x \leq \frac{1}{\mathsf{dec}}\}$ denoting the current price, $\mathsf{t} \in \{-1, 1\}$ denoting whether the position is terminal, $x_a \in \{0, 1\}$ specifying the allocation for agent a, and $\theta_a \in \Theta_a$ specifying her type.

In an initial position, the price starts at 1 and all the allocations are zero. That is, the set of initial positions is $V_\iota = \{\langle 1, 0, ..., 0, 0, \theta_1, ..., \theta_{\mathsf{n}}\rangle \in V\}$.

In non-terminal states, the transition function keeps decreasing the price p as long as it is above zero and every agent performs the action of waiting. If an agent a bids, the good is assigned to her ($x_a = 1$). Since there is only one unit of the good, ties are decided according to the order $\prec$. If the price remains unchanged in the transition, the state is marked as terminal ($\mathsf{t} = 1$). The transition function defines a loop for terminal states, ensuring no change occurs in the auction afterwards (see Figure 1 for a partial illustration). Formally, for each position $v = \langle \mathsf{p}, \{x_a\}, \mathsf{t}, \{\theta_a\}\rangle$ and joint action $\boldsymbol{c} = (c_a)_{a \in \mathrm{Ag}}$, transition $\delta(v, \boldsymbol{c})$ is defined as follows:

- If $\mathsf{t} = -1$, $\delta(v, \boldsymbol{c}) = \langle \mathsf{p}', \{x'_a\}, \mathsf{t}', \{\theta_a\}\rangle$ where:

$$\mathsf{p}' = \begin{cases} \mathsf{p} - \mathsf{dec} & \text{if } \mathsf{p} - \mathsf{dec} \geq 0 \text{ and} \\ & c_a = \mathsf{wait} \text{ for all } a \in \mathrm{Ag} \\ \mathsf{p} & \text{otherwise} \end{cases}$$

$$x'_a = \begin{cases} 1 & \text{if } c_a = \mathsf{bid} \text{ and for all } a' \neq a \\ & \text{either } c_{a'} = \mathsf{wait} \text{ or } a \prec a' \\ 0 & \text{otherwise} \end{cases}$$

$$\mathsf{t}' = \begin{cases} 1 & \text{if } \mathsf{p}' = \mathsf{p}, \\ -1 & \text{otherwise} \end{cases}$$

- Otherwise, $\delta(v, \boldsymbol{c}) = v$.

For each $v = \langle \mathsf{p}, \{x_a\}, \mathsf{t}, \{\theta_a\}\rangle$ and each $a \in \mathrm{Ag}$, the weight function is defined as follows: $\ell(v, \mathsf{price}) = \mathsf{p}$, $\ell(v, \mathsf{all}_a) = x_a$, $\ell(v, \mathsf{pay}_a) = x_a \cdot \mathsf{p}$, $\ell(v, \mathsf{ter}) = \mathsf{t}$, and $\ell(v, \mathsf{type}_a) = \theta_a$.

Finally, for each agent $a \in \mathrm{Ag}$ and for any two positions $v = \langle \mathsf{p}, \{x_a\}, \mathsf{t}, \{\theta_a\}\rangle$ and $v' = \langle \mathsf{p}', \{x'_a\}, \mathsf{t}', \{\theta'_a\}_{a \in \mathrm{Ag}}\rangle$ in V, the observation relation $\sim_a$ is defined as follows:

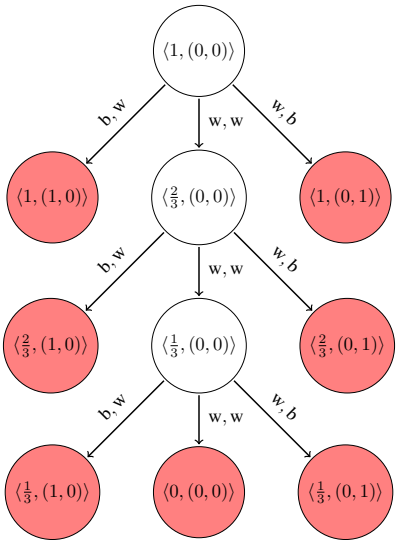

Figure 1: Part of the mechanism for the Dutch auction with two agents and decrement $\mathsf{dec} = \frac{1}{3}$. Terminal states are in red. We only represent one initial state and thus we omit types, which are the same in all states. Action bid is written b and wait is w. Finally, we did not represent ties (bid, bid) or loops.

$v \sim_a v'$ if (i) $\mathsf{p} = \mathsf{p}'$; (ii) $x_b = x'_b$, for all $b \in \mathrm{Ag}$; (iii) $\theta_a = \theta'_a$; and (iv) $\mathsf{t} = \mathsf{t}'$.

Observation relations $\sim_a$ capture the fact that agents do not know other agents' preferences, and thus their actions cannot depend on them. This is reflected in $\mathsf{SLK}[\mathcal{F}]$ by the notion of uniform strategy. It is out of the scope of this work to consider probabilistic beliefs about agents' preferences.

We now show how important concepts of mechanism design can be expressed in $\mathsf{SLK}[\mathcal{F}]$.

4.3 Implementation of Social Choice Functions

Fix a mechanism $\mathcal{G}$. The goal of an agent is to maximize her utility, which is equal to the value of the $\mathsf{SLK}[\mathcal{F}]$ formula

$$\mathsf{util}_a := \mathsf{v}_a((\mathsf{all}_{1,1}, ..., \mathsf{all}_{\mathsf{n},\mathsf{m}}), \mathsf{type}_a) - \mathsf{pay}_a$$

in the terminal situation.

In this section we assume that $\mathcal{F}$ contains the function

$$- : (x, y) \mapsto \min(1, \max(-1, x - y))$$

as well as the valuation function v_a for each agent a, and for readability we use the infix notation $x - y$ in the formula. We also assume that $\mathcal{F}$ contains the comparison function

$$\leq : (x, y) \mapsto \begin{cases} 1 & \text{if } x \leq y, \\ -1 & \text{otherwise,} \end{cases}$$

the comparison function $<$ (defined similarly, with $<$ instead of $\leq$), the equality function

$$= : (x, y) \mapsto \begin{cases} 1 & \text{if } x = y, \\ -1 & \text{otherwise,} \end{cases}$$

and the n-ary sum function

$$\sum : x_1, ..., x_{\mathsf{n}} \mapsto \min(1, \max(-1, \sum_k x_k))$$

Finally we assume that types, allocations, payments and valuations are normalized so that all values remain in $[-1, 1]$.

Before defining what it means for a mechanism to implement a social choice function, we recall two classical concepts of equilibria, Nash equilibria and dominant strategy equilibria, classically used to define implementation.

Nash Equilibria A strategy profile $\sigma = (\sigma_a)_{a\in \mathrm{Ag}}$ is a *Nash equlibrium* (NE) if no agent can increase her utility with a unilateral change of strategy (Parkes and Ungar 2001). Just as Strategy Logic can express Nash equilibria for Boolean objectives, $\mathsf{SLK}[\mathcal{F}]$ can express Nash equilibria with quantitative objectives. Define the formula

$$\mathrm{NE}(\boldsymbol{s}) := \bigwedge_{a\in \mathrm{Ag}} \forall t.\, \big[(\mathrm{Ag}_{-a}, \boldsymbol{s}_{-a})(a,t)\mathbf{F}(\mathrm{ter} \wedge \mathrm{util}_a) \leq (\mathrm{Ag}, \boldsymbol{s})\mathbf{F}(\mathrm{ter} \wedge \mathrm{util}_a)\big]$$

where $\boldsymbol{s} = (s_a)_{a\in \mathrm{Ag}}$ is a profile of strategy variables. The following, stated in (Bouyer et al. 2019b), establishes that this formula is correct.

Lemma 1. *For every assignment $\mathcal{A}$, we have that $[\![NE]\!]^{\mathcal{G}}_{\mathcal{A}}(v) = 1$ iff $(\mathcal{A}(s_a))_{a\in \mathrm{Ag}}$ is a NE in $\mathcal{G}$ from v.*

Dominant Strategy Equilibria A strategy σ_a is a *dominant strategy* (DS) for agent a if it weakly maximizes her utility, for all possible strategies of other agents. Define the formula

$$\mathrm{DS}(s_a, a) := \forall \boldsymbol{t}.\big[(a, t_a)(\mathrm{Ag}_{-a}, \boldsymbol{t}_{-a})\mathbf{F}(\mathrm{ter} \wedge \mathrm{util}_a) \leq (a, s_a)(\mathrm{Ag}_{-a}, \boldsymbol{t}_{-a})\mathbf{F}(\mathrm{ter} \wedge \mathrm{util}_a)\big]$$

For an assignment $\mathcal{A}$, it holds that $[\![\mathrm{DS}(s_a, a)]\!]^{\mathcal{G}}_{\mathcal{A}}(v) = 1$ iff $\mathcal{A}(s_a)$ is a dominant strategy for a in $\mathcal{G}$ from position v.

A strategy profile $\sigma = (\sigma_a)_{a\in \mathrm{Ag}}$ is a *dominant strategy equilibrium* (DSE) if each σ_a is a dominant strategy for agent a (Nisan et al. 2007). Define

$$\mathrm{DSE}(\boldsymbol{s}) := \bigwedge_{a\in \mathrm{Ag}} \mathrm{DS}(s_a, a)$$

Similarly to Nash equilibria, the following holds.

Lemma 2. *For every assignment $\mathcal{A}$, we have that $[\![DSE(\boldsymbol{s})]\!]^{\mathcal{G}}_{\mathcal{A}}(v) = 1$ iff $(\mathcal{A}(s_a))_{a\in \mathrm{Ag}}$ is a DSE in $\mathcal{G}$ from v.*

Remark 3. For verifying the uniqueness of a Nash equilibrium, we can check whether there exist two assignments $\mathcal{A} \neq \mathcal{A}'$ such that $[\![\mathrm{DSE}(\boldsymbol{s})]\!]^{\mathcal{G}}_{\mathcal{A}}(v) = [\![\mathrm{DSE}(\boldsymbol{s})]\!]^{\mathcal{G}}_{\mathcal{A}'}(v) = 1$.

Implementation Informally, a mechanism implements a social choice function if the alternative chosen in *equilibrium* strategies is the same as the one chosen by the social choice function, for all possible agent preferences; in case of multiple equilibria it is required that there exist one equilibrium that agrees with the social choice function (Parkes and Ungar 2001). Different equilibrium concepts may be used, including Nash equilibria and dominant strategy equilibrium.

Definition 6. Let $\mathrm{E} \in \{\mathrm{NE}, \mathrm{DSE}\}$ be a solution concept and f a social choice function. A mechanism $\mathcal{G}$ *E-implements* f if for all type profiles $\boldsymbol{\theta} \in \boldsymbol{\Theta}$ there exists an E-equilibrium $\boldsymbol{\sigma}$ in $\mathcal{G}$ from $v^{\boldsymbol{\theta}}_{\iota}$ such that $\mathcal{G}[\boldsymbol{\theta}, \boldsymbol{\sigma}] = \mathsf{f}(\boldsymbol{\theta})$.

Let us again consider Dutch auctions but from the social choice function perspective.

Example 4. Under the assumption that a Nash equilibrium exists, the Dutch auction (see Example 3) is known to implement the first-price social choice function (Krishna 2009) introduced in Example 2.

For a social choice function $\mathsf{f} = (\mathsf{x}, \{\mathsf{p}_a\})$ and a type profile $\boldsymbol{\theta}$, define the $\mathsf{SLK}[\mathcal{F}]$ formula

$$\varphi_{\mathsf{f}(\boldsymbol{\theta})} := \bigwedge_{a\in \mathrm{Ag}} (\mathrm{pay}_a = p_a \wedge \bigwedge_{g\in Gt} \mathrm{all}_{a,g} = x_{a,g})$$

where $\mathsf{x}(\boldsymbol{\theta}) = ((x_{a,1}, ..., x_{a,\mathsf{m}}))_{a\in \mathrm{Ag}}$, $\mathsf{p}_a(\boldsymbol{\theta}) = p_a$, and values p_a, $x_{a,g}$ are constants (0-ary functions) in $\mathcal{F}$.

Define also, for $\mathrm{E} \in \{\mathrm{NE}, \mathrm{DSE}\}$,

$$\varphi_{\mathrm{impl}}(\mathsf{f}, \mathrm{E}, \boldsymbol{\theta}) := \exists \boldsymbol{s}.\, \mathrm{E}(\boldsymbol{s}) \wedge \mathbf{F}(\mathrm{ter} \wedge \varphi_{\mathsf{f}(\boldsymbol{\theta})})$$

This formula says that there exists an E-equilibrium that leads to choice $\mathsf{f}(\boldsymbol{\theta})$. It can thus be used to express that a mechanism implements a given social choice function:

Theorem 1. *A mechanism $\mathcal{G}$ E-implements a SCF f iff for every type profile $\boldsymbol{\theta} \in \boldsymbol{\Theta}$, $[\![\varphi_{impl}(\mathsf{f}, E, \boldsymbol{\theta})]\!]^{\mathcal{G}}(v^{\boldsymbol{\theta}}_{\iota}) = 1$.*

Proof. Fix a solution concept $\mathrm{E} \in \{\mathrm{NE}, \mathrm{DSE}\}$, a social choice function $\mathsf{f} = (\mathsf{x}, \{\mathsf{p}_a\})$, a mechanism $\mathcal{G}$, any assignment $\mathcal{A}$ and any type profile $\boldsymbol{\theta} \in \boldsymbol{\Theta}$. For each agent $a \in \mathrm{Ag}$ and good type $g \in Gt$, let $x_{a,g}$ and p_a be constants in [-1, 1] denoting the alternative chosen by f, that is $\mathsf{x}(\boldsymbol{\theta}) = ((x_{a,1}, ..., x_{a,\mathsf{m}}))_{a\in \mathrm{Ag}}$, and $\mathsf{p}_a(\boldsymbol{\theta}) = p_a$.

Assume $\mathcal{G}$ E-implements f. By definition there exists a strategy profile $\boldsymbol{\sigma}$ that is an E-equilibrium solution in $\mathcal{G}$ from $v^{\boldsymbol{\theta}}_{\iota}$ and such that $\mathcal{G}[\boldsymbol{\theta}, \boldsymbol{\sigma}] = \mathsf{f}(\boldsymbol{\theta})$. It follows that:

First, because the $\mathsf{SLK}[\mathcal{F}]$ formula $\mathbf{E}(\boldsymbol{s})$ correctly characterises $\mathbf{E}$-equilibria (lemma 1 and 2), letting $\mathcal{A}_{\boldsymbol{\sigma}} : s_a \mapsto \sigma_a$ we have that $[\![\mathrm{E}(\boldsymbol{s})]\!]^{\mathcal{G}}_{\mathcal{A}_{\boldsymbol{\sigma}}}(v^{\boldsymbol{\theta}}_{\iota}) = 1$.

Second, the fact that $\mathcal{G}[\boldsymbol{\theta}, \boldsymbol{\sigma}] = \mathsf{f}(\boldsymbol{\theta})$ implies that in the terminal position $v^{\mathrm{ter}} = v(\boldsymbol{\theta}, \boldsymbol{\sigma})$ (which is reached from $v^{\boldsymbol{\theta}}_{\iota}$ via $\boldsymbol{\sigma}$), we have $\ell(v^{\mathrm{ter}}, \mathrm{all}_{a,g}) = x_{a,g}$ and $\ell(v^{\mathrm{ter}}, \mathrm{pay}_a) = p_a$, for each $a \in \mathrm{Ag}$ and $g \in Gt$. Therefore, we have $[\![\bigwedge_{a\in \mathrm{Ag}}(\mathrm{pay}_a = p_a \wedge \bigwedge_{g\in Gt} \mathrm{all}_{a,g} = x_{a,g})]\!]^{\mathcal{G}}_{\mathcal{A}_{\boldsymbol{\sigma}}}(v^{\mathrm{ter}}) = 1$ or simply $[\![\varphi_{\mathsf{f}(\boldsymbol{\theta})}]\!]^{\mathcal{G}}_{\mathcal{A}_{\boldsymbol{\sigma}}}(v^{\mathrm{ter}}) = 1$. By the semantics of $\mathbf{F}$, it follows that $[\![\mathbf{F}(\mathrm{ter} \wedge \varphi_{\mathsf{f}(\boldsymbol{\theta})})]\!]^{\mathcal{G}}_{\mathcal{A}_{\boldsymbol{\sigma}}}(v^{\boldsymbol{\theta}}_{\iota}) = 1$.

Therefore, $[\![\exists \boldsymbol{s}.\, \mathrm{E}(\boldsymbol{s}) \wedge \mathbf{F}(\mathrm{ter} \wedge \varphi_{\mathsf{f}(\boldsymbol{\theta})})]\!]^{\mathcal{G}}_{\mathcal{A}}(v^{\boldsymbol{\theta}}_{\iota}) = 1$ (the maximal value 1 is attained for strategy profile $\boldsymbol{\sigma}$) and $[\![\varphi_{\mathrm{impl}}(\mathsf{f}, \mathrm{E}, \boldsymbol{\theta})]\!]^{\mathcal{G}}_{\mathcal{A}}(v^{\boldsymbol{\theta}}_{\iota}) = 1$.

Conversely, assume $[\![\exists \boldsymbol{s}.\, \mathrm{E}(\boldsymbol{s}) \wedge \mathbf{F}(\mathrm{ter} \wedge \varphi_{\mathsf{f}(\boldsymbol{\theta})})]\!]^{\mathcal{G}}_{\mathcal{A}}(v^{\boldsymbol{\theta}}_{\iota}) = 1$. By the semantics of the strategy quantifier, there exists a strategy profile $\boldsymbol{\sigma}$ such that $[\![\mathrm{E}(\boldsymbol{s}) \wedge \mathbf{F}(\mathrm{ter} \wedge \varphi_{\mathsf{f}(\boldsymbol{\theta})})]\!]^{\mathcal{G}}_{\mathcal{A}_{\boldsymbol{\sigma}}}(v^{\boldsymbol{\theta}}_{\iota}) = 1$, where $\mathcal{A}_{\boldsymbol{\sigma}} : s_a \mapsto \sigma_a$. It follows that $[\![\mathrm{E}(\boldsymbol{s})]\!]^{\mathcal{G}}_{\mathcal{A}_{\boldsymbol{\sigma}}}(v^{\boldsymbol{\theta}}_{\iota}) = 1$ and $[\![\mathbf{F}(\mathrm{ter} \wedge \varphi_{\mathsf{f}(\boldsymbol{\theta})})]\!]^{\mathcal{G}}_{\mathcal{A}_{\boldsymbol{\sigma}}}(v^{\boldsymbol{\theta}}_{\iota}) = 1$. The former implies that $\boldsymbol{\sigma}$ is an E-equilibrium, and since ter has value -1 in all non-terminal positions, the latter implies that in the terminal position $v^{\mathrm{ter}} = v(\boldsymbol{\theta}, \boldsymbol{\sigma})$ we have $[\![\varphi_{\mathsf{f}(\boldsymbol{\theta})}]\!]^{\mathcal{G}}_{\mathcal{A}_{\boldsymbol{\sigma}}}(v^{\mathrm{ter}}) = 1$ This in turn means that $\mathcal{G}[\boldsymbol{\theta}, \boldsymbol{\sigma}] = \mathsf{f}(\boldsymbol{\theta})$, hence $\mathcal{G}$ E-implements f.

Notice that if there is no $\boldsymbol{\sigma}$ such that $\boldsymbol{\sigma}$ is an E-equilibrium solution in $\mathcal{G}$, we have that $\mathcal{G}$ does not E-implements the SCF f, and $[\![\varphi_{\mathrm{impl}}(\mathsf{f}, \mathrm{E}, \boldsymbol{\theta})]\!]^{\mathcal{G}}_{\mathcal{A}}(v^{\boldsymbol{\theta}}_{\iota}) = -1$. □

In the next section, we show how to express and verify properties of social choice functions by evaluating $\mathsf{SLK}[\mathcal{F}]$ formulas on mechanisms that implement them. We will use the following parameterized formula to capture in a mechanism some equilibrium that implements the social function, and check a property of the resulting alternative. For a social choice function f, a type profile $\boldsymbol{\theta} \in \boldsymbol{\Theta}$, an equilibrium type $\mathrm{E} \in \{\mathrm{NE}, \mathrm{DSE}\}$ and a formula φ expressing a property of the final alternative, define

$$\text{Capture-alt}(\mathsf{f}, \mathrm{E}, \boldsymbol{\theta}, \varphi) := \exists \boldsymbol{s}.\, \mathrm{E}(\boldsymbol{s}) \wedge \mathbf{F}(\mathrm{ter} \wedge \varphi_{\mathsf{f}(\boldsymbol{\theta})} \wedge \varphi)$$

4.4 Mechanism Properties

We show how $\mathsf{SLK}[\mathcal{F}]$ can express a variety of important notions in mechanism design.

A *direct-revelation* mechanism, such as Vickrey auction, is a non-iterative protocol where the agents' possible actions are their possible types. That is, a mechanism $\mathcal{G}$ is a direct revelation one if initial positions lead directly to terminal positions, and $\mathrm{Ac}_a = \Theta_a$ for each a. Equivalently, it is a social choice function, as it maps type profiles to alternatives, and every social choice function can be seen as a direct-revelation mechanism (Jackson 2009).

Strategy proofness One of the core challenges in mechanism design is to ensure that an agent would prefer "telling the truth" by reporting her real type rather than any other (Nisan et al. 2007). Mechanisms that ensure this property are called *strategy-proof* (SP) or *incentive-compatible*.

In a direct-revelation mechanism $\mathcal{G}$, we let $\hat{\theta}_a$ be the truth-revealing strategy for a, defined as $\hat{\theta}_a(v_\iota^{\boldsymbol{\theta}}) = \theta_a$.

Definition 7. A direct-revelation mechanism $\mathcal{G}$ is *strategy-proof* if $(\hat{\theta}_a)_{a \in \mathrm{Ag}}$ is a dominant strategy equilibrium from $v_\iota^{\boldsymbol{\theta}}$, for all $\boldsymbol{\theta} \in \boldsymbol{\Theta}$.

Strategy proofness of a direct-revelation mechanism $\mathcal{G}$ can be expressed in $\mathsf{SLK}[\mathcal{F}]$ by verifying whether the $\mathsf{SLK}[\mathcal{F}]$-formula $\mathrm{DSE}(\boldsymbol{s})$ characterizing dominant strategy equilibrium has satisfaction value 1 on $\mathcal{G}$, where $\boldsymbol{s}$ denotes the joint strategy in which each agent truthfully reports her type. The following holds:

Proposition 1. *A direct-revelation mechanism $\mathcal{G}$ is strategy proof iff $[\![DSE(\boldsymbol{s})]\!]^{\mathcal{G}}_{\mathcal{A}}(v_\iota^{\boldsymbol{\theta}}) = 1$ for all $\boldsymbol{\theta} \in \boldsymbol{\Theta}$, where $\mathcal{A}(s_a) = \hat{\theta}_a$ for each a.*

Proof. Fix a direct revelation mechanism $\mathcal{G}$. We have that $\mathrm{Ac}_a = \Theta_a$ for each a, and $\mathcal{G}$ is strategy-proof iff, for each initial position $v_\iota^{\boldsymbol{\theta}}$, the truth revealing strategy $\hat{\theta}_a$ is a dominant strategy for each $a \in \mathrm{Ag}$. By the semantics of formula $\mathrm{DS}(s)$, for any type profile $\boldsymbol{\theta}$, each strategy $\hat{\theta}_a$ is dominant from $v_\iota^{\boldsymbol{\theta}}$ iff $[\![\mathrm{DS}(s)]\!]^{\mathcal{G}}_{\mathcal{A}_{\hat{\theta}_a}}(v_\iota^{\boldsymbol{\theta}}) = 1$, where $\mathcal{A}_{\hat{\theta}_a} : s \mapsto \hat{\theta}_a$. So for a type profile $\boldsymbol{\theta}$, all strategies $\hat{\theta}_a$ are dominant from $v_\iota^{\boldsymbol{\theta}}$ iff $[\![\mathrm{DSE}(\boldsymbol{s})]\!]^{\mathcal{G}}_{\mathcal{A}}(v_\iota^{\boldsymbol{\theta}}) = 1$, where $\mathcal{A} : s_a \mapsto \hat{\theta}_a$ for all a. □

Individual rationality Individual rationality (IR) expresses the idea that an agent has an incentive to participate (Parkes and Ungar 2001), that is, she can ensure to always get non-negative utility. Hereafter we express in $\mathsf{SLK}[\mathcal{F}]$ the notion of (ex-post) individual rationality (Nisan et al. 2007).

Definition 8. A SCF $\mathsf{f} = (\mathsf{x}, \{\mathsf{p}_a\})$ is *individually rational* if for every $\boldsymbol{\theta} \in \boldsymbol{\Theta}$, $\mathsf{v}_a(\mathsf{x}(\boldsymbol{\theta})) - \mathsf{p}_a(\boldsymbol{\theta}) \geq 0$ for each agent a.

Let us define the following formula:

$$\mathrm{IR} := \bigwedge_{a \in \mathrm{Ag}} 0 \leq \mathrm{util}_a$$

Given a mechanism that E-implements a SCF f, checking that f satisfies IR amounts to checking that formula IR has satisfaction value one in the E-equilibrium that implements f, for every possible type profile $\boldsymbol{\theta}$.

Proposition 2. *Let f be a SCF, $E \in \{NE, DSE\}$, and $\mathcal{G}$ a mechanism that E-implements f. f is individually rational iff $[\![Capture\text{-}alt(\mathsf{f}, E, \boldsymbol{\theta}, IR)]\!]^{\mathcal{G}}(v_\iota^{\boldsymbol{\theta}}) = 1$ for all $\boldsymbol{\theta} \in \boldsymbol{\Theta}$.*

Proof. Fix a solution concept $\mathrm{E} \in \{\mathrm{NE}, \mathrm{DSE}\}$, a social choice function f, a mechanism $\mathcal{G}$ that E-implements f, any assignment $\mathcal{A}$ and any type profile $\boldsymbol{\theta} \in \boldsymbol{\Theta}$.

Assume f is individually rational. Because $\mathcal{G}$ implements f there exists a strategy profile $\boldsymbol{\sigma}$ that is an E equilibrium from $v_\iota^{\boldsymbol{\theta}}$ and such that $\mathcal{G}[v_\iota^{\boldsymbol{\theta}}, \boldsymbol{\sigma}] = \mathsf{f}(\boldsymbol{\theta})$. Let $v^{\mathrm{ter}} = v(\boldsymbol{\theta}, \boldsymbol{\sigma})$. Since f is individually rational, we have that $[\![\mathrm{IR}]\!]^{\mathcal{G}}_{\mathcal{A}}(v^{\mathrm{ter}}) = 1$. Therefore, using $\boldsymbol{\sigma}$ as witness, we obtain that $[\![\exists \boldsymbol{s}.\, \mathrm{E}(\boldsymbol{s}) \wedge \mathbf{F}(\mathrm{ter} \wedge \varphi_{\mathsf{f}(\boldsymbol{\theta})} \wedge \mathrm{IR})]\!]^{\mathcal{G}}_{\mathcal{A}}(v_\iota^{\boldsymbol{\theta}}) = 1$.

The converse is proved in a similar way. □

Efficiency A social choice function is efficient (EF) if it chooses the allocation maximizing the social welfare, i.e., the total value over all agents (Parkes and Ungar 2001).

Definition 9. A social choice function $\mathsf{f} = (\mathsf{x}, \{\mathsf{p}_a\})$ is *allocatively efficient* if for all $\boldsymbol{\theta} \in \boldsymbol{\Theta}$,

$$\sum_{a \in \mathrm{Ag}} \mathsf{v}_a(\mathsf{x}(\boldsymbol{\theta}), \theta_a) = \max_{x \in \mathcal{X}} \sum_{a \in \mathrm{Ag}} \mathsf{v}_a(x, \theta_a)$$

Define formula

$$\mathrm{Eff} := \sum_{a \in \mathrm{Ag}} \mathsf{v}_a(\mathrm{all}_{1,1}, ..., \mathrm{all}_{\mathsf{n},\mathsf{m}}, \mathrm{type}_a) = \mathrm{maxv}_{\boldsymbol{\theta}}$$

where, for each $\boldsymbol{\theta}$, $\mathrm{maxv}_{\boldsymbol{\theta}} = \max_{x \in \mathcal{X}} \sum_{a \in \mathrm{Ag}} \mathsf{v}_a(x, \theta_a)$ is a constant in $\mathcal{F}$. In a terminal position, it means that the social welfare of the allocation it encodes is maximal.

The following proposition shows how one can determinate whether a SCF f is efficient by verifying the satisfaction value of the formula Eff in a mechanism that implements f.

Proposition 3. *Let f be a SCF, $E \in \{NE, DSE\}$, and $\mathcal{G}$ a mechanism that E-implements f. f is allocatively efficient iff $[\![Capture\text{-}alt(\mathsf{f}, E, \boldsymbol{\theta}, Eff)]\!]^{\mathcal{G}}(v_\iota^{\boldsymbol{\theta}}) = 1$ for all $\boldsymbol{\theta} \in \boldsymbol{\Theta}$.*

Proof. Analogous to the proof of Proposition 2. □

Budget-Balance Budget-balance focuses on the monetary transfer between buyers and sellers. Strong budget-balance (SBB) requires strict balance in this transfer. The no-deficit condition, or weak budget-balance (WBB), characterizes no monetary loss. We recall the notions of strong and weak budget balance (Parkes and Ungar 2001):

Definition 10. A social choice function $\mathsf{f} = (\mathsf{x}, \{\mathsf{p}_a\})$ is *strongly budget-balanced* (resp., *weakly budget-balanced*) if $\sum_{a \in \mathrm{Ag}} p_a(\boldsymbol{\theta}) = 0$ (resp., $\sum_{a \in \mathrm{Ag}} p_a(\boldsymbol{\theta}) \geq 0$) for all $\boldsymbol{\theta} \in \boldsymbol{\Theta}$.

Define formula

$$\text{SBB}(\mathsf{f}, \mathrm{E}, \boldsymbol{\theta}) := \exists \boldsymbol{s}.\, \mathrm{E}(\boldsymbol{s}) \wedge \mathbf{F}\big[\text{ter} \wedge \varphi_{\mathsf{f}(\boldsymbol{\theta})} \wedge 0 = \sum_{a \in \text{Ag}} \text{pay}_a\big]$$

and define $\text{WBB}(\mathsf{f}, \mathrm{E}, \boldsymbol{\theta})$ similarly, with $\leq$ instead of $=$. Again, we can prove that

Proposition 4. *Let* f *be a SCF,* $E \in \{NE, DSE\}$*, and* $\mathcal{G}$ *a mechanism that E-implements* f*. It holds that* f *is SBB iff* $[\![\textit{Capture-alt}(\mathsf{f}, E, \boldsymbol{\theta}, \textit{SBB})]\!]^{\mathcal{G}}(v_\iota^{\boldsymbol{\theta}}) = 1$ *for all* $\boldsymbol{\theta} \in \boldsymbol{\Theta}$*, and similarly for WBB.*

Example 5. A Dutch auction is WBB, because the bid price is non-negative, and it is IR because the strategy of waiting in every position leads to zero utility, and thus any equilibrium would not have a negative utility. One can check that in the mechanism $\mathcal{G}_{\text{dut}}$ from Example 3, with the first-price social choice function f_{fp}, for any type profile $\boldsymbol{\theta}$ we have $[\![\text{Capture-alt}(\mathsf{f}_{\text{fp}}, \text{NE}, \boldsymbol{\theta}, \text{IR})]\!]^{\mathcal{G}_{\text{dut}}}(v_\iota^{\boldsymbol{\theta}}) = 1$ and $[\![\text{Capture-alt}(\mathsf{f}_{\text{fp}}, \text{NE}, \boldsymbol{\theta}, \text{WBB})]\!]^{\mathcal{G}_{\text{dut}}}(v_\iota^{\boldsymbol{\theta}}) = 1$.

Pareto Optimality A social choice function is *Pareto optimal* (PO) if it chooses an alternative for which no other alternative is strongly preferred by at least one agent, and weakly preferred by all others (Parkes and Ungar 2001). Formally:

Definition 11. A social choice function $\mathsf{f} = (\mathsf{x}, \{\mathsf{p}_a\})$ is *Pareto optimal* if, for all $\boldsymbol{\theta} \in \boldsymbol{\Theta}$, for all $a \in \text{Ag}$ and for all $\alpha \neq \mathsf{f}(\boldsymbol{\theta})$, if $u_a(\alpha, \theta_a) > u_a(\mathsf{f}(\boldsymbol{\theta}), \theta_a)$ then there exists an agent $b \in Ag$ such that $u_b(\alpha, \theta_b) < u_b(\mathsf{f}(\boldsymbol{\theta}), \theta_b)$.

For every alternative $\alpha \in \text{Alt}$, every type profile $\boldsymbol{\theta}$ and agent a, let $\text{utilalt}_{a,\alpha} : \theta_a \mapsto u_a(\alpha, \theta_a)$ be a function in $\mathcal{F}$. Define formula (recall formula util_a, defined in Section 4.3):

$$\text{PO} := \bigwedge_{a \in \text{Ag}, \alpha \in \text{Alt}} \big(\text{util}_a < \text{utilalt}_{\alpha,a}(\text{type}_a) \rightarrow (\bigvee_{b \in \text{Ag}} \text{utilalt}_{\alpha,b}(\text{type}_a) < \text{util}_b)\big)$$

As for Proposition 2, we can prove:

Proposition 5. *Let* f *be a SCF,* $E \in \{NE, DSE\}$*, and* $\mathcal{G}$ *a mechanism that E-implements* f*. It holds that* f *is PO iff* $[\![\textit{Capture-alt}(\mathsf{f}, E, \boldsymbol{\theta}, \textit{PO})]\!]^{\mathcal{G}}(v_\iota^{\boldsymbol{\theta}}) = 1$ *for all* $\boldsymbol{\theta} \in \boldsymbol{\Theta}$*.*

4.5 Revenue Benchmarks with Knowledge

Let us now go further by considering the interplay between the agents' epistemic state and mechanism properties. Hereafter, we focus on the *auctioneer's revenue*, i.e., the total payment among the agents. Guaranteeing a revenue is an important MD issue (Krishna 2009). To address this problem, Chen and Micali (2015; 2016) exhibit an auction mechanism based on "possibilistic beliefs", i.e., beliefs an agent may hold about other agents' types. The mechanism then sets a clear link between the revenue and the agents' epistemic state. We show that this can be represented in a natural way in $\mathsf{SLK}[\mathcal{F}]$.

Second-belief benchmark Let us consider the *second-belief benchmark* (Chen and Micali 2015) for single good mechanisms. Given a set of possible type profiles $\boldsymbol{\Theta}$, a set $\mathcal{B}_a \subset \boldsymbol{\Theta}$ denotes a belief for agent a about all agents' types.

Given a tuple $S \in [-1, 1]^{\mathsf{n}}$, let $\text{2nd-max}(S)$ be the second maximum value in S, and assume that $\text{2nd-max} \in \mathcal{F}$. Given a correct belief profile $\boldsymbol{\mathcal{B}}$ (i.e., a profile in which the true type is considered possible), the second-belief benchmark (for single-good auctions) is defined as follows:

$$2^{\text{nd}}(\boldsymbol{\mathcal{B}}) := \text{2nd-max}(\text{smv}_{a_1}(\boldsymbol{\mathcal{B}}), ..., \text{smv}_{a_n}(\boldsymbol{\mathcal{B}}))$$

where $\text{smv}_a(\boldsymbol{\mathcal{B}}) := \min_{\boldsymbol{\theta} \in B_a}(\max_{b \in \text{Ag}}(\theta_b))$ denotes the *sure maximum value* according to a.

Let $\mathcal{G}$ be a mechanism. To each position $v \in V$ we can associate a correct belief $\mathcal{B}_a(v)$ for each agent a as follows: $\mathcal{B}_a(v) := (\{\ell(v', \text{type}_b) : v' \sim_a v\})_{b \in \text{Ag}}$. We then let $\boldsymbol{\mathcal{B}}(v) = (\mathcal{B}_a(v))_{a \in \text{Ag}}$. The sure maximum value for an agent and the second-belief benchmark in a position correspond to the semantics of the following epistemic $\mathsf{SLK}[\mathcal{F}]$-formulas:

$$\varphi_a^{\text{smv}} := K_a \max_{a' \in \text{Ag}}(\text{type}_{a'})$$

$$\varphi_{\text{2nd}} := \text{2nd-max}(\varphi_{a_1}^{\text{smv}}, ..., \varphi_{a_n}^{\text{smv}})$$

It follows directly that:

Proposition 6. *Given a mechanism* $\mathcal{G}$*, a position* v *and a belief profile* $\boldsymbol{\mathcal{B}}(v)$*, it holds that* $[\![\varphi_{2nd}]\!]^{\mathcal{G}}(v) = 2^{nd}(\boldsymbol{\mathcal{B}}(v))$.

Proof. Fix an assignment $\mathcal{A}$. We have that $[\![\varphi_{\text{2nd}}]\!]^{\mathcal{G}}_{\mathcal{A}}(v) = [\![\text{2nd-max}(\varphi_{a_1}^{\text{smv}}, ..., \varphi_{a_n}^{\text{smv}})]\!]^{\mathcal{G}}_{\mathcal{A}}(v)$. By the semantics of K_a, φ_a^{smv} denotes the minimum value of $[\![\max_{a \in \text{Ag}}(\text{type}_a)]\!]^{\mathcal{G}}_{\mathcal{A}}(v')$, for all $v' \sim_a v$. Since $\mathcal{B}_a(v) = (\{\ell(v', \text{type}_b) : v' \sim_a v\})_{b \in \text{Ag}}$, it holds that $[\![\varphi_a^{\text{smv}}]\!]^{\mathcal{G}}_{\mathcal{A}}(v) = \min_{\boldsymbol{\theta} \in B_a}(\max_{b \in \text{Ag}}(\theta_b))$. Therefore, $[\![\varphi_{\text{2nd}}]\!]^{\mathcal{G}}_{\mathcal{A}}(v) = \text{2nd-max}(\text{smv}_{a_1}(\boldsymbol{\mathcal{B}}), ..., \text{smv}_{a_n}(\boldsymbol{\mathcal{B}}))$ or simply $[\![\varphi_{\text{2nd}}]\!]^{\mathcal{G}}_{\mathcal{A}}(v) = 2^{\text{nd}}(\boldsymbol{\mathcal{B}}(v))$. □

Chen and Micali (2015) design a reward-based single-stage mechanism that ensures that, in equilibrium, the revenue is greater than the second belief minus ϵ, where $\epsilon > 0$ is a reward factor associated to the mechanism. In this one-stage mechanism, agents' beliefs are constant. But should we devise a multi-stage mechanism to achieve a similar result, one may ask the question whether the revenue in equilibrium is (modulo ϵ) greater than the *initial* second belief, or the *last* second belief (before termination) for instance. Such properties can be expressed in $\mathsf{SLK}[\mathcal{F}]$, as we show for the latter one (the former one is easier). Define formulas

$$\varphi_{\text{revenue}} := \mathbf{F}(\text{ter} \wedge \sum_{a \in \text{Ag}}(\mathsf{p}_a))$$

$$\varphi_{\text{last}-\text{2nd}} := \mathbf{F}(\mathbf{X}\text{ter} \wedge \varphi_{\text{2nd}})$$

which compute the final revenue and the second belief before the last round, respectively. Now to check whether a given mechanism satisfies the property in all equilibria of a given kind $\mathrm{E} \in \{\text{NE}, \text{DSE}\}$, one can check whether the following formula has value 1 on this mechanism:

$$\varphi_{\text{2nd},\epsilon} := \forall \boldsymbol{s}.\, \mathrm{E}(\boldsymbol{s}) \rightarrow (\varphi_{\text{last}-\text{2nd}} - \epsilon \leq \varphi_{\text{revenue}})$$

Best-belief benchmark for combinatorial auctions Chen and Micali (2016) propose the *best-belief* benchmark for combinatorial auctions. This benchmark maximizes, over all agents, the maximum revenue each one would be sure to obtain if she were to sell all her currently allocated goods to her opponents, based on her beliefs about their preferences over bundles of goods.

Similar to the second-belief benchmark, one could express the best-belief benchmark using $\mathsf{SLK}[\mathcal{F}]$-formulas. The main difference is that the formula would consider the agents beliefs' about each other's valuations over possible choices. Notice that bundles can be easily encoded as the allocative choices introduced on Section 4.2.

5 Model Checking

In this section we show that model checking $\mathsf{SLK}[\mathcal{F}]$ with imperfect information and memoryless agents is no harder than model checking LTL or classical SL with memoryless agents. Let us first define formally the quantitative model-checking problem for $\mathsf{SLK}[\mathcal{F}]$.

Definition 12. The *model-checking problem* for $\mathsf{SLK}[\mathcal{F}]$ consists in deciding, given a sentence φ, wCGS $\mathcal{G}$, position v in $\mathcal{G}$ and predicate $P \subseteq [-1, 1]$, whether $[\![\varphi]\!]^{\mathcal{G}}(v) \in P$.

For $\mathsf{LTL}[\mathcal{F}]$ model checking is in PSPACE (Almagor, Boker, and Kupferman 2016), and so is model checking SLK with memoryless agents (Cermák et al. 2018). We show that it is also the case for $\mathsf{SLK}[\mathcal{F}]$, as long as the functions $f \in \mathcal{F}$ can be computed in polynomial space. Otherwise, they become the computational bottleneck.

Theorem 2. *Assuming that functions in $\mathcal{F}$ can be computed in polynomial space, model checking $\mathsf{SLK}[\mathcal{F}]$ with imperfect information and memoryless agents is* PSPACE*-complete.*

Proof. We first show that each recursive call only needs at most polynomial space. First, observe that each assignment $\mathcal{A}$ can be stored in space $O((|\text{free}(\varphi)|+|\text{Ag}|)\cdot|V|\cdot\log|\text{Ac}|)$. Next, for the base case it is clear that $[\![p]\!]^{\mathcal{G}}_{\mathcal{A}}(v)$ can be computed in constant space. For strategy quantification $[\![\exists s_a.\,\varphi]\!]^{\mathcal{G}}_{\mathcal{A}}(v)$, besides the recursive call to $[\![\varphi]\!]^{\mathcal{G}}_{\mathcal{A}[s\mapsto\sigma]}(v)$ we need space $O(|V|\cdot\log|\text{Ac}|)$ to store the current strategy and the current maximum value computed. The case for $[\![K_a\varphi]\!]^{\mathcal{G}}_{\mathcal{A}}(v)$ is clear. For $[\![f(\varphi_1,...,\varphi_m)]\!]^{\mathcal{G}}_{\mathcal{A}}(v)$, by assumption f is computed in polynomial space. For $[\![\mathbf{X}\varphi]\!]^{\mathcal{G}}_{\mathcal{A}}(v)$, we only need to observe that the next position in $\text{Out}(\mathcal{A}, v)$ is computed in constant space.

Finally we detail how $[\![\varphi_1\mathbf{U}\varphi_2]\!]^{\mathcal{G}}_{\mathcal{A}}(v)$ is computed. Let $\pi = \text{Out}(v, \mathcal{A})$. Since $\mathcal{G}$ has finitely many positions, there exist two indices $k < l$ such that $\pi_k = \pi_l$, and since strategies depend only on the current position, the suffix of π starting at index l is equal to the suffix starting at index k. So there exist $\rho_1 = v_0...v_{k-1}$ and $\rho_2 = v_k...v_{l-1}$ such that $\pi = \rho_1 \cdot \rho_2^{\omega}$. It follows that

$$[\![\varphi_1\mathbf{U}\varphi_2]\!]^{\mathcal{G}}_{\mathcal{A}}(v) = \sup_{i\geq 0} \min\left([\![\varphi_2]\!]^{\mathcal{G}}_{\mathcal{A}}(\pi_i), \min_{0\leq j<i}[\![\varphi_1]\!]^{\mathcal{G}}_{\mathcal{A}}(\pi_j)\right)$$
$$= \max_{0\leq i<l} \min\left([\![\varphi_2]\!]^{\mathcal{G}}_{\mathcal{A}}(\pi_i), \min_{0\leq j<i}[\![\varphi_1]\!]^{\mathcal{G}}_{\mathcal{A}}(\pi_j)\right)$$

This can be computed by a while loop that increments i, computes $[\![\varphi_2]\!]^{\mathcal{G}}_{\mathcal{A}}(\pi_i)$, $\min_{0\leq j<i}[\![\varphi_1]\!]^{\mathcal{G}}_{\mathcal{A}}(\pi_j)$ and their minimum, records the result if it is bigger than the previous maximum, and stops upon reaching a position that has already been visited. This requires to store the current value of $\min_{0\leq j<i}[\![\varphi_1]\!]^{\mathcal{G}}_{\mathcal{A}}(\pi_j)$, the current maximum, and the list of positions already visited, which are at most $|V|$.

Next, the number of nested recursive calls is at most $|\varphi|$, so the total space needed is bounded by $|\varphi|$ times a polynomial in the size of the input, and is thus polynomial. □

When the set of possible type profiles is finite, as it is the case here, our results from the previous section show that verifying the key MD properties SP, IR, EF, BB and PO on mechanisms can be done by model checking $\mathsf{SLK}[\mathcal{F}]$ formulas for all type profiles of interest.

6 Conclusion

In this paper, we demonstrate how Strategy Logic provides a formal framework expressive enough to reason about core concepts from Mechanism Design in an intuitive way. The ability of SL to naturally express key strategic concepts such as Nash Equilibria, and the possibility to extend it with quantitative aspects and epistemic operators, as we do with $\mathsf{SLK}[\mathcal{F}]$, make it a perfect candidate to become a standard logic for mechanism design, as called for in (Pauly and Wooldridge 2003).

We demonstrate the usefulness of $\mathsf{SLK}[\mathcal{F}]$ with auctions because they "provide a good example of mechanisms which are sufficiently complex to demonstrate the usefulness of formal verification" (Pauly and Wooldridge 2003). We used allocations and payments because of the example considered, but it is enough to replace them with abstract choices to capture any kind of deterministic mechanisms. In addition we showed how our setting allows capturing properties that are central in the design of many types of mechanisms other than auctions, including efficiency and Pareto optimality.

The present setting is enough to capture many kinds of auction mechanisms where memoryless strategies are sufficient to represent the bidders' behaviour, such as one-shot or English auctions. However, when participating in sequential auctions, agents could gather information from other agents' behaviour and act based on what happened in previous steps of the game (Jeitschko 1998). For such situations we plan to study the model-checking problem for $\mathsf{SLK}[\mathcal{F}]$ with memoryful strategies. In the qualitative setting already, imperfect information yields undecidability, but known decidable cases exist (Berthon et al. 2021; Belardinelli et al. 2020). We will investigate them in the quantitative case.

We aim to investigate how the concurrent game structures modelling mechanisms may be generated via compact representations, for instance expressed in the Auction Description Language (Mittelmann and Perrussel 2020), and study how this would impact the complexity of model checking.

Yet another line of future research concerns the impact of bounded rationality on mechanism design (de Clippel, Saran, and Serrano 2018), that could be investigated thanks to the epistemic feature of $\mathsf{SLK}[\mathcal{F}]$.

Acknowledgements

This research is supported by the ANR project AGAPE ANR-18-CE23-0013.

References

Almagor, S.; Boker, U.; and Kupferman, O. 2016. Formally reasoning about quality. *Journal of the ACM* 63(3).

Alur, R.; Henzinger, T. A.; and Kupferman, O. 2002. Alternating-time temporal logic. *Journal of the ACM* 49(5):672–713.

Barthe, G.; Gaboardi, M.; Arias, E.; Hsu, J.; Roth, A.; and Strub, P.-Y. 2016. Computer-aided verification for mechanism design. In *Proc. of the International Conference on Web and Internet Economics (WINE 2016)*.

Belardinelli, F., and Lomuscio, A. 2016. Abstraction-based verification of infinite-state reactive modules. In *Proc. of the European Conference on Artificial Intelligence (ECAI 2016)*.

Belardinelli, F.; Lomuscio, A.; Murano, A.; and Rubin, S. 2020. Verification of multi-agent systems with public actions against strategy logic. *Artif. Intell.* 285.

Berthon, R.; Maubert, B.; Murano, A.; Rubin, S.; and Vardi, M. 2021. Strategy logic with imperfect information. *ACM Trans. Comput. Logic* 22(1).

Bouyer, P.; Kupferman, O.; Markey, N.; Maubert, B.; Murano, A.; and Perelli, G. 2019a. Reasoning about Quality and Fuzziness of Strategic Behaviours. In *Proc. of the International Joint Conference on Artificial Intelligence (IJCAI 2019)*.

Bouyer, P.; Kupferman, O.; Markey, N.; Maubert, B.; Murano, A.; and Perelli, G. 2019b. Reasoning about quality and fuzziness of strategic behaviours. In *Proc. of the International Joint Conference on Artificial Intelligence (IJCAI 2019)*, 1588–1594.

Caminati, M.; Kerber, M.; Lange, C.; and Rowat, C. 2015. Sound auction specification and implementation. In *Proc. of the ACM Conference on Economics and Computation (EC 2015)*.

Cermák, P.; Lomuscio, A.; Mogavero, F.; and Murano, A. 2018. Practical verification of multi-agent systems against slk specifications. *Inf. Comput.* 261:588–614.

Chatterjee, K.; Henzinger, T.; and Piterman, N. 2010. Strategy logic. *Inf. Comput.* 208(6):677–693.

Chen, J., and Micali, S. 2015. Mechanism design with possibilistic beliefs. *Journal of Economic Theory* 156:77–102.

Chen, J., and Micali, S. 2016. Leveraging possibilistic beliefs in unrestricted combinatorial auctions. *Games* 7(4).

de Clippel, G.; Saran, R.; and Serrano, R. 2018. Level-k Mechanism Design. *The Review of Economic Studies* 86(3):1207–1227.

Genesereth, M., and Thielscher, M. 2014. *General game playing*. Synthesis Lectures on Artificial Intelligence and Machine Learning. Morgan & Claypool Publishers.

Jackson, M. O. 2009. *Optimization and Operations Research -Volume III*. EOLSS Publications. chapter Mechanism Theory.

Jamroga, W., and Bulling, N. 2011. Comparing variants of strategic ability. In *Proc. of the International Joint Conference on Artificial Intelligence (IJCAI 11)*, 252–257. AAAI Press.

Jeitschko, T. D. 1998. Learning in sequential auctions. *Southern Economic Journal* 98–112.

Kerber, M.; Lange, C.; and Rowat, C. 2016. An introduction to mechanized reasoning. *Journal of Mathematical Economics* 66:26 – 39.

Krishna, V. 2009. *Auction Theory*. Academic Press.

Laroussinie, F., and Markey, N. 2015. Augmenting ATL with strategy contexts. *Inf. Comput.* 245:98–123.

Maubert, B., and Murano, A. 2018. Reasoning about knowledge and strategies under hierarchical information. In *Proc. of the International Conference on Principles of Knowledge Representation and Reasoning (KR 2018)*, 530–540. AAAI Press.

Mittelmann, M., and Perrussel, L. 2020. Auction description language (ADL): a general framework for representing auction-based markets. In *Proc. of the European Conference on Artificial Intelligence (ECAI 2020)*.

Mogavero, F.; Murano, A.; Perelli, G.; and Vardi, M. 2014. Reasoning about strategies: On the model-checking problem. *ACM Trans. Comput. Log.* 15(4).

Nisan, N.; Roughgarden, T.; Tardos, É.; and Vazirani, V. 2007. *Algorithmic Game Theory*. Cambridge University Press.

Padgham, L.; Thangarajah, J.; and Winikoff, M. 2008. Prometheus design tool. In *Proc. of the AAAI Conference on Artificial Intelligence (AAAI 2008)*.

Parkes, D., and Ungar, L. 2001. *Iterative combinatorial auctions: Achieving economic and computational efficiency*. University of Pennsylvania Philadelphia, PA.

Pauly, M., and Wooldridge, M. 2003. Logic for mechanism design–a manifesto. In *Proc. of the 2003 Workshop on Game Theory and Decision Theory in Agent Systems (GTDT 2003)*.

Sandholm, T. 2003. Automated mechanism design: A new application area for search algorithms. In *Proc. of the International Conference on Principles and Practice of Constraint Programming (CP 2003)*.

Troquard, N.; van der Hoek, W.; and Wooldridge, M. 2011. Reasoning about Social Choice Functions. *JPL* 40(4):473––498.

Wooldridge, M.; Agotnes, T.; Dunne, P.; and Van der Hoek, W. 2007. Logic for automated mechanism design-a progress report. In *Proc. of the AAAI Conference on Artificial Intelligence (AAAI 2007)*.

Yadav, N., and Thangarajah, J. 2016. Checking the conformance of requirements in agent designs using atl. In *Proc. of the European Conference on Artificial Intelligence (ECAI - 2016)*.